Genstat®

Statistics

www.vsni.co.uk

VSNi

The Guide to the Genstat® Command Language (Release 23)

Part 2: Statistics

Genstat is developed by VSN International Ltd, in collaboration with practising statisticians at Rothamsted and other organisations in Britain, Australia, New Zealand and The Netherlands.

Published by: VSN International, 2 Amberside, Wood Lane,
Hemel Hempstead, Hertfordshire HP2 4TP, UK
E-mail: info@genstat.co.uk
Website: http://www.genstat.co.uk/

First published 2000, as *The Guide to Genstat*
This edition published 2023, for Genstat Release 23

Citation: VSN International (2023). *The Guide to the Genstat Command Language (Release 23), Part 2 Statistics*. VSN International, Hemel Hempstead, UK.

Contents

1 Introduction 1
- 1.1 Syntax 1
- 1.2 Data structures 3
- 1.3 Input and output 5
- 1.4 Calculations and manipulation 7
- 1.5 Programming in Genstat 13
- 1.6 Graphics 16

2 Basic statistics and exploratory analysis 19
- 2.1 Summary statistics 22
  - 2.1.1 The DESCRIBE procedure 22
  - 2.1.2 Circular data: the CDESCRIBE procedure 24
- 2.2 Exploring the distribution of the data 26
  - 2.2.1 Histograms 26
  - 2.2.2 Boxplots 29
  - 2.2.3 Rugplots 32
  - 2.2.4 Stem-and-leaf plots 34
  - 2.2.5 Tally tables and plots 36
  - 2.2.6 Dotplots 38
  - 2.2.7 Probability plots 40
  - 2.2.8 Kernel density estimation 43
  - 2.2.9 Plots of circular data 46
  - 2.2.10 Estimating the parameters of a distribution 48
  - 2.2.11 Tests for Normality 59
  - 2.2.12 Goodness of fit tests for other continuous distributions 61
- 2.3 Comparison of groups of data 66
  - 2.3.1 The t-test 66
  - 2.3.2 One-way analysis of variance 71
  - 2.3.3 Two-way analysis of variance 74
  - 2.3.4 Binomial data 81
  - 2.3.5 Poisson data 84
- 2.4 One-sample nonparametric tests 86
  - 2.4.1 The Wilcoxon test 86
  - 2.4.2 The sign test 87
  - 2.4.3 The runs test 88
- 2.5 Two-sample nonparametric tests 89
  - 2.5.1 The Mann-Whitney test 90
  - 2.5.2 The Kolmogorov-Smirnoff test 91
- 2.6 Nonparametric analysis of variance 93
  - 2.6.1 The Kruskal-Wallis one-way analysis of variance 93
  - 2.6.2 Friedman's nonparametric analysis of variance 94
  - 2.6.3 Steel's many-one rank test 95
- 2.7 Plotting relationships between variables 97
  - 2.7.1 Scatter plots 97
  - 2.7.2 Parallel coordinates 98
- 2.8 Correlation 100
  - 2.8.1 Product-moment correlation coefficient 100
  - 2.8.2 Spearman's rank correlation coefficient 102
  - 2.8.3 Kendall's rank correlation coefficient τ 103
  - 2.8.4 Kendall's coefficient of concordance 105
  - 2.8.5 The kappa coefficient 106
  - 2.8.6 The gamma statistic 108
  - 2.8.7 Lin's concordance correlation coefficient 109
  - 2.8.8 Bland-Altman plots 111
- 2.9 Tests for independence and changes in two-way tables 114
  - 2.9.1 The chi-square test 115
  - 2.9.2 Fisher's exact test and permutation tests 116
  - 2.9.3 McNemar's test 119
  - 2.9.4 Cochran's Q test 120
  - 2.9.6 Cochran-Armitage chi-square test for trend 123
- 2.10 Six sigma 124
  - 2.10.1 Control charts for mean, standard deviation or range 125
  - 2.10.2 CUSUM tables 128
  - 2.10.3 Moving-average control charts 130
  - 2.10.4 Control charts for proportions of defective items 131
  - 2.10.5 Control charts for numbers of defects 133
  - 2.10.6 Capability statistics 135
- 2.11 Ecological data 136
  - 2.11.1 Diversity measures 137
  - 2.11.2 Plotting species abundance data 139
  - 2.11.3 Species abundance models 141
  - 2.11.4 Niche-based models 143
  - 2.11.5 Rarefaction 145
  - 2.11.6 Species accumulation curves 148
  - 2.11.7 Nonparametric estimation of species richness 150
  - 2.11.8 Lorenz curve and Gini coefficient 154

3 Regression analysis 156
- 3.1 Simple linear regression 160
  - 3.1.1 The MODEL directive 162
  - 3.1.2 The FIT directive 166
  - 3.1.3 Further output: the RDISPLAY directive 171
  - 3.1.4 Storing the results: the RKEEP directive 172

3.1.5 Saving the results to a spreadsheet the RSPREADSHEET procedure 176
3.1.6 Displaying the model: the RGRAPH procedure 178
3.1.7 Diagnostics: the RCHECK procedure 180
3.1.8 Power calculations: the RPOWER procedure 182
3.1.9 Permutation and exact tests: the RPERMTEST procedure 185
3.2 Multiple linear regression 187
3.2.1 Extensions to the FIT and RDISPLAY directives in multiple linear regression 188
3.2.2 Saving information about individual regression terms: the RKESTIMATES directive 189
3.2.3 Defining the maximal model: the TERMS directive 191
3.2.4 Modifying the model: the ADD, DROP and SWITCH directives 193
3.2.5 Evaluating changes to the model: the TRY directive 197
3.2.6 Wald tests to assess whether terms can be dropped: the RWALD procedure 198
3.2.7 Stepwise regression: the STEP directive 200
3.2.8 Searching for the best regression model: the RSEARCH procedure 205
3.2.9 Screening tests for terms in a regression model: the RSCREEN procedure 210
3.3 Linear regression with grouped or qualitative data 212
3.3.1 Formulae in parameters of regression directives 214
3.3.2 Parameterization of factors 215
3.3.3 Parameterization of interactions, and marginality 217
3.3.4 Forming predictions: the PREDICT directive 218
3.3.5 Comparisons between predictions: the RCOMPARISONS and RTCOMPARISONS procedures 229
3.3.6 Plots of estimates: the RDESTIMATES procedure 235
3.4 Polynomials and additive models 237
3.4.1 Polynomial regression 237
3.4.2 Orthogonal polynomials and general functions 241
3.4.3 Cubic smoothing splines 243
3.4.4 Locally weighted regression 246
3.4.5 Interactions with SSPLINE or LOESS functions 249
3.4.6 The RLOESSGROUPS procedure 255
3.5 Generalized linear models 258
3.5.1 Introduction to generalized linear models 262
3.5.2 The deviance 269
3.5.3 Modifications to output and the RKEEP and PREDICT directives 271
3.5.4 The RCYCLE directive 274
3.5.5 Models for multinomial and ordinal responses 276
3.5.6 Non-standard distributions and link functions 278
3.5.7 Generalized additive models 280
3.5.8 Generalized nonlinear models 281
3.5.9 Probit analysis 288
3.5.10 Generalized linear mixed models 292
3.5.11 Hierarchical generalized linear models 312
3.5.12 Generalized estimating equations 331
3.5.13 Zero-inflated regression models 338
3.6 Generalized least-squares 343
3.7 Standard nonlinear curves 345
3.7.1 The FITCURVE directive 347
3.7.2 Distributions and constraints in curve fitting 351
3.7.3 Parallel curve analysis 352
3.7.4 Modifications to regression output and the RKEEP directive 355
3.7.5 Functions of parameters: the RFUNCTION directive 357
3.7.6 Controlling the start of the search with the RCYCLE directive 358
3.7.7 Standard errors for linear parameters: the RCURVECOMMONNONLINEAR procedure 359
3.8 General nonlinear regression, and minimizing a function 361
3.8.1 Fitting nonlinear models 363
3.8.2 Nonlinear regression for models with some linear parameters 365
3.8.3 Nonlinear regression models with no linear parameters 367
3.8.4 General nonlinear models 369
3.9 Regression trees 372
3.9.1 Constructing a regression tree 372
3.9.2 Displaying a regression tree 374
3.9.3 Pruning a regression tree 375

3.9.4 Predictions from a regression tree 378
3.9.5 Predictions from a regression tree 379
3.10 Quantile regression 380
3.10.1 The RQLINEAR procedure 380

4 Analysis of variance and design of experiments 386
4.1 Designs with a single error term 393
4.1.1 The TREATMENTSTRUCTURE directive 395
4.1.2 The ANOVA directive 397
4.1.3 Output from ANOVA 401
4.1.4 Procedures for examining residuals 410
4.1.5 Displaying tables of means 413
4.1.6 Checking the assumptions 418
4.1.7 Permutation and exact tests for analysis of variance 419
4.1.8 Simultaneous confidence intervals for means 421
4.1.9 Multiple comparison tests 423
4.1.10 Simultaneous confidence limits around a control 426
4.2 Designs with several error terms 427
4.2.1 The BLOCKSTRUCTURE directive 427
4.2.2 The ABLUPS procedure 434
4.2.3 Multitiered designs 436
4.3 Analysis of covariance 440
4.3.1 The COVARIATE directive 442
4.3.2 The ACOVARIATES procedure 445
4.4 Missing values 447
4.5 Contrasts between treatments 448
4.5.1 The APOLYNOMIAL procedure 455
4.5.2 The ADPOLYNOMIAL procedure 456
4.6 Saving information from an analysis of variance 458
4.6.1 The AKEEP directive 458
4.6.2 The ASTATUS procedure 467
4.6.3 The ASPREADSHEET procedure 468
4.7 Non-orthogonality and balance 470
4.7.1 Efficiency factors 470
4.7.2 Balance 475
4.7.3 Pseudo-factors 476
4.7.4 Non-orthogonality between treatment terms 480
4.7.5 The method of analysis 481
4.7.6 Screening tests for unbalanced designs 483
4.8 Unbalanced designs 485
4.8.1 The AUNBALANCED procedure 486
4.8.2 The AUDISPLAY procedure 490
4.8.3 The AUGRAPH procedure 497
4.8.4 The AUKEEP procedure 500
4.8.5 The AUPREDICT procedure 502
4.8.6 The AUSPREADSHEET procedure 504
4.8.7 The AOVANYHOW procedure 506
4.8.8 The AN1ADVICE procedure 509
4.9 Selecting and generating an experimental design 511
4.9.1 Orthogonal hierarchical designs 514
4.9.2 Complete and fractional factorial designs 518
4.9.3 Factorial designs with confounding 523
4.9.4 Latin and Youden squares 527
4.9.5 Semi-Latin squares 537
4.9.6 Square lattice and lattice square designs 541
4.9.7 Alpha designs 544
4.9.8 Balanced-incomplete-block designs 546
4.9.9 Cyclic designs 549
4.9.10 Neighbour-balanced designs 551
4.9.11 Central composite designs 554
4.9.12 Box-Behnken designs 556
4.9.13 Plackett Burman (main effect) designs 558
4.9.14 Response surface designs 560
4.9.15 Designs for nonlinear and generalized linear models 564
4.9.16 Reference-level designs 567
4.9.17 Loop designs 569
4.10 Displaying a design 571
4.10.1 Printing a design: the PDESIGN procedure 571
4.10.2 Plotting the plan of a design: the DDESIGN procedure 572
4.10.3 Plans and data forms in spreadsheets: the ADSPREADSHEET procedure 574
4.11 Randomization 577
4.11.1 The RANDOMIZE directive 578
4.11.2 The ARANDOMIZE procedure 581
4.12 Sample size and power calculations 582
4.12.1 Sample size for t-tests 583
4.12.2 Sample size for analysis of variance 589
4.12.3 Power for analysis of variance 593
4.12.4 Sizes of effects and contrasts detectable in an analysis of

variance 596
4.12.5 Sample size for binomial tests 599
4.12.6 Sample size for Poisson tests 601
4.12.7 Sample size for sign tests 603
4.12.8 Sample-size for McNemar's test 604
4.12.9 Sample size for the Mann-Whitney test 606
4.12.10 Sample size for correlations 608
4.12.11 Sample size for Lin's concordance correlation coefficient 610
4.13 Design tools 612
4.13.1 Generating factor values: the GENERATE directive 612
4.13.2 Generating factor values using design keys: the AKEY directive 614
4.13.3 Adding extra units to a design: the AMERGE procedure 617
4.13.4 Taking the product of two experimental designs: the APRODUCT procedure 619
4.13.5 Augmented designs: the AFAUGMENTED procedure 621
4.13.6 Construction of design keys 627
4.13.7 Forming pseudo-factors from a design key 636
4.13.8 Forming the basic contrasts of a model term 640
4.13.9 Minimum aberration designs 640

5 REML analysis of mixed models 642
5.1 Models for REML estimation 647
5.1.1 Fixed and random effects 647
5.1.2 The linear mixed model with independent random effects 648
5.1.3 REML estimation 650
5.2 Specifying linear mixed models 651
5.2.1 The VCOMPONENTS directive 651
5.2.2 The fixed model 652
5.2.3 The random model 654
5.2.4 Setting initial values and constraints on variance components 655
5.3 Analysing linear mixed models 657
5.3.1 The REML directive 658
5.3.2 Further output: the VDISPLAY directive 662
5.3.3 Tables of means and effects for fixed and random terms 664
5.3.4 Plots of means and effects 669
5.3.5 Residual plots 675
5.3.6 Assessing and plotting fixed effects 678
5.3.7 Assessing random effects 697
5.3.8 Examining sources of variability 707
5.3.9 Technical details of the Fisher method and absorbing factors 714
5.3.10 Controlling advanced features of the REML algorithm 717
5.4 Modelling variance structures 718
5.4.1 The VSTRUCTURE directive 718
5.4.2 Displaying the model: the VSTATUS directive 726
5.4.3 A repeated measurements example 727
5.4.4 An example of spatial analysis of a field experiment 735
5.4.5 An example of random coefficient regression 742
5.4.6 Direct products 745
5.5 Predictions from a REML analysis 747
5.5.1 The VPREDICT directive 747
5.5.2 The VTCOMPARISONS procedure 751
5.6 Generating an inverse relationship matrix from a pedigree 753
5.6.1 The VPEDIGREE directive 753
5.6.2 The VFPEDIGREE procedure 755
5.7 Including cubic spline terms in the random model 756
5.8 Combined analyses of several experiments 760
5.8.1 The VRMETAMODEL procedure 761
5.8.2 The VRESIDUAL directive 763
5.9 Saving information from a REML analysis 765
5.9.1 The VKEEP directive 765
5.9.2 The VFRESIDUALS procedure 771
5.9.3 The VSPREADSHEET procedure 773
5.9.4 The VFIXEDTESTS procedure 774

6 Multivariate and cluster analysis 776
6.1 Measures of association 779
6.1.1 Forming sums of squares and products 779
6.1.2 Forming similarity matrices: the FSIMILARITY directive 784
6.1.3 Forming similarities between groups: the HREDUCE directive 788
6.1.4 Forming associations using CALCULATE 790

6.1.5 Assessing the association between similarity matrices: the MANTEL procedure 791
6.1.6 Nonparametric analysis of similarities: the ECANOSIM procedure 793
6.2 Principal components analysis 795
6.2.1 The PCP directive 795
6.2.2 Scree diagrams of latent roots: the LRVSCREE procedure 801
6.3 Canonical variates analysis 803
6.3.1 The CVA directive 803
6.3.2 Canonical variate scores: the CVASCORES procedure 807
6.3.3 Plotting canonical variate scores: the CVAPLOT procedure 808
6.3.4 Plotting large numbers of canonical variate scores: the CVATRELLIS procedure 810
6.4 Factor rotation: the FACROTATE directive 811
6.5 Discriminant analysis 814
6.5.1 The DISCRIMINATE procedure 815
6.5.2 The SDISCRIMINATE procedure 818
6.5.3 The QDISCRIMINATE procedure 821
6.6 Multivariate analysis of variance and regression 824
6.6.1 The MANOVA procedure 824
6.6.2 The RMULTIVARIATE procedure 827
6.6.3 The MVAOD procedure 829
6.7 Ridge and principal component regression: the RIDGE procedure 831
6.8 Partial least squares: the PLS procedure 834
6.9 Canonical correlation analysis: the CANCORRELATION procedure 839
6.10 Principal coordinates analysis 840
6.10.1 The PCO directive 842
6.10.2 The ADDPOINTS directive 848
6.10.3 Relating associations to data variables: the PCORELATE directive 850
6.11 Factor analysis: the FCA directive 853
6.12 Multidimensional scaling: the MDS directive 856
6.13 Correspondence analysis 860
6.13.1 The CORANALYSIS procedure 860
6.13.2 The MCORANALYSIS procedure 864
6.13.3 The CABIPLOT procedure 866
6.14 Redundancy analysis: the RDA procedure 869
6.15 Canonical correspondence analysis: the CCA procedure 873
6.16 Biplots 878
6.16.1 The DBIPLOT procedure 878
6.16.2 The CRBIPLOT procedure 881
6.16.3 The CRTRIPLOT procedure 883
6.17 Analysis of skew-symmetry: the SKEWSYMMETRY procedure 886
6.18 Procrustes rotation 888
6.18.1 The ROTATE directive 889
6.18.2 Generalized Procrustes rotation: the GENPROCRUSTES procedure 892
6.18.3 Multiple Procrustes analysis: the PCOPROCRUSTES procedure 896
6.19 Hierarchical cluster analysis 899
6.19.1 The HCLUSTER directive 900
6.19.2 Displaying and saving information from a cluster analysis: the HDISPLAY directive 902
6.19.3 Examining the data by groups: the HLIST directive 906
6.19.4 Relating groups to the original data variables: the HSUMMARIZE directive 909
6.19.5 Plotting the dendrogram: the DDENDROGRAM procedure 911
6.19.6 Plotting a minimum spanning tree: the DMST procedure 916
6.19.7 Comparing clusterings: the HCOMPAREGROUPINGS procedure 918
6.19.8 Bootstrap analyses to assess the reliability of the clusters: the HBOOTSTRAP procedure 921
6.20 Non-hierarchical classification 924
6.20.1 The CLUSTER directive 925
6.20.2 Determining an initial classification: the CLASSIFY procedure 934
6.20.3 Large data sets: the PCPCLUSTER procedure 936
6.21 Classification trees 942
6.21.1 Constructing a classification tree 942
6.21.2 Displaying a classification tree 944
6.21.3 Pruning a classification tree 946
6.21.4 Identification using a classification tree 948
6.21.5 Saving information from a classification tree 949
6.22 Identification 950
6.22.1 Constructing an identification key

950
6.22.2 Displaying an identification key 953
6.22.3 Identification using a key 954
6.22.4 Saving information from a key 955
6.22.5 Interactive identification 956
6.22.6 Irredundant test sets 958

7 Analysis of time series 963
7.1 Correlation 965
7.1.1 The CORRELATE directive 965
7.1.2 Autocorrelation 966
7.1.3 The BJIDENTIFY procedure 968
7.1.4 Partial autocorrelation 969
7.1.5 Cross-correlation 970
7.2 Fourier transformation 972
7.2.1 The FOURIER directive 972
7.2.2 Cosine transformation of a real series 973
7.2.3 Fourier transformation of a real series 973
7.2.4 Fourier transformation of a complex series 974
7.2.5 Fourier transformation of a conjugate sequence 975
7.2.6 The SMOOTHSPECTRUM procedure 976
7.2.7 The DFOURIER procedure 979
7.2.8 The MCROSSPECTRUM procedure 983
7.3 ARIMA modelling 986
7.3.1 The BJESTIMATE procedure 988
7.3.2 Defining ARIMA models for time series with the TSM directive 991
7.3.3 The TFIT directive 993
7.3.4 Technical information about how Genstat fits ARIMA models 999
7.3.5 The TDISPLAY directive 1001
7.3.6 The TKEEP directive 1002
7.3.7 The TFORECAST directive 1005
7.3.8 The BJFORECAST procedure 1009
7.4 Regression with autocorrelated (ARIMA) errors 1011
7.4.1 The TRANSFERFUNCTION directive 1013
7.4.2 Extensions to the TFIT directive for regression with ARIMA errors 1014
7.4.3 Extensions to the TFORECAST directive for regression with ARIMA errors 1015
7.5 Multi-input transfer-function models 1016
7.5.1 Declaring transfer-function models with the TSM directive 1018
7.5.2 Extensions to the TRANSFERFUNCTION directive for multi-input models 1019
7.5.3 Extensions to the TFIT directive for multi-input models 1020
7.5.4 Extensions to the TKEEP directive for multi-input models 1020
7.5.5 Extensions to the TFORECAST directive for multi-input models 1020
7.6 Filtering time series 1021
7.6.1 The TFILTER directive 1022
7.7 Forming preliminary estimates and displaying models 1024
7.7.1 Preliminary estimation of ARIMA model parameters 1025
7.7.2 Preliminary estimation of transfer-function model parameters 1026
7.7.3 The TSUMMARIZE directive 1027
7.7.4 Deriving the generalized form of a time-series model 1028

8 Spatial and temporal modelling 1030
8.1 Repeated measurements 1033
8.1.1 Plotting repeated measurements 1033
8.1.2 Analysis of polynomial contrasts 1035
8.1.3 Repeated-measures analysis of variance 1037
8.1.4 Multivariate analysis of variance 1040
8.1.5 Ante-dependence structure 1041
8.1.6 Regression with correlated errors 1045
8.2 Survival analysis 1050
8.2.1 Kaplan-Meier estimation 1050
8.2.2 Nonparametric tests 1054
8.2.3 Life-table estimates 1055
8.2.4 Survival distributions 1058
8.2.5 Proportional hazards model 1062
8.3 Geostatistics 1066
8.3.1 The FVARIOGRAM directive 1068
8.3.2 The MVARIOGRAM procedure 1072
8.3.3 The DVARIOGRAM procedure 1078
8.3.4 The KRIGE directive 1079
8.3.5 Coregionalization and cokriging 1084

8.3.6 The FCOVARIOGRAM directive 1089
8.3.7 The MCOVARIOGRAM directive 1093
8.3.9 The COKRIGE directive 1097
8.4 Analysis of spatial point patterns 1103
8.5 Clusters of points in multi-dimensional space 1107
8.5.1 The PTFCLUSTERS procedure 1107
8.5.2 The PTFILLCLUSTERS procedure 1112

References 1115

Index 1125

1 Introduction

This book, Part 2 of the *Guide to the Genstat Command Language*, describes the statistical facilities in Genstat, reviewing the underlying methodology, explaining the output, and describing the relevant Genstat commands. Most of the analyses supported by Genstat can be run using the menus in Genstat *for Windows*. However, the menus themselves operate by generating Genstat commands – and you can see these recorded in the Input log. So even if you are using Genstat in a Windows environment, you may still want to examine the commands, or to save them as an audit trail of the analyses that have been done. You may also want to issue your own commands, in order to gain additional flexibility, to use more specialized methods, or simply to access the desired analysis more directly. Alternatively, you may want to develop your own methods of analysis, using the Genstat command language as a high-level programming language. Programs can be formed into *procedures* for convenient future use – in fact, many of the advanced analyses in Genstat are implemented in this way, and distributed as part of the refereed and officially supported Genstat Procedure Library (see Part 3 of the *Genstat Reference Manual*).

Unlike the *Genstat Reference Manual*, which describes the commands one at a time, here the information is categorized by type of analysis. The facilities are introduced by means of examples, which illustrate the commonest analyses and explain the output that can be obtained. First, though, this chapter gives a brief description of the syntax of the Genstat language, and summarizes the facilities for data manipulation, which were described in Part 1 of this Guide. References below to Part 1 are prefixed by "1:". So, for example, 1:1.2 refers to Part 1 Section 1.2, while 1.2 refers to Section 1.2 in this book.

1.1 Syntax

Input to Genstat is known as a Genstat *program*. This is made up of statements each of which may use one of the standard Genstat commands (known as *directives*); alternatively, it may use a Genstat *procedure*, that is, a subprogram of statements. You can write your own procedures, or use those in the Library distributed with Genstat, or in the library provided at your site.

Whether the statement uses a directive or a procedure, the syntax is identical. First you give the name of the directive (or procedure), then options, and then parameters. Finally, you indicate the end of the statement, either by typing a colon or by ending the line (by typing <RETURN>). Long statements can be continued onto succeeding lines by typing the continuation character (\) before <RETURN>.

Some statements will have neither options nor parameters: for example

```
PAGE
```

to start a new page in output. Others may have no options: for example

```
PRINT STRUCTURE=X,Y; DECIMALS=0,2
```

prints the contents of data structures X and Y with zero and two decimal places respectively. In this statement, there are two parameter settings defining two lists running in parallel. Parameter settings are always in parallel like this, and are separated from one another by semicolons. Options are enclosed in square brackets, and set aspects that apply to all the (parallel) parameter values. They are also separated from one another by semicolons. For example

```
PRINT [CHANNEL=2; INDENTATION=5] STRUCTURE=X,Y;\
  DECIMALS=0,2
```

prints X and Y to output channel 2 with a five-character indentation at the start of each line. Nearly all options, and some parameters, have default values chosen to be those required most often, and so will usually not need to be set.

Settings of options and parameters can be lists (as above), expressions or formulae. Lists may

be of numbers (as with DECIMALS above), or identifiers (as with STRUCTURE) or strings. An *identifier* is the name that you give to a Genstat data structure (for example X or Y), and which you then use to refer to it in the program. They must start with a *letter* (for Genstat this means the alphabetic characters A to Z, in capitals or lower case, as well as the percent and underline characters) and then contain either letters or *digits* (the numerical characters 0 to 9); Genstat takes notice of only the first 32 characters. (This is the default in Releases 4.2 onwards, but you can use the SET directive to request that Genstat take notice of only the first eight characters as in earlier releases.) Where a list of identifiers provides *input* to a directive or procedure, you can put an expression instead; this will then be evaluated (to give a list of identifiers containing the results) before the directive or procedure is used. A string is a list of characters. Usually the start and end of the string must be marked by a single quote ('). Strings occur within the text data structure. Also, the settings of some options and parameters are lists of *string tokens* that can be chosen from a defined list; these do not need to start and end with single quotes. The separator between items in lists is comma; spaces can be included anywhere between items but do *not* act as separators. Formal definitions of expressions, formulae, and all the other concepts of the Genstat language are in 1:1.2.

Names of directives, procedures, options and parameters are examples of Genstat *system words*. They can be given in capital or small letters (or in mixtures of both) and, provided you are only using directives and official Genstat Library procedures, they can always be abbreviated to four characters. But of course, if you or your site have defined your own procedures, you may have chosen names that differ only in the fifth or subsequent characters. If you supply more characters, Genstat will check the name up to the 32nd character, and ignore any characters after that. (You can, however, use the SET directive to request that Genstat also ignores the ninth and subsequent characters, as in releases before 4.2.)

Names of options and parameters can often be abbreviated to fewer than four characters. Each option name can be abbreviated to the minimum number of letters needed to distinguish it from the options that precede it in the prescribed order for the directive or procedure concerned. Characters up to the 32nd (or the eighth if short wordlengths have been requested) must match the appropriate part of the full form; subsequent characters are ignored. For example, here are the options of the FIT directive (3.1.2), with the minimum form of each name printed in bold:

PRINT, **C**ALCULATION, **O**WN, **CO**NSTANT, **F**ACTORIAL, **P**OOL,
DENOMINATOR, **N**OMESSAGE, **FP**ROBABILITY, **T**PROBABILITY,
SELECTION, **NG**RIDLINES, **SELI**NEAR, **I**NOWN, **OU**TOWN

Notice for example that the minimum for FPROBABILITY is FP, since F on its own would not distinguish it from FACTORIAL which precedes it in this prescribed order. Likewise, each option name can be abbreviated to the minimum number of letters needed to distinguish it from the options that precede it in the prescribed order for the directive or procedure concerned.

There are also rules by which the option or parameter name, with its accompanying equals character, can be omitted altogether. The most useful of these is that, if the first parameter of the directive is the one that comes first in the statement, then the name of the parameter can be omitted: for example

```
PRINT [CHANNEL=2; INDENTATION=5] X,Y; DECIMALS=0,2
```

as STRUCTURE is the first parameter of PRINT. The same rule holds for options:

```
PRINT [2; INDENTATION=5] X,Y; DECIMALS=0,2
```

as CHANNEL is the first option of PRINT. Full details of the rules are in 1:1.2.

A final point about the first parameter is that its setting determines the length of the parallel lists. The lists for other parameters will be repeated (or recycled) if they are shorter. (If they are longer, Genstat gives an error diagnostic.) For example

```
PRINT A,B,C,D; DECIMALS=0,2
```

prints A with zero decimal places, B with two, and then (recycling the DECIMALS list), C with

zero and D with two.

To make the language easier to learn and remember, the "vocabulary" of the directives and Library procedures has been standardized, for example, to avoid using the same option or parameter name for different purposes in different commands, or the same name for different purposes. In particular, commands that produce output should have a PRINT option with a list of available *string tokens* that correspond to the various output components (and, where it occurs, PRINT will always be the first option). For example, the PRINT option of the FIT directive (3.1.2) is defined as follows:

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, grid); default mode, summ, esti or grid if NGRIDLINES is set

So, to print the model, parameter estimates and a table of fitted values, residuals etc you would need to specify

```
PRINT=model,estimates,fittedvalues
```

The same rules apply for the string-token settings of options and parameters as for the option and parameter names. They may be typed in capital or small letters (or mixtures), and each one can be abbreviated to the minimum number of characters necessary to distinguish it from earlier tokens in the list given in the definition of the option or parameter. If more than that number are given, the extra characters must match the full form up to the 32nd character (or the eighth if short wordlengths have been requested). So the model token above can be abbreviated to just m, as this is listed first in the definition of the syntax; whereas monitoring can be abbreviated only to mon. To suppress printed output from FIT, or from any other command with a PRINT option, you should specify

```
PRINT=*
```

The setting * denotes an empty (or missing) string token, implying no output.

There are also various standard prefixes: for example, A for analysis of variance and design of experiments, R for regression and generalized linear models, V for variance components and REML, and so on. So, there are directives ADISPLAY, RDISPLAY and VDISPLAY which allow you to display further output from an analysis of variance, regression or REML analysis, respectively, and directives AKEEP, RKEEP and VKEEP that allow you to copy results from these analyses into Genstat data structures. The full currently used prefixes are listed in the Instructions for Authors of Library Procedures, obtainable from the NOTICE procedure

```
NOTICE [PRINT=instructions]
```

Genstat programs can thus be presented in a wide range of styles and formats. For clarity, however, we have imposed some conventions on the examples in this book. The use of spaces is standardized. System words are given in full and in capitals; the only exception is that the name, and corresponding equals character, of the main parameter of a directive will usually be omitted. String tokens are given in full and in small letters. Identifiers will begin with a capital; any other letters are in lower case. There is usually only one statement per line, unless this is very wasteful of space; continuation lines are indented. We hope these conventions will help you to recognize the items, both in the descriptions of syntax and in the examples. However, in your own programs, you can use whatever style you find most convenient.

1.2 Data structures

Data structures store the information on which a Genstat program operates. Examples include data for statistical analyses, coordinates for graphs, text for annotation, and so on. You can also store almost anything that can be printed in an analysis. This enables you to extend the range of facilities that Genstat offers, by taking information from one directive and using it as input for

another. To allow you to do this, Genstat has a comprehensive set of different structures. You can define the identifier of a structure, together with its type, using a directive known as a *declaration*. The directive for declaring each type of structure has the same name as given to that type of structure, for example SCALAR to declare a scalar (or single-valued numerical structure), and so on. These are the directives, with details of their corresponding data structures and references to the sections where they are described.

SCALAR	single number (1:2.2.1)
VARIATE	series of numbers (1:2.3.1)
TEXT	series of character strings i.e. lines of text (1:2.3.2)
FACTOR	series of group allocations, using a pre-defined set of numbers or strings to indicate the groups (1:2.3.3)
MATRIX	rectangular matrix (1:2.4.1)
DIAGONALMATRIX	diagonal matrix (1:2.4.2)
SYMMETRICMATRIX	symmetric matrix (1:2.4.3)
TABLE	table – to store tabular summaries like means, totals etc (1:2.5)
DUMMY	single identifier (1:2.2.2)
POINTER	series of identifiers e.g. to represent a set of structures (1:2.6)
EXPRESSION	arithmetic expression (1:2.2.3)
FORMULA	model formula – to be fitted in a statistical analysis (1:2.2.4)
LRV	latent roots and vectors (1:2.7.1)
SSPM	sums of squares and products with associated information such as means (1:2.7.2)
TSM	model for Box-Jenkins modelling of time series (1:2.7.3)
TREE	tree, as used to represent classification trees, identification keys and regression trees (1:2.8, 3.9, 6.20, 6,21)

You can also define data structures whose contents are customized for particular tasks (1:2.7.4).

STRUCTURE	defines a customized data structure
DECLARE	declares one or more customized data structures

In the standard version of Genstat, your program can contain as many data structures of each type as you like, limited only by the total amount of workspace that they occupy. Student Versions may have additional constraints, explained in the accompanying on-line help or documentation.

Chapter 2 of Part 1 also describes several additional commands that are useful for managing your data structures.

DELETE	allows values of data structures to be deleted to save space within Genstat; attributes can also be deleted so that the structure can be redefined, for example as another type (1:2.10.1)
RENAME	renames a data structure, to give it a new identifier (1:2.10.2)
DUPLICATE	forms new data structures with attributes taken from an existing structure (1:2.10.3)
PDUPLICATE	duplicates a pointer, with all its components (1:2.10.4)
LIST	lists details of the data structures currently in store (1:2.11.1)
DUMP	prints attributes and values of data structures (1:2.11.2)
GETATTRIBUTE	accesses attributes of data structures such as their types,

sizes and so on (1:2.11.3)

1.3 Input and output

Genstat supports a wide variety of styles and formats for data entry. The simplest method is provided by the FILEREAD procedure, which provides the basis of the Read Data from ASCII file in Genstat *for Windows*. The Windows implementation also allows a wide range of spreadsheet files to be imported, as well as save-files from many other statistical systems and data bases. The most general facilities are provided by the READ directive, which caters for a wide variety of styles and formats, and can also rescale and sort the data values as they are read:

READ	provides general facilities for reading data from the keyboard, an input file or a Genstat text structure (see Sections 1:3.1.2 to 1:3.1.12, and 1:3.7)
FILEREAD	provides a convenient way of reading values into a set of variates, factors and or texts which all have equal lengths; the data values are provided in a rectangular layout, in a separate file (1:3.1.1)
TX2VARIATE	reads values into a variate from a text structure (1:4.5.3)

Genstat can produce output in either plain-text or a "formatted" style written in either RTF, HTML or LaTeX. The style of an output channel is set when the channel is opened, either by the OPEN directive (1:3.3.1) or by the command used to run Genstat (1:1.1.2). You can also switch a formatted output channel temporarily into the plain-text style (and back into its formatted style) using the OUTPUT directive (1:3.4.4). Alternatively, in Genstat *for Windows*, this is done using the View menu.

The plain-text style assumes that every character occupies an identical width on the page. This was the situation with the line printers that were originally used for computer output. In more modern environments, such as Microsoft® Windows™, this can be achieved by using a "non-proportional" font such as Courier. In plain text, columns of output are lined up by inserting space characters. The formatted styles insert tab characters or use tabular modes of output, which are likely to be more convenient if you want to import the output into a wordprocessor, web page or scientific publication. In the formatted styles, you can also include "typesetting commands" inside a textual string to generate italic or bold fonts, subscripts or superscripts, and Greek or mathematical symbols (1:1.4.2).

Genstat's analysis commands produce output in formats appropriate to the current style. You can generate your own output by "printing" the contents of data structures into output files (or into text structures) using the PRINT directive. Titles in Genstat's standard formats can be printed using the CAPTION directive. The PAGE directive starts future output at the top of the next page, the SKIP directive allows blank lines to be inserted in output files (or lines to be skipped in input files), and the PLINK procedure allows you to include graphics in an HTML file. The DECIMALS and MINFIELDWIDTH procedures help you to define appropriate output formats.

PRINT	prints data in tabular form to an output file or a text (1:3.2.1, 1:3.2.2 and 1:3.7)
CAPTION	prints various types of caption and title (1:3.2.3)
PAGE	moves to the top of the next page of an output file (1:3.2.4)
SKIP	skips lines of input or output files (1:3.3.3)
PLINK	prints a link to a graphics file into an HTML file
DECIMALS	sets the number of decimals for a structure, using its round-off (1:3.2.5)
MINFIELDWIDTH	calculates minimum field widths for printing data structures (1:3.2.6)

You can open and close external files from within your Genstat program. Each file is connected to a *channel* (input, output, backing-store, and so on) through which it is accessed by the Genstat commands that read input or generate output. Files can also be copied, deleted and renamed.

OPEN	opens files, connects them to Genstat input or output channels and specifies aspects such as the line width and output style (1:3.3.1)
CLOSE	closes files, freeing the channels to which they were attached (1:3.3.2)
ENQUIRE	provides details about external files attached to Genstat (1:3.3.4)
FCOPY	makes copies of files
FDELETE	deletes files
FRENAME	renames files

The channel from which input statements are taken can be changed, as can the channel to which output is sent. It is also possible to send a transcript (or copy) of input and/or output to output files.

INPUT	specifies the channel from which subsequent statements should be read (1:3.4.1)
RETURN	returns to the previous input channel (1:3.4.2)
OUTPUT	specifies the channel to which future output should be sent, and allows you to switch between plain-text and formatted styles for channels opened as RTF, HTML or LaTeX (1:3.4.4)
COPY	requests a transcript of subsequent input and/or output (1:3.4.4)

The values of a data structure, with all its defining information, can be stored in a sub-file of a "backing-store" file (1:3.5). It can then be retrieved in a later job, without the need to repeat the definitions.

STORE	stores data structures in a backing-store file (1:3.5.3)
RETRIEVE	retrieves data structures from a backing-store file (1:3.5.4)
CATALOGUE	displays the contents of a backing-store file (1:3.5.5)
MERGE	copies sub-files of backing-store files into a single file (1:3.5.6)

The current state of the whole job can also be stored, so that it can be picked up and continued on a later occasion.

RECORD	saves the complete details of a job (1:3.6.1)
RESUME	reads and restarts a recorded job (1:3.6.2)

Genstat *for Windows*, has several additional commands for accessing data from spreadsheets, databases and other systems. However, these may be unavailable in other implementations.

EXPORT	Outputs data structures in foreign file formats, or as plain or comma-delimited text
IMPORT	Reads data in a foreign file format, and loads it into Genstat or into a Genstat spreadsheet file
SPLOAD	loads a Genstat spreadsheet file
SPCOMBINE	combines spreadsheet and data files, without reading them into Genstat
CSPRO	reads a data set from a CSPro survey data file and

	dictionary, loads it into Genstat or puts it into a spreadsheet file
DBCOMMAND	runs an SQL command on an ODBC database
DBEXPORT	Update an ODBC database table using data from Genstat
DBIMPORT	Loads data into Genstat from an ODBC database
DBINFORMATION	loads information on the tables and columns in an ODBC database
DDEEXPORT	Sends data or commands to a Dynamic Data Exchange server
DDEIMPORT	Gets data from a Dynamic Data Exchange (DDE) server
GRIBIMPORT	reads data from a GRIB2 meteorological data file, and loads it or converts it to a spreadsheet file
%CD	Changes the current directory

Details are in the on-line help.

1.4 Calculations and manipulation

Genstat has many directives for doing calculations or for manipulating data, and a full range of mathematical and statistical functions (1:4.2). There is also a directive to link to algorithms in the Numerical Algorithms Group (NAG) Library (1:4.13). Other facilities are provided by procedures, mainly in the Manipulation module of the procedure library.

The CALCULATE directive (1:4.1) can perform straightforward arithmetic operations on any numerical data structure. It also enables you to make logical tests on data: for example, you may want to check whether two variates contain the same values; similar checks can be done with factors, texts and pointers. You can use CALCULATE for matrix operations: for example, matrix multiplication, inversion and Choleski decompositions (1:4.1.3 and 1:4.2.4). CALCULATE can do calculations with tables, and these need not have identical sets of classifying factors (1:4.1.4). When you use CALCULATE, the results are stored in appropriate data structures (which may be defined for you automatically: 1:4.1.5). However, if you want to use the results only once, do not forget that you can use an expression anywhere that Genstat expects a list of identifiers (1:1.5.3).

CALCULATE	performs arithmetic and logical calculations (1:4.1)

In Genstat *for Windows*, the Calculate menu provides a convenient interfact to CALCULATE. The menu allows you to assemble the calculation by selecting data structures from an Available Data window, and clicking appropriate buttons to select the various operators (addition, multiplication and so on).

Other general directives include:

EQUATE	copies values between sets of data structures; they need not have same type, but their values must have the same mode, for example, numbers or text (1:4.3.1)
SETRELATE	compares the sets of values in two data structures; again they need not have same type, but their values must have the same mode (1:4.3.2)
SETCALCULATE	performs Boolean set calculations on the contents of vectors and pointers (1:4.3.3)
SETALLOCATIONS	runs through all ways of allocating a set of objects to subsets with specified sizes (1:4.3.4)
GETLOCATIONS	finds locations of an identifier within a pointer, or a string within a factor or text, or a number within any numerical data structure (1:4.3.5)

There are several commands for manipulating vectors (variates, factors or texts). A "restriction" can be associated with a vector, so that subsequent statements operate on only a subset of its units. Alternatively, you may wish to store the subset, in a data structure on its own. Units of vectors can be sorted into systematic order or into random order, and you can select random samples of a set of units. You can form a vector containing the values of a set of vectors of the same type, appended together, along with a factor which indicates the vector from which each unit came. Similarly, data matrices can be combined by "stacking" (or appending) their corresponding vectors. Another type of combination is to "join" (or merge) new vectors into a data matrix according to the values of one or more "key" vectors. You can also form a set of variates, each of which contains the values from one of the units of every member of a set of structures.

RESTRICT	defines a "restriction" on the units of a vector (1:4.4.1)
SUBSET	forms vectors containing subsets of the values in other vectors (1:4.4.2)
FREGULAR	expands vectors onto a regular two-dimensional grid (procedure)
FRESTRICTEDSET	forms vectors with the restricted subset of a list of vectors (procedure)
SORT	sorts units of vectors into alphabetic or numerical order of an index vector, or forms a factor from a variate or text (1:4.4.3)
RANDOMIZE	puts the units of a set of vectors into random order, or randomizes the units of an experimental design (4.10.1)
SAMPLE	samples from a set of units, possibly stratified by factors (procedure)
SVSAMPLE	constructs stratified random samples (procedure)
APPEND	appends values of a list of vectors of compatible types (1:4.4.4)
STACK	combines several data sets by "stacking" the corresponding vectors (1:4.4.5)
UNSTACK	splits vectors into individual vectors according to levels of a factor (1:4.4.6)
JOIN	joins or merges two sets of vectors together, based on classifying keys (1:4.4.7)
FUNIQUEVALUES	redefines a variate or text so that its values are unique (procedure)
MVFILL	replaces missing values in a vector with the previous non-missing value (procedure)
RESHAPE	reshapes a data set with classifying factors for rows and columns, into a reorganized data set with new identifying factors (procedure)
VEQUATE	equates values across a set of data structures (procedure)

The spreadsheet facilities of Genstat *for Windows* also provide several convenient menus for data manipulation, accessed by clicking Spread on the menu bar and then selecting Manipulate. For example, you can stack and unstack columns, transpose the sheet, append new data onto the ends of the columns, and so on. These facilities will generally be easier to use than the corresponding Genstat commands. Details can be found in the Spreadsheet Help file (click Help on the menu bar, and then select Spreadsheet).

There are several commands for calculations and manipulation that form variates.

INTERPOLATE	calculates variates of interpolated values (1:4.5.1)

MONOTONIC	fits an increasing monotonic regression (1:4.5.2)
TX2VARIATE	converts a text structure into a variate (1:4.5.3)
ORTHPOLYNOMIAL	calculates orthogonal polynomials (procedure)
QUANTILE	calculates quantiles of the values in a variate (procedure)
RANK	produces ranks, from the values in a variate, allowing for ties (procedure)
VINTERPOLATE	performs linear and inverse linear interpolation between variates (procedure)

Other commands are designed specifically for factors.

GROUPS	forms a factor (or grouping variable) from a variate or text, together with the set of distinct values that occur (1:4.6.1)
FACAMEND	permutes the levels and labels of a factor (procedure)
FACDIVIDE	represents a factor by factorial combinations of a set of factors (procedure)
FACEXCLUDEUNUSED	redefines the levels and labels of a factor to exclude those that are unused
FACGETLABELS	obtains the labels for a factor if it has been defined with labels, or constructs labels from its levels otherwise (procedure)
FACLEVSTANDARDIZE	redefines a list of factors so that they have the same levels or labels (procedure)
FACMERGE	merges levels of factors (procedure)
FACPRODUCT	forms a factor with a level for every combination of other factors (procedure)
FACSORT	sorts the levels of a factor according to an index vector (procedure)
FACUNIQUE	redefines a factor so that its levels and labels are unique (procedure)
FDISTINCTFACTORS	checks sets of factors to remove any that define duplicate classifications (procedure)
FREPLICATEFACTOR	forms a factor to indicate observations with identical values of a set of variates, texts or factors (procedure)

Text handling facilities include the ability to omit complete lines, or to append one text onto the end of another, using the non-specialist commands EQUATE and APPEND already mentioned. You can also form a text each of whose lines is made up from sections of lines from several texts concatenated together, form progressions of strings, and perform more general operations using Genstat's text editor.

CONCATENATE	concatenates together lines of text vectors (1:4.7.1)
TXBREAK	breaks a text structure into individual words (1:4.7.6)
TXCONSTRUCT	forms a text structure by appending or concatenating values of from scalars, variates, texts, factors or pointers; allows the case of letters to be changed or values to truncated and reversed (1:4.7.2)
TXFIND	finds a subtext within a text structure (1:4.7.4)
TXPAD	pads strings of a text structure with extra characters so that their lengths are equal
TXPOSITION	locates strings within the lines of a text structure (1:4.7.3)
TXREPLACE	replaces strings within a text structure (1:4.7.5)
TXSPLIT	splits a text into individual texts, at positions on each line marked by separator characters (1:4.7.7)

TXINTEGERCODES	converts textual characters to and from their corresponding integer codes (1:4.7.8)
TXPROGRESSION	forms a text containing a progression of strings (1:4.7.9)
EDIT	line editor for units of text vectors (1:4.7.10)
FVSTRING	forms a string listing the identifiers of a set of data structures

Formulae can be interpreted, modified to operate on different data structures, or constructed automatically from pointers.

FCLASSIFICATION	forms classification sets for the terms in a formula, or breaks a formula up into separate formulae one for each term (1:4.8.1)
REFORMULATE	modifies a formula or an expression to operate on a different set of data structures (1:4.8.4)
SET2FORMULA	forms a model formula with the structures contained in a pointer (1:4.8.3)

You can find out which data structures are used in an expression.

FARGUMENTS	forms lists of data structures used as arguments in an expression (1:4.8.2)

Values can be assigned to dummies and pointers by the ASSIGN directive.

ASSIGN	sets values of dummies and pointers (1:4.9.1)

There are several procedures for calculating or fitting splines, and for manipulating series of observations of a theoretical curve.

SPLINE	calculates a set of basis functions for M-, B- or I-splines
LSPLINE	calculates design matrices to fit a natural polynomial or trignometric L-spline as a linear mixed model
NCSPLINE	calculates natural cubic spline basis functions (for use e.g. in REML)
PENSPLINE	calculates design matrices to fit a penalized spline as a linear mixed model
PSPLINE	calculates design matrices to fit a P-spline as a linear mixed model
RADIALSPLINE	calculates design matrices to fit a radial-spline surface as a linear mixed model
TENSORSPLINE	calculates design matrices to fit a tensor-spline surface as a linear mixed model
ALIGNCURVE	forms an optimal warping to align an observed series of observations with a standard series
BASELINE	estimates a baseline for a series of numbers whose minimum value is drifting
PEAKFINDER	finds the locations of peaks in an observed series

There are several commands for calculations on matrices (either as individual structures, or as elements of a compound structure such as an LRV or an SSPM).

SVD	calculates the singular-value decomposition of a matrix (1:4.10.1)
FLRV	calculates latent roots and vectors – that is, eigenvalues and eigenvectors (1:4.10.2)
FSSPM	calculates values for SSPM structures i.e. sums of squares and products, means, etc. (1:4.10.3)

QRD	calculates the QR decomposition of a matrix (1:4.10.4)
FCORRELATION	forms and tests the correlation matrix for a list of variates (procedure)
FROWCANONICALMATRIX	puts a matrix into row canonical, or reduced row echelon, form (procedure)
FVCOVARIANCE	forms the variance-covariance matrix for a list of variates (procedure)
LINDEPENDENCE	finds the linear relations associated with matrix singularities (procedure)
MPOWER	forms integer powers of a square matrix (procedure)
PARTIALCORRELATIONS	calculates a matrix of partial correlations between a set of variates (procedure)
POSSEMIDEFINITE	calculates a positive semi-definite approximation of a non-positive semi-definite symmetric matrix (procedure)
STANDARDIZE	standardizes columns of a matrix, or a set of variates, to have mean 0 and variance 1 (procedure)
VMATRIX	copies values and row/column labels from a matrix to variates or texts

Tables can be formed containing summaries of values in variates: totals, minimum and maximum values, quantiles, numbers of missing and non-missing values, means and variances. The table manipulation facilities include the ability to add various types of marginal summaries to tables, and to combine "slices" of tables (and also of matrices or variates), calculation of tables of percentages, identification of outliers, and formation of a data matrix (variate and factors) from a table. You can also tabulate results from stratified surveys and surveys involving multiple-response factors.

TABULATE	forms tables of summaries of the values of a variate (1:4.11.1)
MARGIN	calculates or deletes margins of tables (1:4.11.2)
COMBINE	combines or omits "slices" of tables, matrices or variates (1:4.11.4)
MEDIANTETRAD	gives robust identification of multiple outliers in 2-way tables (procedure)
PERCENT	expresses the body of a table as percentages of one of its margins (1:4.11.3)
T%CONTROL	expresses tables as percentages of control cells (1:4.11.3)
TABINSERT	inserts the contents of a sub-table into a table (1:4.11.5)
TABMODE	forms summary tables of modes (procedure)
TABSORT	sorts tables so their margins are in ascending or descending order, as in a Pareto chart (1:4.11.5)
TCOMBINE	combines several tables into a single table (procedure)
DTABLE	plots tables (1:4.11.7)
VTABLE	forms a variate and a set of classifying factors from a table (procedure)
FMFACTORS	forms a pointer of factors representing a multiple-response (1:4.11.8)
FFREERESPONSEFACTOR	forms multiple-response factors from free-response data (1:4.11.9)
MTABULATE	forms tables classified by multiple-response factors (1:4.11.10)
SVBOOT	bootstraps data from random surveys (procedure)
SVCALIBRATE	performs generalized calibration of survey data

(procedure)

SVGLM	fits generalized linear models to survey data (procedure)
SVHOTDECK	performs hot-deck and model-based imputation for survey data (procedure)
SVMERGE	merges strata prior to survey analysis (procedure)
SVREWEIGHT	modifies survey weights adjusting to ensure that their overall sum weights remains unchanged (procedure)
SVSAMPLE	constructs stratified random samples (procedure)
SVSTRATIFIED	analyses stratified random surveys by expansion or ratio raising (procedure)
SVTABULATE	tabulates data from random surveys, including multistage surveys and surveys with unequal probabilities of selection (procedure)
SVWEIGHT	forms survey weights (procedure)

Directives are available for adding and removing branches of trees. There are also procedures for displaying and pruning trees, which provide basic utilities for Genstat's tree-based analysis including classification trees, identification keys and regression trees (6.20, 6.21, 3.9).

BCUT	cuts a tree at a defined node, discarding nodes and information below it (1:4.12.4)
BJOIN	extends a tree by joining another tree to a terminal node (1:4.12.5)
BGROW	adds new branches to a node of a tree (1:4.12.3)
BCONSTRUCT	constructs a tree (1:4.12.6)
BASSESS	assesses potential splits for regression and classification trees (1:4.12.7)
BGRAPH	plots a tree (1:4.12.2)
BPRINT	displays a tree (1:4.12.1)
BPRUNE	prunes a tree using minimal cost complexity (1:4.12.8)
BIDENTIFY	identifies specimens using a tree (1:4.12.9)

There are also various specialist mathematical facilities

NAG	calls an algorithm from the NAG Library (1:4.13)
FHADAMARDMATRIX	forms Hadamard matrices (procedure)
FPARETOSET	forms the Pareto optimal set of non-dominated groups
FPROJECTIONMATRIX	forms a projection matrix for a set of model terms (procedure)
FRTPRODUCTDESIGNMATRIX	forms summation, or relationship, matrices for model terms (procedure)
GALOIS	forms addition and multiplication tables for a Galois finite field (procedure)
BPCONVERT	converts bit patterns between integers, pointers of set bits and textual descriptions (procedure)
NCONVERT	converts integers between base 10 and other bases (procedure)
PERMUTE	forms all possible permutations of the integers 1...*n* (procedure)
PRIMEPOWER	decomposes a positive integer into its constituent prime powers (procedure)

1.5 Programming in Genstat

A Genstat program consists of a sequence of one or more *jobs*. The first job starts automatically at the start of the program. Later, if you want, you can begin a subsequent job using the JOB and ENDJOB directives. The effect is equivalent to restarting Genstat (data structures are deleted, the graphics environment is reset, and so on) except that any files that have been attached to Genstat retain their current status. So, for example, Genstat will continue to add output to the end of an output file, and will continue reading from the current point of an input file.

JOB	starts a Genstat job, ending the previous one if necessary (1:5.1.1)
ENDJOB	ends a job (1:5.1.2)

The whole program is terminated by a STOP directive:

STOP	ends a Genstat program (1:5.1.3)

Statements within a program can be repeated using a FOR loop. The loop is introduced by a FOR statement. This is followed by the series of statements that is to repeated (that is, the contents of the loop), and the end of the loop is marked by an ENDFOR statement. Parameters of the FOR directive allow lists of data structures to be specified so that the statements in the loop operate on different structures each time that it is executed.

FOR	indicates the start of a loop (1:5.2.1)
ENDFOR	marks the end of a loop (1:5.2.1)

Genstat has two ways of choosing between sets of statements. The block-if structure consists of one or more alternative sets of statements. The first set is introduced by an IF statement. There may then be further sets introduced by ELSIF statements. Then there may be a final set introduced by an ELSE statement, and the whole structure is terminated by an ENDIF structure. The IF statement, and each ELSIF statement, contains a single-valued logical expression. Genstat evaluates each one in turn and executes the statements following the first TRUE logical found; if none of them is true, Genstat executes the statements following the ELSE statement (if any).

IF	introduces a block-if structure (1:5.2.2)
ELSIF	introduces an alternative set of statements in a block-if structure (1:5.2.2)
ELSE	introduces a default set of statements for a block-if structure (1:5.2.2)
ENDIF	marks the end of a block-if structure (1:5.2.2)

The multiple-selection structure consists of several sets of statements. The first is introduced by a CASE statement. Subsequent sets are introduced by OR statements. There can then be a final, default, set introduced by an ELSE statement, and the end of the structure is indicated by an ENDCASE statement. The parameter of the CASE statement is an expression which must produce a single number. Genstat rounds this to the nearest integer, *n* say, and then executes the *n*th set of statements. If there is no *n*th set, the statements following the ELSE statement are executed (if any).

CASE	introduces a multiple-selection structure (1:5.2.3)
OR	introduces an alternative set of statements for a multiple-selection structure (1:5.2.3)
ELSE	introduces a default set of statements for a multiple-selection structure (1:5.2.3)
ENDCASE	marks the end of a multiple-selection structure (1:5.2.3)

Any control structure (job, block-if structure, loop, multiple-selection structure or procedure – see below) can be abandoned using an EXIT statement.

EXIT	exits from a control structure (1:5.2.4)

Sequences of statements can be formed into Genstat procedures. This not only makes them simpler for you to use; it also means that you can make them easily available to other users. The use of a procedure looks just like one of the Genstat directives, with its own options and parameters, which transfer information to and from the procedure. Otherwise the procedure is completely self-contained. There is a standard, officially-supported procedure library, which is automatically available whenever you run Genstat. Details are available on-line from the procedures in the help module of the library. You can also write your own procedures (1:5.3.2), and form your own libraries with their own on-line help (1:5.3.4).

LIBHELP	provides help information for Library procedures (1:5.3.1)
LIBEXAMPLE	accesses examples and source code of Library procedures (1:5.3.1)
LIBVERSION	provides the name of the current Genstat Procedure Library (1:5.3.1)

The start of a procedure is indicated by a PROCEDURE statement. Then OPTION and PARAMETER statements can be given to define the arguments of the procedure. These are followed by the statements to be executed when the procedure is called, terminated by an ENDPROCEDURE statement.

PROCEDURE	introduces a procedure, and defines its name (1:5.3.2)
OPTION	defines the options of a procedure (1:5.3.2)
PARAMETER	defines the parameters of a procedure (1:5.3.2)
CALLS	lists the procedures called by a procedure (1:5.3.2)
ENDPROCEDURE	indicates the end of a procedure (1:5.3.2)

Commands are available to enable procedure writers to provide their own error handing, to define and access private data structures, to execute macros, and to increment counters. You can also discover whether and how a particular command has been implemented.

FAULT	checks whether to issue a diagnostic, i.e. a fault, warning or message (1:5.4.1)
DISPLAY	repeats the last Genstat diagnostic (1:5.4.1)
WORKSPACE	accesses "private" data structures for use in procedures (1:5.4.2)
EXECUTE	executes the statements contained within a text (1:5.4.3)
COUNTER	increments a multi-digit counter using non base-10 arithmetic (1:5.4.4)
COMMANDINFORMATION	provides information about whether (and how) a command has been implemented (1:5.4.5)
SPSYNTAX	puts details about the syntax of commands into a spreadsheet
SYNTAX	obtains details about the syntax of a command (1:5.4.6)

Genstat has commands to help you debug your programs. The execution of any control structure (job, block-if structure, loop, multiple-selection structure or procedure) can be interrupted explicitly (so that you can enter other commands such as PRINT) using a BREAK statement, or implicitly by using DEBUG. Once DEBUG has been entered, Genstat will produce breaks automatically at regular intervals, until it meets an ENDDEBUG statement.

BREAK	suspends execution of a control structure (1:5.5.1)
ENDBREAK	continues execution of a control structure, following a break (1:5.5.1)
DEBUG	can cause a break to take place after the current statement (and at specified intervals thereafter), or immediately after

	the next fault (1:5.5.2)
ENDDEBUG	cancels DEBUG (1:5.5.2)

You can modify aspects of the "environment" of the current Genstat job, such as whether or not Genstat starts output from a statistical analysis at the top of a new page, or whether it should pause during interactive output. You can also copy details of these environmental settings into Genstat data structures so that, for example, you can react appropriately within a procedure. User-defined defaults can be specified for the options and parameters of any directive or procedure.

SET	sets details of the "environment" of a Genstat job (1:5.6.1)
GET	accesses information about the Genstat environment (1:5.6.2)
SETOPTION	sets or modifies defaults of options of Genstat directives or procedures (1:5.6.3)
SETPARAMETER	sets or modifies defaults of parameters of Genstat directives or procedures (1:5.6.3)

In many implementations of Genstat, you can suspend the execution of Genstat and return to the operating system of the computer to execute commands, for example to list or edit files on the computer. Likewise, it may be possible to halt the execution of Genstat to execute some other computer program. Some implementations also allow you to incorporate your own programs into Genstat. You can also execute code within an external DLL using the EXTERNAL directive and the OWN function. The OWN directive calls a subroutine called OWN, within the Fortran code of Genstat, which may be modified to call the program. The new code must then be recompiled and linked into a new version of Genstat.

SUSPEND	suspends the execution of Genstat to carry out operating-system commands (1:5.7.1)
PASS	runs another computer program, taking data from Genstat and transferring results back (1:5.7.2)
EXTERNAL	declares an external function in a DLL for use by the OWN function (1:5.7.3)
OWN	executes the user's own code linked into Genstat

There are also several specialised directives, with names prefixed by %, that may be useful.

%LOG	adds text into the Input Log window in the Genstat client
%MESSAGEBOX	displays text in a dialog in the Genstat client
%OPEN	open a binary file for use with %WRITE
%FPOSITION	returns the current position in the binary file opened by %OPEN
%WRITE	writes values of data structures to a binary file opened by %OPEN
%CLEAR	clears the client Output window
%CLOSE	closes the binary file opened by %OPEN
%SLEEP	pauses execution of the server for a time specified in seconds
%TEMPFILE	creates a unique temporary file in the Genstat temporary folder

Details are in the *Genstat Reference Manual, Part 2 Directives*.

1.6 Graphics

Genstat can produce graphical output in two distinctively different styles. These are *line-printer* graphics and *high-resolution* graphics. The line-printer style uses the ordinary characters of textual output, and is available in every Genstat implementation. Most implementations also support high-resolution graphics as a more attractive alternative. Lines and points are plotted with far greater precision, and a wider range of plotting symbols can be used to enhance the output. Also most devices allow the use of colour. Plots can be saved in files using standard formats that are suitable for plotters or laser printers or for importing into word-processed documents. Genstat *for Windows* has a Graphics Wizard that allows you to select a high-resolution graph and customize its appearance. You can also modify many aspects of the graph, such as colours, line styles, plotting symbols, fonts and axes, interactively after it has been plotted.

For high-resolution graphics, the directives have two main purposes. There are those that define the "graphics environment" for subsequent plots, and those that do the plotting. Often the default environment, set up at the start of a program, will be satisfactory. However, to change the graphics environment, the following commands can be used:

DEVICE	switches between graphics devices (1:6.9.1)
FRAME	defines the positions of the windows within the frame (1:6.9.3)
FFRAME	forms multiple windows in a plot-matrix for high-resolution graphics
XAXIS	defines the x-axis in a graphical window (1:6.9.4)
YAXIS	defines the y-axis in a graphical window (1:6.9.5)
ZAXIS	defines the z-axis in a graphical window (1:6.9.6)
AXIS	defines an oblique axis for high-resolution graphics (1:6.9.7)
PEN	defines properties of graphics "pens" (1:6.9.8)
GETRGB	provides a standard sequence of colours, defined by the initial defaults of the Genstat pens (1:6.9.9)
DCOLOURS	forms a band of contiguous colours for graphics (1:6.9.9)
DFONT	defines the default graphics font (1:6.9.12)
DHELP	provides information about the graphics environment (1:6.9)
DKEEP	copies details of the graphics environment into Genstat data structures (1:6.9.10)
DLOAD	loads the graphics environment settings from an external file (1:6.9.11)
DSAVE	saves the current graphics environment settings to an external file (1:6.9.11)

The directives for plotting high-resolution graphs are:

DGRAPH	produces scatter plots and line graphs (1:6.2.1)
D3GRAPH	plots a 3-dimensional graph (1:6.2.2)
DHISTOGRAM	plots histograms (1:6.3.1)
BARCHART	plots bar charts (1:6.3.2)
DCONTOUR	plots contour maps (1:6.4.1)
DSHADE	plots a shade diagram of three-dimensional data (1:6.4.2)
DSURFACE	draws a perspective plot of a two-way array of numbers (1:6.4.3)
D3HISTOGRAM	plots three-dimensional histograms (1:6.4.4)

DBITMAP	plots a bit map of RGB colours (1:6.5)
DPIE	plots pie charts (1:6.6.1)
DCLEAR	clears a graphics screen (1:6.8.1)
DSTART	starts a sequence of related plots (1:6.8.2)
DFINISH	ends a sequence of related plots (1:6.8.2)
DDISPLAY	redraws the current graphical display (1:6.9.2)
DCLOSE	closes windows in the Genstat Graphics Viewer (1:6.9.13)
DVIEW	views windows in the Genstat Graphics Viewer (1:6.9.13)

You can add arrows, annotation, error bars, reference lines and customized keys to graphs:

DARROW	adds arrows to an existing plot (1:6.7.3)
DERRORBAR	adds error bars to a graph (1:6.7.4)
DKEY	adds a key to a graph (1:6.7.5)
DTEXT	adds text to a graph (1:6.7.1)
DFRTEXT	adds text to the graphics frame
DREFERENCELINE	adds reference lines to a graph (1:6.7.2)

Some implementations support interactive graphics devices that allow information to be read from the screen:

DREAD	reads locations of points from an interactive graphics device

Other facilities, provided by procedures in the `graphics` module of the Library include:

BANK	calculates the optimum aspect ratio for a graph
BOXPLOT	draws box-and-whisker diagrams (2.2.2)
DCOMPOSITIONAL	plots 3-part compositional data within a barycentric triangle
DELLIPSE	draws a 2-dimensional scatter plot with confidence, prediction and/or equal-frequency ellipses superimposed
DMASS	plots discrete data like mass spectra, discrete probability functions
DMOSAIC	produces a mosaic plot to display a table of counts
DMSCATTER	produces a scatter-plot matrix for one or two sets of variables (1:6.8.4)
DPROBABILITY	plots probability distributions, and estimates their parameters (2.2.7)
DOTPLOT	displays a dot-plot (2.2.6)
DPARALLEL	displays multivariate data using parallel coordinates (2.7.2)
DSPIDERWEB	displays spider-web and star plots
DTIMEPLOT	produces horizontal bars displaying a continuous time record
DXDENSITY	produces one-dimensional density (or violin) plots
DXYDENSITY	produces density plots for large data sets (1:6.4.5)
DXYGRAPH	draws two-dimensional graphs with marginal distribution plots alongside the y- and x-axes
DYPOLAR	produces polar plots
D2GROUPS	displays the distribution of groups in a plane using a trellis of bar or pie charts (1:6.8.5)
RUGPLOT	draws "rugplots" to display the distribution of one or more samples (2.2.3)
STEM	plots a stem-and-leaf chart (2.2.4)

TRELLIS	produces trellis plots for each level of one or more factors (1:6.8.3)

The relevant directives for line-printer graphics are:

LPCONTOUR	produces contour maps of two-way arrays of numbers (1:6.10.1)
LPGRAPH	produces scatter plots and line graphs (1:6.10.2)
LPHISTOGRAM	plots histograms (1:6.10.3)

2 Basic statistics and exploratory analysis

Before embarking on a full statistical analysis, it can be useful to investigate your data, for example by calculating some summary statistics or studying exploratory plots. Genstat provides a wide range of possibilities. Some are available through specially-designed commands (usually procedures in the Genstat Procedure Library). Others simply use basic options of more powerful commands (usually directives). Many of the relevant commands are described in this chapter, and cross references are given to others.

DESCRIBE	forms summary statistics for variates (2.1.1)
CDESCRIBE	calculates summary statistics and tests of circular data (2.1.2)
FCORRELATION	forms correlations between variates, and calculates their probabilities (2.8.1)
TABULATE	forms tables of summaries of the values in a variate (1:4.11.1)
PERCENT	expresses the body of a table as percentages of one of its margins (1:4.11.3)
MTABULATE	forms tables classified by multiple-response factors (1:4.11.10)
SVSTRATIFIED	analyses stratified random surveys by expansion or ratio raising
SVTABULATE	tabulates data from random surveys, including multistage surveys and surveys with unequal probabilities of selection
TALLY	forms a simple tally table of the distinct values in a vector (2.2.5)
VSUMMARY	Summarizes variate, with classifying factors, into a data matrix of variates and factors
DGRAPH	produces (high-resolution) scatter plots and line graphs (2.7.1)
LPGRAPH	produces (character-based) scatter plots and line graphs (1:6.10.2)
DCIRCULAR	plots circular data (2.2.9)
DHISTOGRAM	plots (high-resolution) histograms (2.2.1)
LPHISTOGRAM	plots (character-based) histograms (1:6.10.2)
BARCHART	plots a bar chart (1:6.3.2)
DPIE	produces pie charts (1:6.6.1)
BOXPLOT	draws box-and-whisker diagrams or schematic plots (2.2.2)
DCOMPOSITIONAL	plots 3-part compositional data within a barycentric triangle
DMASS	plots discrete data like mass spectra, discrete probability functions
DPROBABILITY	plots probability distributions, and estimates their parameters (2.2.7)
DOTPLOT	produces a dot-plot (2.2.6)
DPARALLEL	displays multivariate data using parallel coordinates (2.7.2)
DMSCATTER	produces a scatter-plot matrix (1:6.8.4)
DSHADE	produces a pictorial representation of a data matrix (1:6.4.2)

DTIMEPLOT	produces horizontal bars displaying a continuous time record
KERNELDENSITY	uses kernel density estimation to estimate a sample density (2.2.8)
RUGPLOT	draws "rugplots" to display the distribution of one or more samples (2.2.3)
STEM	produces a simple stem-and-leaf chart (2.2.4)
TRELLIS	produces trellis plots for each level of one or more factors (1:6.8.3)
WINDROSE	plots rose diagrams of circular data like wind speeds

This chapter also covers some of the more straightforward statistical analyses, in particular the t-test and a range of nonparametric tests, as well as describing how you can fit probability distributions to random samples of data, and test whether data come from a Normal distribution. (Commands to determine sample sizes for many of these tests are described later, in Section 4.12.)

TTEST	performs a one- or two-sample t-test (2.3.1)
AONEWAY	provides one-way analysis of variance (2.3.2)
A2WAY	performs analysis of variance of a balanced or unbalanced design with up to two treatment factors (2.3.3)
A2DISPLAY	provides further output from an A2WAY analysis (2.3.3)
A2KEEP	saves information from an A2WAY analysis (2.3.3)
CHIPERMTEST	does a random permutation test for a two-dimensional contingency table (2.9.2)
CHISQUARE	calculates chi-square statistics for one- and two-way tables (2.9.1)
CMHTEST	performs the Cochran-Mantel-Haenszel test (2.9.5)
FEXACT2X2	does Fisher's exact test for 2×2 tables (2.9.2)
FRIEDMAN	performs Friedman's nonparametric analysis of variance (2.6.2)
BNTEST	calculates one- and two-sample binomial tests (2.3.4)
PNTEST	calculates one- and two-sample Poisson tests (2.3.5)
GSTATISTIC	calculates the gamma statistic of agreement for ordinal data (2.8.6)
KAPPA	calculates a kappa coefficient of agreement for nominally scaled data (2.8.5)
KCONCORDANCE	calculates Kendall's Coefficient of Concordance, synonym CONCORD (2.8.4)
KOLMOG2	performs a Kolmogorov-Smirnoff two-sample test (2.5.2)
KRUSKAL	carries out a Kruskal-Wallis one-way analysis of variance (2.6.1)
KTAU	calculates Kendall's rank correlation coefficient τ (2.8.3)
LCONCORDANCE	calculates Lin's concordance correlation coefficient (2.8.7)
MANNWHITNEY	performs a Mann-Whitney U test (2.5.1)
MCNEMAR	performs McNemar's test for the significance of changes (2.9.3)
QCOCHRAN	performs Cochran's *Q* test for differences between related samples (2.9.4)
RUNTEST	performs a test of randomness of a sequence of observations (2.4.3)
SIGNTEST	performs a one or two sample sign test (2.4.2)
SPEARMAN	calculates Spearman's rank correlation coefficient (2.8.2)

STEEL	performs Steel's many-one rank test (2.6.3)
WILCOXON	performs a Wilcoxon Matched-Pairs (Signed-Rank) test (2.4.1)
DISTRIBUTION	estimates the parameters of continuous and discrete distributions (2.2.10)
NORMTEST	performs tests of univariate and/or multivariate Normality (2.2.11)
WSTATISTIC	calculates the Shapiro-Wilk test for Normality (2.2.11)

Section 2.10 describes some of the Genstat facitities for supporting the six-sigma approach to quality improvement. These include a wide range of control charts and the calculation of capability statistics.

SPCAPABILITY	calculates capability statistics (2.10.6)
SPCCHART	plots c or u charts representing numbers of defective items (2.10.5)
SPCUSUM	prints CUSUM tables for controlling a process mean (2.10.2)
SPEWMA	plots exponentially weighted moving-average control charts (2.10.3)
SPPCHART	plots p or np charts for binomial testing for defective items (2.10.4)
SPSHEWHART	plots control charts for mean and standard deviation or range (2.10.1)

Finally, Section 2.11 describes some procedures that can be used to study species diversity and abundance.

ECDIVERSITY	calculates measures of diversity with jackknife or bootstrap estimates (2.11.1)
ECABUNDANCEPLOT	produces rank/abundance, *ABC* and *k*-dominance plots (2.11.2)
ECFIT	fits models to species abundance data (2.11.3)
ECNICHE	generates relative abundance of species for niche-based models (2.11.4)
ECRAREFACTION	calculates individual or sample-based rarefaction (2.11.5)
ECACCUMULATION	plots species accumulation curves for samples or individuals (2.11.6)
ECNPESTIMATE	calculates nonparametric estimates of species richness (2.11.7)
ECANOSIM	compares communities between sites by a nonparametric analysis of similarities known as ANOSIM (6.1.6)
LORENZ	plots the Lorenz curve and calculates the Gini and asymmetry coefficients (2.11.8)

The analyses in this chapter can all be obtained through menus in Genstat *for Windows*, mainly in the Summary Statistics, Statistical Tests, Six sigma and Distributions categories.

2.1 Summary statistics

2.1.1 The DESCRIBE procedure

DESCRIBE procedure

Saves and/or prints summary statistics for variates (R.C. Butler & D.A. Murray).

Options

PRINT = *string token*	Controls whether or not the summaries are printed (summaries); default summ
SELECTION = *string tokens*	Selects the statistics to be produced (nval, nobs, nmv, mean, median, min, max, range, q1, q3, sd, sem, var, sevar, %cv, sum, ss, uss, skew, seskew, kurtosis, sekurtosis, all); default mean, min, max, nobs, nmv, medi, q1, q3
GROUPS = *factor*	Allows groups to be defined, so that summaries are produced for each group in turn

Parameters

DATA = *variates*	Data to summarize
SUMMARIES = *variates* or *pointers*	To save summaries for each DATA variate, in a variate if GROUPS is unset, or in a pointer to a set of variates (one for each group) if groups have been specified; will be redefined if necessary

The DESCRIBE procedure (used by the Summary of Variates menu of Genstat *for Windows*) provides a wide range of summary statistics for grouped or ungrouped data.

Example 2.1.1a produces summary statistics from a set of data specifying the heights of active volcanos around the world. As well as the heights and names of the volcanos, the data set also contains their latest eruption dates and geographical regions. The DATA parameter of DESCRIBE specifies the data variate for which the statistics are to be calculated. The PRINT option controls whether or not they are printed. By default they will be printed (so PRINT is not set in the example); to suppress printing you need to put PRINT=*. The statistics to be calculated are indicated by the SELECTION option. Here we keep the default selection.

Example 2.1.1a

```
  2  " Heights of active volcanoes. Data from The World Almanac (1992);
 -3    previous version of data also displayed in Tukey (1977) p.40."
  4  OPEN '%GENDIR%/Examples/GuidePart2/Volcano.dat'; CHANNEL=2
  5  TEXT Volcano
  6  TEXT [VALUES=America,'Asia/Oceania',Elsewhere] Regname
  7  FACTOR [LABELS=Regname] Region
  8  READ [CHANNEL=2] Volcano,Year,Height,Region

  Identifier   Minimum      Mean   Maximum    Values   Missing
     Volcano                                   126         0
        Year      1960      1983      1991       126         0
      Height     10.00     77.19     199.0       126         0

  Identifier    Values   Missing    Levels
      Region       126         0         3

  9  CLOSE 2
 10  DESCRIBE Height
```

```
Summary statistics for Height
=============================

      Number of observations = 126
    Number of missing values = 0
                        Mean = 77.19
                      Median = 67
                     Minimum = 10
                     Maximum = 199
              Lower quartile = 49
              Upper quartile = 100
```

The available settings of `SELECTION` are:

`nval` number of values	`sem` standard error of mean
`nobs` number of non-missing values	`var` variance
`nmv` number of missing values	`sevar` standard error of variance
`mean` arithmetice mean	`%cv` coefficient of variation
`median` median	`sum` total of values
`min` minimum	`ss` corrected sum of squares
`max` maximum	`uss` uncorrected sum of squares
`range` range (max-min)	`skew` skewness
`q1` lower quartile	`seskew` standard error of skewness
`q3` upper quartile	`kurtosis` kurtosis (centred around zero i.e. subtracting the expected value of 3 for a Normal distribution; see 2.2.10)
`sd` standard deviation	`sekurtosis` s.e. of kurtosis
`all` all 22 summaries	

by default the mean, min, max, nobs, nmv, median and both quartiles are calculated.

The `GROUPS` option allows groups of observations to be defined, so that the summaries are calculated separately for each group. This is illustrated in Example 2.1.1b, which continues Example 2.1.1a: in the `DESCRIBE` statement in line 10, `GROUPS` is set to `Region` to produce summaries for each geographical region.

Example 2.1.1b

```
  11  DESCRIBE [GROUPS=Region] Height

Summary statistics for Height: Region America
=============================================

      Number of observations = 50
    Number of missing values = 0
                        Mean = 91.92
                      Median = 82.5
                     Minimum = 34
                     Maximum = 199
              Lower quartile = 53
              Upper quartile = 124


Summary statistics for Height: Region Asia/Oceania
==================================================

      Number of observations = 61
    Number of missing values = 0
                        Mean = 66.95
                      Median = 60
                     Minimum = 10
                     Maximum = 156
              Lower quartile = 49
              Upper quartile = 81.5
```

```
Summary statistics for Height: Region Elsewhere
===============================================

      Number of observations = 15
    Number of missing values = 0
                        Mean = 69.73
                      Median = 75
                     Minimum = 17
                     Maximum = 134
              Lower quartile = 23.25
              Upper quartile = 108.2
```

DESCRIBE allows for only one grouping classification (i.e. a single factor). If you have several factors, you could use the FACPRODUCT procedure to generate a new factor with a level for every combination of the original factors, and specify that as the GROUPS factor. Alternatively, the TABULATE directive (1:4.11.1) allows you to produce multi-way tables of summary statistics such as means, medians, totals, minima, maxima, replications, variances, standard deviations, skewness and kurtosis.

The SUMMARIES parameter of DESCRIBE allows the statistics to be saved in a variate, or in a pointer to a set of variates if there are groups. These need not be declared in advance. The units of the variate(s) are labelled by the corresponding strings from the settings (in capital letters) of the SELECTION option, to simplify the subsequent access of any individual statistic. For example, the minimum value can be copied from a SUMMARIES variate v into a scalar m by

```
CALCULATE m = v$['MIN']
```

2.1.2 Circular data: the CDESCRIBE procedure

CDESCRIBE procedure

Calculates summary statistics and tests of circular data (P.W. Goedhart & R.W. Payne).

Options

PRINT = *string tokens*	What to print (summary, fittedvalues); default summ
SEGMENT = *scalar*	Width of sectors (in degrees) into which to group an ANGLES variate for calculation of the test of randomness and the chi-square goodness of fit statistic for the von Mises distribution; default 20
MSEGMENT = *scalar*	Defines the centre (in degrees) of the sectors; default 0
DIRECTION = *scalar*	Direction (in degrees) of the unimodal alternative distribution for the Rayleigh test; default * i.e. not known

Parameters

ANGLES = *factors* or *variates*	Directional observations (in degrees)
RESULTS = *variates*	Saves the summary statistics
VONMISESCOUNTS = *pointers*	Saves structures relevant for calculation of the chi-square goodness of fit statistic for the von Mises distribution

CDESCRIBE summarizes data values that consist of directional observations recorded as angles between 0 and 360 degrees. These are supplied using the ANGLES parameter, in either a variate or a factor. If ANGLES is restricted, only the unrestricted units are analysed. The procedure mainly uses the methods presented in the book by Fisher (1993). The various statistics are cross-referenced below with the relevant page numbers.

CDESCRIBE prints the following summary statistics: number of observations, mean direction (page 31), circular standard deviation (page 32), mean resultant length (page 32), skewness (page 34) and estimate of the parameter Kappa (which provides the concentration parameter of the von Mises distribution for circular data; pages 39 and 88). If the angles are supplied in a factor, a grouping correction is applied to the mean resultant length and to the skewness (page 35).

Two tests of uniformity are presented. The null hypothesis for both of these is that the observations come from a uniform distribution around the circle. The first is a test of randomness against any alternative model. The test is based on counts of the number of observations in a set of angular sectors of equal size (page 67). If ANGLES is set to a variate, the width of the sectors is defined by the SEGMENT option (in degrees), with centres defined by the MSEGMENT option. The sectors are centred at MSEGMENT, MSEGMENT+SEGMENT, MSEGMENT+2*SEGMENT, and so on. The default values for SEGMENT and MSEGMENT are 20 and 0 respectively. If ANGLES is set to a factor with equidistant levels, it is assumed that the levels define the centres of the segments and that the limits of the sectors are at the midpoints between each pair of factor levels. If ANGLES is set to factor with non-equidistant levels, the SEGMENT and MSEGMENT options are used to define the angular sectors.

The second is Rayleigh's test of uniformity against a unimodel alternative. The test is based on the mean resultant length and has two forms which differ according to whether or not the mean direction of the alternative distribution is known (pages 69 and 70). The direction, if known, is specified using the DIRECTION option.

Finally a goodness of fit test is calculated to assess whether the observations follow a von Mises distribution. This is a chi-square test, which compares the observed distribution with the expected distribution from a von Mises distribution with mean direction and concentration parameter (kappa) taking the values estimated from the observations. The observed and expected values are calculated for grouped directional data defined by the (M)SEGMENT options for a variate or by the factor levels if ANGLES is set to a factor.

The PRINT options controls whether the summary statistics are printed and whether a table of observed and expected counts for the fit of the von Mises distribution is printed. The summary statistics can be saved by means of the RESULTS parameter. The VONMISESCOUNTS parameter saves the grouped directional data used for calculation of the chi-square goodness of fit test and tables of observed and expected counts. Note that when ANGLES is set to factor, the saved grouped directional data set is identical to ANGLES.

Example 2.1.2 calculates summary statistics for data concerning the directions chosen by 100 ants in response to an evenly illuminated black target placed at 180 degrees (see Fisher 1993, pages 60, 61, 83, 85 and 243). The results show that the data are not uniform, nor can they be modelled by a von Mises distribution. In Section 2.2.9, procedure DCIRCULAR is used to plot the data (Figure 2.2.9).

Example 2.1.2

```
  2  VARIATE [NVALUES=100] Direction
  3  READ    Direction

   Identifier   Minimum      Mean   Maximum    Values   Missing
    Direction     10.00     181.1     360.0       100         0

 14  CDESCRIBE Direction

Summary statistics and tests for circular data
==============================================

Variate: Direction
Number of equidistant sectors: 18
Number of observations: 100
Mean direction: 183.14
Circular standard deviation: 56.96
```

```
Mean resultant length: 0.6101
Skewness: 0.2304
Kappa estimate: 1.5576
Prob. test of randomness: 0.000
Prob. Rayleigh test of uniformity: 0.000
Chi-square von Mises: 31.20 with 15 df
Prob. Chi-square von Mises: 0.008


Goodness of fit for von Mises distribution
------------------------------------------
```

Midpoint	Observed	Expected	ChiSquare
0	2	0.69	2.46
20	1	0.74	0.09
40	3	0.95	4.45
60	3	1.41	1.80
80	2	2.32	0.04
100	1	3.97	2.22
120	4	6.63	1.04
140	4	10.15	3.73
160	12	13.57	0.18
180	23	15.31	3.86
200	21	14.37	3.05
220	13	11.31	0.25
240	2	7.67	4.19
260	2	4.69	1.54
280	3	2.74	0.02
300	3	1.63	1.14
320	0	1.06	1.06
340	1	0.79	0.06

Part 3 of the *Genstat Reference Manual* describes three other procedures for circular data. Measures of association for circular data can be calculated by the CASSOCIATION procedure, and you can test whether samples from circular distributions have a common mean direction or have identical distributions using the CCOMPARE procedure. Circular regressions can be fitted using the RCIRCULAR procedure.

2.2 Exploring the distribution of the data

Many plots are concerned with studying the empirical distribution, that is the observed distribution, of the data values. You might want to do this in order to decide on a suitable analysis, or as an initial check of the assumptions prior to an analysis (although, you will find later that most Genstat analyses also have their own diagnostic plots). The values for plotting may be either continuous measurements (specified as variates) or categorical observations (specified as factors). If you have a single random sample of data, you might want to see whether it could have been generated by a specific probability distribution. Probability plots for a wide range of distributions can be plotted by the DPROBABILITY procedure (2.2.7). Kernel density plots can also be useful (procedure KERNELDENSITY 2.2.8). Once you have identified a plausible distribution, you can estimate its parameters using the DISTRIBUTION directive (2.2.10).

Most of these plots can be constructed using the Graphics Wizard of Genstat *for Windows* (click Graphics on the menu bar, and then select Create Graph), but the probability plots are accessed from the summary statistics section (click Stats on the menu bar, then Summary Statistics and then Probability Plots).

2.2.1 Histograms

A histogram provides a simple and effective way of studying the distribution of a set of data. It is formed by splitting the range of the data into contiguous categories. It displays the number of observations falling into successive categories, thus showing whether they are tightly-packed or

spread-out, symmetrically distributed or skew, and whether there are observations separated, or outlying, from the mass of the data.

High-resolution plots are produced by the DHISTOGRAM directive.

DHISTOGRAM directive

Draws histograms or bar charts on a plotter or graphics monitor.

Options

TITLE = *text*	General title; default *
WINDOW = *scalar*	Window number for the histograms; default 1
KEYWINDOW = *scalar*	Window number for the key (zero for no key); default 2
LIMITS = *variate*	Variate of group limits for classifying DATA variates into groups; default *
LOWER = *scalar*	For a DATA variate, this specifies the lower limit of the first bar; default * takes the minimum value of the variate
UPPER = *scalar*	For a DATA variate, this specifies the upper limit of the last bar; default * takes the maximum value of the variate
NGROUPS = *scalar*	When LIMITS and BINWIDTH are not specified, this defines the number of groups into which a DATA variate is to be classified; default is then 10, or the integer value nearest to the square root of the number of values in the variate if that is smaller
BINWIDTH = *scalar*	When LIMITS is unset the range of a DATA variate is split into equal intervals known as "bins" to form the groups, this option can set the bin widths (alternative is to set the number of groups using NGROUPS)
FIXEDBARWIDTH = *string token*	Whether to plot the histogram with bars of equal width (yes, no); default no
BARCOVERING = *scalar*	What proportion of the space allocated along the x-axis each bar should occupy; default * gives proportion 1 for a DATA variate, and 0.8 for a factor or table (thus giving a gap between each bar)
BARSCALE = *scalar*	Width of bar for which one unit of bar length represents one unit of data; default * uses the width of the narrowest bar
LABELS = *text*	Group labels; default *
APPEND = *string token*	Whether or not the bars of the histograms are appended together (yes, no); default no
ORIENTATION = *string token*	Direction of the plot (horizontal, vertical); default vert
OUTLINE = *string token*	Where to draw outlines (bars, perimeter); default bars
PENOUTLINE = *scalar*	Pen to use for the outlines; default -8
SCREEN = *string token*	Whether to clear the screen before plotting or to continue plotting on the old screen (clear, keep); default clea
KEYDESCRIPTION = *text*	Overall description for the key; default *
ENDACTION = *string token*	Action to be taken after completing the plot (continue, pause); default * uses the setting from the last DEVICE

statement

Parameters

DATA = *identifiers*	Data for the histograms; these can be either a factor indicating the group to which each unit belongs, a variate whose values are to be grouped, or a one-way table giving the height of each bar
NOBSERVATIONS = *tables*	One-way table to save numbers in the groups
GROUPS = *factors*	Factor to save groups defined from a variate
PEN = *scalars* or *variates*	Pen number(s) for each histogram; default * uses pens 2, 3, and so on for the successive structures specified by DATA
DESCRIPTION = *texts*	Annotation for key

Here we illustrate only the simple use of DHISTOGRAM (a full description is given in 1:6.3.1). Example 2.2.1 plots a histogram of the heights of the volcanoes, discussed earlier in Example 2.1.1.

Example 2.2.1

```
12  TEXT [VALUES='Height distribution of active volcanoes'] Head
13  & [VALUES='Height in 100s of feet'] Scale
14  DHISTOGRAM [TITLE=Head] Height; DESCRIPTION=Scale
```

The DHISTOGRAM statement in this example draws the picture in Figure 2.2.1a, showing that the distribution of heights is positively skewed. The statement automatically chooses the number of classes into which to divide the observations, and uses the default colours, brush-types, and so on. These details can be changed by setting options in the DHISTOGRAM statement, or by explicitly setting the graphical environment. For example, to specify a more spread-out picture, the NGROUPS option of DHISTOGRAM could have been set to get 20 groups instead of the 10 produced by default:

```
DHISTOGRAM [NGROUPS=20] Height
```

DHISTOGRAM can also be used to plot a bar chart showing the distribution of categorical data supplied in factors. The histogram then has a bar for each level of the factor, with height equal to the number of observations with that level. For example, the following statement draws such a histogram displaying the number of active volcanos in each region; see Figure 2.2.1b.

```
DHISTOGRAM [TITLE=\
  'Active volcanoes in three regions of the world'] Region
```

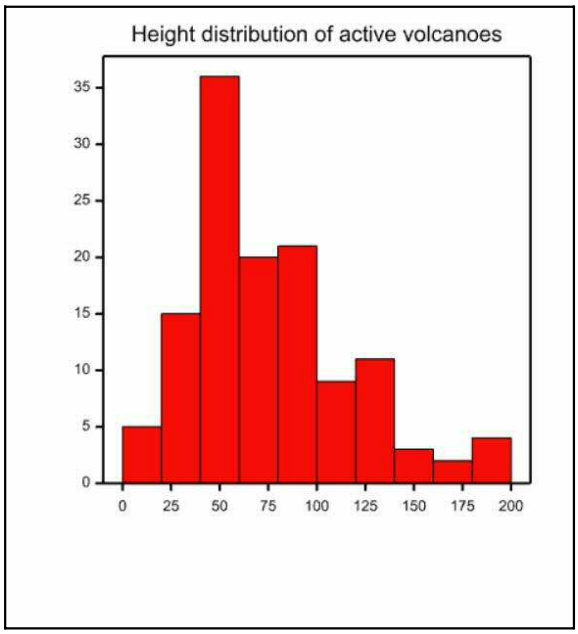


Figure 2.2.1a

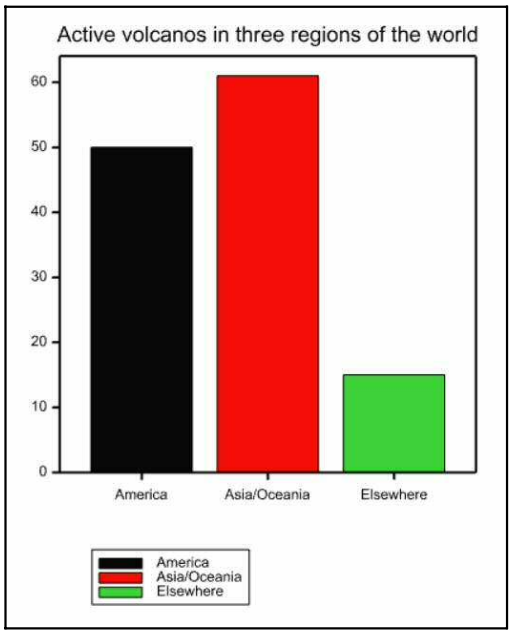


Figure 2.2.1b

Character-based (line-printer style) histograms can be drawn using the `LPHISTOGRAM` directive (see 1:6.10.2).

2.2.2 Boxplots

An alternative diagram for studying the distribution of observations is the boxplot, or box-and-whisker plot, which can be drawn using the `BOXPLOT` procedure.

`BOXPLOT` procedure

Draws box-and-whisker diagrams or schematic plots (P.W. Lane & S.D. Langton).

Options

`GRAPHICS` = *string token*	What type of graphics to use (`highresolution`, `lineprinter`); default `high`
`TITLE` = *text*	Title for diagram; default `*`
`AXISTITLE` = *text*	Title for axis representing data values; default `*`
`WINDOW` = *scalar*	Window in which to draw a high-resolution plot; default 4
`ORIENTATION` = *string token*	Orientation of plots (`horizontal`, `vertical`, `across`, `down`); default `vert`
`YORIENTATION` = *string token*	Direction of the y-axis for horizontal plots (`reverse`, `normal`); default `reve`
`METHOD` = *string token*	Type of representation of data in a high-resolution plot (`boxandwhisker`, `schematic`); default `boxa`
`SCREEN` = *string token*	Whether to clear screen before a high-resolution plot (`clear`, `keep`); default `clea`
`BOXTITLE` = *text*	Title for axis representing different variates or groups; default `*`
`BOXWIDTH` = *string token*	Whether to relate box width to size of sample in high-resolution plot (`fixed`, `variable`); default `fixe`

WHISKER = *number*	Linestyle for whiskers (0...10); default 1
BAR% = *scalar*	Size of bar at the end of the whiskers, as a percentage of the box-width; default 0 (i.e. no bar)
WIDTH% = *scalar*	Width of the boxes, expressed as a percentage of the default width; default 100

Parameters

DATA = *variates*	Data to be summarized; no default
GROUPS = *factor*	Factor to divide values of a single variate into groups; default *
BOXLABELS = *texts*	Labels for individual boxes; default *, i.e. identifiers of variates or labels or levels of factor
UNITLABELS = *texts*	Labels for extreme points in schematic plot; default is to use unit labels
BOXPOSITIONS = *variates*	Positions of the boxes on the appropriate axis; default defines positions in an equal spacing

BOXPLOT draws pictures to display the distribution of one or more sets of data. In the simplest case, with the DATA parameter set to a single variate, BOXPLOT will draw a box-and-whisker diagram, as defined by Tukey (1977). The box spans the inter-quartile range of the values in the variate, so that the middle 50% of the data lie within the box, with a line indicating the median. Whiskers extend beyond the ends of the box as far as the minimum and maximum values. If several variates are supplied, a box is drawn for each of them using the same scale. Alternatively, if a single variate is supplied by the DATA parameter, a factor with the same number of values as the variate may be provided by the GROUPS parameter, and a box will be drawn for each level of the factor.

The GRAPHICS option allows you to request a line-printer style plot, instead of a high-resolution plot. The TITLE, AXISTITLE and BOXTITLE options can be set to specify the titles displayed at the top of the plot, along the axis representing the data values, and along the axis representing separate boxes when there are several variates or groups, for either graphics mode. For high-resolution plots, the WINDOW and SCREEN options control the placement of the picture in the graphical frame.

It is not possible to produce line-printer plots with more than 14 boxes. If the page size is small, as in interactive mode, vertical line-printer plots may be very cramped: the PAGE option of the OUTPUT directive can be used to increase the depth of the graphs.

The ORIENTATION option controls the orientation of the boxes, with the following settings:

vertical	plots the boxes vertically i.e. down the screen (default),
horizontal	plots the boxes horizontally i.e. across the screen,
down	synonym of vertical, and
across	synonym of horizontal.

When ORIENTATION=horizontal, the horizontal axis is taken to be the y-axis, so the same XAXIS and YAXIS settings can be used however the boxes are oriented.

The YORIENTATION option controls the orientation of the y-axis when the boxes are plotted horizontally. By default this is reversed, so that the first box is at the top of the screen.

For example, the following statement draws a boxplot of the volcano heights:

```
BOXPLOT [TITLE=Head; AXISTITLE=Scale] Height
```

The resulting plot is shown in Figure 2.2.2a.

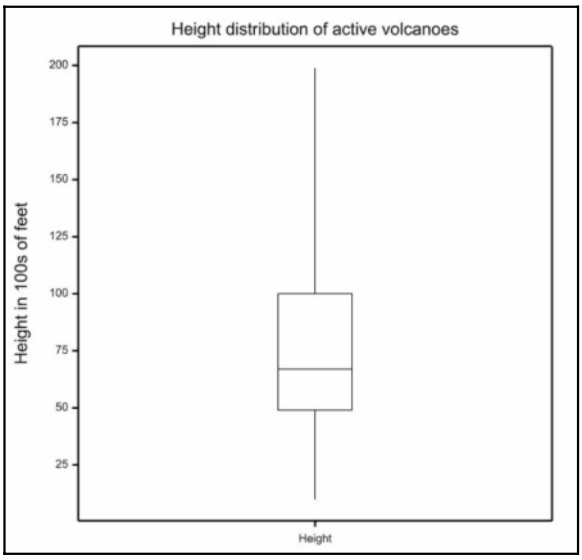


Figure 2.2.2a

Schematic plots can be drawn (high-resolution only) by setting option `METHOD=schematic`. These diagrams (also defined by Tukey 1977) are modifications of box-and-whisker diagrams which display individual outlying points as well as the box. The whiskers extend only to the most extreme data values within the inner "fences", which are at a distance of 1.5 times the interquartile range beyond the quartiles, or the maximum value if that is smaller. Individual outliers are plotted with a cross by default, and labelled under control of the `UNITLABELS` parameter. "Far" outliers, beyond the outer "fences" which are at a distance of three times the interquartile range beyond the quartiles, are plotted with a different pen. By default, all boxes have equal width. High-resolution diagrams can be modified to indicate the number of values being represented by each box. The option `BOXWIDTH=variable` will scale the box widths by the square root of the number of values represented.

Figure 2.2.2b shows an example of a schematic boxplot of the volcano heights within each region. This was generated by the statement

```
BOXPLOT [TITLE=Head; AXISTITLE=Scale; METHOD=schematic;\
  BOXWIDTH=variable] Height; GROUPS=Region;\
  UNITLABELS=Volcano
```

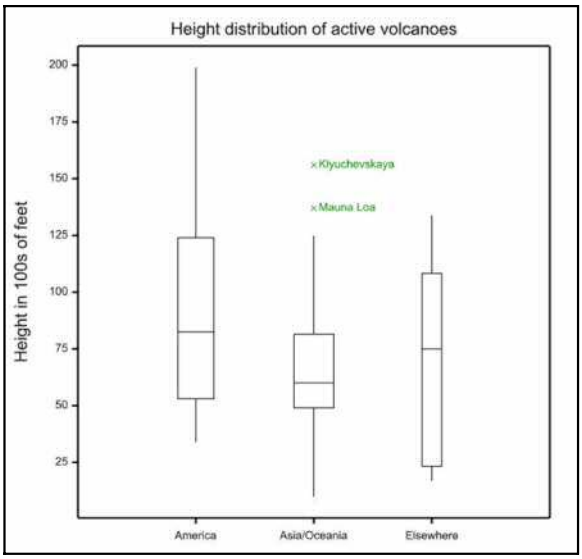


Figure 2.2.2b

The style of the whiskers can be controlled by setting the `WHISKER` option to a graphical linestyle in the range 0 to 10. These styles are device dependent, but 0 and 1 always give a solid line (the default) and 2 usually gives a dashed line. The `BAR%` option allows you to add bars at the end of the whiskers. For example, the setting 100 gives a bar as wide as the box, and 25 would give one a quarter the width. The default is 0, giving no bars. The `WIDTH%` option specifies the width of the boxes, as a percentage of the default width (default 100).

Four pens are used to draw the high-resolution displays, apart from the axes: Pen 1 for the boxes and median line (default colour black), Pen 2 for far outliers (red crosses), Pen 3 for outliers (green crosses) and Pen 4 for the whiskers (set to match the colour of Pen 1). You can customize the pictures by setting some aspects of these pens with the `PEN` directive before calling the procedure: in particular, the colours, symbols and line-thicknesses.

The `BOXLABELS` parameter allows you to specify labels that will identify each box.

The `UNITLABELS` parameter allows you to specify labels that will be used to identify outlying

observations in schematic plots (but this is not available if you gave a list of variates in the DATA parameter).

The BOXPOSITIONS parameter defines the positions of the boxes on the appropriate axis. If this is unset, the positions are defined with an equal spacing.

2.2.3 Rugplots

Procedure RUGPLOT can display the distribution of a set of data, either on the axis of an existing graph (looking like a "rug" of vertical lines on the x-axis), or as a picture by itself.

RUGPLOT procedure

Draws "rugplots" to display the distribution of one or more samples (P.W. Lane).

Options

GRAPHICS = *string token*	What type of graphics to use (highresolution, lineprinter); default high
TITLE = *text*	Title for diagram; default *
AXISTITLE = *text*	Title for axis; default *
WINDOW = *scalar*	Window in which to draw high-resolution plot; default *, taken as 11 if SCREEN=clear, or 1 if SCREEN=keep
SCREEN = *string token*	Whether to clear screen before high-resolution plot (clear, keep); default clea
ORIENTATION = *string token*	Orientation of plots (down, across); default down
JITTER = *number*	Ratio of jitter width to range of data in high-resolution plot; default 0.01
SEED = *number*	Seed for generating random numbers used in jittering; default 0, i.e. continue from last generation, or initialize from system clock

Parameters

DATA = *variates*	Data to be summarized; no default
GROUPS = *factor*	Factor to divide values of a single variate into groups; default *
RUGLABELS = *texts*	Labels for individual rugs; default *, i.e. identifiers of variates or labels or levels of factor
POSITION = *scalar* or *variate*	Position on *x*-axis (or on *y*-axis if ORIENTATION=across) at which to plot each rug; if GROUPS is set, positions for each level of the factor are taken from a variate; default is to draw a single rug on the axis, and to spread multiple rugs across the window

---

In the simplest case, with the DATA parameter set to a single variate, RUGPLOT draws a single vertical "rug": that is, a series of short horizontal lines on the vertical axis, positioned at each value of the variate. Setting option ORIENTATION=across produces a horizontal rug. A rug can be added to an existing plot by specifying SCREEN=keep, and setting the WINDOW option to specify the window where the rug is to be drawn. With SCREEN=keep, the default window is 1; with SCREEN=clear, window 11 is used after defining it to fill the whole graphical frame.

If several variates are supplied, a rug is drawn for each of them using the same scale. Alternatively, if a single variate is specified by the DATA parameter, a factor with the same number of values as the variate may be defined by the GROUPS parameter, and a box will be drawn for each level of the factor. The rug plots are spread out across the window by default. The POSITION parameter can be set to specify where each rug is to be positioned on the *x*-axis

(or *y*-axis if ORIENTATION=across). The setting should be in the range (0, *n*) for a plot with SCREEN=clear, where n is the number of rugs to be drawn; with SCREEN=keep, the position should be specified in the units of the axis last drawn in the window.

Line-printer rugplots can be drawn by setting option GRAPHICS=lineprinter. The plot is drawn with asterisks, or digits to represent points that are effectively coincident. If the page size is small, as in interactive mode, line-printer plots with ORIENTATION=down are very cramped: the PAGE option of the OUTPUT directive (1:3.4.3) can be used to increase the depth of the graphs. The option ORIENTATION=down cannot be selected for line-printer plots with more than 14 rugs. The TITLE and AXISTITLE options can be set to specify the titles displayed at the top of the plot and along the axis, for either graphics mode. The RUGLABELS parameter allows you to specify labels that will identify each rug, in place of the default labels taken from the variate identifiers, or factor labels or levels if the GROUPS parameter is set. Long identifiers or labels may overlap each other if ORIENTATION=down, or they may overlap the rug-plots if ORIENTATION=across; a maximum of eight characters is recommended.

In high-resolution plots, all data values are "jittered" to try to remove ties. This involves adding a small random value: by default the ratio of the maximum adjustment to the range of all the data is 1:100. This can be modified by setting the JITTER option to 0 to suppress jittering, or to some other ratio than the default of 0.01. The SEED option can be set to specify the seed of the random-number generation, if a reproducible plot is required.

For example, this statement draws a boxplot of the volcano heights from Example 2.1.1:

```
RUGPLOT  [TITLE='Volcano heights'] Height; GROUPS=Region
```

The plot is shown in Figure 2.2.3a.

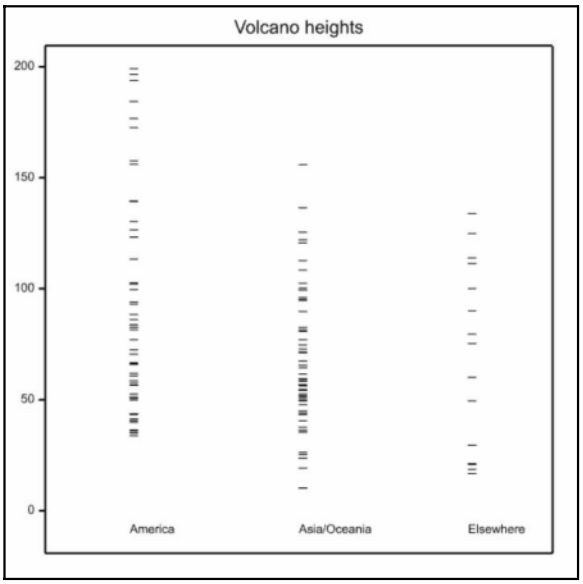


Figure 2.2.3a

Example 2.2.3 and Figure 2.2.3b show how you can plot rugplots alongside the axes to illustrate the distributions of the x- and y-variates.

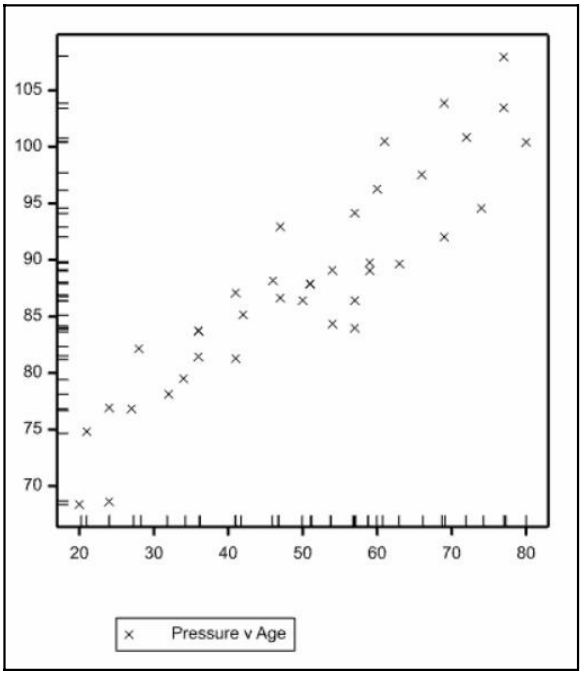


Figure 2.2.3b

Example 2.2.3

```
" A scatter plot is drawn of blood-pressure against age for 38 women.
  Then two rugplots are added to show the distribution of ages and
  pressures along the axes."
VARIATE  [VALUES=82.17,88.19,89.66,81.45,85.16,89.77,89.11,107.96,\
  74.82,83.98,92.95,79.51,87.86,76.85,76.93,87.09,97.55,92.04,100.85,\
  96.30,86.42,94.16,78.12,89.06,94.58,103.48,81.30,83.71,68.38,86.64,\
  87.91,86.42,103.87,83.76,84.35,68.64,100.50,100.42] Pressure
VARIATE  [VALUES=28,46,63,36,42,59,54,77,21,57,47,34,51,27,24,41,66,\
  69,72,60,50,57,32,59,74,77,41,36,20,47,51,57,69,36,54,24,61,80] Age
DGRAPH  Pressure; Age
RUGPLOT [SCREEN=keep] Pressure
RUGPLOT [SCREEN=keep; ORIENTATION=across] Age
```

2.2.4 Stem-and-leaf plots

`STEM` procedure

Produces a simple stem-and-leaf chart (J. Ollerton & S.A. Harding).

No options

Parameters

`DATA` = *variates*	Data values for each plot
`NDIGITS` = *scalars*	Number of digits in the leaves of each plot
`STEMUNITS` = *scalars*	Scale units for the stem values in each plot

The `STEM` procedure also displays the distribution of a variate of data, but in the form of a simple stem-and-leaf diagram. The stems indicate leading digits and the leaves indicate subsequent digits. By default, the leaves are formed from single digits; the parameter `NDIGITS` can be used to specify the number of digits in each leaf if more than one is required. The `STEMUNITS`

parameter can be used to specify the units represented by the stem values. By default, this is determined from the data so that the display will fit within a single screen or page of output. Small values of STEMUNITS (in comparison to the range of the data) should be avoided as they may generate far too many lines of output. The display produced by STEM is restricted to the current output width; any lines that have to be truncated at the right-hand margin are terminated by >, indicating their continuation.

Example 2.2.4

```
   2  VARIATE [NVALUES=18] Prices
   3  READ [PRINT=data] Prices

   4   250  150  795  895  696 1699 1499 1099 1693
   5  1166  688 1333  895 1775  895 1895  795  806 :
   6  STEM Prices; NDIGIT=1


Stem-and-leaf display for Prices
Number of observations: 18. Minimum: 150.0. Maximum: 1895.0.
Stem units: 100, leaf digits: 1 (the value 150.0 is represented by  1|5)

1     1|5
1     2|5
0     3|
0     4|
0     5|
2     6|89
2     7|99
4     8|0999
0     9|
1    10|9
1    11|6
0    12|
1    13|3
1    14|9
0    15|
2    16|99
1    17|7
1    18|9

   7  STEM Prices; NDIGIT=2


Stem-and-leaf display for Prices
Number of observations: 18. Minimum: 150.0. Maximum: 1895.0.
Stem units: 100, leaf digits: 2 (the value 150.0 is represented by  1|50)

1     1|50
1     2|50
0     3|
0     4|
0     5|
2     6|88,96
2     7|95,95
4     8|06,95,95,95
0     9|
1    10|99
1    11|66
0    12|
1    13|33
1    14|99
0    15|
2    16|93,99
1    17|75
1    18|95
```

2.2.5 Tally tables and plots

TALLY procedure

Forms a simple tally table of the distinct values in a vector (D.B. Baird & R.D. Stern).

Options

PRINT = *string tokens*	What to print out for each vector (frequencies, percentages, cumfrequencies, cumpercentages, cumgraph, all); default freq, perc
GRAPH = *string tokens*	What to display as graphs (cumulative, %cumulative); default * i.e. no graphs
NGROUPS = *scalar*	Number of groups to form from a DATA variate or factor (ignored for texts); default * forms a group for each distinct value allowing for rounding (see DECIMALS)
DECIMALS = *scalar*	Number of decimal places to which to round the DATA before forming the groups; default * i.e. no rounding
BOUNDARIES = *string token*	Whether to interpret the LIMITS as upper or lower boundaries (upper, lower); default lowe
DIRECTION = *string token*	Order in which to sort (ascending, descending); default asce
OMITEMPTY = *string token*	Whether empty groups are omitted (yes, no); default no
WEIGHTS = *variate*	Weights to be used in the tabulations; default * indicates that all units have weight 1
PQUANTILES = *string token*	Whether to include quantiles on the plot (yes, no); default no
WINDOW = *scalar*	Window in which to plot the graphs; default 1 if GROUPS is set, or 3 otherwise
KEYWINDOW = *scalar*	Window in which to display the key when GROUPS is set; default 2
SCREEN = *string token*	Whether to clear screen before the plot (clear, keep); default clea

Parameters

DATA = *variates*, *factors* or *texts*	Data to be tallied
GROUPS = *factors*	Defines groupings of the data, to be tallied into separate tables; default * i.e. none
LIMITS = *variates* or *texts*	Limits to define the groups within the tally tables
FREPRESENTATION = *string tokens*	Specifies the representation used to define the sort order of a DATA factor (ordinals, levels, labels); default leve
VALUES = *variates*, *texts* or *pointers*	Saves the distinct groups formed for the tally tables
FREQUENCIES = *variates* or *pointers*	Saves the frequencies of the groups in the tally tables
PERCENTAGES = *variates* or *pointers*	Saves the percentage occurrences of the groups
CUMFREQUENCIES = *variates* or *pointers*	Saves the cumulative frequencies of the groups
CUMPERCENTAGES = *variates* or *pointers*	Saves the cumulative percentages of the groups
TITLE = *texts*	Title for plot; default automatically forms a title

	containing the identifiers of the DATA vector and any GROUPS factor
XTITLE = *texts*	Title for the axis representing data values; default uses the identifier of the DATA vector

The TALLY procedure provides another way of displaying the distribution of a set of data. This is in the form of a tally table, giving the counts, percentages, and cumulative counts and percentages of each distinct value. The data can be supplied, using the DATA parameter in a variate, factor or text. So the values can be either numerical or textual. You can also define groups, by specifying a factor using the GROUPS parameter. Separate tables are then formed for each group.

By default, the factor classifying the groups within the tally tables contains a level for each distinct data value. You can decrease the number of groups formed from a DATA variate or text by specifying the NGROUPS and DECIMALS options, or the LIMITS parameter. These work exactly as in the GROUPS directive (1:4.6.1). If limits are specified the BOUNDARIES option controls whether these are interpreted as upper or lower boundaries of the groups; by default they are lower limits. The value to represent each group is the median of the units in the group. The WEIGHTS option can supply a variate of weights for the units of the vector, to be used when calculating the table. If this is not set, the units are all assumed to have weights equal to one.

The PRINT option controls which summaries are printed. The DIRECTION option controls the order of the tally table (ascending or descending). For a factor, the FREPRESENTATION parameter controls which attribute is used to sort the groups (ordinals, levels or labels); by default the levels are used. the OMITEMPTY option can be set to omit empty groups.

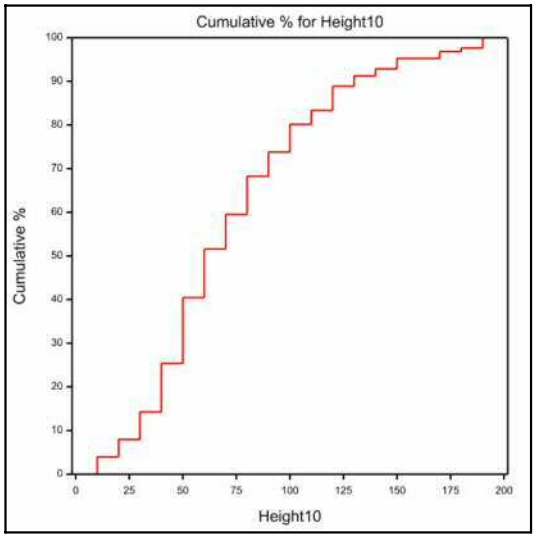


Figure 2.2.5

The GRAPH option may be set to cumulative to produce a cumulative frequency graph, or %cumulative to produce a percentage graph. The PQUANTILES option controls whether or not the graphs include quantiles. The WINDOW and KEYWINDOW options specify the numbers of the windows to use for the plot and key respectively, and the SCREEN option controls whether the screen is cleared first. The TITLE parameter allows you to define an overall title for the graphs, and the XTITLE parameter allows you to define a title for their x-axes. If these are not set, suitable titles are defined automatically.

The VALUES, FREQUENCIES, PERCENTAGES, CUMFREQUENCIES, CUMPERCENTAGES parameters can be used to save the information. This is in variates or texts, if there are no GROUPS; otherwise it is in pointers, containing a variate or text for each group.

Example 2.2.5 prints a tally table for the heights of the volcanos from Example 2.1.1. The CALCULATE statement in line 21 rounds the heights down to multiples of ten. The TALLY statement (line 22) prints the tally table and plots a graph of the cumulative percentages (Figure 2.2.5).

Example 2.2.5

```
21  CALCULATE Height10 = INTEGER(Height / 10) * 10
22  TALLY     [GRAPH=%cumulative] Height10
```

```
Tally of Height10
=================

        Value    Frequency   Percentage   Cumulative Cumulative %
           10            5          4.0            5          4.0
           20            5          4.0           10          7.9
           30            8          6.3           18         14.3
           40           14         11.1           32         25.4
           50           19         15.1           51         40.5
           60           14         11.1           65         51.6
           70           10          7.9           75         59.5
           80           11          8.7           86         68.3
           90            7          5.6           93         73.8
          100            8          6.3          101         80.2
          110            4          3.2          105         83.3
          120            7          5.6          112         88.9
          130            3          2.4          115         91.3
          140            2          1.6          117         92.9
          150            3          2.4          120         95.2
          170            2          1.6          122         96.8
          180            1          0.8          123         97.6
          190            3          2.4          126        100.0
```

2.2.6 Dotplots

DOTPLOT procedure

Produces a dot-plot using line-printer or high-resolution graphics (J. Ollerton & S.A. Harding).

Options

GRAPHICS = *string token*	Whether to use high-resolution graphics or line-printer graphics (`lineprinter`, `highresolution`); default `high`
TITLE = *text*	Title for the Dot Plot; default *
WINDOW = *scalar*	Window number for the graph; default 1
SCREEN = *string token*	Whether to clear the screen before plotting or to or continue plotting on the old screen (`clear`, `keep`); default `clea`
ENDACTION = *string token*	Action to be taken after completing the plot (`continue`, `pause`); default * uses the current setting
DIRECTION = *string token*	Order in which to sort the data before plotting, `DIRECTION=*` implies plot unsorted data (`ascending`, `descending`); default `asce`
LINES = *string token*	How to draw guide lines on the plot, `LINES=*` omits the guide lines (`todot`, `full`); default `todot` draws lines from the *x*-origin to the dots

Parameters

YLABELS = *texts*	Text specifying Y labels for each dotplot
X = *variates*	Data to be plotted
PENDOTS = *scalars*	Pen to draw the dots; default 1
PENLINES = *scalars*	Pen to draw the lines; default 2

Procedure DOTPLOT produces a "dot-plot". Two parameters need to be set: YLABELS supplies a text containing *y*-labels, and X supplies a variate of *x*-data. The display takes the form of a vertical histogram, with a single row for each value of YLABELS. The length of line for each row is specified by the corresponding value of *x*. It is customary to sort the data according to the *x*-values, into either ascending or descending order. This is controlled by the DIRECTION option,

which by default is `ascending`; setting `DIRECTION=*` will plot the data unsorted.

By default a high-resolution graph is given, but you can set option `GRAPHICS=high` to obtain a line-printer plot instead. The guide lines can then also be drawn across the full width of the plot (option `LINES=full`) or can be omitted (`LINES=*`). By default, pens are set up to draw the dots and lines in a form appropriate for the output device. For an interactive display, solid guide lines in pale grey are used; for other devices dashed or dotted lines are used. The plotting symbol is symbol 2 (circle), except for PostScript output which uses a solid dot (`SYMBOL=-9`). The parameters `PENDOTS` and `PENLINES` can be used to specify pens which have been set up with different attributes. The dot-plot is usually produced in window 1, but this can be changed using the `WINDOW` option. A `FRAME` statement can be used before using `DOTPLOT` to change the size and position of the display (for example to widen the x lower margin to allow more space for the *y*-labels). The `SCREEN` option controls whether or not the screen is cleared before plotting and the `ENDACTION` option determines what action to take after completing the plot. An `XAXIS` or `YAXIS` statement can be used to set axis titles, and modify the upper and lower bounds of the *x*-axis. If `TITLE` is unset and axis titles are not set explicitly, they will be generated from the identifier names of the `YLABEL` and `X` parameters.For high-resolution plots, the default window size specifies a lower *x*-margin of size 0.12. This allows room for a title and labels of up to about 10 characters. To produce a dot-plot with longer labels, a `FRAME` statement should be used to specify new dimensions for the window that include a larger value for `XMLOWER`. A full-size window, with standard margins, has room for about 48 rows before the labels start to overlap. To produce a dot-plot with more rows the margins should be reduced or the axis pen size reduced.

Example 2.2.6

```
 2  " Dotplot: data from Cleveland, William S. (1985). The Elements
-3    of Graphing Data, Wadsworth Advanced Books and Software. P.115."
 4  TEXT [NVALUES=22] Cities
 5  VARIATE [NVALUES=22] Population
 6  READ Cities,Population

   Identifier   Minimum      Mean   Maximum    Values   Missing
       Cities                                    22         0
   Population     60.00     259.5      1100        22         0    Skew

11  DOTPLOT [TITLE=\
12    'Populations (in thousands) of cities at end of the 1700s';\
13    GRAPHICS=lineprinter] Cities; Population

 Populations (in thousands) of cities at end of the 1700s

Edinburgh     ...60
Stockholm     ....65
Florence      ....75
Turin         ....80
Genoa         ....80
Warsaw        ....80
Lisbon        .......120
Palermo       .......130
Madrid        ........140
Berlin        ........145
Rome          .........160
Petersburgh   ..........180
Copenhagen    ..........190
Venice        ...........200
Dublin        ...........210
Amsterdam     ............220
Moscow        ..............250
Vienna        ..............255
Naples        ....................380
Paris         ...................................690
Constantinople.............................................900
```

```
London ..................................................1100
```

2.2.7 Probability plots

DPROBABILITY procedure

Creates a probability distribution plot of the values in a variate (D.B. Baird).

Options

PRINT = *string tokens*	Controls whether to print estimated parameters of the distribution or test statistics (parameters, tests); default para
DISTRIBUTION = *string token*	Distribution for expected values against which to plot values (normal, stdnormal, lognormal, exponential, expnormal, gamma, weibull, beta, b2, pareto, chisquare, cauchy, logistic, ev1, ev2, ev3, gev, invnormal, t, f, uniform, skewnormal, stduniform, laplace, gpareto, ubetamix, ugammamix, loggamma, loglogistic, paralogistic, igamma, iweibull, burr, iburr); default norm
METHOD = *string token*	Method used for the plot axes (quantile, probability, stabilizedprobability); default quan
GRAPHICS = *string token*	Type of graphics (highresolution, lineprinter); default high
PLOT = *string tokens*	Whether to plot differences from expectations or the 1-1 reference line (differences, reference); default refe
CONSTANT = *string token*	Whether to estimate the constant for the distribution (estimate, omit) default omit
BANDS = *string token*	What type of confidence bands to plot, if any (simultaneous, pointwise); default simu
NSIMULATIONS = *scalar*	Number of simulations for pointwise bands; default 100
ALPHA = *scalar*	Acceptance limits for confidence bands; default 0.95
DF = *scalar*	Number of degrees of freedom of chi-square or t distribution; default 1
DFNUMERATOR = *scalar*	Numerator degrees of freedom of F distribution; default 1
DFDENOMINATOR = *scalar*	Denominator degrees of freedom of F distribution; default 1
WINDOW = *scalar*	Window to use for the plot; default 3
XMETHOD = *string token*	Scaling of X / Expected Plot axes (quantile, probability, stabilizedprobability); if unset, takes the same setting as METHOD
QMETHOD = *string token*	Whether to standardize plotted score in expected quantiles (standardized, unstandardized); default stan
TMETHOD = *string tokens*	Specifies the method used to perform the goodness-of-fit tests (likelihoodratio, traditional); default like
NTIMES = *scalar*	Number of Monte-Carlo simulations to perform for

	likelihood-ratio tests; default 999
SEED = *scalar*	Seed for random number generation for the likelihood-ratio tests; default 0 continues an existing sequence or, if none, selects a seed automatically

Parameters

DATA = *variates*	Values to plot
TITLE = *text*	Title for the graph; default * generates an appropriate title automatically
ESTIMATES = *variates*	Saves the estimated parameters for the distribution
SE = *variates*	Saves standard errors for the estimated parameters
LOWERTRUNCATION = *scalars*	Lower truncation points for Loss distributions
UPPERTRUNCATION = *scalars*	Upper truncation points for Loss distributions
DEVIANCE = *scalars*	Saves the deviance for the fitted distribution
PROBABILITIES = *variates*	Saves the probabilities from the goodness-of-fit tests

DPROBABILITY produces plots to help you assess whether the distribution of an observed set of data might be modelled by a particular theoretical distribution. The idea is to plot the sorted values (the order statistics, X_i) against the expected values of the order statistics E_i from the given distribution. However, usually the particular parameters of the distribution are not known, and these have to be estimated within DPROBABILITY using the directives DISTRIBUTION (2.2.10) or FITNONLINEAR (3.8.2) to obtain the expected values.

If the distribution has a cumulative density function of F(x), and the inverse of this function is G(x) (i.e. G(F(x)) = x), then the expected values of the order statistics, are approximately G((i−0.5)/n), where i = 1...n, and n is the number of values in the sample. A plot of X_i versus E_i is known as a Quantile-Quantile (or Q-Q) plot. The data can also be plotted on the probability scale by plotting the cumulative probabilities of the data under the assumed distribution against their expected probabilities, i.e. F(X(i)) versus (i−0.5)/n. This is known as a Probability-Probability (or P-P) plot.

A third plot called the stabilized probability (SP) plot (Michael 1983), was introduced, which rescales the probabilities using the transformation

$$sp = (2/\pi) \times \text{ARCSIN}(\text{SQRT}(p))$$

so that the variance of the plotted points is approximately equal over the range of probability values. In the SP plot the scaled values sp are plotted rather than the unscaled p values. The METHOD option allows the choice of which scale is used in the graph (quantile, probability or stabilizedprobability for the Q-Q, P-P or SP plots respectively).

By default the x-value used in plotting Q, P or SP is the corresponding expected value of these statistics. Alternative x-values can be used by setting the XMETHOD option to quantile, probability, or stabilizedprobability. So for example a Q-P plot can be obtained with the option settings METHOD=quantile and XMETHOD=probability or a P-Q plot with the settings METHOD=probability and XMETHOD=quantile.

The QMETHOD option allows the scaling of the expected quantiles plotted on the x-axis to be set. By default quantiles are standardized to have a mean of zero and variance of one (as in a normal score plot) but, if QMETHOD=unstandardized, the quantiles are scaled to the same mean and variance as the data.

The DATA parameter specifies the data values, in a variate. The TITLE parameter can specify a title for the graph. The ESTIMATES parameter can be used to save the values estimated for the parameters for the distribution, and the SE parameter can save their standard errors.

The distribution for the expected values against which to plot the data is specified by the DISTRIBUTION option. Some distributions (Log-Normal, Gamma, Weibull and Pareto) can have an extra parameter (a) estimated, so that $X-a$ follows the specified distribution. Setting option

CONSTANT=estimate estimates a value for *a*. Some of the distributions (Chi Square, T and F) cannot have the parameters estimated by the usual DISTRIBUTION directive, so the procedure provides 3 options (DF, DFNUMERATOR, DFDENOMINATOR) for specifying the parameters of these distributions. However, if for example you set DF=*, the degrees of freedom are estimated along with the other parameters of the distribution.

Some distributions (normal, loggamma, loglogistic, paralogistic, igamma, iweibull, burr, iburr) can be estimated and plotted in a truncated form. The values in the distribution less than LOWERTRUNCATION and greater than UPPERTRUNCATION are removed (if either of these are set), and the distribution between these limits is rescaled to have an area of one. If only LOWERTRUNCATION is set, the distribution is left-truncated, and it is right-truncated if only UPPERTRUNCATION is set.

The BANDS option allows two forms of confidence intervals to be displayed in the graph. BANDS=pointwise simulates NSIMULATIONS distributions of the same size as the data, from the theoretical distribution, and plots the range of values at each value of the order statistics that contain the proportion specified by the option ALPHA of simulated values. Thus a sample drawn from the assumed distribution has approximately a probability ALPHA of lying within the limits at each point. However, overall there will be a probability of less than ALPHA that a sample will completely lie within the confidence bands. The BANDS=simultaneous uses a statistic given by Michael (1983) for which the overall probability of plotted data lying completely within the confidence bands is approximately the specified value of ALPHA, under the null hypothesis that the data is a random iid sample from the specified distribution. This form of confidence limits has the advantage that it is much faster to calculate and that probability of the data points falling outside the limits is approximately constant over the range of the data.

When plotting the data against the expected values, setting option PLOT=reference allows the 1-1 line to be added to the graph, so that departures from this can be more easily observed. The other PLOT setting, difference, plots the difference between the data and the expected values, so that departures can be observed more easily in a horizontal direction rather than on a 45 degree slant. Setting option GRAPHICS=lineprinter produces a character based graph in the output window rather than in the high-resolution graphics window as usual. The WINDOW option can be used to specify which graphics window to use for a high-resolution graph.

The PRINT option control of the output that is printed. The parameters setting prints the fitted parameters of the specified distribution, and some sample statistics of the observed data. The test setting provides output from three empirical distribution tests, namely the Anderson-Darling, Cramer-von Mises and Watson statistics. The method used to perform these tests is specified by the TMETHOD option, with settings likelihoodratio for the Zhang (2002) likelihood-ratio based method, and traditional for the traditional approach. The default is to use the likelihood-ratio based tests, which are generally more powerful. Monte-Carlo simulations are used to calculate the empirical probability values of the test statistics under the likelihood-ratio based method. The NTIMES option defines how many Monte-Carlo simulations are used; default 999. The SEED option specifies the seed for the random-number generator used during the Monte-Carlo simulations. The default of zero continues the sequence of random numbers from a previous generation or, if this is the first use of the generator in this run of Genstat, the seed is initialized automatically. The test probabilities can be saved, in a variate, by the PROBABILITIES parameter.

Further information about the distributions fitted in this procedure can be found in the books by Hogg & Klugman (1984) and Johnson, Kotz & Balakrishnan (1994, 1995).

Figure 2.2.7 shows a Q-Q plot with the gamma distribution for the volcano data introduced in Section 2.1.1. This was produced by the statement

```
DPROBABILITY [PRINT=*;\
DISTRIBUTION=gamma]\
  Height
```

(The `PRINT` option is set to suppress output as the distribution will be fitted explicitly by the `DISTRIBUTION` directive, in Section 2.2.10).

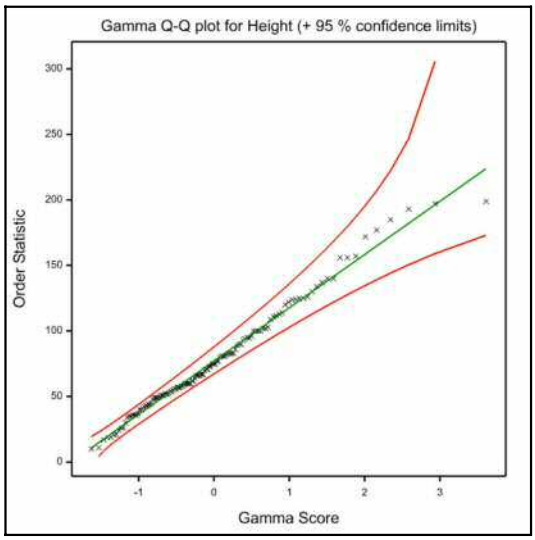


Figure 2.2.7

2.2.8 Kernel density estimation

`KERNELDENSITY` procedure

Uses kernel density estimation to estimate the underlying density of a sample (P.W. Goedhart).

Options

`PRINT` = *string token*	What to print (`integral`, `summary`, `monitoring`, `graph`); default `inte`
`METHOD` = *string token*	Which automatic bandwidth selection method should be used when the `BANDWIDTH` option is not set (`s1`, `s2`, `s3`, `sj`); default `sj`
`BANDWIDTH` = *scalar* or *variate*	Which bandwidth value or values are to be used; default *
`NGRIDEXPONENT2` = *scalar*	Defines the number of grid points as 2**`NGRIDEXPONENT2`; default 11
`SAVEGRIDEXTENT` = *scalar*	Defines the lower and upper limit of the interval on which the kernel density is saved; the default value of 4 uses the full interval on which the kernel density is calculated
`NFOURIER` = *scalar*	Defines the upper limit of the sample size for which the kernel density is calculated directly (when the sample size exceeds the setting of this option, the fast Fourier transform is used to calculate the kernel density); default 100
`PROPORTION` = *variate*	Proportions at which to calculate quantiles of the kernel density estimate; default !(0.025, 0.25, 0.5, 0.75, 0.975)
`PLOT` = *string tokens*	Specifies the graphs to be plotted (`kerneldensity`, `histogram`, `sample`); default `kern`, `hist`, `samp`
`TITLE` = *text*	General title(s) for the graph(s); default *
`WINDOW` = *scalar* or *variate*	Window number(s) for the graph(s); default 1
`SCREEN` = *string token*	Whether to clear the screen before plotting into the first window, or whether to or continue plotting on the old screen (`clear`, `keep`); default `clea`

Parameters

SAMPLE = *variates*	The sample for which to calculate the kernel density estimate
GRID = *variates*	Saves the grid of equidistant points at which the kernel density is calculated
DENSITY = *variates* or *pointers*	Saves the kernel density estimate
CUMULATIVE = *variates* or *pointers*	Saves the estimated cumulative distribution
QUANTILE = *variates* or *pointers*	Saves the quantiles calculated from the estimated cumulative distribution
SAVEBANDWIDTH = *scalars*	Saves the automatically selected bandwidths as specified by the METHOD option

Kernel density estimation is a way of estimating the probability density curve for a sample, without assuming that they come from any specific probability distribution.
See for example Silverman (1986) for a general introduction in density estimation. The kernel method constructs an estimate $f_h(t)$ of the true density function by placing a kernel function $K(t; x_i, h)$ over each observation x_i in the sample. The kernel function $K(t; x, h)$ is itself a density function with location parameter x and scale parameter h, also called bandwidth in this context. The density estimate is then given by

$$\hat{f}_H(t) = \frac{1}{nh} \sum_{I=1}^{N} K\left(\frac{t - x_I}{h}\right) \tag{1}$$

where n denotes the sample size. It turns out that the choice of kernel function K is not very critical for the resulting estimate $f_h(t)$, see Section 3.3 of Silverman (1986). The Gaussian kernel is commonly used and is therefore adopted here as kernel function, i.e.

$$K(t) = \frac{1}{\sqrt{2\pi}} e^{-t^2/2} \tag{2}$$

For this choice of kernel function K, there is an efficient algorithm available for the calculation of $f_h(t)$. This algorithm employs the fast Fourier transform of the data.

The choice of bandwidth h is of crucial importance in kernel density estimation. A large value of h will give rise to an oversmoothed density estimate, while a small value of h will produce a very ragged density with many spikes at the observations. Silverman (1986) recommends examining kernel density estimates for several values of h, since this will highlight different features of the data. For automatic use of kernel density estimation, estimation of the bandwidth h from the data is very helpful. Silverman (1986) suggests the following normal-based estimates:

S1 = 1.06 × (standard deviation) × $n^{-1/5}$
S2 = 0.79 × (interquartile range) × $n^{-1/5}$
S3 = 0.90 × minimum(standard deviation, interquartile range/1.34) × $n^{-1/5}$

These estimates are popular due to their simplicity. Jones, Marron & Sheather (1996), who provide an extensive review of the many automatic methods for choosing the bandwidth, advise against these estimates. They recommend the method of Sheather & Jones (1991) for general purposes. This method, denoted below by SJ, is therefore the default method used in the KERNELDENSITY procedure.

The sample, for which to estimate the underlying density, must be specified by means of the SAMPLE parameter. The METHOD and BANDWIDTH options determine which bandwidths h are used. When the BANDWIDTH option is set to a scalar or variate, then these values are used for the bandwidth h. When the BANDWIDTH option is unset, the METHOD option determines which automatic bandwidth selection method is used. The default setting of the METHOD option is sj, which indicates that the method of Sheather & Jones (1991) is to be used. The automatically

selected bandwidth can be saved by means of the SAVEBANDWIDTH parameter.

The kernel density estimate is calculated on an interval at a grid of equidistant points. The grid is returned using the GRID parameter, and the density estimate and corresponding cumulative density can be saved with the DENSITY and CUMULATIVE parameters. When the BANDWIDTH option is set to a variate, the DENSITY and CUMULATIVE parameters are pointers to variates: one variate for each bandwidth value. The number of grid points can be set using the NGRIDEXPONENT2 option as 2**NGRIDEXPONENT2. The lower and upper limit of the interval on which the kernel density is calculated are given by:

```
CALCULATE lower = MINIMUM(SAMPLE) - 4*MAXIMUM(BANDWIDTH)
CALCULATE upper = MAXIMUM(SAMPLE) + 4*MAXIMUM(BANDWIDTH)
```

This ensures that the integral of the kernel density will be very close to one. The SAVEGRIDEXTENT option can be used to save the grid and the (cumulative) density at a more limited interval defined by

```
CALCULATE lowsave = MINIMUM(SAMPLE) \
                    - SAVEGRIDEXTENT*MAXIMUM(BANDWIDTH)
CALCULATE uppsave = MAXIMUM(SAMPLE) \
                    + SAVEGRIDEXTENT*MAXIMUM(BANDWIDTH)
```

The setting of the NFOURIER option determines whether the kernel density is calculated directly by means of equation (1) or by employing the fast Fourier transform of the data. When the sample size n exceeds the setting of the NFOURIER option, the fast Fourier transform is used.

The parameter QUANTILES can be used to save quantiles of the kernel density estimate, for proportions specified by means of the PROPORTION option. When the BANDWIDTH option is set to a variate, the QUANTILES are saved in a pointer containing a set of variates.

The PRINT option controls the output displayed by KERNELDENSITY. The integral setting prints the integral of the kernel density, which should be close to one, while the summary setting print summary statistics of the sample and of the kernel density estimate. The monitoring setting can be used to monitor the iterative bandwidth estimation method SJ. Finally, the setting graph produces a high-resolution plots of the kernel densities, superimposed over a rough estimate of the density calculated as the proportion of the sample falling into CEILING(SQRT(number of samples))+1 equal intervals across the range of sample values. (There will be as many plots as there were bandwidths.) The sample values are also plotted, using the symbol +, along the bottom of the plots. The PLOT option controls which elements (kerneldensity, histogram, sample) are plotted. The TITLE option can provide a title for each graph. The WINDOW option specifies the windows to be used for the plots (default 1), and the SCREEN option controls whether or not the screen is cleared before plotting into the first window (default clear).

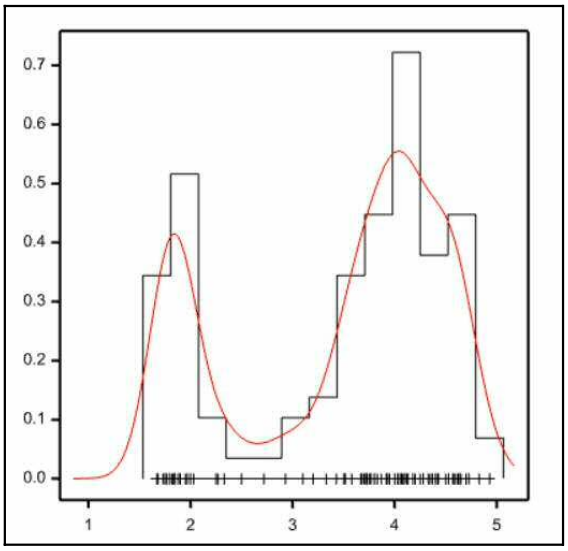


Figure 2.2.8

Example 2.2.8 produces a kernel density estimate, plotted in Figure 2.2.8, of the distribution of the eruption lengths (in minutes) of the Old Faithful geyser (from Table 2.2 of Silverman 1986).

Example 2.2.8

```
  2  VARIATE [NVALUES=107] Eruption
  3  READ    Eruption

   Identifier   Minimum      Mean   Maximum    Values   Missing
     Eruption     1.670     3.460     4.930       107         0

 12  KERNELDENSITY [PRINT=summary,graph] Eruption

Summary statistics for Eruption with bandwidth 2.0371E-01

                          Sample   Kernel density
                Mean       3.460            3.460
  Standard deviation       1.040            1.055
              Median       3.800            3.810
      Lower quartile       2.285            2.340
      Upper quartile       4.250            4.279
            Integral           -          1.00000
```

2.2.9 Plots of circular data

DCIRCULAR procedure

Plots circular data (P.W. Goedhart & R.W. Payne).

Options

PLOT = *string tokens*	Information to be plotted (counts, kerneldensity, lines, mean, rose); default coun, mean, rose
TITLE = *text*	Title for the graph; default * i.e. none
SEGMENT = *scalar*	Width of sectors (in degrees) into which to group an ANGLES variates before plotting; default 20
MSEGMENT = *scalar*	Defines the centre (in degrees) of the sectors; default 0
BANDWIDTH = *scalar*	Bandwidth to use for the kernel density estimate; if this is unset, the value h_0 suggested by Fisher (1993, page 26) is used
NGRID = *scalar*	Defines the number of grid points for the kernel density estimate; default 180
WINDOW = *scalar*	Window for the graph; default 3
SCREEN = *string token*	Whether to clear screen before displaying the graph (keep, clear); default clea

Parameters

ANGLES = *factors* or *variates*	Directional observations to be plotted
GRID = *variates*	Saves the grid (in degrees) on which the kernel density is estimated
DENSITY = *variates*	Saves the kernel density estimate
SAVEBANDWIDTH = *scalar*	Saves the calculated bandwidth h_0 when BANDWIDTH is unset

DCIRCULAR plots data values that consist of directional observations recorded as angles between 0 and 360 degrees. The data values are supplied by the ANGLES parameter, in either a variate or a factor. With a variate, the observations are grouped for plotting into sectors of width specified (in degrees) by the SEGMENT option, with centres defined by the MSEGMENT option. The sectors are centred at MSEGMENT, MSEGMENT+SEGMENT, MSEGMENT+2*SEGMENT, and so on. The default

value for SEGMENT and MSEGMENT is 20 and 0 respectively. If ANGLES is set to a factor, its levels define the midpoints of the sectors and these must be in clockwise order.

The graph contains a circle with marks at every 10 degrees, and labels at 0, 90, 180 and 270 degrees. The representations of the observations are determined by the settings supplied for the PLOT option as follows

counts	plots counts of the number of observations in each sector.
kerneldensity	plots estimates of the probability distribution of the data, using a quartic kernel function with bandwidth specified by the BANDWIDTH option. If BANDWIDTH is unset, a default is calculated based on the estimated concentration of the data (this is the value h_0 suggested by Fisher, 1993, page 26). The kernel is calculated on a grid of values with number of values defined by the NGRID option.
lines	plots lines in each direction with lengths proportional to the number of observations in that direction.
mean	plots the mean vector (see Fisher 1993, page 31).
rose	plots a "rose" diagram in which the observations in each sector are represented as a triangle with apex at the centre of the circle and area proportional to the number of observations there.

By default PLOT=counts,mean,rose.

The options TITLE, WINDOW and SCREEN allow you to define a title for the plot, specify which window to use, and indicate whether or not to clear the screen beforehand. Parameters GRID, DENSITY and SAVEBANDWIDTH can be used to save the grid (in degrees), kernel estimate and bandwidth h_0. The latter is saved only when BANDWIDTH is unset.

Figure 2.2.9 shows a plot containing the counts, the mean direction and a rose diagram for the data concerning directions taken by ants, discussed in Section 2.1.2. This confirms the impression, given by the summary statistics calculated in Example 2.1.2, that the ants are attracted by the target at 180 degrees. The plots were generated by the statement

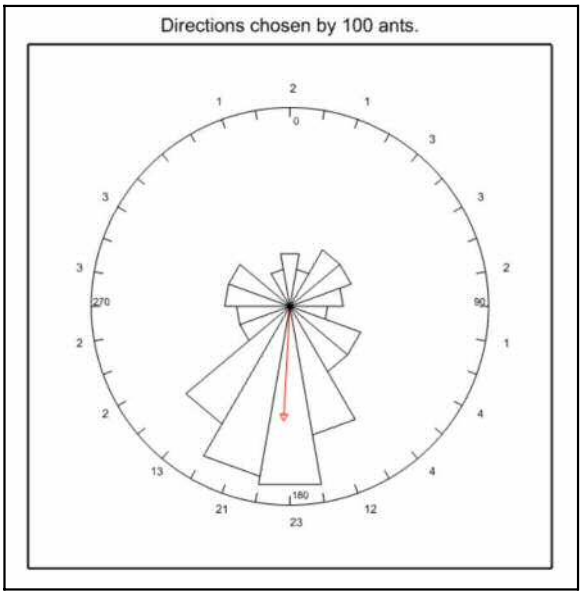


Figure 2.2.9

```
DCIRCULAR [TITLE='Directions chosen by 100 ants.'] Direction
```

Other plots of circular data can be obtained using the WINDROSE procedure, which provides the wind-rose diagrams that are often used to represent climatic data.

2.2.10 Estimating the parameters of a distribution

DISTRIBUTION directive

Estimates the parameters of continuous and discrete distributions.

Options

PRINT = *string tokens*	Printed output required from each individual fit (parameters, samplestatistics, fittedvalues, proportions, monitoring); default para, samp, fitt
CBPRINT = *string tokens*	Printed output required from a fit combining all the input data (parameters, samplestatistics, fittedvalues, proportions, monitoring); default *
DISTRIBUTION = *string token*	Distribution to be fitted (Poisson, geometric, logseries, negativebinomial, NeymanA, PolyaAeppli, PlogNormal, PPascal, Normal, dNvequal, dNvunequal, logNormal, exponential, gamma, Weibull, b1, b2, Pareto); default * i.e. fit nothing
CONSTANT = *string token*	Whether to estimate a location parameter for the gamma, logNormal, Pareto, or Weibull distributions (estimate, omit); default omit
LIMITS = *variate*	Variate to specify or save upper limits for classifying the data into groups; default *
NGROUPS = *scalar*	When LIMITS is not specified, this defines the number of groups (of approximately equal size) into which the data are to be classified; default is the integer value nearest to the square root of the number of data values
XDEVIATES = *variate*	Variate to specify points up to which the CUMPROPORTIONS are to be estimated
JOINT = *string token*	Requests joint estimates from the combined fit to be used for a re-fit to the separate data sets (dispersion, variancemeanratio, Poissonindex); default *
PARAMETERS = *variate*	Estimated parameters from the combined fit
SE = *variate*	Standard errors for the estimated parameters of the combined fit
VCOVARIANCE = *symmetric matrix*	Variance-covariance matrix for the estimated parameters of the combined fit
CUMPROPORTIONS = *variate*	Estimated cumulative proportions of the combined distribution up to the values specified by the XDEVIATES option
MAXCYCLE = *scalar*	Maximum number of iterations; default 30
TOLERANCE = *scalar*	Convergence criterion; default 0.0001

Parameters

DATA = *variates* or *tables*	Data values either classified (table) or unclassified (variate)
NOBSERVATIONS = *tables*	One-way table to save the data classified into groups
RESIDUALS = *tables*	Residuals from each (individual) fit
FITTEDVALUES = *tables*	Fitted values from each fit

PARAMETERS = *variates*	Estimated parameters from each fit
SE = *variates*	Standard errors of the estimates
VCOVARIANCE = *symmetric matrices*	Variance-covariance matrix for each set of estimated parameters
CUMPROPORTIONS = *variates*	Estimated cumulative proportions of each distribution up to the values specified by the XDEVIATES option
CBRESIDUALS = *tables*	Residuals from the combined fit
CBFITTEDVALUES = *tables*	Fitted values from the combined fit
STEPLENGTH = *variates*	Initial step lengths for each fit
INITIAL = *variates*	Initial values for each set fit

The DISTRIBUTION directive (which corresponds to the Fit Distribution menu of Genstat *for Windows*) is used to fit an observed sample of data to a theoretical distribution function, in order to estimate the parameters of the distribution and test the goodness of fit. The data consists of observations x_i of a random variable X, which has a distribution function $F(x)$ defined by $F(x)=\Pr(X \le x)$. A selection of both discrete and continuous distributions are available, and full details are given later in this section.

For discrete distributions X may take non-negative integer values only, except for the log-series distribution where only positive integer values are allowed. For continuous distributions the random variable X may take any values, subject to constraints for certain distributions, for example, data values must be strictly positive in order to fit a log-Normal distribution. Constraints are detailed with the individual distributions described below.

The data can be supplied to DISTRIBUTION as a variate or as a one-way table of counts. The raw data should be supplied (as a variate) if they are available, since these provide more information than grouped data.

If raw data are not available, a one-way table of counts (or frequencies) should be given. The factor classifying the table must have its levels vector declared explicitly, since the levels are used to indicate the boundary values of the raw data used to create the grouping. For example, if the discrete variable X takes the values 0...8, with numbers of observations 2,6,7,4,2,1,0,1,0 respectively, a table of counts can be declared by

```
FACTOR [LEVELS=!(0...8)] F
TABLE [CLASSIFICATION=F; VALUES=2,6,7,4,2,1,0,1,0] T
```

The factor levels do not have to specify single data values: often it will be desirable to group certain values together, and indeed for continuous data this is the only sensible way to proceed. In general, for a classifying factor with levels $l_1, l_2, \ldots, l_f$, the count n_k for the kth cell of the table will be the number of observations x_i such that

$x_i \le l_1$, $k=1$
$l_{k-1} < x_i \le l_k$, $2 \le k \le f-1$
$l_{f-1} < x_i$, $k=f$

This means that, for all except the last cell of the table, the factor level represents the upper limit on values in that cell. The final class of the table is termed the *tail*; it is formed by combining the frequencies for all values of X greater than l_{f-1}, and the upper limit on values in the tail is infinity. For continuous distributions with no lower bound, the first class will be the lower tail. You will often want to form the tail(s) by amalgamating groups with low numbers of counts. In the example above, you might amalgamate the groups for values 6-8:

```
FACTOR [LEVELS=!(0...5,99)] F2
TABLE [CLASSIFICATION=F2; VALUES=2,6,7,4,2,1,1] T2
```

Note that the final factor level, for the tail, can be given a dummy value of 99 to indicate that it has no upper limit, since this value is never used in calculations.

When the data are supplied as a table instead of as a variate, the computed log-likelihood is only an approximation to the full log-likelihood and the solution obtained will depend to some extent on the choice of class limits. More reliable results will be achieved with a larger number of classes, since this gives more information on the data distribution, so only classes with very few observations should be amalgamated. In general, care should be taken to choose class limits that give a reasonable number of counts in each class, but with none of the individual classes holding a disproportionately large number of observations.

The `DISTRIBUTION` option should be set to indicate which distribution is to be fitted to the data. The following distributions are available:

Discrete	**Continuous**
Binomial (as a special case of the negative binomial)	Normal
Poisson	Double Normal (equal variances)
Geometric	Double Normal (unequal variances)
Log-series	Log-Normal
Negative binomial	Exponential
Neyman type A	Gamma
Pólya-Aeppli	Weibull
Poisson-log-Normal	Beta type I and type II
Poisson-Pascal	Pareto

The first step of the fitting process is to compute and print various sample statistics. Examining these may help in the selection of appropriate distributions for fitting – properties of the various distributions are listed at the end of this section. The setting `DISTRIBUTION=*` can be used to produce this output without any model fitting. The following sample statistics are calculated:

Sample size	n	
Sample mean	$m = \Sigma\, x_i/n$	
Sample variance	$s^2 = \Sigma\, x_i^2/n - m^2$	discrete distributions
	$s^2 = \Sigma\, (x_i-m)^2 / (n-1)$	continuous distributions
Sample skewness	$g_1 = \Sigma\, (x_i-m)^3 / (n-1)s^3$ $= m_3/s^3$	
Sample kurtosis	$g_2 = \Sigma\{(x_i-m)^4/(n-1)s^4\} - 3$	continuous distributions only
Sample quartiles	x_p: $F(x_p)=p$	
Poisson index	$(s^2-m)/m^2$	discrete distributions only
Negative binomial index	$m(m_3-3s^2+2m)/(s^2-m)^2$	discrete distributions only

If the original data are not available, the sample statistics are calculated by substituting class mid-points in place of the data. For the lower tail, the class "mid-point" is taken to be $l_1-\frac{1}{2}(l_2-l_1)$ and for the upper tail, $l_{f-1}+\frac{1}{2}(l_{f-1}-l_{f-2})$. No corrections are made for groupings. When a distribution has been fitted to data, the relevant theoretical statistics of that distribution are printed for comparison with the sample statistics, as a check on the appropriateness of the model for the data.

If a distribution has been specified, it is then fitted to the data to obtain maximum-likelihood estimates of the parameters, as in Example 2.2.10a below, which fits a gamma distribution to the volcano data introduced in Section 2.1.1.

Example 2.2.10a

```
24  VARIATE [VALUES=20,40...180] Limits
25  DISTRIBUTION [DISTRIBUTION=gamma; LIMITS=Limits] Height
```

```
Fit continuous distribution
===========================

Sample statistics
-----------------

Sample Size          126
Mean               77.19
Variance         1702.91
Skewness            0.96
Kurtosis            0.62

Quartiles:
         25%         50%         75%
        49.0        67.0       100.0


Summary of analysis
-------------------

Observations: Height
              Parameter estimates from individual data values
Distribution: Gamma
              f(x) = (b**k).(x**(k-1)).exp(-bx)/Gamma(k), x>0
Deviance:     8.83 on 7 d.f.


Estimates of parameters
-----------------------

          estimate      s.e.     correlations
k           3.5014    0.4221     1.0000
b           0.0454    0.0059     0.9301  1.0000


Fitted quartiles
----------------

              25%         50%         75%
           46.925      69.979      99.651


Fitted values (expected frequencies) and residuals
--------------------------------------------------

              x      Number      Number    Weighted
                   observed    expected    residual
        < 20.0            5        3.85        0.56
        < 40.0           15       18.63       -0.87
        < 60.0           36       27.12        1.62
        < 80.0           20       25.64       -1.16
       < 100.0           21       19.54        0.33
       < 120.0            9       13.10       -1.20
       < 140.0           11        8.06        0.98
       < 160.0            3        4.67       -0.83
       < 180.0            2        2.58       -0.38
       > 180.0            4        2.81        0.67
```

A summary is given of the fit: the parameter estimates are printed with their standard errors and correlations, including the *working parameters*, which are *stable* functions of the parameters defining the distribution and are used in the internal algorithm (Ross 1990). The goodness of fit for the chosen distribution is indicated by the residual deviance which has an asymptotic chi-square distribution with the specified degrees of freedom. The deviance is also the preferred statistic for comparison of nested models, for example the double Normal distribution with equal and unequal variances. This is followed by a table of observed and fitted values (expected frequencies), together with weighted residuals. If raw data are supplied, by default this table is formed by dividing the data into $\sqrt{n}$ groups of approximately equal observed frequency, which

are therefore likely to be of unequal widths. The NGROUPS option may be used to set the number of groups for this table. If data are supplied as a table, as in Example 2.2.10b, the fitted values use the classification from that table. In either case the LIMITS option may be used to supply a different set of limits, with the constraint that if tabulated data are analysed these limits should be a subset of the original limits so that the new groups are formed by aggregation. In Example 2.2.10a, evenly spaced limits were specified.

Example 2.2.10b shows the analysis of some tabulated data: this is disease data, indicating the number of leaves on which zero, one, up to seven red mites were found. A further cell, containing 0, for eight mites is included in the table as the tail. A negative binomial distribution is fitted to investigate the distribution of mites on leaves.

Example 2.2.10b

```
   2  " Negative binomial fit to counts of European red mites on apple
  -3    leaves, Bliss (1953). The data are recorded as the number of leaves
  -4    having no mites, number with one mite, and so on."
   5  FACTOR [LEVELS=!(0...8)] Mites; DECIMALS=0
   6  TABLE [CLASSIFICATION=Mites] Leaves; DECIMALS=0
   7  READ [PRINT=*] Leaves
   9  PRINT [ACROSS=Mites] Leaves; FIELDWIDTH=6

      Leaves
Mites     0     1     2     3     4     5     6     7     8
         70    38    17    10     9     3     2     1     0

  10  DISTRIBUTION [DISTRIBUTION=negativebinomial] Leaves

Fit discrete distribution
=========================

Sample Statistics
-----------------

Sample size                      150
Mean                            1.15
Variance                        2.26
Skewness                        1.53
Poisson index                   0.85
Negative binomial index         0.66


Summary of analysis
-------------------

Observations: Leaves
              Parameter estimates from tabulated data values
Distribution: Negative Binomial
              Pr(X=r) = (r+k-1)C(k-1).(m/(m+k))**r.(1+m/k)**(-k)
    Deviance: 4.22 on 6 d.f.


Estimates of working parameters
-------------------------------

         estimate       s.e.     Correlations
mean       1.1467     0.1273     1.0000
variance   2.4301     0.5379     0.7663  1.0000


Estimates of defining parameters
--------------------------------

         estimate       s.e.     Correlations
m          1.1467     0.1273     1.0000
k          1.0246     0.2758     0.0001  1.0000
1/k        0.9760     0.2628     Poisson Index
```

```
Fitted values (expected frequencies) and residuals
--------------------------------------------------

   r            Number      Number   Weighted
              Observed    Expected   Residual
   0                70       69.49       0.06
   1                38       37.60       0.07
   2                17       20.10      -0.71
   3                10       10.70      -0.22
   4                 9        5.69       1.28
   5                 3        3.02      -0.01
   6                 2        1.60       0.30
   7                 1        0.85       0.16
   8+                0        0.95      -1.38
```

The NOBSERVATIONS, RESIDUALS and FITTEDVALUES parameters can be used to save the number of observations in each cell, the fitted number and the residual respectively (all in tables). The parameter estimates and their standard errors can be saved in variates specified by PARAMETERS and SE. The variance-covariance matrix for the estimated parameters can be saved as a symmetric matrix using the VCOVARIANCE parameter.

Having fitted the required distribution, the estimated cumulative distribution function (CDF) can be evaluated at specified values of X. These are defined using the XDEVIATES option. The values of the CDF can be printed (by selecting PRINT=proportions) or saved in a variate by setting the CUMPROPORTION parameter.

If you have several sets of data you may be interested in fitting the distribution individually to each set; this can be done by setting the DATA parameter to a list of identifiers. A separate analysis is then performed for each set of data, but of course any option settings are common to all the data sets. The data sets should all be specified in the same way, either as raw data or as tabulated counts. For tabulated counts, the same categories must be used for defining every table. You can also carry out one final fit to the combined data set, in order to investigate whether the data can be adequately modelled as coming from a single population. This combined fit is produced if any of the options relating to the combined fit have been set (that is, options CBPRINT, PARAMETERS, SE, VCOVARIANCE, or CUMPROPORTION which print or save information from the combined analysis). For each individual data set you can also save fitted values and residuals based on the parameters estimated from the combined data set, using the CBRESIDUALS and CBFITTEDVALUES parameters. The JOINT option can be used to specify that certain parameters should be held constant at their estimated values from the combined analysis during refits to the individual data sets. For continuous distributions only, a common dispersion parameter can be requested; for discrete distributions a common value can be requested for either the Poisson index or the ratio of variance to mean. An analysis of deviance is printed to compare the nested models.

If the original data are available, the full log-likelihood is used in the optimization algorithm. Otherwise, an approximate log-likelihood is optimized, using representative values for each class. For some distributions, it is necessary to use stable *working parameters* in the optimization algorithm (Ross 1990), and the *defining parameters* for the distribution are then evaluated by a simple transformation.

The deviance and corresponding degrees of freedom that are printed as part of the model summary are based on the table of fitted values, and thus may be affected by the choice of limits. The residuals computed are deviance residuals (McCullagh & Nelder 1989), and the deviance is therefore the sum of squared residuals. The degrees of freedom are $n-p-1$, where n is the number of cells in the table of fitted values and p is the number of parameters estimated in the model. The default limits for grouping the raw data are designed to avoid small expected frequencies (for example in the tail cells) which can have an inflationary affect on the deviance; however, if the tails are important, because of the origin of the data, it may be important to specify the limits explicitly.

An iterative Gauss-Newton optimization method is used to estimate the parameters of the distribution. The parameterization is chosen for each model so that the optimization is stable, but if there are any problems with particular data sets it may be necessary to control this process. The MAXCYCLE and TOLERANCE options allow you to increase the number of iterations and alter the convergence criterion for data sets that fail to converge. You can also specify initial values and step lengths for the parameters for each set of data using the STEPLENGTH and INITIAL parameters. These parameters should be set to variates of length appropriate for the distribution being fitted; for example, if DISTRIBUTION=Poisson they should have just one value. Another use of INITIAL and STEPLENGTH is to constrain a parameter to a particular value; for example when fitting a double Normal the proportion parameter p could be fixed at 0.5 by setting the initial value to 0.5 and the step length to 0, thus fitting a double Normal in equal proportions. Note that the degrees of freedom are not adjusted to take account of this. Optimization problems are discussed further in 3.7 and 3.8.

We now discuss the distributions that can be fitted, looking first at the discrete and then the continuous distributions. A summary of the theoretical properties of the discrete distributions is given in Table 2.2.10.

Table 2.2.10: Theoretical properties of discrete distributions

	Mean (μ)	Variance (V)	Parameters estimated	Poisson index	Neg.bin. index
Poisson	μ	$V=\mu$	μ	0	–
Geometric	$\mu=(1-p)/p$	$V=(1-p)/p^2$	μ	1	2
Log-series	$\mu=\theta/z(1-\theta)$	$V=\mu[(1-\theta)^{-1}-\mu]$	$z=-\log(1-\theta)$	$z-1$	–
Negative binomial	μ	$V=\mu+\mu^2/k$	μ,V	$1/k$	2
Neyman type A	$\mu=\mu_1\mu_2$	$V=\mu_1\mu_2(1+\mu_2)$	μ,V	$1/\mu_1$	1
Pólya-Aeppli	$\mu=\mu_1/p$	$V=\mu_1(2-p)/p^2$	μ,V	$2(1-p)/\mu_1$	1.5
Poisson-log-Normal	$\mu=\exp(\mu_1+\frac{1}{2}\sigma^2)$	$V=\mu+\mu^2(e^{\sigma^2}-1)$	μ,V	$\exp(\sigma^2)-1$	2
Poisson-Pascal	$\mu=\lambda pk$	–	μ $(k+1)/\lambda k$ $(k+2)/(k+1)$	$(k+1)/\lambda k$	$(k+2)/(k+1)$

The *negative binomial* distribution is applicable in many different situations, and can be derived in several ways. For example: waiting times for the rth success in a sequence of Bernoulli trials have a negative binomial distribution; random sampling from a heterogeneous population described by a mixture of Poisson distributions with means varying according to a gamma distribution will produce negative binomial data; and the distribution can also describe the number of events per unit interval given underlying Poisson and log-series distributions. Further explanation can be found in Ross (1987 and 1990) and Johnson & Kotz (1969). The negative binomial distribution can be defined in terms of the expansion of $(q-p)^{-n}$ with $q-p=1$. In Genstat, it is specified in the form obtained by setting $\mu=np$ and $k=n$, so that the probability of observing the value $X=r$ is given by:

$$p_r = \Pr(X=r) = \binom{r+k-1}{k-1}\left[\frac{\mu}{\mu+k}\right]^r\left[1+\frac{\mu}{k}\right]^{-k} \qquad r=0,1\ldots$$

The parameters estimated are the mean (μ) and the variance ($V=\mu+\mu^2/k$) from which the defining parameters μ and k are derived.

When the sample variance is less than the mean (indicated by a negative value of the Poisson index), the usual (positive) binomial distribution will be fitted where

$$p_r = \binom{N}{r} p^r (1-p)^{N-r} \qquad r=0,1...N$$

In this case, a negative value of k will be estimated, and the index of the binomial distribution, N, will be estimated to be $-k$, where $-k$ exceeds the largest value present in the sample. The probability of a success, p, is derived from $\mu=Np$.

The negative binomial distribution also generates the Poisson distribution (as $k\to\infty$), the geometric distribution (with $k=1$) and log-series distribution (as $k\to 0$) as special cases. Although the estimated parameters μ and k are independent, estimated standard errors for k are not reliable since the confidence interval for k is skew: the deviance should therefore be used to compare the fit of the negative binomial distribution with nested models for particular values of k.

The *Poisson* distribution with mean μ arises as the number of events per unit time, assuming that events are distributed randomly and independently in time (or space), with mean number of events per unit interval equal to μ. The probability of observing r independent events in a unit interval is then:

$$p_r = \frac{\mu^r}{r!} e^{-\mu} \qquad r=0,1...$$

The distribution is described by the single parameter μ, equal to the mean and variance. The skewness is $g_1=1/\sqrt{\mu}$. For a sample from a Poisson distribution with mean μ, the expected value of the Poisson index is 0, with variance $2/n\mu^2$.

The *geometric* distribution is a discrete analogue of the continuous exponential distribution described later in this section, and can be interpreted as the waiting time in a series of Bernoulli trials before an event occurs. The probability that r trials occur before an event is given by:

$$p_r = p\,(1-p)^r \qquad 0<p<1,\ r=0,1...$$

where p is the probability that the event occurs in a single trial. The parameter estimated is the mean ($\mu=(1-p)/p$), from which the defining parameter p is derived.

The *logarithmic series* (or *log-series*) distribution is applicable when there is no zero cell, for example when events are not reported unless they occur at least once. This might occur when a crop survey records numbers of parasites per host for infected plants only. The series is also important in the study of species diversity. The distribution is given by

$$p_r = \frac{\theta^r}{zr} \qquad \text{where} \quad z=-\log(1-\theta),\ r=1,2...,\quad 0<\theta<1$$

The parameter estimated is z, from which θ is derived.

The *Neyman type A* distribution is a *contagious* distribution; that is, one allowing for heterogeneity, in which events are aggregated into groups. The number of groups per unit interval has a Poisson distribution (with mean μ_1), and the number of events per group has an independent Poisson distribution with mean μ_2. The Neyman type A distribution is generated by compounding the two Poisson distributions. The probabilities, p_r, can be described by the recurrence relation:

$$p_0 = \exp(-\mu_1(1-\exp(-\mu_2)))$$

$$p_r = \frac{\mu_1}{r} e^{-\mu_2} \sum_{j=1}^{r} \mu_2^j \frac{p_{r-j}}{(j-1)!} \qquad r>0$$

This distribution is less skew than the negative binomial, and cannot be fitted if the variance is less than the mean. When μ_2 tends to zero the distribution becomes a simple Poisson; if μ_2 tends

to infinity whilst the mean $\mu=\mu_1\mu_2$ remains constant the distribution tends to a Poisson with added zeroes. The distribution is fitted by estimating the mean and the variance from which the defining parameters μ_1 and μ_2 are obtained. These parameters may be highly negatively correlated, since the mean $\mu=\mu_1\mu_2$ is usually well-defined.

The *Pólya-Aeppli* distribution is a contagious distribution where the number of groups per unit interval has a Poisson distribution with mean μ_1 and the number of events per group has a geometric distribution with parameter p. The probabilities are generated by the recurrence relation:

$$p_0 = e^{-\mu_1}$$

$$p_r = \frac{p\mu_1}{r}\sum_{j=0}^{r-1}(r-j)(1-p)^{r-j-1}p_j \qquad r>0$$

As p tends to 1 the distribution becomes Poisson. As μ_1 tends to 0 the distribution becomes geometric with added zeroes. The distribution is fitted by estimating the mean (μ_1/p) and variance $(\mu_1(2-p)/p^2)$, from which estimates of the defining parameters μ_1 and p are obtained.

The *Poisson-Pascal* distribution is a more general three-parameter contagious distribution in which the number of groups per unit interval has a Poisson distribution (with mean λ) and the number of events per group has a negative binomial (or Pascal) distribution. The distribution is defined by the parameters k, p (with $q=1+p$), and λ in the following recurrence relations:

$$p_0 = \exp(-\lambda(1-q^{-k}))$$

$$p_r = \frac{\lambda pk}{r\,q^{k+1}}\sum_{j=1}^{\infty}\left(\frac{p}{q}\right)^{j-1}\binom{k+j-1}{k}p_{r-j} \qquad r>0$$

The distribution is fitted by estimating the mean, the Poisson index and the negative binomial index: the defining parameters can then be derived. This distribution contains several others as special cases:

	k	Negative binomial index
Neyman type A	∞	1
Pólya-Aeppli	1	1.5
Negative binomial	0	2

The *Poisson-log-Normal* distribution is an aggregated distribution which is more skew than the negative binomial. It is generated as a mixture of Poisson distributions whose means are log-Normally distributed with mean μ_1 and variance σ^2. Then the probabilities are obtained as follows:

$$p_r = \frac{1}{r!\sigma\sqrt{2\pi}}\int_0^{\infty} e^{-z}z^{r-1}\exp(-(\log(z)-\mu_1)^2/2\sigma^2)\,dz$$

The mean and variance of the distribution are fitted, from which the defining parameters μ and σ^2 are obtained. The probabilities are computed by numerical integration when r is small and by an approximation formula when r is large.

Other discrete distributions could be fitted to data using the facilities for fitting nonlinear models: see 3.8 for more details. We now go on to look at the continuous distributions in more detail. For these the density function $f(x)=F'(x)$ is used instead of point probabilities.

Several of the continuous distribution functions available are based around the *Normal* distribution, which has density function:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} = \Phi(x;\mu;\sigma)$$

The parameters to be estimated are μ and σ. The sample skewness is 0 with variance $6/n$ and the sample kurtosis is 0 with sampling variance $24/n$.

The *double Normal* distribution can be used when an observation may come from either of two Normal populations with different means. If a proportion p of the population is Normally distributed with mean μ_1 and variance σ_1^2 and a proportion $(1-p)$ is Normally distributed with mean μ_2 and variance σ_2^2 the density function is:

$$f(x) = p\Phi(x;\mu_1;\sigma_1) + (1-p)\Phi(x;\mu_2;\sigma_2)$$

with mean $p\mu_1+(1-p)\mu_2$ and variance $p\sigma_1^2+(1-p)\sigma_2^2+p(1-p)(\mu_1-\mu_2)^2$. There may be one mode or two, depending on the separation of μ_1 and μ_2. There are two cases of the Double Normal that can be fitted. The variances can be constrained to be equal, by setting `DISTRIBUTION=dNvequal`, so that four parameters (p, μ_1, μ_2 and σ) are fitted. As p tends to 0 or 1 the limiting case of a single Normal is reached, and as μ_1 tends to μ_2, p becomes indeterminate. The more general five-parameter model (p, μ_1, μ_2, σ_1 and σ_2) can be fitted by setting `DISTRIBUTION=dNvunequal`. Unless there is good separation between the two underlying distributions, local maxima may cause problems during the fitting process.

The *log-Normal* distribution assumes that $\log(X)$ (the natural logarithm) is Normally distributed with mean μ and variance σ^2. An additional location parameter a can be included in the model so that the Normal distribution is fitted to $\log(X-a)$, by setting `CONSTANT=estimate`. By default, the constant is omitted and the two-parameter model is fitted (that is, with a=0). The density function is

$$f(x) = \frac{1}{x-a}\Phi(\log(x-a);\mu;\sigma) \qquad x>a$$

with mean $a+\exp(\mu+\sigma^2/2)$ and variance $\exp(2\mu+\sigma^2)(\exp(\sigma^2)-1)$. The distribution must have positive skewness; if the sample skewness is negative an automatic switch is made to the Normal distribution, which is the limit as a tends to minus infinity.

The *exponential* (or *negative exponential*) distribution can be used to model *lifetime* distributions, for example the time to failure of a process or death of an organism, where the failure rate can be assumed constant. The density function is

$$f(x) = b\,e^{-bx} \qquad x>0,\ b>0$$

where b is the failure rate per unit time. The mean is $1/b$, the variance is $1/b^2$, and the median is $\log(2)/b$.

The *Weibull* distribution is a generalization of the exponential distribution in which the failure rate can vary monotonically with time. It can be derived using a power transformation, so that X^c is assumed to have an exponential distribution. The density function is given by

$$f(x) = cb^c x^{c-1}\exp(-(bx)^c) \qquad x>0,\ b,c>0$$

which has mean $(1/b)\Gamma((c+1)/c)$ and median $(1/b)(\log 2)^{1/c}$. For $0<c<1$ the failure rate decreases with time and has a single mode at 0. If $c>1$ the failure rate increases with time and the mode is at $(1/b)(1-c^{-1})^c$. The skewness decreases as c increases, until c=3.6 when the skewness is 0, then becomes negative. The Weibull distribution is fitted by holding the median fixed to the sample estimate, whilst obtaining an initial estimate of c; the full model is then fitted. If the option `CONSTANT=estimate` is set, an additional location parameter is estimated, so that the Weibull is fitted to $(X-a)$. By default, `CONSTANT=omit`.

The *gamma* distribution is useful as a general empirical distribution. It is similar in form to the Weibull, and is closely related to other standard distributions. By default, it is fitted with two

parameters. An additional location parameter can be fitted by setting CONSTANT to estimate, and you must do this if X can take negative values. The density function for the two-parameter model is

$$f(x) = b^k x^{k-1} \frac{e^{-bx}}{\Gamma(k)} \qquad \text{where} \quad \Gamma(k) = \int_0^\infty x^{k-1} e^{-x}\, dx\,, \quad x>0.$$

The parameter k is known as the *shape* parameter, and is sometimes represented by the Greek letter kappa (κ). The parameter b is known as the *rate* parameter, and is sometimes represented by the Greek letter beta (β). The mean of the distribution μ is k/b and the variance V is kb^{-2}. Note: the parameterization here differs from that used in the gamma probability functions (1:4.2.9). Instead of the rate, these use the *scale* parameter t (or theta), which is the reciprocal of the rate ($t=1/b$). If the shape parameter $k=1$, the gamma distribution becomes an exponential distribution. If the rate parameter $b=½$, it is a χ^2 distribution with $2k$ degrees of freedom. If $b=1$, the gamma distribution tends to a standard Normal distribution as k tends to infinity. The distribution is fitted using the sample median, approximately $(k+1)/b$, to provide initial estimates for the parameters before the full model is fitted.

The *beta* distribution is suitable for fitting proportions and ratios. Two forms are available in Genstat, denoted *type I* and *type II*. The type I distribution is a two-parameter model restricted to values in the range $0<x<1$ and is thus used to fit proportions. The density function is

$$f(x) = \frac{1}{\mathrm{B}(p,q)} x^{p-1} (1-x)^{q-1} \qquad 0 < x < 1, \quad p,q > 0$$

$$\text{where} \quad \mathrm{B}(p,q) = \int_0^1 x^{p-1} x^{q-1}\, dx, \qquad \mathrm{B}(p,q) = \frac{\Gamma(p)\,\Gamma(q)}{\Gamma(p+q)}.$$

This distribution has mean $\mu=p/(p+q)$ and variance $pq/\{(p+q)^2(p+q+1)\}$. If $p>1$ and $q>1$ then there is a single mode at $x=(p-1)/(p+q-2)$; whilst if $p<1$ and $q<1$ there is a minimum at this point. For large values of p and q the distribution is approximately Normal. The parameters p and q are often represented in the literature as α and β. In the probability functions, PRBETA etc, they are represented as a and b (see 1:4.2.9). Parameters of the beta-binomial distribution can be estimated by the BBINOMIAL procedure.

The type II beta distribution is suitable for any positive continuous data, and has density

$$f(x) = \frac{b^p x^{p-1}}{(1+bx)^{p+q}\, \mathrm{B}(p,q)} \qquad x > 0$$

now with three parameters b, p and q. The distribution has mean $\mu=p/\{b(q-1)\}$ and variance $V=p(p+q-1)/\{b^2(q-1)^2(q-2)\}$. The mode is at 0 for $p<1$ and at $(p-1)/(q+1)$ otherwise. For large values of q the distribution tends to a gamma distribution with index p. For $p=m/2$, $q=n/2$ and $b=m/n$ we have the F distribution with m and n degrees of freedom.

For either form of the beta distribution it is possible to include an additional location parameter, so that the distribution is fitted to $(X-a)$. This is specified by setting the CONSTANT option to estimate. By default, CONSTANT=omit, so no location parameter is fitted.

The *Pareto* distribution originates in economics where it is used for modelling the distribution of incomes in a population. Like the log-Normal it is suitable for data with very long upper tails; it provides a better fit to the tail but performs less well over the whole range. The "Pareto distribution of the first kind" is defined by its distribution function, only for positive data greater than a minimum value c:

$$F(x) = 1 - \left(\frac{c}{x}\right)^b \qquad x \geq c, \quad b,c > 0.$$

An additional location parameter can be requested, by setting `CONSTANT=estimate`. This fits a Pareto distribution "of the second kind", which has the distribution function

$$F(x) = 1 - \left(\frac{c-a}{x-a}\right)^b \qquad x \geq c > a, \quad b,c > 0.$$

The mean is $\mu = a + (c-a)b/(b-1)$ if $b>1$, and the variance is $b(c-a)^2(b-1)^{-2}(b-2)^{-1}$ if $b>2$.

2.2.11 Tests for Normality

Genstat provides several tests for assessing whether a sample of data comes from a Normal distribution. The Shapiro-Wilk test can be obtained using the `WSTATISTIC` procedure. Alternatively the `NORMTEST` procedure uses the Anderson-Darling statistic, the Cramer-von Mises statistic and the Watson statistic to assess either the Normality of a single measurement, or the multivariate Normality of several measurements.

`WSTATISTIC` procedure

Calculates the Shapiro-Wilk test for Normality (R.W. Payne).

Option

`PRINT` = *string tokens*	What to print (`test`); default `test`

Parameters

`DATA` = *variates*	Samples of data to be tested for Normality
`W` = *scalars*	Saves the Shapiro-Wilk W statistic for each sample
`PROBABILITY` = *scalars*	Saves the probability for W under the assumption that the data are Normal

---

The data values for `WSTATISTIC` must be supplied, in a variate, using the `DATA` parameter. By default `WSTATISTIC` prints the statistic, W, with its probability value under the assumption that the data are Normal. (So a low probability indicates that the data are unlikely to be from a Normal distribution.) The printed output can be supressed by setting option `PRINT=*`. The test statistic can be saved, in a scalar, using the `W` parameter, and its probability can similarly be saved using the `PROBABILITY` parameter.

Example 2.2.11 uses `WSTATISTIC` to assess the Normality of the variate `Glucose`.

Example 2.2.11

```
  2  " Data from Royston (1995), A remark on Algorithm AS 181:
 -3    the W-test for Normality. Applied Statistics, 44, 547-551. "
  4  VARIATE    [VALUES=4.2,4.9,5.2,5.3,6.7,6.7,7.2,7.5,8.1,8.6,\
  5             8.8,9.3,9.5,10.3,10.8,11.1,12.2,12.5,13.3,15.1,\
  6             15.3,16.1,19.0,19.5] Glucose
  7  WSTATISTIC Glucose

Shapiro-Wilk test for Normality
-------------------------------

Test statistic W: 0.9453
Probability:      0.213
```

NORMTEST procedure

Performs tests of univariate and/or multivariate Normality (M.S. Ridout).

Option

PRINT = *string tokens*	Allows the required printed output to be selected: test statistics, tables of critical values and the flagging of significant values with stars (marginal, bivariateangle, radius, critical, stars); default marg, biva, radi

Parameter

DATA = *variates* or *pointers*	Variates whose univariate Normality is to be tested or pointers, each to a set of variates whose Normality and/or multivariate Normality are to be tested

NORMTEST provides three types of test of Normality:

1 Marginal (univariate) tests – assess the Normality of each variate in turn. The variates are standardized to have mean=0, variance=1 and then transformed with the NORMAL function. The test is based on the idea that, assuming Normality, these transformed values should look like a sample from a uniform distribution on (0,1).
2 Bivariate angle tests – assess the bivariate Normality of each pair of variates in turn. The variates are standardized so that they are uncorrelated and have mean=0 and variance=1. The test is based on the following idea: if x and y are the standardized values, then the angle between the x-axis and the line joining (0,0) to (x,y) should, assuming Normality, be uniformly distributed on $(0,2\pi)$.
3 Radius test – provides a single overall test of multivariate Normality. The variates are again standardized to have mean=0 and so that their covariance matrix is the identity matrix. The test uses the fact that if $z_1, z_2, ..., z_n$ are the standardized values then $z_1^2 + z_2^2 + ... + z_n^2$ should, under multivariate Normality, be approximately distributed as chi-square on n degrees of freedom.

The calculations are as described in Aitchison (1986; Section 7.3). Bivariate angle and radius tests are described by Andrews, Gnanadesikan & Warner (1973). Stephens (1974) describes the EDF statistics used and gives tables of critical values and information on their comparative power.

For each type of test, the test statistics are empirical distribution function (EDF) statistics – i.e. they compare the empirical distribution function of the sample with the theoretical distribution expected under the null hypothesis. Three EDF statistics are provided for each type of test – the Anderson-Darling statistic, the Cramer-von Mises statistic and the Watson statistic. The idea is to provide good power against a wide range of alternatives. The test statistics are adjusted so that their null distribution is independent of the sample size; critical values can be printed by the procedure (option PRINT=critical).

The DATA parameter is used to indicate the variate(s) whose Normality is to be assessed. If a single variate is supplied, its Normality is tested using the marginal test. Alternatively, DATA can supply a pointer to a set of variates to be tested for multivariate Normality.

The PRINT option can be used to select the type of test using the settings marginal, bivariateangle and radius. The setting critical allows tables of critical values to be printed, and stars requests that significant values of the test statistics be flagged with stars. Settings bivariateangle and radius are relevant only when testing for multivariate Normality. The default settings are marginal, bivariateangle and radius.

2.2.12 Goodness of fit tests for other continuous distributions

EDFTEST procedure

Performs empirical-distribution-function goodness-of-fit tests (V.M. Cave).

Options

PRINT = *string tokens*	Controls printed output (summary, tests); default summ, test
PLOT = *string tokens*	What graphs to plot (kerneldensity, histogram); default *
TEST = *string tokens*	Specifies the type of goodness-of-fit test to perform (andersondarling, cramervonmises, kolmogorovsmirnov); default ande, cram, kolm
DISTRIBUTION = *string tokens*	Continuous distribution that is hypothesized to have generated the DATA; (beta, b2, burr, cauchy, chisquare, ev1 (or gumbel), ev2 (or frechet), ev3, expnormal, exponential, fdistribution, gamma, gev, gpareto, iburr, igamma, invnormal, iweibull, laplace, loggamma, logistic, loglogistic, lognormal, normal, paralogistic, pareto, skewnormal, stdnormal, stduniform, tdistribution, ubetamix, ugammamix, uniform, weibull, calculated); default norm
CONSTANT = *string tokens*	Whether to estimate a constant for the distribution, when the parameter values are estimated from the DATA (estimate, omit); default omit
TMETHOD = *string tokens*	Specifies the method used to perform the goodness-of-fit tests (likelihoodratio, traditional); default like
PARAMETERS = *scalar* or *variate*	Parameter values for the hypothesized distribution; if this is not set, parameter values are estimated from the DATA
NAMES = *text*	Names to identify the parameters in PARAMETERS; if this is not set, the default parameter ordering is assumed
CDFCALCULATION = *expression*	Expression, formed using argument X, that defines the cumulative distribution function of the hypothesized distribution; must be specified when DISTRIBUTION = calculated
MCPARAMETERS = *string tokens*	Whether the parameters are re-estimated or fixed during the Monte-Carlo simulations, when the parameter values are estimated from the DATA (fix, estimate); default esti
NTIMES = *scalar*	Number of Monte-Carlo simulations to perform; default 999
SEED = *scalar*	Seed for random number generation; default 0 continues an existing sequence or, if none, selects a seed automatically
TITLE = *text*	Title for the graphs; default generates the title automatically
YTITLE = *text*	Y-axis title for the graphs; default generates the title automatically

XTITLE = *text*	X-axis title for the graphs; default generates the title automatically
WINDOW = *scalar*	Window to use for the graphs; default 3
SCREEN = *string tokens*	Whether to clear the screen before plotting the graph or to continue plotting on the old screen, when a single graph is requested (clear, keep); default clear

Parameters

DATA = *variate*	Identifier of the variate holding the data
STATISTIC = *pointer*	Pointer to scalar(s) to save the test statistic(s)
MCSTATISTICS = *pointer*	Pointer to variates(s) to save the Monte-Carlo simulated test statistic(s)
PROBABILITY = *pointer*	Pointer to scalar(s) to save the probability value(s) of the test statistic(s)

EDFTEST performs one-sample two-sided empirical-distribution-function goodness-of-fit tests to assess whether a sample of data comes from a specified continuous distribution. The data values must be supplied, in a variate, using the DATA parameter. This can be restricted to assess only a subset of the data.

The distribution from which the data are assumed to arise is specified using the DISTRIBUTION option; default normal. Values for the parameters can be supplied, in either a scalar or a variate, by the PARAMETERS option. If parameter values are not supplied, they are estimated from the DATA, using the methods in the DPROBABILITY procedure (2.2.7), except when DISTRIBUTION is set to stdnormal, stduniform or calculated.

The NAMES option specifies a text to identify the individual parameter values within a variate of PARAMETERS. The parameter names associated with each distribution are given below. When the names are not supplied, the default ordering of the parameters is assumed. (This matches the ordering in which parameter estimates are saved using the ESTIMATES parameter of the DPROBABILITY procedure,) The parameter names are listed below, in the default parameter ordering for each distribution:

Beta Type I (beta)	ashape, bshape;
Beta Type II (b2)	ashape, bshape, rate;
Burr (burr)	ashape, scale, bshape;
Cauchy (cauchy)	location, scale;
Chi-square (chisquare)	df;
Extreme Value Type I (ev1 or gumbel)	location, scale;
Extreme Value Type II (ev2 or frechet)	location, scale, shape;
Extreme Value Type III (ev3)	location, scale, shape;
Exponential (exponential)	rate;
Exponentially modified Normal (expnormal)	mean, sd, rate (default) or emean;
F (fdistribution)	ndf, ddf;
Gamma (gamma)	shape, rate, constant (optional);
Generalized Extreme Value (gev)	shape, location, scale;
Generalized Pareto (gpareto)	shape, scale;
Inverse Burr (iburr)	ashape, scale, bshape;

Inverse Gamma (`igamma`)	shape, scale;
Inverse Normal (`invnormal`)	mean, shape;
Inverse Weibull (`iweibull`)	scale, shape;
Laplace (`laplace`)	location, scale;
Log-Gamma (`loggamma`)	shape, rate;
Logistic (`logistic`)	location, scale;
Log-Logistic (`loglogistic`)	shape, scale;
Log-Normal (`lognormal`)	mean, sd, constant (optional);
Normal (`normal`)	mean, sd;
Paralogistic (`paralogistic`)	shape, scale;
Pareto (`pareto`)	shape, scale, constant (optional);
Skew Normal (`skewnormal`)	mean, sd, skewness parameter alpha;
t (`tdistribution`)	df;
Uniform-Beta mixture (`ubetamix`)	weight, ashape, bshape;
Uniform-Gamma mixture (`ugammamix`)	weight, shape, scale;
Uniform (`uniform`)	min, max;
Weibull (`weibull`)	shape, rate, constant (optional);

The Gamma, Log-Normal, Pareto and Weibull distributions can have an extra constant parameter, so that the data values minus the constant then follow the specified distribution. When `PARAMETERS` are not supplied, you can set option `CONSTANT` = `estimate` to estimate a constant from the `DATA`. The default is not to estimate a constant.

The Exponentially modified Normal can have two parameterizations, with the third parameter as either *emean* (mean of the exponential distribution) or the exponential rate (reciprocal of the mean). `DPROBABILITY` estimates and returns the exponential rate, but in some case it is easier to provide the mean. The third unit of `NAMES` indicates whether rate or *emean* has been provided; if is not set a rate parameter is assumed.

The types of test to perform are specified by the `TEST` option, with settings `andersondarling` (Anderson-Darling), `cramervonmises` (Cramér-von Mises) and `kolmogorovsmirnov` (Kolmogorov-Smirnov). The method used to perform these tests is specified by the `TMETHOD` option, with settings `likelihoodratio` for the Zhang (2002) likelihood-ratio based method, and `traditional` for the traditional approach. The default is to use the likelihood-ratio based tests, which are generally more powerful.

If `TMETHOD=traditional`, `EDFTEST` calculates the traditional Anderson-Darling, Cramér-von Mises and Kolmogorov-Smirnov goodness-of-fit tests. When `PARAMETERS` are supplied (or if `MCPARAMETERS` = `fix`), the probability of the Anderson-Darling test statistic is calculated using the fast algorithm (adinf) of Marsaglia & Marsaglia (2004), the probability of the Cramér-von Mises test statistic is calculated using the one-term linking approximation (equation 1.8) of Csörgő & Faraway (1996), and the probability of the Kolmogorov-Smirnov test statistic is calculated using the method of Carvalho (2015) for data sets with fewer than 171 values or using the Wang *et al.* (2003) approximation for larger data sets. When `PARAMETERS` are not supplied, Monte-Carlo simulation is used by default to obtain empirical probability values of the test statistics. However, empirical probability values are not available for `DISTRIBUTION` = `ubetamix` or `ugammamix`.

If `TMETHOD` = `likelihoodratio`, `EDFTEST` calculates likelihood-ratio based goodness-of-fit test statistics using the method of Zhang (2002). (Note, however, that the likelihood-ratio based

method is not available for DISTRIBUTION = ubetamix, ugammamix, or calculated.) The resulting tests are generally more powerful than their traditional analogues. Monte-Carlo simulation is used to obtain empirical probability values of the test statistics.

The DISTRIBUTION option provides the common distributions. Alternatively, for traditional tests (i.e. TMETHOD = traditional) you can set DISTRIBUTION=calculated to define your own distribution. You must then use the CDFCALCULATION option to provide an expression, formed using argument X, to calculate the cumulative distribution function. For example, the exponential distribution with rate parameter of 2 could be specified by setting options

```
DISTRIBUTION=calculated
```

and

```
CDF=!E(X=1-EXP(-2*X))].
```

Monte-Carlo simulations are used to calculate the empirical probability values of the test statistics under the likelihood-ratio based method (i.e. TMETHOD = likelihoodratio), or, by default, under the traditional method when the parameters are estimated from the DATA. The NTIMES option defines how many Monte-Carlo simulations are used; default 999. The SEED option can be set to initialize the random-number generator used during the Monte-Carlo simulations; if the procedure is called again with the same settings, you will get identical results. The default of zero continues the sequence of random numbers from a previous generation or, if this is the first use of the generator in this run of Genstat, the seed is initialized automatically.

By default, when parameters are estimated from the DATA during the Monte-Carlo simulations, the parameters are re-estimated to ensure that the correct probability values are obtained. However, this can be overridden by setting the MCPARAMETERS option to fix.

Printed output is controlled by the PRINT option, with settings:

summary	to print summary information; and
tests	to print the test statistic(s), with its probability value(s) under the assumption that the data are from the hypothesized distribution (so a low probability indicates that the data are unlikely to be from the hypothesized distribution).

The default is to print the summary and the tests.

The PLOT option controls graphical output, with settings:

histogram	to plot a histogram of the Monte-Carlo simulated test statistics; and
kerneldensity	to produce a kernel density plot of the Monte-Carlo simulated test statistics.

By default, nothing is plotted.

The TITLE, YTITLE and XTITLE options can supply an overall title, a y-axis title and a x-axis title for the graphs, respectively. If these are not supplied, suitable titles are generated automatically. When a single plot is requested, you can set option SCREEN = keep to plot the graph on an existing screen; by default the screen is cleared first. The WINDOW option defines the window to use for the plots; default 3.

The STATISTIC, PROBABILITY and MCSTATISTICS parameters allow the test statistics, their probabilities and the Monte-Carlo simulated test statistics, respectively, to be saved in pointers.

Example 2.2.12 confirms that it is reasonable to assume that distribution of the volcano heights, introduced in Section 2.1.1, can be represented by a gamma distribution. The test statistics are non-significant, and this is confirmed by the fact that the test statistics lie well within the histograms of the simulated values.

Example 2.2.12

```
26  EDFTEST [PLOT=histogram; DISTRIBUTION=gamma; SEED=73197; NTIMES=999] Height
```

```
Likelihood-ratio based empirical-distribution-function goodness-of-fit tests
============================================================================

Distribution: Gamma
f(x) = b^k.x^(k-1).exp(-b.x)/Gamma(k)
shape (k) = 3.501
rate (b) = 0.04536
Parameters estimated from the observations

Variate: Height
Observations: 126
Monte-Carlo simulations: 999
Seed: 73197
Parameters re-estimated during simulations

                              Statistic  Probability
     LR-based Anderson-Darling     3.304        0.646
     LR-based Cramer-von Mises     6.453        0.617
   LR-based Kolmogorov-Smirnov     0.706        0.843
```

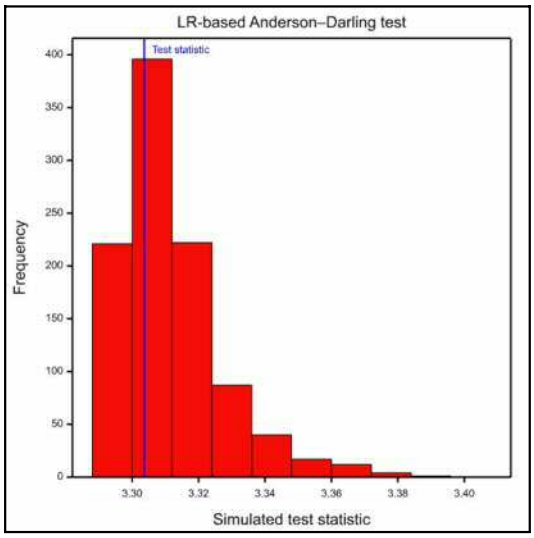


Figure 2.2.12a

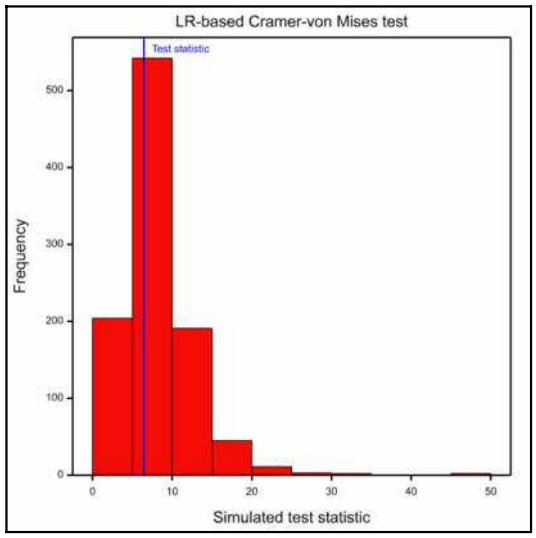


Figure 2.2.12b

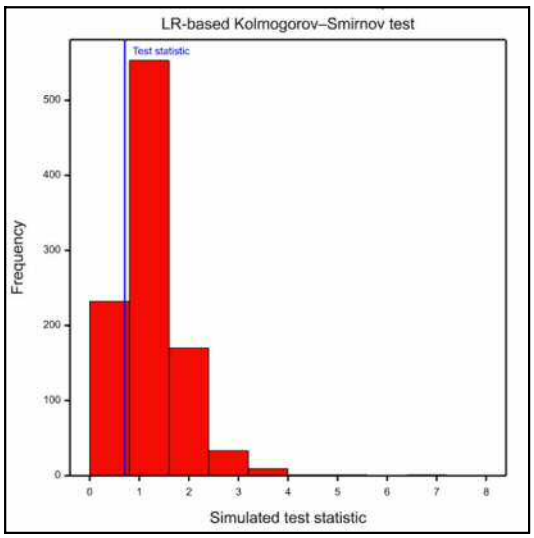


Figure 2.2.12c

2.3 Comparison of groups of data

The aim of many statistical studies is to compare different groups of observations. These groups may differ because they have been selected from separate populations, or perhaps because they have received different experimental treatments. It is often possible to fit statistical models containing different parameters for each group, which enable you to answer questions like "How much better is treatment A than treatment B?". These methods of estimation generally also provide extra information such as standard errors, sums of squares, or perhaps deviances to allow you to check statistically whether the treatments genuinely differ in their effects. Many of the estimation procedures in Genstat provide formal probability levels associated with these *hypothesis tests*.

In this section we start with the simplest statistical test, the t-test (2.3.1). This allows you to compare the means of two samples or, if you have only one sample, to assess whether its mean differs from some specified value (usually zero). The test assumes that the samples have Normal distributions; sometimes you may need to transform the data for this assumption to be reasonable (see 1:4.2). The simplest generalization of the t-test is one-way analysis of variance, which is described in Section 2.3.2. More sophisticated types of analysis of variance, for example factorial treatment structures and multiple sources of error, are covered in Chapters 4 and 5. Methods for determining sample sizes for analysis of variance are described in Section 4.12.2.

It is not always possible to make sensible assumptions about the models or the probability distributions from which the observations have been generated. So as an alternative `TTEST` can use a permutation test, or an exact test if there are few data values. Genstat also contains a range of procedures for performing nonparametric or distribution-free tests that require only relatively simple assumptions. These are described in Sections 2.4, 2.5 and 2.6.

2.3.1 The t-test

`TTEST` procedure

Performs a one- or two-sample t-test (S.J. Welham).

Options

`PRINT` = *string tokens*	Controls printed output (`confidence`, `summary`, `test`, `variance`, `permutationtest`); default `conf`, `summ`, `test`, `vari`
`METHOD` = *string token*	Type of test required (`twosided`, `greaterthan`, `lessthan`, `equivalence`, `noninferiority`, `nonsuperiority`); default `twos`
`GROUPS` = *factor*	Defines the groups for a two-sample test if only the `Y1` parameter is specified
`CIPROBABILITY` = *scalar*	The probability level for the confidence interval; for a one-sided test this will be for the mean and for a two-sided test for the difference in means; default `*`, i.e. no confidence interval is produced
`NULL` = *scalar*	The value of the mean under the null hypothesis; default 0
`VMETHOD` = *string token*	Selects between the standard two-sample t-test, with a pooled estimate of the variances of the samples, and the use of separate estimates for the sample variances (`automatic`, `pooled`, `separate`); default `auto` uses a pooled estimate unless there is evidence of unequal variances

PLOT = *string token*	How to plot the statistics from a permutation test (histogram); default * i.e. no plots
NTIMES = *scalar*	Number of random allocations to make when PRINT=perm; default 999
PERMMETHOD = *string token*	Which statistic to use in a permutation test (difference, t); default t
SEED = *scalar*	Seed for the random number generator used to make the allocations; default 0 continues from the previous generation or (if none) initializes the seed automatically
NTIMES = *scalar*	Number of random allocations to make when PRINT=perm; default 999
SEED = *scalar*	Seed for the random number generator used to make the allocations; default 0 continues from the previous generation or (if none) initializes the seed automatically
EQLIMITS = *scalar* or *variate*	Limits for equivalence, non-inferiority or non-superiority

Parameters

Y1 = *variates*	Identifier of the variate holding the first sample
Y2 = *variates*	Identifier of the variate holding the second sample
TESTRESULTS = *variates*	Identifier of variate (length 3) to save test statistic, d.f. and probability value
LOWER = *scalars*	Identifier of scalar to save the lower limit of each confidence interval
UPPER = *scalars*	Identifier of scalar to save the upper limit of each confidence interval
W1 = *variates*	Weights (replications) of the values in Y1; default * i.e. all 1
W2 = *variates*	Weights (replications) of the values in Y2; default * i.e. all 1
SAVEPERMUTATIONS = *variates*	Saves the permutation statistics

The data for TTEST are specified by the parameters Y1 and Y2 and the option GROUPS. For a one-sample test, the Y1 parameter should be set to a variate containing the data. TTEST then performs a one-sample t-test for the mean of a Normal distribution. The value of the mean under the null hypothesis can be specified by the option NULL; by default NULL=0. By default, TTEST does a two-sided test, but the METHOD option (described below) can select one-sided tests or equivalence tests instead. Example 2.3.1a tests whether the mean of a set of diffusion data differs from 20.

Example 2.3.1a

```
   2  " Rates of diffusion of carbon dioxide through two soils.
  -3    Data from Smith & Brown (1933); also analysed by Snedecor &
  -4    Cochran (1989) p.94. (who give wrong reference for the data."
   5  VARIATE [VALUES=20,31,18,23,23,28,23,26,27,26,12,17,25] Fine
   6  TTEST [NULL=20] Fine

One-sample t-test
=================

Variate: Fine.

Summary
-------
```

```
                                                  Standard   Standard error
Sample           Size        Mean    Variance   deviation           of mean
Fine               13       23.00       26.50       5.148             1.428

95% confidence interval for mean: (19.89, 26.11)


Test of null hypothesis that mean of Fine is equal to 20.00
-----------------------------------------------------------

Test statistic t = 2.10 on 12 d.f.

Probability = 0.057
```

For Normally distributed observations, the statistic t is distributed as Student's t distribution on $n-1$ degrees of freedom. (It is possible to use the DPROBABILITY procedure or DISTRIBUTION directive, described in 2.2.7 and 2.2.10, to assess the distribution of a sample; however, with small samples like this one, there is rarely enough evidence to determine the distribution clearly.) The probability level quoted is the theoretical probability of getting a result as extreme as the value calculated, given that the null hypothesis is true (that is, that the sample mean equals the target value). For this example, the probability of getting a value as large as 2.10 is 0.057 under the null hypothesis. Since this probability is small, there is evidence that the sample mean is different from 20, but (since the probability is greater than 0.05) there is not enough evidence to reject the null hypothesis at the 5% level. Further details on hypothesis testing can be found in any book covering basic statistical methods, such as Snedecor & Cochran (1989).

The data for a two-sample test can be specified in two separate variates using the parameters Y1 and Y2. Alternatively, they can be given in a single variate, with the GROUPS option set to a factor to identify the two samples; the GROUPS option is ignored when the Y2 parameter is set. The assumption here is that the individual measurements have been made independently. The test is not appropriate for measurements taken in a series where it is likely that neighbouring measurements are more correlated than measurements further apart in the series; in this situation you could use the procedures in the repeatedmeasures module of the procedure library (8.1), or the facilities for modelling correlations by REML in Section 5.4, or the time-series methods in Chapter 7. Likewise, if the two samples have been taken in a paired way, so that each measurement in one sample is matched with a measurement in the other, the test procedure must reflect this structure, for example by treating the pairs of observations as blocks in an analysis of variance or by subtracting one set of values from the other and then doing a one-sample test. This structure often arises when several samples are taken from a single set of individuals.

Printed output is controlled by the PRINT option with settings:

summary	number of observations, mean, variance, standard deviation and standard error of mean;
test	t-statistic and probability level;
confidence	confidence interval for the difference between mean and NULL for a one-sample test, or the two means for a two-sample test;
variance	F test for equality of the sample variances in a two-sample test; and
permutationtest	probabilities calculated by a random permutation test (relevant only for two-sample tests).

The default is PRINT=summary,test,confidence,variance. By default a 95% confidence interval is calculated, but this can be changed by setting the CIPROBABILITY option to the required value (between 0 and 1) or leaving it unset to suppress the interval. For equivalence tests, the confidence interval is an amalgamation of two one-sided intervals, as you are making two one-sided tests. Each limit is therefore calculated for twice the distance from 100% (e.g.

90% instead of 95%, corresponding to a significance level of 5% for the test of equivalence).

By default, for the permutation test, TTEST makes 999 random allocations of the data to the two samples (using a default seed), and determines the probability from the distribution of the t-statistic over these randomly generated data sets. Alternatively, you can set option PERMMETHOD=difference to use the difference between the means instead of the t-statistic. The NTIMES option allows you to request another number of allocations, and the SEED option allows you to specify another seed. TTEST checks whether NTIMES is greater than the number of possible ways in which the data values can be allocated. If so, it does an exact test instead, which takes each possible allocation once. For a visual indication, you can set option PLOT=histogram to display a histogram of the statistics from the permuted data sets, with a vertical line to show the position of the statistic from the original data set.

Example 2.3.1b compares measurements of diffusion of carbon dioxide through two soils of different porosity (the first set was used in Example 2.3.1a, the second is defined in line 7). There is no pairing of the measurements – indeed, there are different numbers of measurements in each sample – and we assume that the measurements are independent. The test shows that there is a probability of 10.9% of obtaining a result this extreme under the null hypothesis of no difference between sample means. So there is some, but not strong, evidence that the mean of the second sample is higher; it is conventional, however, to reject the hypothesis of no difference only at the 5% level. Equivalently, the 95% confidence interval for the difference between the two means includes zero – showing that a zero difference is not inconsistent with the data at this significance level.

Example 2.3.1b

```
  7  VARIATE [VALUES=19,30,32,28,15,26,35,18,25,27,35,34] Coarse
  8  TTEST [PRINT=confidence,summary,test,variance,permutation] Fine; Coarse

Two-sample t-test
=================

Variates: Fine, Coarse.


Test for equality of sample variances
-------------------------------------

Test statistic F = 1.74 on 11 and 12 d.f.

Probability (under null hypothesis of equal variances) = 0.36


Summary
-------

                                                Standard   Standard error
Sample           Size        Mean    Variance   deviation          of mean
Fine               13       23.00       26.50       5.148            1.428
Coarse             12       27.00       46.00       6.782            1.958

Difference of means:           -4.000
Standard error of difference:  2.396

95% confidence interval for difference in means: (-8.957, 0.9567)


Test of null hypothesis that mean of Fine is equal to mean of Coarse
--------------------------------------------------------------------

Test statistic t = -1.67 on 23 d.f.

Probability = 0.109

* MESSAGE: Default seed for random number generator used with value 247685
```

```
Probability determined from 999 random permutations = 0.114
```

The standard two-sample t-test assumes that the two samples arise from Normal distributions with equal variances and forms a pooled estimate for the variance of both samples. If, however, the variances are unequal, a separate estimate can be used for the variance of each sample. This is known as Welch's t-test or Welch's analysis of variance (Welch 1947). The degrees of freedom of the test are then only approximate (see, for example, Snedecor & Cochran 1989, page 97) but these seem to work well in practice. The VMETHOD option specifies how to estimate the variances for the test. The default setting, automatic, uses a pooled estimate unless there is evidence of unequal variances, pooled always uses a pooled estimate and separate always uses separate estimates. If either pooled or automatic are selected, TTEST will print a warning if there is evidence of inequality of variances. Alternatively, if you do not want to assume that the data come from Normal distributions, you can use the permutation test (obtained from the permutationtest setting of the PRINT option). In Example 2.3.1b, this confirms that there is no evidence that the means of the samples differ.

The W1 and W2 parameters can supply variates of weights to accompany Y1 or Y2, respectively. You can use these to specify replicate observations. For example, instead of specifying variate for Y1 with values (11, 12, 12, 13, 14, 14, 14, 15) you could give Y1 the values (11, 12, 13, 14, 15) together with weight variate W1 containing values (1, 2, 1, 3, 1) indicating the number of replications of each of the values in Y1. The calculation of the t-test assumes that the weights are positive integers defining the
replications of the values inside Y1 or Y2 (or zero or missing values to exclude the corresponding values in Y1 or Y2). A warning is given if any positive weight is given that is not an integer.

By default, TTEST does a two-sided test, as shown in the examples above, but other types of test can be selected by the METHOD option . This has the following settings:

twosided	does a two-sided test (default);
greaterthan	does a one-sided test of the null hypothesis that mean(Y1) is not greater than mean(Y2) or NULL (for a two-sample or one-sample test, respectively);
lessthan	does a test of the null hypothesis that mean(Y1) is not less than mean(Y2) or NULL;
equivalence	does an equivalence test;
noninferiority	does a non-inferiority test;
nonsuperiority	does a non-superiority test.

A small "p-value" indicates that the data is inconsistent with the null hypothesis. If any sample has fewer than six values, a warning is given that the sample size is too small and the test may not be valid.

For a two-sample equivalence test, the null hypothesis is that the difference between the mean of the first sample and the mean of the second sample lies outside two limits specified, in a variate, by the EQLIMITS option. For a one-sample test, the difference is the mean of the sample minus NULL. TTEST does two tests: first to test whether the difference is outside the lower limit (specified by the first element of the variate), then to test whether it is outside the upper limit. The p-value is the larger of the values from the two tests.

For a two-sample non-inferiority test, the null hypothesis is that the mean of the first sample minus the mean of the second sample is less than the negative value specified, in a scalar, by the EQLIMITS option. For a one-sample test, the null hypothesis is that the mean of the sample minus NULL is less than that value.

For a two-sample non-superiority test, the null hypothesis is that the mean of the first sample minus the mean of the second sample is greater than the positive value specified, in a scalar, by the EQLIMITS option. For a one-sample test, it is that the mean of the sample minus NULL is greater than that value.

Results can be saved using the TESTRESULTS, LOWER and UPPER parameters. TESTRESULTS saves the t-statistic, its degrees of freedom and probability level in a variate of length 3. LOWER and UPPER save the lower and upper limits of the confidence interval. The SAVEPERMUTATIONS parameter can save the values of the statistics from the permutation tests in a variate; the final value in the variate is the statistic from the original data set.

Nonparametric alternatives to the t-test are described in Sections 2.4.1, 2.4.2 and 2.5.1. Methods for determining sample sizes for t-tests are described in Section 4.12.1.

2.3.2 One-way analysis of variance

One-way analysis of variance can be regarded as a simple extension of the two-sample t-test in which several samples of data are compared. Section 4.1 explains how to use Genstat's general analysis-of-variance commands to do a one-way analysis. It is well worth learning these so that you can exploit the wider facilities that they offer. However, if you need no more than a one-way analysis, there is also a special-purpose procedure AONEWAY which is specially customized for this particular type of analysis. (Note: in Genstat *for Windows* one-way analysis of variance can be obtained with the One- and two-way Analysis of Variance menu, or with the general Analysis of Variance menu by selecting One-way ANOVA (no Blocking) in the Design list box.)

AONEWAY procedure

Performs one-way analysis of variance (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output from the analysis of variance (aovtable, information, covariates, effects, residuals, contrasts, means, cbeffects, cbmeans, stratumvariances, %cv, missingvalues, homogeneity, permutationtest); default aovt, mean, miss
GROUPS = *factor*	Defines the treatments for the analysis
COVARIATES = *variates*	Covariates (if any) for analysis of covariance
PLOT = *string tokens*	Which residual plots to provide (fittedvalues, normal, halfnormal, histogram, absresidual); default fitt, norm, half, hist
GRAPHICS = *string token*	Type of graphs (lineprinter, highresolution); default high
FPROBABILITY = *string token*	Probabilities for variance ratio (yes, no); default no
PSE = *string tokens*	Types of standard errors to be printed with the means (differences, lsd, means); default diff
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
NTIMES = *scalar*	Number of random allocations to make when PRINT=perm; default 999
SEED = *scalar*	Seed for the random number generator used to make the allocations; default 0 continues from the previous generation or (if none) initializes the seed automatically

Parameters

Y = *variates*	Each of these contains the data values for an analysis
RESIDUALS = *variates*	Saves the residuals from each analysis

FITTEDVALUES = *variates*	Saves the fitted values from each analysis

The Y parameter supplies a variate containing the data values to be analysed. The factor defining the groups to be compared is supplied by the GROUPS option. You can either specify just the factor to produce a simple one-way anova, or you can put it within a POL, REG or COMPARISON function to fit some contrasts at the same time (see 4.5). There is also a COVARIATES option which can supply one or more variates to be used as covariates in an analysis of covariance (4.3).

Printed output is requested by listing the required components with the PRINT option. The most relevant settings are:

aovtable	to print the analysis-of-variance table;
means	to print the table of means;
effects	to print the effects (means minus grand mean);
%cv	to print the coefficient of variation;
missingvalues	to print estimates for missing values (if any);
homogeneity	to print tests for the homogeneity of the variances within the groups;
permutationtest	analysis-of-variance table with the probabilities calculated by a random permutation test.

For compatibility all the settings of the PRINT option of ANOVA are included, but the others are not particularly useful with one-way analysis of variance. Note, though, that ANOVA does not have a setting of homogeneity.

By default, when PRINT=perm, AONEWAY makes 999 random allocations of the data to the two samples (using a default seed), and determines the probabilities of the variance ratios from their distribution over these randomly generated datasets. (It therefore makes no assumptions about the distribution of the data values.) The NTIMES option allows you to request another number of allocations, and the SEED option allows you to specify another seed. AONEWAY checks whether NTIMES is greater than the number of possible ways in which the data values can be allocated. If so, it does an exact test instead, which takes each possible allocation once.

The FPROBABILITY option can be set to yes to print of probabilities for variance ratios in the analysis-of-variance table. The PSE option controls the standard errors printed with the tables of means. The default setting is differences, which gives standard errors of differences of means. The setting means produces standard errors of means, LSD produces least significant differences, and by setting PSE=* the standard errors can be suppressed altogether. The significance level to use in the calculation of the least significant differences can be changed from the default of 5% using the LSDLEVEL option.

The PLOT option allows up to four of the following residual plots to be requested:

fittedvalues	for a plot of residuals against fitted values;
normal	for a Normal plot;
halfnormal	for a half-Normal plot;
histogram	for a histogram of residuals; and
absresidual	for a plot of the absolute values of the residuals against the fitted values.

By default the first four are produced. The GRAPHICS option determines the type of graphics that is used, with settings highresolution (the default) and lineprinter.

Variates of residuals and fitted values can be saved using the RESIDUALS and FITTEDVALUES parameters, respectively. Directive AKEEP (4.6.1) can be used to save other information from the analysis of the last data variate to be analysed by AONEWAY.

The use of AONEWAY is illustrated by Example 2.3.2, which analyses some measurements on fat absorbance of doughnuts during cooking (from Snedecor & Cochran 1989). As we have set option FPROBABILITY=yes, the F pr. column is included, with the results of an F test of the null hypothesis that there are no differences between the groups (here the different types of fat).

In this example, the probability of the statistic under the null hypothesis is 0.007, indicating differences between the fat types which can be seen in the table of means: fat type 2 tends to have a higher absorbance and fat type 4 has a lower absorbance. Chapter 4 gives further details about the output (and explains how to analyse more complex designs). We have included `homogeneity` in the settings of the `PRINT` option to request Bartlett's test for homogeneity of the variances in the four groups. This statistic is small compared to a chi-square distribution on three degrees of freedom, and so there is no evidence against the assumption of equal variation across the groups.

Example 2.3.2

```
  2  "  Absorbance of four types of fat while cooking doughnuts.
 -3     Data from Lowe (1935) analysed by Snedecor & Cochran (1989) p.217."
  4  VARIATE [VALUES=64,72,68,77,56,95,  78,91,97,82,85,77, \
  5                  75,93,78,71,63,76,  55,66,49,64,70,68] Absorb
  6  FACTOR  [LEVELS=4; VALUES=6(1...4)] Fat
  7  AONEWAY [PRINT=aov,means,homogeneity,permutationtest; GROUPS=Fat;\
  8           PLOT=*; FPROBABILITY=yes; SEED=325691] Absorb

Analysis of variance
====================

Variate: Absorb

Source of variation     d.f.       s.s.       m.s.    v.r.  F pr.
Fat                        3     1636.5      545.5    5.41  0.007
Residual                  20     2018.0      100.9
Total                     23     3654.5


Analysis of variance
====================

Variate: Absorb
Probabilities determined from 999 random permutations

Source of variation  d.f.       s.s.       m.s.    v.r.   prob.
Fat                  3.00     1636.5      545.5   5.406   0.008
Residual            20.00     2018.0      100.9
Total               23.00     3654.5


Table of means
==============

 Grand mean       73.75

Fat           1           2           3           4
          72.00       85.00       76.00       62.00

Replication  6

Standard error of differences of means 5.799


Bartlett's Test for homogeneity of variances
--------------------------------------------

Chi-square 1.75 on 3 degrees of freedom: probability 0.626
```

Note that the output may differ slightly from that given by `ANOVA` to take advantage of the special features of the situation. If the treatments have unequal replication, a standard error is printed for each mean, rather than the summary for comparisons of means with minimum and maximum replication as given by `ANOVA` (4.1.3, 4.3).
Similarly, any missing values are excluded from the analysis by `AONEWAY`. In `ANOVA` they need

to be included, to ensure balance in the more general situations that it covers, and are estimated as part of the analysis (4.4).

2.3.3 Two-way analysis of variance

Often you may wish to study more than one type of treatment at a time. For example, in a medical trial you might want to consider the type of drug as well as the size of dose, or in a field trial you might want to look at the variety of a crop as well as the amount of fertiliser. Procedures A2WAY, A2DISPLAY, A2KEEP and A2RESULTSUMMARY provide customized facilities for analysing designs with two treatment factors, like these. They automatically determine the type of design and use the appropriate method: the ANOVA directive (Chapter 4) if the design is balanced, or the regression directives (Chapter 3) if it is unbalanced. So you need very little technical knowledge to use them.

The A2WAY procedure does the analysis.

A2WAY procedure

Performs analysis of variance of a balanced or unbalanced design with up to two treatment factors (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output from the analysis (aovtable, information, covariates, effects, residuals, means, %cv, missingvalues); default aovt, mean
TREATMENTS = *factors*	Defines either one or two treatment factors
BLOCKS = factor	Can specify a blocking factor e.g. for a randomized block design
COVARIATES = *variates*	Specifies any covariates
FACTORIAL = *scalar*	Can be set to 1 to fit only the main effects of the treatments factors; default 2 also fits their interaction
FPROBABILITY = *string token*	Probabilities for variance ratio (yes, no); default no
PLOT = *string tokens*	Which residual plots to provide (fittedvalues, normal, halfnormal, histogram, absresidual); default fitt, norm, half, hist
GRAPHICS = *string token*	Type of graphs (lineprinter, highresolution); default high
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
PSE = *string tokens*	Types of standard errors to be printed with the means (differences, lsd, means); default diff
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
RMETHOD = *string token*	Type of residuals to save or display (simple, standardized); default simp
MVINCLUDE = *string token*	Whether to include units with missing y-values when using ANOVA (yvariate); default * i.e. not included
EXIT = *scalar*	Saves an exit code indicating the properties of the design

Parameters

Y = *variates*	Each of these contains the data values for an analysis

RESIDUALS = *variates*	Saves the residuals from each analysis
FITTEDVALUES = *variates*	Saves the fitted values from each analysis
SAVE = *pointers*	Save structure for each analysis (to use in A2DISPLAY or A2KEEP)

The Y parameter supplies a variate containing the data values to be analysed. The treatment factor or factors are specified by the TREATMENTS option. The FACTORIAL option sets a limit in the number of factors in each treatment term. So you can set FACTORIAL=1 to fit only the main effects when there are two treatment factors; the default FACTORIAL=2 also fits their interaction. The BLOCKS option can supply a blocking factor, for example to define a randomized-block design (see 4.2.1). There is also a COVARIATES option which can supply one or more variates to be used as covariates in an analysis of covariance (4.3).

Printed output is controlled by the PRINT option, with settings:

aovtable	analysis-of-variance table (probabilities are given for the variance ratios if option FPROBABILITY=yes);
information	information about the design (non-orthogonality &c);
covariates	covariate regression coefficients);
effects	treatment parameters in the linear model;
means	table of means;
%cv	to print the coefficient of variation;
missingvalues	to print estimates for any missing values.

The PSE option controls the standard errors printed with the tables of means. The default setting is differences, which gives standard errors of differences of means. The setting means produces standard errors of means, lsd produces least significant differences, and by setting PSE=* the standard errors can be suppressed altogether. The significance level to use in the calculation of least significant differences can be changed from the default of 5% using the LSDLEVEL option.

For unbalanced designs (analysed using Genstat regression), the means are produced using the PREDICT directive (3.3.4). The first step (A) of the calculation forms the full table of predictions, classified by all the treatment and blocking factors. The second step (B) averages the full table over the factors that do not occur in the table of means. The COMBINATIONS option specifies which cells of the full table are to be formed in Step A. The default setting, estimable, fills in all the cells other than those that involve parameters that cannot be estimated. Alternatively, setting COMBINATIONS=present excludes the cells for factor combinations that do not occur in the data. The ADJUSTMENT option then defines how the averaging is done in Step B. The default setting, marginal, forms a table of marginal weights for each factor, containing the proportion of observations with each of its levels; the full table of weights is then formed from the product of the marginal tables. The setting equal weights all the combinations equally. Finally, the setting observed uses the WEIGHTS option of PREDICT to weight each factor combination according to its own individual replication in the data.

The PLOT option allows up to four of the following residual plots to be requested:

fittedvalues	for a plot of residuals against fitted values;
normal	for a Normal plot;
halfnormal	for a half-Normal plot;
histogram	for a histogram of residuals; and
absresidual	for a plot of the absolute values of the residuals against the fitted values.

By default the first four are produced. The GRAPHICS option determines the type of graphics that is used, with settings highresolution (the default) and lineprinter.

The EXIT option can save an exit code indicating how the analysis was done. For the exact

meanings of the values see the ANOVA directive. Essentially, it has the values 0 or 1 if the analysis has been done using ANOVA (0 if design orthogonal and 1 if it is balanced). Other values indicate that it has been done using the regression directives.

In A2WAY, any units with missing values in the y-variate are excluded from the analysis. This differs from the situation in ANOVA, where they need to be included to ensure balance in the more general situations that it covers. So ANOVA estimates them as part of the analysis (see 4.4). You can reproduce the analysis that you would get by using ANOVA directly, by setting option MVINCLUDE=yvariate.

The RESIDUALS parameter can save the residuals from the analysis, and the FITTEDVALUES parameter can save the fitted values. The RMETHOD option controls whether simple or standardized residuals are saved or displayed; by default RMETHOD=simple. The SAVE parameter can save a "save" structure that can be used as input to procedure A2DISPLAY to produce further output, or to procedure A2KEEP to copy output into Genstat data structures.

A2DISPLAY procedure

Provides further output following an analysis of variance by A2WAY (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output from the analysis (aovtable, information, covariates, effects, residuals, means, %cv, missingvalues); default *
FPROBABILITY = *string token*	Probabilities for variance ratio (yes, no); default no
PLOT = *string tokens*	Which residual plots to provide (fittedvalues, normal, halfnormal, histogram, absresidual); default *
GRAPHICS = *string token*	Type of graphs (lineprinter, highresolution); default high
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
PSE = *string tokens*	Types of standard errors to be printed with the means (differences, lsd, means); default diff
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
RMETHOD = *string token*	Type of residuals to display (simple, standardized); default simp

Parameter

SAVE = *pointers*	Save structure (from A2WAY) for the analysis; if omitted, output is from the most recent A2WAY analysis

---

Procedure A2DISPLAY allows you to display further output from the analysis. By default the output is from the most recent analysis performed by A2WAY. Alternatively, you can set the SAVE parameter to a save structure (saved using the SAVE parameter of A2WAY) to obtain output from an earlier analysis. The options of A2DISPLAY control what is printed, in the same way as those of A2WAY.

A2KEEP allows you to save information from the analysis.

A2KEEP procedure

Copies information from an A2WAY analysis into Genstat data structures (R.W. Payne).

Options

FACTORIAL = *scalar*	Sets a limit on the number of factors in the terms formed from the TERMS formula; default 2
RESIDUALS = *variate*	Saves the residuals
FITTEDVALUES = *variate*	Saves the fitted values
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
AOVTABLE = *pointer*	To save the analysis-of-variance table as a pointer with a variate or text for each column (source, d.f., s.s., m.s. etc)
RMETHOD = *string token*	Type of residuals to display (simple, standardized); default simp
EXIT = *scalar*	Saves an exit code indicating the properties of the design
SAVE = *pointer*	Save structure (from A2WAY) for the analysis; if omitted, output is from the most recent A2WAY analysis

Parameters

TERMS = *formula*	Specifies the treatment terms whose means &c are to be saved
MEANS = *table* or *pointer* to *tables*	Saves tables of means for the terms or pointer to tables
SEMEANS = *table* or *pointer* to *tables*	Saves approximate effective standard errors of means
SEDMEANS = *table* or *pointer* to *tables*	Saves standard errors of differences between means
LSD = *table* or *pointer* to *tables*	Saves least significant differences

By default A2KEEP saves information from the most recent analysis performed by A2WAY. Alternatively, you can set the SAVE option to a save structure (saved using the SAVE parameter of A2WAY) to save information from an earlier analysis.

You can use the parameters of A2KEEP to save means, standard errors and least significant differences for the treatment main effects and interactions. The TERMS parameter should be set to a model formula to define the main effects and interactions whose means &c you want to save. The MEANS parameter saves tables of means. The SEMEANS parameter saves their standard errors (also in a table). The SEDMEANS parameter saves standard errors for differences between the means (in a symmetric matrix), and the LSD parameter saves least significant differences (also in a symmetric matrix). The significance level for the least significant differences can be change from the default of 5% using the LSDLEVEL option. If you have a single term, you can supply a table or symmetric matrix for each of these parameters, as appropriate. However, if you have several terms, you must supply a pointer which will then be set up to contain as many tables or symmetric matrices as there are terms. The LSDLEVEL option sets the significance level (as a percentage) for the least significant differences.

The FACTORIAL option sets a limit in the number of factors in the terms generated from the

TERMS model formula. So

```
A2KEEP [FACTORIAL=1] A*B; MEANS=!p(MA,MB)
```

would save only the main effects of A and B. The option is provided for compatibility with the AKEEP directive. However, an alternative (and simpler) way of saving means only for the main effects would be to put

```
A2KEEP [FACTORIAL=1] A+B; MEANS=!p(MA,MB)
```

The default for FACTORIAL is 2.

As in A2WAY and A2DISPLAY, the COMBINATIONS and ADJUSTMENT option control how the means are formed from an unbalanced design. The RESIDUALS option can save the residuals from the analysis, and the FITTEDVALUES option can save the fitted values. The RMETHOD option controls whether simple or standardized residuals are saved; by default RMETHOD=simple. The AOVTABLE option saves the analysis-of-variance table, as a pointer with a variate or a text for each column of the table. The pointer elements are labelled with the column labels of the table, and the variates contain missing values where the table has blanks. These can be printed as blanks by setting option MISSING=' ' in the PRINT directive. The EXIT option saves the exit code, as defined by A2WAY.

A2RESULTSUMMARY procedure

Provides a summary of results from an analysis by A2WAY (R.W. Payne).

Options

PRINT = *string tokens*	What to print (description, means, significant); default desc, mean, sign
PSE = *string tokens*	Standard errors to be printed with the means (sed, sedsummary, lsd, lsdsummary, dfmeans); default sed, dfme
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
SAVE = *pointer*	Save structure from A2WAY; default uses the save structure from the most recent A2WAY analysis

No parameters

A2RESULTSUMMARY can provide a summary of the results. The output is controlled by the PRINT option, with settings:

description	prints the name of the y-variate, any covariates and the block and treatment models,
means	prints relevant tables of means, and
significant	lists the significant treatment terms.

By default, it is all printed.

The relevant tables of means are those that contain significant treatment effects. If the interaction is significant in an analysis with two treatment factors, the relevant table is just the two-way table of means. Otherwise the relevant tables consist of the one-way tables of means for any significant main effect.

The PSE option controls the information provided with the tables of means:

sed	standard errors for differences between means,
sedsummary	summary of the standard errors for differences,
dfmeans	degrees of freedom for the standard errors of differences,
lsd	least significant differences between the means, and

`lsdsummary` summary of the least significant differences.

The default is to print the standard errors of differences and their degrees of freedom.

The `LSDLEVEL` option specifies the significance level (%) to use in the calculation of least significant differences (default 5%).

The use of the very similar `ARESULTSUMMARY` procedure is shown in Example 4.1.3g. (This procedure provides a summary of the output from the `ANOVA` directive.)

Example 2.3.3 illustrates the use of `A2WAY` with an unbalanced design set up to study the effects of genetics versus environment in the development of rats. The `Mother` factor indicates the natural mother (i.e. the genetic background) of each rat, while the `Litter` factor indicates the litter in which it was brought up. It was not possible to balance these two treatment factors (e.g. by ensuring that every combination of `Mother` and `Litter` was equally replicated), so the order in which they are fitted may be important. The analysis of variance table presents both orders: the line *Litter ignoring Mother* presents the effect of fitting `Litter` first, whereas the line *Litter eliminating Mother* is the effect of fitting `Litter` after `Mother` (so it represents all the effects of `Litter` than cannot be explained by `Mother` effects). Ideally, as here, the lines will be either both significant or non-significant. If they are contradictory, the conclusion would be that there are effects in the data that could be explained by either `Litter` or `Mother` effects (or by both). However, the non-orthogonality between these factors makes it impossible to determine which one is responsible. The conclusion would then be to design a more balanced experiment! The line *Litter.Mother* represents the interaction between `Litter` and `Mother`. In the example, approximate effective standard errors are presented using the option setting `PSE=means`. These ese's are calculated to allow good approximations to the standard errors of differences (sed's) between means *i* and *j* to be obtained by the usual formula:

$$\text{sed} = \sqrt{(\text{ese}_i^2 + \text{ese}_j^2)}$$

The output below shows that here there is virtually no discrepancy between the true sed's and the values calculated from the ese's. If the approximation is poor, you should set `PSE=differences` to print the (rather larger) triangular array of sed's instead.

Example 2.3.3

```
  2  " Experiment on foster feeding of rats from Scheffe (1959),
  3    The Analysis of Variance; also see McConway, Jones & Taylor
  4    (1999), Statistical Modelling using GENSTAT, Example 7.6. "
  5  FACTOR  [NVALUES=61; LABELS=!t(A,B,I,J)] Litter,Mother
  6  VARIATE [NVALUES=61] Littwt
  7  READ    Litter,Mother,Littwt; FREPRESENTATION=labels

    Identifier   Minimum      Mean   Maximum    Values   Missing
        Littwt     36.30     53.97     69.80        61         0

    Identifier    Values   Missing    Levels
        Litter        61         0         4
        Mother        61         0         4

  8  A2WAY [PRINT=aovtable,means; PSE=means; PLOT=*;\
  9        TREATMENTS=Litter,Mother] Littwt

Analysis of variance
====================

Source                      d.f.        s.s.        m.s.    v.r.   F pr.
Litter ignoring Mother         3       60.16       20.05    0.37   0.775
Litter eliminating Mother      3       63.63       21.21    0.39   0.760
Mother ignoring Litter         3      771.61      257.20    4.74   0.006
Mother eliminating Litter      3      775.08      258.36    4.76   0.006
Litter.Mother                  9      824.07       91.56    1.69   0.120
Residual                      45     2440.82       54.24
Total                         60     4100.13       68.34
```

```
Predictions from regression model
---------------------------------

Response variate: Littwt

              Prediction
       Litter
            A       54.97
            B       53.07
            I       52.82
            J       53.50


Approximate effective standard errors
-------------------------------------

Litter
     A       1.813
     B       2.019
     I       2.009
     J       1.946


Discrepancy between sed and value calculated from ese's
-------------------------------------------------------

Maximum discrepancy              0
Maximum % discrepancy         0.00


Predictions from regression model
---------------------------------

Response variate: Littwt

              Prediction
       Mother
            A       54.79
            B       58.08
            I       53.60
            J       48.34


Approximate effective standard errors
-------------------------------------

Mother
     A       1.853
     B       2.026
     I       1.881
     J       2.023


Discrepancy between sed and value calculated from ese's
-------------------------------------------------------

Maximum discrepancy              0
Maximum % discrepancy         0.00


Predictions from regression model
---------------------------------

Response variate: Littwt

              Prediction
       Mother            A           B           I           J
       Litter
            A       63.68       52.40       54.13       48.96
            B       52.33       60.64       53.93       45.90
            I       47.10       64.37       51.60       49.43
            J       54.35       56.10       54.53       49.06
```

```
Approximate effective standard errors
-------------------------------------

Mother           A           B           I           J
Litter
     A       3.294       4.252       3.682       3.294
     B       3.682       3.294       3.682       5.208
     I       4.252       4.252       3.294       4.252
     J       3.682       4.252       4.252       3.294


Discrepancy between sed and value calculated from ese's
-------------------------------------------------------

Maximum discrepancy             0
Maximum % discrepancy        0.00
```

2.3.4 Binomial data

`BNTEST` procedure

Calculates one- and two-sample binomial tests (D.A. Murray).

Options

`PRINT` = *string tokens*	Controls printed output (`test`, `summary`, `confidence`); default `test`, `summ`, `conf`
`METHOD` = *string token*	Type of test required (`twosided`, `greaterthan`, `lessthan`); default `twos`
`TEST` = *string token*	Form of the test for one-sample test (`exact`, `normalapproximation`) or for two-sample (`normalapproximation`, `oddsratio`); default `norm`
`CIPROBABILITY` = *scalar*	The probability level for the confidence interval; default 0.95
`NULL` = *scalar*	The value of the probability of success under the null hypothesis for the one-sample test; default 0.5

Parameters

`R1` = *scalars* or *variates*	Number of successes (scalar) or results (variate) for the first sample
`N1` = *scalars*	Sample size of the first sample
`R2` = *scalars* or *variates*	Number of successes (scalar) or results (variate) for the second sample
`N2` = *scalars*	Sample size of the second sample
`STATISTIC` = *scalars*	Saves the Normal approximation from the one-sample or two-sample tests, or the odds ratio
`PROBABILITY` = *scalars*	Saves the probability value from the one-sample or two-sample tests
`LOWER` = *scalars*	Saves the lower limit of the confidence interval
`UPPER` = *scalars*	Saves the upper limit of the confidence interval

`BNTEST` calculates one- and two-sample binomial tests, and odds ratios. For a one-sample test, the number of successes r_1 can be specified using the `R1` parameter, and the sample size n_1 using the `N1` parameter (both as scalars). Alternatively you can supply the raw data, by setting `R1` to a variate containing one in the units corresponding to successful trials and zero in those for unsuccessful trials. The test is for the probability of success under a binomial distribution. The value for the probability under the null hypothesis is 0.5 by default, but you can specify other

probabilities using the NULL option. With a two-sample test, R1 and N1 similarly provide the number of successes and sample size for the first sample (r_1 and n_1), and R2 and N2 those for the second sample (r_2 and n_2).

For both one- and two-sample cases, the test is assumed to be two-sided unless otherwise requested by the METHOD option. Setting METHOD=greaterthan gives a one-sided test of the null hypothesis that $r_1/n_1 > r_2/n_2$ or NULL (for a two-sample or one-sample test, respectively). Similarly, METHOD=lessthan produces a test of the null hypothesis $r_1/n_1 < r_2/n_2$ or NULL. A small "p-value" indicates that the data are inconsistent with the null hypothesis.

The TEST option specifies the form of test to be used. The default is to use a standard Normal approximation. Alternatively, for a one-sample test you can set TEST=exact to obtain an exact test (Arimitage, Berry & Matthews 1994, page 121). For a two-sample test you can set TEST=oddsratio to obtain an odds ratio. This is estimated by

$$p_1 (1 - p_1) / p_2 (1 - p_2)$$

where p_1 and p_2 are the success probabilities in the two sets of data. The calculation of the approximate standard error of the estimated log-odds ratio and the confidence interval is described on page 36 of Collett (1991). By default a 95% confidence interval is calculated, but this can be changed by setting the CIPROBABILITY option to the required value (between 0 and 1).

Printed output is controlled by the PRINT option with settings:

summary	number of successes, sample size, proportion, standard error (for Normal approximation and odds ratio) and odds ratio (when TEST=ODDSRATIO is selected);
test	Normal approximation and probability level;
confidence	confidence interval for the difference between the probability of success and NULL for one-sample test, or the two proportions for a two-sample test; for the odds ratio the confidence interval is displayed for the true log-odds ratio and odds ratio.

The default is to print everything.

Example 2.3.4 first tests whether a sample with 65 successes out of 100 trials can reasonably be generated by a success probability of 0.5. It then tests for the equality of probabilities of success of two samples, one with 41 successes out of 257 and the other with 64 out of 244.

Example 2.3.4

```
   2  BNTEST 65; N1=100

One-sample binomial test
========================

Summary
-------

 Sample Size   Successes    Proportion
         100          65          0.65

Approx s.e. of proportion:   0.04770


Test of null hypothesis that proportion  is equal to 0.5000
------------------------------------------------------------

Normal Approximation =    2.900
Probability          =    0.004

95% confidence interval for proportion: (0.5565, 0.7435)

   3  BNTEST [TEST=exact] R1=65; N1=100
```

```
One-sample binomial test
========================

Summary
-------

 Sample Size   Successes    Proportion
         100          65          0.65


Test of null hypothesis that proportion  is equal to 0.5000
-----------------------------------------------------------

Exact probability    =    0.004

95% confidence interval for proportion: (0.5482, 0.7427)

   4  BNTEST R1=41; N1=257; R2=64; N2=244

Two-sample binomial test
========================

Summary
-------

  Sample     Size   Successes    Proportion
       1      257          41        0.1595
       2      244          64        0.2623

Approx s.e. of difference between proportions:  0.03626


Test of null hypothesis that proportion 1 is equal to proportion 2
-------------------------------------------------------------------

Normal Approximation =   -2.825
Probability          =    0.005

95% confidence interval for difference between proportions: (-0.1738, -0.03170)

   5  BNTEST [TEST=oddsratio] R1=41; N1=257; R2=64; N2=244

Odds Ratio
==========

Summary
-------

  Sample     Size   Successes    Proportion
       1      257          41          0.16
       2      244          64          0.26

Odds ratio                      =  0.534
Log of Odds ratio               = -0.628
Standard error of log(ratio)    =  0.224

95% confidence interval for odds ratio: (0.3441, 0.8282)
95% confidence interval for log odds ratio: (-1.067, -0.1885)
```

Results can be saved using the STATISTIC, PROBABILITY, LOWER and UPPER parameters. STATISTIC saves the Normal approximation for the one- and two-sample tests or the odds ratio, PROBABILITY saves the probability level. LOWER and UPPER save the lower and upper limits, respectively, of the confidence interval; for the odds ratio the confidence interval is saved for the true odds ratio.

Binomial data can also be analysed by Genstat's facilities for generalized linear models, which cover much more than the one- and two-sample situations considered here. Full details are in Section 3.5. Methods for determining sample sizes for binomial tests are described in Section 4.12.5.

2.3.5 Poisson data

PNTEST procedure

Calculates one- and two-sample Poisson tests (D.A. Murray).

Options

PRINT = *string tokens*	Controls printed output (test, summary, confidence); default test, summ, conf
METHOD = *string token*	Type of test required (twosided, greaterthan, lessthan); default twos
TEST = *string token*	Form of the test for one-sample test (exact, normalapproximation); default norm
S1 = *scalar*	Sample size for sample 1; default 1
S2 = *scalar*	Sample size for sample 2; default 1
CIPROBABILITY = *scalar*	The probability level for the confidence interval; default 0.95
NULL = *scalar*	The value of the probability of success under the null hypothesis for the one-sample test

Parameters

MU1 = *scalars* or *variates*	Numbers recorded in the first sample
MU2 = *scalars* or *variates*	Numbers recorded in the second sample
NORMAL = *scalars*	Saves the Normal approximation
PROBABILITY = *scalars*	Saves the probability value from the one-sample or two-sample tests
LOWER = *scalars*	Saves the lower limit of the confidence interval
UPPER = *scalars*	Saves the upper limit of the confidence interval

PNTEST calculates one- and two-sample Poisson tests. The value for the mean under the null hypothesis for a one-sample test is specified by the option NULL. You can supply the sample mean m_1 as a scalar using the MU1 parameter. The sample size is then specified by the S1 option (with default 1). Alternatively, you can set MU1 to a variate containing the counts in the individual samples (and the sample size is then the number of non-missing values that it contains). With a two-sample test, parameters MU1 and MU2 similarly provide the means (m_1 and m_2) for samples 1 and 2 respectively, and the sample sizes can be specified using the S1 and S2 options.

For both one- and two-sample cases, the test is assumed to be two-sided unless otherwise requested by the METHOD option. Setting METHOD=greaterthan will give a one-sided test of the null hypothesis that $m_1 > m_2$ or NULL (for a two-sample or one-sample test, respectively). Similarly, METHOD=lessthan will produce a test of the null hypothesis $m_1 < m_2$ or NULL. A small "p-value" indicates that the data are inconsistent with the null hypothesis. The TEST option specifies the form of test used for the one-sample test; either an exact test or a Normal approximation can be selected.

The TEST option specifies the form of test used for the one-sample test. The default is to use a Normal approximation can be selected, but you can set TEST=exact to obtain an exact test. The exact test and confidence intervals are based on the methodology described in Chapter 4 (page 141) of Arimitage, Berry & Matthews (1994). By default a 95% confidence interval is calculated, but this can be changed by setting the CIPROBABILITY option to the required value (between 0 and 1).

Printed output is controlled by the PRINT option with settings:

summary	mean, sample size, standard error (for Normal

approximation);

`test` Normal approximation and probability level, or just probability level for the exact test;

`confidence` confidence interval for the difference between the mean and `NULL` for a one-sample test, or the two means for a two-sample test.

The default is to print everything.

Example 2.3.5 illustrates the various tests.

Example 2.3.5

```
   2  PNTEST [NULL=20] MU1=33

One-sample Poisson test
=======================

Summary
-------

  Sample Size        Mean
            1       33.00

Approx s.e. of mean:  5.745


Test of null hypothesis that mean is equal to 20.00
---------------------------------------------------

Normal Approximation =    2.907
Probability          =    0.004

95% confidence interval for mean: (23.50, 46.34)

   3  PNTEST [NULL=20; TEST=exact] MU1=33

One-sample Poisson test
=======================

Summary
-------

  Sample Size        Mean
            1       33.00


Test of null hypothesis that mean is equal to 20.00
---------------------------------------------------

Exact probability    =    0.009

95% confidence interval for mean: (22.72, 46.34)

   4  PNTEST [TEST=exact] MU1=13; MU2=31

Two-sample Poisson test
=======================

Summary
-------

  Sample     Size        Mean
       1        1       13.00
       2        1       31.00

 Difference between means:      -18
Approx s.e. of difference:    6.633
```

```
Test of null hypothesis that mean 1 is equal to mean 2
------------------------------------------------------

Simple Normal Approximation    =   -2.714
Exact probability              =    0.007

95% confidence interval for difference: (-31.00, -4.999)
```

Results can be saved using the NORMAL, PROBABILITY, LOWER and UPPER parameters. NORMAL saves the Normal approximation for the one- and two-sample tests, PROBABILITY saves the probability level. LOWER and UPPER save the lower and upper limits, respectively, of the confidence interval.

Poisson data can also be analysed by Genstat's facilities for generalized linear models, which cover much more than the one- and two-sample situations considered here. Full details are in Section 3.5. Methods for determining sample sizes for Poisson tests are described in Section 4.12.6.

2.4 One-sample nonparametric tests

Genstat provides several one-sample nonparametric tests. The Wilcoxon test (procedure WILCOXON, Section 2.4.1) provides a nonparametric alternative to the one sample t-test. The test is based upon the ranked data values, and so depends only on their order, rather than on the actual distribution of the data. Another possibility is the sign test (procedure SIGNTEST, Section 2.4.2) which tests the location of the sample against a specified value using the *signs* (positive or negative) of the differences between the members of the sample and the specified value; there is also a two-sample version which tests for any difference in location between two matched samples. Finally, the runs test (procedure RUNTEST, Section 2.4.3) assesses the randomness of a sequence of observations. These tests are all accessible through the One-sample Nonparametric Tests menu of Genstat *for Windows*.

2.4.1 The Wilcoxon test

The Wilcoxon test is a nonparametric equivalent to the one sample t-test, and can be performed using the WILCOXON procedure.

WILCOXON procedure

Performs a Wilcoxon Matched-Pairs (Signed-Rank) test (S.J. Welham, N.M. Maclaren & H.R. Simpson).

Option

PRINT = *string tokens*	Output required (test, ranks): test gives the relevant test statistics, ranks prints out the signed ranks for the vector of differences; default test

Parameters

DATA = *variates*	Variates holding the differences between each pair of samples
RANKS = *variates*	Variate to save the signed ranks
STATISTIC = *scalars*	Scalar to save the value of the test statistic
PROBABILITY = *scalars*	Saves the probability for each test statistic
SIGN = *scalars*	Scalar to indicate the sign of the total sum of the signed ranks: 1 if the sum is positive, 0 otherwise

WILCOXON performs a Wilcoxon Matched-Pairs test on a variate holding differences between two paired samples. It does not have a NULL option like TTEST, so you need to use the CALCULATE directive first to form the differences with the target value: see line 9 of Example 2.4.1, which continues Example 2.3.1b.

Example 2.4.1

```
   9  CALCULATE Fine20 = Fine - 20
  10  WILCOXON Fine20

Wilcoxon Matched-Pairs Test
===========================

Variate: Fine20

Test Statistic:    15.00  (sum of signed ranks is positive)
Sample size:          12  (zero values have been excluded)
Probability:       0.062  (two-sided test)
```

The variate to be analysed is specified using the (first) parameter, DATA. Output is controlled by the PRINT option: test produces the relevant test statistics, and ranks prints the vector of signed ranks for the data. By default, WILCOXON prints the test statistic and sample size, excluding zero values. Here, the sample size is 12 since one of the original data values was 20 which then gave a zero value in Fine20. It also prints the probability of the statistic under the null hypothesis. This is calculated using the PRWILCOXON procedure, and is for a two-sided test: i.e. no assumption is made about whether the differences should be positive or negative. In Example 2.4.1 the conclusions are the same as from the t-test.

The value of the test statistic can be saved in the parameter STATISTIC, and the probability can be saved using the PROBABILITY parameter. The SIGN parameter saves an indicator of whether the total sum of signed ranks is positive (SIGN=1) or negative (SIGN=0), and the RANKS parameter can save a variate of the signed ranks of the differences (i.e. of DATA).

2.4.2 The sign test

The sign test is a nonparametric test for a difference in location between two related samples, or for testing the location of a single sample. The test is based on the *signs* (positive or negative) of the differences between corresponding members of the two samples, or on the sign of the differences between the sample members and the proposed location.

SIGNTEST procedure

Performs a one or two sample sign test (E. Stephens & P.W. Goedhart).

Options

PRINT = *string token*	Whether to print the test statistic with the associated probability and sample size (test); default test
METHOD = *string token*	Type of test (twosided, greaterthan, lessthan); default twos
GROUPS = *factor*	Defines the groups for a two-sample test if only the Y1 parameter is specified
NULL = *scalar*	Median value or difference in medians under the null hypothesis; default 0

Parameters

Y1 = *variates*	Data values for a one-sample sign test (neither Y2 nor

	GROUPS specified), or for the first sample of a two-sample test (Y2 also specified) or the values in both samples of a two-sample test (GROUPS specified but not Y2)
Y2 = *variates*	Data values for the second sample of a two-sample test
STATISTIC = *scalars*	To save the sign test statistic
NBINOMIAL = *scalars*	To save the effective sample size
PROBABILITY = *scalars*	To save the probability level of the test

The data values are specified by the parameters Y1 and Y2 and the option GROUPS. For a one-sample test, the Y1 parameter should be set to a variates containing the data. The data for a two-sample test can either be specified in two separate variates using the parameters Y1 and Y2. Alternatively, they can be given in a single variate, with the GROUPS option set to a factor to identify the two samples; the units are then assumed to be specified in the same order within each group. The GROUPS option is ignored when the Y2 parameter is set. The NULL option defines the size of the median under the null hypothesis for a one-sample test, or the difference between the two medians in a two-sample test. By default NULL=0.

The test is assumed to be two-sided unless otherwise requested by the METHOD option. Settings greaterthan or lessthan will give one-sided tests for the median or the difference between medians greater than, or less than, the null hypothesis value respectively.

In a one-sample test, units that are equal to the null hypothesis median are excluded and the effective sample-size is reduced. Similarly, in a two-sample test, units are excluded where the differences between the pairs of values are equal to that required by the null hypothesis. Units with missing values are also excluded.

By default, SIGNTEST prints the test statistic, the effective sample size and the (exact) probability level. This information can also be saved in named scalars using the STATISTIC, NBINOMIAL and PROBABILITY parameters repectively, and printing can be suppressed by setting option PRINT=*.

Example 2.4.2 continues Example 2.4.1, using a sign test to assess whether the median of the Fine soil values differ from 20.

Example 2.4.2

```
  11  SIGNTEST [NULL=20] Fine

One-sample sign test
====================

         Variate        Size      Median
            Fine          12       23.00

Test if median equals 20.00

Test statistic: 9
Effective sample size: 12
Two-sided probability level: 0.146
```

Methods for determining sample sizes for sign tests are described in Section 4.12.7.

2.4.3 The runs test

The runs test checks the randomness of a sequence of observations. The sample is assumed to be an ordered sequence of observations of two types, n_1 of the first type and n_2 of the second type. A run is defined to be a succession of observations of the same type. A clue to lack of randomness is provided by the total number of runs in the sequence. If the data are in random

order, the expected number of runs is $1 + 2n_1n_2/(n_1+n_2)$. A low number of runs might indicate positive serial correlation while a high number might arise from negative serial correlation.

RUNTEST procedure

Performs a test of randomness of a sequence of observations (P.W. Goedhart).

Options

PRINT = *string token*	Controls printed output (results); default resu
NULL = *scalar*	Defines the boundary between the two types; default 0

Parameters

DATA = *variates*	Sequences of observations
SAVE = *pointers*	To save the number of runs, the number of positive and negative observations and the lower and upper tail probabilities of the test

The DATA parameter is used to specify the sequence of observations. Observations larger than option NULL are considered to be of the first type (positive) while observation smaller than NULL are of the second type (negative). Missing values and observations that equal NULL are not taken into account. The PRINT option controls printed output, while the SAVE parameter can be used to specify a pointer containing five scalars to save the number of runs, the number of positive observations (that is, those larger than NULL), the number of negative observations and the lower and upper tail probabilities of the number of runs.

Example 2.4.3 performs a runs test on a set of random numbers generated by the function URANDOM.

Example 2.4.3

```
   2  CALCULATE uniform = URAND(43671; 5000)
   3  RUNTEST   [NULL=0.5] uniform

Runs test
=========

Number of runs in uniform: 2523
  expected number of runs: 2500.45
      right sided P-value: 0.266
       left sided P-value: 0.743
```

2.5 Two-sample nonparametric tests

This section describes some of the two-sample nonparametric tests in Genstat. The Mann-Whitney *U* test (procedure MANNWHITNEY, Section 2.5.1) provides a nonparametric alternative to the two-sample t-test, based on the ranks of the data values. An alternative for two matched samples (i.e. the situation where the data consist of pairs of observations, one from each sample) is the sign test, already described in Section 2.4.2. These procedures test for differences between the locations of the sample distributions. There are of course other aspects that can be compared. The Kolmogorov-Smirnoff test (procedure KOLMOG2, Section 2.5.2) assesses the overall similarity between the distributions of two samples. These tests are accessible through the Two-sample Nonparametric Tests menu of Genstat *for Windows*.

2.5.1 The Mann-Whitney test

MANNWHITNEY procedure

Performs a Mann-Whitney U test (S.J. Welham, N.M. Maclaren & H.R. Simpson).

Options

PRINT = *string tokens*	Output required (test, ranks, hodgeslehmann, confidence); default test
METHOD = *string token*	Type of test required (twosided, greaterthan, lessthan); default twos
GROUPS = *factor*	Defines the samples for a two-sample test if the Y2 parameter is not set
CIPROBABILITY = *scalar*	Probability for the confidence interval for the median difference between the samples; default 0.95
CONTROL = *scalar* or *text*	Identifies the control group against which to make comparisons if GROUPS is set; default uses the reference level of GROUPS

Parameters

Y1 = *variates*	Identifier of the variate holding the first sample if Y2 is set, or both samples if Y2 is unset (the GROUPS option must then also be set)
Y2 = *variates*	Identifier of the variate holding the second sample
R1 = *variates*	Saves the ranks of the first sample if Y2 is set, or both samples if Y2 is unset
R2 = *variates*	Saves the ranks of the second sample if Y2 is set
STATISTIC = *scalars* or *tables*	Saves the test statistics *U*
PROBABILITY = *scalars* or *tables*	Probability values for the test statistics
SIGN = *scalars* or *tables*	Saves indicators: 1 if the first sample scores the highest ranks on average, 0 otherwise
HODGESLEHMANN = *scalars* or *tables*	Saves the Hodges-Lehmann estimates for the differences in location of the two samples (i.e. the median differences between the samples)
LOWER = *scalars* or *tables*	Saves lower confidence values for median differences between the samples
UPPER = *scalars* or *tables*	Saves upper confidence values for median differences between the samples

The Mann-Whitney *U* test is a nonparametric test for differences in location between two samples. The data samples can be stored in two separate variates, and supplied by the parameters Y1 and Y2. Alternatively, they can be stored in a single variate, supplied by Y1, with the GROUPS option set to a factor to identify which unit belongs to each sample. The GROUPS option is ignored when the Y2 parameter is set. If GROUPS has more than 2 levels, each group is compared against a control group. You can define which level (or label) of GROUPS represents the control by setting the CONTROL option to a scalar or text. If CONTROL is not set, the reference level of GROUPS is used.

MANNWHITNEY calculates the test statistic *U*, along with its its associated probability value. An exact probability is calculated (using procedure PRMANNWHITNEYU) if the size of either sample is less than 51 and the statistic *U* is less than 10000; otherwise a Normal approximation is used. The statistic and the probability can be saved using the STATISTIC and PROBABILITY

parameters respectively. Parameter SIGN holds an indicator which takes the value 1 if the ranks in the first sample are higher on average than those in the second sample, and takes the value 0 otherwise. Usually STATISTIC, PROBABILITY and SIGN will save scalars, but they will save tables classified by the GROUPS factor when GROUPS is set to a factor with more than two levels. The ranks (with respect to the combined data set) for each sample can be saved using the R1 and R2 parameters.

Printed output is controlled by the PRINT option, with settings

test	test statistic and probability,
ranks	ranks (with respect to the whole data set) for each sample, and
hodgeslehmann	Hodges-Lehmann estimate of the difference in the locations of the samples, with confidence limits, and
confidence	synonym of hodgeslehmann.

The Hodges-Lehmann estimate is calculated as the median of all the differences between pairs of units (with one unit from each sample). The probability for the confidence limits is specified by the CIPROBABILITY option; the default, of 0.95, gives a 95% interval. The calculation of the interval may be slow when there are ties amongst the values, as essentially MANNWHITNEY then has to invert the probability function. The Hodges-Lehmann estimates can be saved by the HODGESLEHMANN parameter. The lower and upper confidence values can be saved by the LOWER and UPPER parameters, respectively.

By default a two-sided test is done (to assess that samples are unequal) but the METHOD option can be set to greaterthan to test that the first sample is greater than the than the second, or lessthan to test that it is smaller.

Example 2.5.1 illustrates the use of MANNWHITNEY to analyse the soil diffusion data previously assessed using a t-test, in Example 2.3.1b. For this data set, the results of the t-test and the Mann-Whitney test are similar with probabilities of 0.109 and 0.098, respectively, of obtaining a result this extreme under the null hypothesis of no difference between sample means.

Example 2.5.1

```
  12  MANNWHITNEY Fine; Coarse

Mann-Whitney U (Wilcoxon rank-sum) test
=======================================

Variates: Fine, Coarse.

Value of U: 47.0 (second sample has higher rank score).

Exact probability (adjusted for ties): 0.094
(under null hypothesis that Fine is equal to Coarse).

Sample sizes: 13, 12.
```

2.5.2 The Kolmogorov-Smirnoff test

The KOLMOG2 procedure performs a Kolmogorov-Smirnoff test of the overall similarity between the distributions of two samples, without assuming that these distributions follow any particular shape.

KOLMOG2 procedure

Performs a Kolmogorov-Smirnoff two-sample test (S.J. Welham, N.M. Maclaren & H.R. Simpson).

Options

PRINT = *string tokens*	Output required (test, differences, ranks): test gives the test statistic, differences gives signed differences, and ranks produces the ranks for each sample; default test
GROUPS = *factor*	Defines the groups for a two-sample test if only the Y1 parameter is specified

Parameters

Y1 = *variates*	Identifier of the variate holding the first sample
Y2 = *variates*	Identifier of the variate holding the second sample
R1 = *variates*	Saves the ranks of the first sample
R2 = *variates*	Saves the ranks of the second sample
STATISTIC = *scalars*	Scalar to save the test statistic (the maximum absolute difference between the cumulative distribution functions)
CHISQUARE = *scalars*	Scalar to save the chi-square approximation to the test statistic
DIFFERENCES = *variates*	Variate to save the signed differences between the cumulative distribution functions

The Kolmogorov-Smirnoff test assesses the similarity between the underlying distributions of the two samples, by comparing their cumulative distribution functions; the test statistic is the maximum absolute difference between the cumulative distribution functions. The samples can either be specified in two separate variates using the parameters Y1 and Y2. Alternatively, they can be given in a single variate, with the GROUPS option set to a factor to identify the samples. The GROUPS option is ignored when the Y2 parameter is set.

Output from the procedure is controlled by the PRINT option: test prints the relevant test statistic, differences prints the signed differences, and ranks prints a vector of ranks for each of the samples.

The test statistic and its chi-square approximation can be saved using the parameters STATISTIC and CHISQUARE respectively. The parameter DIFFERENCES can be used to save the differences between the cumulative distributions. The R1 and R2 parameters allow the ranks of the samples to be saved.

Example 2.5.2 continues Example 2.5.1, and applies the test to the soil diffusion data, finding no significant difference between the cumulative distribution functions of the two samples.

Example 2.5.2

```
  13  KOLMOG2 Fine; Coarse

Kolmogorov-Smirnov two-sample test
==================================

Variates: Fine, Coarse.

Maximum difference: 0.3526
Chi-square: 3.10 on 2 d.f. (p=0.212)
```

```
Sample Sizes: 13, 12.
```

2.6 Nonparametric analysis of variance

This section presents two procedures for nonparametric analysis of variance. The KRUSKAL procedure (2.6.1) performs the Kruskal-Wallis one-way analysis of variance, a nonparametric method based on the ranks of the data. Friedman's test (procedure FRIEDMAN, Section 2.6.2) is also based on ranks, but here the data are from a randomized complete block design: that is, the data set consists of observations on k treatments assessed under n different conditions (blocks). Section 2.6.3 then describes another test for several treatments based on ranks: Steel's many-one rank test (procedure STEEL), which compares several treatments with a control.

Custom menus are available for both these analyses in Genstat *for Windows*: click Stats on the menu bar, select Statistical Tests, and then the analysis required.

2.6.1 The Kruskal-Wallis one-way analysis of variance

KRUSKAL procedure

Carries out a Kruskal-Wallis one-way analysis of variance (S.J. Welham, N.M. Maclaren & H.R. Simpson).

Options

PRINT = *string tokens*	Output required (test, ranks): test produces the relevant test statistics, ranks produces a vector of ranks for each sample relative to the whole data set; default test
GROUPS = *factor*	Defines the sample membership if only one variate is specified by DATA
STATISTIC = *scalar*	Scalar to save the Kruskal-Wallis test statistic
MEANRANKS = *variate*	Variate to save the mean ranks of the samples
DF = *scalar*	Scalar to save the degrees of freedom for the statistic

Parameters

DATA = *variates*	List of variates containing the data for each sample, or a single variate containing the data from all the samples (the GROUPS option must then be set to indicate the sample to which each unit belongs)
RANKS = *variates*	Allow the ranks to be saved (relative to the combined data)

KRUSKAL carries out a Kruskal-Wallis one-way analysis of variance based on the ranks (relative to the whole data set) of a set of k samples. The analysis assesses the hypothesis that the samples come from distributions with the same mean (but without making any assumptions about the distributions themselves). The samples can be stored in different variates and supplied as a list in the DATA pointer. Alternatively, they can all be placed in a single variate, and the GROUPS option set to a factor to indicate the sample to which each unit belongs.

Output from the procedure is controlled by the PRINT option: test (the default setting) prints the relevant test statistics, and ranks prints the vector of ranks for each sample. When there are at least five observations in each of the samples, the test statistic approximately follows a Chi-square distribution on $k-1$ degrees of freedom. When this condition is not satisfied, and there are three samples, KRUSKAL uses a table of calculated values of the distribution of the statistic.

The test statistic, vector of mean ranks and degrees of freedom can be saved using the

STATISTIC, MEANRANKS and DF options, respectively. Parameter RANKS can be set to a variate, or variates, to store the ranks of the data relative to the whole data set.

Example 2.6.1 shows the use of the procedure KRUSKAL to analyse the doughnut data from Example 2.3.2. The chi-square test indicates that differences do exist between groups, and the mean ranks show which samples tend to have higher or lower scores: in this case sample 2 tends to have higher and group 4 lower scores, as in the analysis of variance in Example 2.3.2.

Example 2.6.1

```
   8  KRUSKAL [GROUPS=Fat] Absorb

Kruskal-Wallis One-Way Analysis of Variance
===========================================

Variate: Absorb
Group factor: Fat
Value of H = 11.81
Adjusted for ties = 11.83

 Sample         Size   Mean rank
Group 1            6       11.25
Group 2            6       19.50
Group 3            6       13.58
Group 4            6        5.67

Degrees of freedom = 3
Chi-square p-value = 0.008
```

2.6.2 Friedman's nonparametric analysis of variance

FRIEDMAN procedure

Performs Friedman's nonparametric analysis of variance (S. Langton).

Options

PRINT = *string tokens*	Output required (test, ranks); default test
TREATMENTS = *factor*	Treatment factor
BLOCKS = *factor*	Block factor

Parameters

DATA = *variates*	Identifier of the variate holding the data values
RANKS = *variates*	Saves the ranks
STATISTIC = *scalars*	Saves the test statistic
DF = *scalars*	Saves the degrees of freedom for the chi-square approximation
PROBABILITY = *scalars*	Saves the probability value for the chi-square statistic

Friedman's test is a nonparametric test for analysing a randomized complete block design. That is, the data set contains observations on k treatments assessed under n different conditions (or blocks). The test assesses the hypothesis that, under each condition, the samples arise from distributions with the same mean versus the alternative that the distribution means differ according to the treatment.

The variate of observations is specified using the DATA parameter, whilst options TREATMENTS and BLOCKS supply the treatment and blocking factors. Each block is checked in turn to ensure that it consists of exactly one replicate of each treatment, after excluding any units which are restricted out or which have missing values for DATA, TREATMENTS or BLOCKS. Any

block not meeting this condition is excluded from analysis and a warning is printed.

FRIEDMAN calculates the test statistic together with a probability value based on a chi-square approximation. If sample sizes are small, stored tabulated values are printed as well. The PRINT option controls printed output, with settings test to print the various test statistics, and ranks to print the ranks (together with the BLOCKS, TREATMENTS and DATA). Parameters RANKS, STATISTIC, DF and PROBABILITY can be used to save the ranks, the test statistic (adjusted for ties), the degrees of freedom for the chi-square approximation, and the probability value for the chi-square approximation.

Example 2.6.2

```
  2  " Example from Siegel & Castellan (1988), p.179."
  3  VARIATE Rank
  4  READ [PRINT=data,errors] Rank

  5  1 3 2   2 3 1     1 3 2     1 2 3   3 1 2   2 3 1
  6  3 2 1   1 3 2     3 1 2     3 1 2   2 3 1   2 3 1
  7  3 2 1   2 3 1   2.5 2.5 1   3 2 1   3 2 1   2 3 1 :
  8  FACTOR [LEVELS=18; VALUES=3(1...18)] Group
  9  & [LEVELS=3; LABELS=!t(RR,RU,UR); VALUES=(1...3)18] Type
 10  FRIEDMAN [TREATMENTS=Type; BLOCKS=Group] Rank

Friedman's test
===============

Data variate: Rank
Blocks:       Group
Treatments:   Type

Based on 18 blocks of 3 treatments
Friedman's statistic = 8.58
Adjusted for ties = 8.70
P-value using chi-square approximation (2 d.f.) = 0.013
Based on 2 degrees of freedom
```

2.6.3 Steel's many-one rank test

STEEL procedure

Performs Steel's many-one rank test (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (description, sumranks, critical, permutationtest); default desc, sumr, crit
METHOD = *string token*	Form of the alternative hypothesis (twosided, greaterthan, lessthan); default twos
TREATMENTS = *factor*	Defines the treatments
CONTROL = *scalar* or *text*	Treatment level corresponding to the control; default takes the reference level of TREATMENTS
NTIMES = *scalar*	Number of permutations for the permutation test; default 999
SEED = *scalar*	Seed to use to generate the random numbers for the permutation test; default 0
DATA = *variates*	Data values for the tests
SUMRANKS = *tables*	Saves the sum of the ranks within the treatments from each test

RANKS = *variates*	Saves the ranks of the data values for each test

Steel's test (Steel 1959) is a multiple-comparison test for comparing several treatments with a control treatment. The data are assumed to come from a one-way classification where all the treatments (and the control) have equal replication. The data values are specified, in a variate, using the DATA parameter. The TREATMENTS option species a factor to indicate the allocation of data values to treatments. The CONTROL option indicates which level of the TREATMENTS factor is the control; if this is not set, the reference level of TREATMENTS is used.

The METHOD option defines the type of test that is done. By default STEEL does a two-sided test, so the test is against the alternative hypothesis that the treatments may be either less than or greater than the control. If you set METHOD=lowerthan, STEEL does a one-sided test of the null hypothesis that the treatment values are not lower than the control. Alternatively, you can set METHOD=greaterthan, to do a one-sided test of the null hypothesis that the treatment values are not greater than the control.

The test operates by comparing the data values from each treatment in turn with the control. The comparison is made by pooling the data values from the treatment and control, forming their ranks, and calculating the sum of the ranks for the treatment data values. For METHOD=greaterthan, the test statistic for each treatment is simply the sum of the ranks for each treatment. For METHOD=lessthan, each rank sum must be subtracted from the total sum of ranks $(2n + 1) \times n$, where n is the replication of the treatments. For METHOD=twosided, the statistic is the minimum of the greaterthan and the lessthan statistics.

The PRINT option controls printed output, with settings:

description	description of the data and test;
sumranks	the test statistics (sums of ranks for each treatment);
critical	critical value as provided by Steel (1959);
permutationtest	uses a random permutation test to forms critical values and the probability that any treatment differs from control (according to the test specified by METHOD).

By default these are all produced.

By default, when PRINT=perm, STEEL makes 999 random allocations of the data to the treatment and control groups (using a default seed), and determines critical values for the test from the distribution of the minimum rank sum over these randomly generated datasets. The NTIMES option allows you to request another number of allocations, and the SEED option allows you to specify another seed. STEEL checks whether NTIMES is greater than the number of possible ways in which the data values can be allocated. If so, it does an exact test instead, which takes each possible allocation once. The results should be more reliable than Steel's critical values, which are based on a multivariate Normal approximation.

The rank sums can be saved using the SUMRANKS parameter, and the ranks of the individual treatment data values can be saved using the RANKS parameter.

Example 2.6.3 analyses data from Steel (1959). These are Binnet IQ scores of 3-year old female, white, private patients. classified as Normal. The aim is to test the suggestion that the IQ's of the Anoxic, Rh negative or Premature patients are less than those in the "Normal" control group. The permutation test concludes that no groups have IQ's significantly less than control: the 5% critical value is 27, but the minimum rank sum (for Anoxic) is 28.

Example 2.6.3

```
2  FACTOR  [NVALUES=24; LABELS=!t(Normal,Anoxic,'Rh negative',Premature);\
3          VALUES=(1...4)6] Treatment
4  VARIATE [NVALUES=24] IQ
5  READ    IQ

 Identifier   Minimum      Mean   Maximum    Values   Missing
```

```
            IQ     86.00     108.0     136.0        24         0

  12  STEEL   [METHOD=less; TREATMENTS=Treatment; CONTROL='Normal'] IQ

Steel's many-one rank test
==========================

Data variate: IQ
Treatments:   Treatment

Test against alternative hypothesis that treatments are less than control
(Normal).

               Sum of ranks
    Treatment
       Normal             -
       Anoxic          28.0
  Rh negative          38.0
    Premature          33.5

Minimum sum of ranks 28

* MESSAGE: Default seed for random number generator used with value 574750

Probability determined from 999 random permutations = 0.085


Critical values formed by a permutation test
--------------------------------------------

   5% 27.00
   1% 22.25
 0.1% 21.00


Critical values from Steel (1959)
---------------------------------

   5%    26
   1%    22
```

2.7 Plotting relationships between variables

Many investigations are concerned with understanding, and perhaps then modelling, relationships between variables. In this section we show a few of the techniques provided by Genstat for displaying relationships graphically (all accessible using the Graphics Wizard of Genstat *for Windows*). You can also use Genstat's very flexible graphics facilities to generate your own types of display.

2.7.1 Scatter plots

The scatterplot is a very effective method for displaying the relationship between pairs of variables (see Tufte 1983, page 47). To study a single pair of variables, you can use the DGRAPH directive (1:6.2.1) for a high-resolution plot, or the LPGRAPH directive (1:6.10.1) for the line-printer equivalent. Example 2.7.1 draws a scatterplot showing cancer death rates and cigarette consumption, as discussed by Tufte (1983, page 47). The resulting picture is shown in Figure 2.7.1.

Example 2.7.1

```
   2  " Display the relationship between death rates from lung cancer and
  -3    per capita cigarette consumption. Data from Doll (1955); also
  -4    displayed by Tufte (1983)."
   5  TEXT   Country
   6  READ   [PRINT=data] Country,Deaths,Cigarettes
```

```
 7  Australia       172  452
 8  Canada          151  508
 9  Denmark         168  379
10  Finland         353 1113
11  'Great Britain' 468 1145
12  Holland         244  468
13  Iceland          60  226
14  Norway           95  258
15  Sweden          116  315
16  Switzerland     252  540
17  U.S.A.          194 1290 :
18  PEN    1; SYMBOLS=9; LABELS=Country
19  XAXIS  3; TITLE='Cigarette consumption';\
20         LOWER=0; UPPER=1500; MARKS=!(0,250...1500)
21  YAXIS  3; TITLE='Deaths per million';\
22         LOWER=0; UPPER=500; MARKS=!(0,50...500)
23  DGRAPH [TITLE=\
24         'Lung cancer deaths 1950 vs cigarette consumption 1930';\
25         WINDOW=3; KEY=0] Deaths; Cigarettes; PEN=1
```

A single DGRAPH statement is all that would have been necessary to produce a simple unlabelled scatterplot (see 1:6.2.1). The PEN, XAXIS and YAXIS statements here provide labelling for the axes and the points (see 1:6.9). We could have reproduced Tufte's picture exactly, but this would have required a slightly more complicated program, to fit and display a regression line, and further refine the graphical environment).

The scatter-plot matrix provides a generalization of the simple scatter plot for the situation of more than two variables. A symmetric scatter-plot matrix is a triangular array of scatter plots showing every variable plotted against every other variable. Alternatively, a rectangular scatter-plot matrix plots one ser of variables against another set. Scatter-plot matrices are often studied prior to a multivariate analysis (see Chapter 6), and can be produced (in high-resolution graphics) by the DMSCATTER procedure. Details and an example are given in 1:6.8.4.

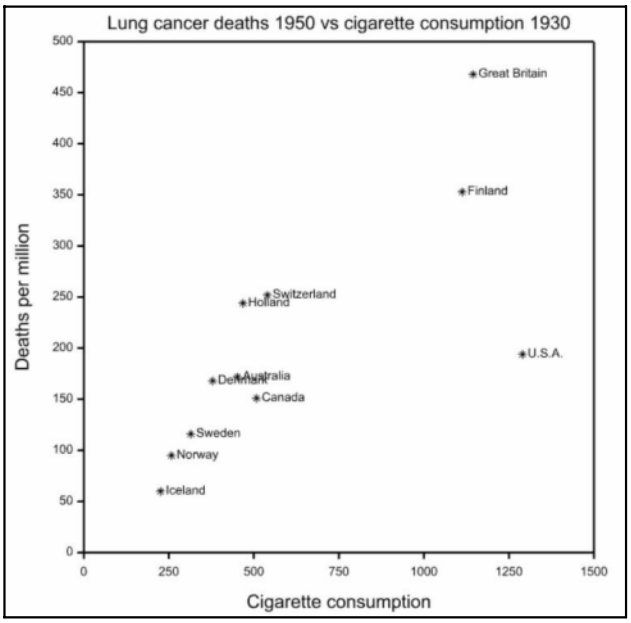


Figure 2.7.1

2.7.2 Parallel coordinates

DPARALLEL procedure

Displays multivariate data using parallel coordinates (Z. Karaman).

Options

TITLE = *text*	Title for the plot
GROUPS = *factor*	Defines grouping of the units (if any); by default, different pens are used for the observations in different groups
PERMUTATIONSALL = *string token*	Whether to display all necessary permutations so that any two variates will be adjacent in at least one plot, or

	just display once in the order given by the DATA pointer (yes, no); default no
SCALING = *string token*	Whether to do scaling overall (scale all variates on the same scale), or to scale each variate separately (overall, separate); default sepa
PEN = *variate*	Pens to be used for different groups (if any); default * uses pens from 1 up to the number of groups (number of levels of the GROUPS factor)

Parameter

DATA = *variates*	Data variables to be plotted

The DPARALLEL procedure displays the relationship between a set of variates using parallel coordinates. The dimensions are not represented by orthogonal lines as is customary when plotting scatter diagrams, but are represented by a series of parallel lines (either horizontal or vertical), with each point in multidimensional space represented by a broken line connecting its coordinates in each dimension. The only limit on the number of dimensions that can be displayed simultaneously by such plot is its readability, which is a function of the underlying graphics display (hardware).

The relationship between two variables can be visually assessed by inspecting the plot. When the correlation between two variables is close to −1, the lines will cross over so, in the limit, we would have a pencil of lines. (A pencil of lines is a set of lines that are coincident at a single point.) On the other hand, when the correlation approaches +1, we will have fewer and fewer crossovers, so that in the limit we will have a set of parallel lines.

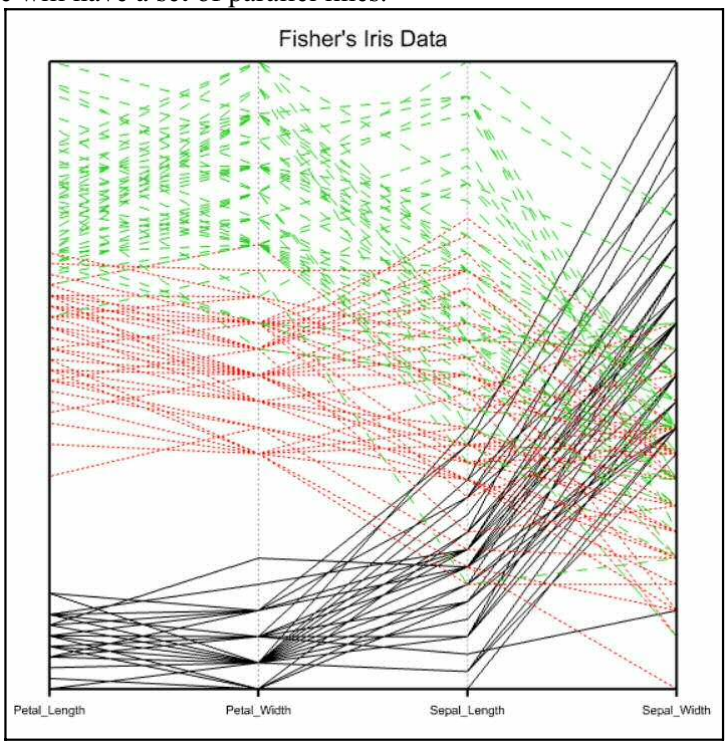


Figure 2.7.2

The pairwise comparisons are easy for variables represented by adjacent axes; however, they are much more difficult for the axes far away on the graph. If the PERMUTATIONSALL option is set to yes, several plots will be produced so that every pair of variables is adjacent in at least one plot.

The data are specified, in a list of variates, using the DATA parameter. The GROUPS option can be used to specify a grouping factor. The lines for observations in each group are then plotted using different pens, thus giving an immediate insight to any patterns in data. By default, pens 1 upwards are used for the different groups, but the PEN option can be used to specify other pens, in a variate with as many values as groups. If the GROUPS option is not set, the PEN option can be set to a scalar, to select the pen to be used for all the points. The TITLE option can be used to supply a title for the plots.

Example 2.7.2 produces a parallel coordinates plot of Fisher's Iris Data; see Figure 2.7.2.

Example 2.7.2

```
1  SET [WORKINGDIRECTORY='D:/G5/Proclib/PL23']
2  SPLOAD    [PRINT=*] '%GENDIR%/Data/Iris.gsh'
3  PEN       1...3; LINESTYLE=1...3
4  DPARALLEL [TITLE=!t('Fisher''s Iris Data'); GROUPS=Species] \
5            Petal_Length,Petal_Width,Sepal_Length,Sepal_Width
```

Genstat also provides several graphical displays specifically for examining the way in which one variable changes with two other variables, namely contour plots (DCONTOUR or CONTOUR), perspective views of surfaces (DSURFACE), three-dimensional graphs (D3GRAPH) and three-dimensional histograms (D3HISTOGRAM).

2.8 Correlation

Correlation is a measure of the association between two variables. The most commonly used correlation coefficient is the product-moment correlation coefficient which measures linear association (2.8.1), but Genstat also has some nonparametric alternatives: Spearman's rank correlation coefficient (2.8.2), Kendall's rank correlation coefficient τ (2.8.3), Kendall's coefficient of concordance (2.8.4), the kappa coefficient of agreement for nominally scaled data (2.8.5), the gamma statistic of agreement for ordinal data (2.8.6) and Lin's concordance correlation coefficient (2.8.7). Finally, if your aim is to assess the agreement between two sets of measurements, an alternative to correlation is to plot the differences between the measurements against their mean, in a *Bland-Altman* plot; see 2.8.8.

2.8.1 Product-moment correlation coefficient

Product-moment correlation coefficients between variates can be calculated by the FCORRELATION procedure.

FCORRELATION procedure

Forms the correlation matrix for a list of variates (R.W. Payne).

Options

PRINT = *string tokens*	Printed output (correlations, test); default corr
METHOD = *string token*	Type of test to make (against zero) for the correlations (twosided, greater, lessthan); default twos
WEIGHTS = *variate*	Provides weights for the units of the variates; default * assumes that they all have weight one
CORRELATIONS = *symmetric matrix*	Saves the correlations
PROBABILITIES = *symmetric matrix*	Saves the test probabilities
NOBSERVATIONS = *scalars*	Saves the number of observations from which the correlations have been calculated

Parameter

DATA = *variates*	Variates for which the matrix is to be calculated

The variates are listed by the DATA parameter. The WEIGHTS option can provide a variate of weights for the units of the variates; by default these are all assumed to have weight one.

Printed output is controlled by the PRINT option with settings:

correlations	prints the correlation matrix;
tests	prints tests for the correlations.

By default PRINT=correlation.

The METHOD option indicates the type of test to be done, with settings:

twosided	for a two-sided test of the null hupothesis that that the correlation is zero;
greaterthan	for a one-sided test of the null hypothesis that the correlation is not greater than zero;
lessthan	for a one-sided test of the null hypothesis that the correlation is not less than zero.

Tests cannot be produced if there are fewer than two observations.

The correlation matrix can be saved using the CORRELATIONS option, the (symmetric) matrix of test probabilities can be saved using the PROBABILITIES option, and the number of observations upon which it is based can be saved using NOBSERVATIONS option.

Example 2.8.1 shows how to use FCORRELATION to display a matrix of correlation coefficients between three measures of phosphorus in soil, and test the null hypothesis that they are not greater than zero.

Example 2.8.1

```
  2  " Correlations between inorganic phosphorus, organic phosphorus,
 -3    and estimated plant-available phosphorus. Data from Eid et al.
 -4    (1954); also analysed by Snedecor & Cochran (1989) p.335."
  5  READ [PRINT=data] InorganicP,OrganicP,PlantavailableP

  6   0.4 53 64   0.4 23 60   3.1 19 71   0.6 34 61   4.7 24 54
  7   1.7 65 77   9.4 44 81  10.1 31 93  11.6 29 93  12.6 58 51
  8  10.9 37 76  23.1 46 96  23.1 50 77  21.6 44 93  23.1 56 95
  9   1.9 36 54  29.9 51 99 :
 10  FCORRELATION [PRINT=correlations,test; METHOD=greaterthan]\
 11               InorganicP,OrganicP,PlantavailableP

Correlations
============

     InorganicP   1        -
       OrganicP   2   0.3989        -
PlantavailableP   3   0.7201   0.2118        -
                          1        2        3

Number of observations: 17


One-sided test of correlations greater than zero


     InorganicP   1           -
       OrganicP   2      0.0563           -
PlantavailableP   3      <0.001      0.2072           -
                             1           2           3
```

Note, however, that the product-moment correlation coefficient is assessing the linear relationship between the variables. It may not be effective with non-linear relationships. So, it is sensible also the plot the variables (see Section 2.7.1). If the relationship is causal (i.e. one variable represents a response and the other a "treatment") it is usually more informative to fit a model, using the methods for linear, generalized linear or non-linear regression described in Chapter 3. Methods for determining sample sizes for correlations are described in Section 4.12.10.

2.8.2 Spearman's rank correlation coefficient

This is the nonparametric equivalent of the product-moment correlation coefficient, based on the ranks of the data values rather than on the values themselves.

SPEARMAN procedure

Calculates Spearman's rank correlation coefficient (S.J. Welham, N.M. Maclaren & H.R. Simpson).

Options

PRINT = *string tokens*	Output required (test, correlations, ranks): test produces the correlation coefficient/matrix and relevant test statistics, correlations prints out just the correlation coefficients for each pair of variates; ranks produces the vectors of ranks for each sample; default test
GROUPS = *factor*	Defines the sample membership if only one variate is specified by DATA
CORRELATION = *scalar* or *symmetric matrix*	Scalar to save the rank correlation coefficient if there are two samples, or symmetric matrix to save the coefficients between all pairs of samples if there are several
T = *scalar* or *symmetric matrix*	Scalar to save the Student's t approximation to the correlation coefficient if there are two samples, or symmetric matrix to save the t approximations for all pairs of samples if there are several (calculated only if the sample size is 8 or more)
DF = *scalars*	Save the degrees of freedom for each t-statistic

Parameters

DATA = *variates*	List of variates containing the data for each sample, or a single variate containing the data from all the samples (the GROUPS option must then be set to indicate the sample to which each unit belongs)
RANKS = *variates*	Saves the ranks

SPEARMAN calculates Spearman's rank correlation coefficient between pairs of samples. The samples can be stored in different variates and supplied as a list with the DATA parameter. Alternatively, they can all be placed in a single variate, and the GROUPS option set to a factor to indicate the sample to which each unit belongs.

If the sample size is less than 50, an exact two-sided probability is calculated using the PRSPEARMAN procedure. Note, though, that the probability will be approximate if the variates contain ties; the probability is calculated for the adjusted correlation, but the calculation itself takes no account of the ties. SPEARMAN also calculates a Student's t approximation if the sample size is 8 or more (i.e. large enough for the approximation to be valid).

Printed output is controlled by the PRINT option, with settings:

correlation	to display correlations;
test	to display tests and correlations; and
ranks	to display the ranks for each sample.

The results can also be saved using the CORRELATION, T and DF options and the RANKS parameter.

Example 2.8.2 illustrates SPEARMAN, using the same data as in Example 2.8.1.

Example 2.8.2

```
  12  SPEARMAN InorganicP,OrganicP,PlantavailableP

Spearman's rank correlation coefficient
=======================================

Sample size = 17
Degrees of freedom = 15


Correlation matrix (adjusted for ties)
--------------------------------------

      InorganicP   1   1.000
        OrganicP   2   0.360   1.000
 PlantavailableP   3   0.663   0.245   1.000
                           1       2       3


Exact probabilities
-------------------

      InorganicP   1       *
        OrganicP   2   0.038       *
 PlantavailableP   3   0.001   0.084       *
                           1       2       3

(calculate for adjusted correlations, but ignoring ties)


t Approximation
---------------

      InorganicP   1       *
        OrganicP   2   1.496       *
 PlantavailableP   3   3.426   0.978       *
                           1       2       3

P-values
--------

      InorganicP   1       *
        OrganicP   2   0.155       *
 PlantavailableP   3   0.004   0.343       *
                           1       2       3
```

2.8.3 Kendall's rank correlation coefficient τ

Kendall's rank correlation coefficient (known as τ i.e. tau) provides an alternative to the Spearman correlation coefficient (2.8.2).

KTAU procedure

Calculates Kendall's rank correlation coefficient τ (R.W. Payne & D.B. Baird).

Options

PRINT = *string tokens* — Output required (correlations, probabilities); default corr, prob

GROUPS = factor — Defines the sample membership if only one variate is specified by DATA

CORRELATIONS = *scalar* or *symmetric matrix* — Scalar to save the rank correlation coefficient if there

are two samples, or symmetric matrix to save the coefficients between all pairs of samples if there are several

PROBABILITIES = *scalar* or *symmetric matrix*
: Scalar to save the probability for the correlation coefficient if there are two samples, or symmetric matrix to save the probabilities for all pairs of samples if there are several

NORMAL = *scalar* or *symmetric matrix*
: Scalar to save a transformation of tau that approximately follows a Normal distribution with mean zero and variance if there are two samples, or symmetric matrix to save the transformation for all pairs of samples if there are several

Parameter

DATA = *variates*
: List of variates containing the data for each sample, or a single variate containing the data from all the samples (the GROUPS option must then be set to indicate the sample to which each unit belongs)

The samples are specified as with SPEARMEN: as a list of DATA variates (one for each sample), or as a single DATA variate with the GROUPS option set to a factor to indicate the sample to which each unit belongs.

The PRINT option controls the printed output, with settings:

correlations
: to print the correlations between the samples; and

probabilities
: to print the corresponding probabilities (calculated under the assumption, or null hypothesis, that there is no association between the samples).

By default these are both printed.

The CORRELATIONS option allows the correlations to be saved, in a scalar if there are only two samples or in a symmetric matrix if there are three or more. Similarly, the probabilities can be saved using the PROBABILITIES option. These are calculated by procedure PRKTAU, which uses an exact formula for samples of size less than 35. For larger samples a Normal approximation can be used, which gives results practically identical to the exact values.

A drawback of the exact method is that is does not take account of ties. As an alternative, you can use the NORMAL option to save a transformation of τ that approximately follows a Normal distribution with mean zero and variance; this provides reasonably accurate probabilities when the number of units N is no smaller than 8 (see Kendall 1948). Example 2.8.3 shows how you can use the CUNORMAL function (1:4.2.9) to obtain probabilities from the Normal transformation of τ.

Example 2.8.3

```
  13  KTAU [NORMAL=Knorm] InorganicP,OrganicP,PlantavailableP

Kendall's rank correlation coefficient tau
==========================================

     InorganicP          1.0000
       OrganicP          0.3071          1.0000
PlantavailableP          0.5247          0.1805          1.0000
                     InorganicP        OrganicP  PlantavailableP
```

```
Probabilities
-------------

     InorganicP               *
       OrganicP          0.0381               *
PlantavailableP          0.0009          0.1541               *
                      InorganicP       OrganicP  PlantavailableP

* MESSAGE: probabilities of tau not adjusted for ties.

  14  PRINT CUNORMAL(Knorm)

             CUNORMAL(Knorm)

           1           *
           2     0.04266           *
           3     0.00164     0.15600           *
                       1           2           3
```

2.8.4 Kendall's coefficient of concordance

This coefficient, which can be calculated by procedure KCONCORDANCE and the Kendall's Coefficient of Concordance menu of Genstat *for Windows*, measures the overall level of association between several different sets of measurements taken on a single set of subjects.

KCONCORDANCE procedure

Calculates Kendall's Coefficient of Concordance, synonym CONCORD (S.J. Welham, N.M. Maclaren & H.R. Simpson).

Options

PRINT = *string tokens*	Output required (test, ranks): test produces the relevant test statistics, ranks produces the vector of mean ranks and the ranks for each sample; default test
GROUPS = *factor*	Defines the variable stored in each unit if only one variate is specified by DATA
STATISTIC = *scalar*	Scalar to save the coefficient of concordance
CHISQUARE = *scalar*	Scalar to save the chi-square approximation to the coefficient (calculated only if the sample size is at least 8)
MEANRANKS = *variate*	Variate to save the mean ranks for individuals over variables
DF = *scalar*	Scalar to save the degrees of freedom for CHISQUARE

Parameters

DATA = *variates*	List of variables to be compared, or a single variate containing the data for all the variables (the GROUPS option must then be set to indicate the variable recorded in each unit belongs)
RANKS = *variates*	Save the ranks of the variables

Kendall's Coefficient of Concordance is a measure of association between k rankings on n individuals. So, we have a set of N individuals that have been ranked on each of k variables in turn, and wish to compare the rankings. The variables can be stored in separate variates, with the DATA parameter set to list them all. Alternatively, all the data can be provided in a single variate, with the GROUPS option set to a factor to indicate which variable is recorded in each unit of the variate. (KCONCORDANCE then assumes that the individuals are recorded in the same order for

each variable.)

KCONCORDANCE calculates the chi-square approximation to the statistic if the sample sizes are large enough (i.e. 8 or more). Otherwise, for $2<k<21$ and $2<n<8$, KCONCORDANCE looks up the probability from a stored table. The results of these calculations can be printed using the test setting of PRINT, or saved using the options STATISTIC (for the coefficient), CHISQUARE (for the chi-square statistic) and DF (degrees of freedom). The ranks setting of PRINT causes the vector of mean ranks (over all variates) and the ranks for each variate individually to be displayed, and these can be saved using the MEANRANKS option and the RANKS parameter.

Example 2.8.4 calculates the overall concordance between the three different measures of phosphorus in soil, indicating evidence of association between the orderings of the three variables.

Example 2.8.4

```
  15  KCONCORDANCE InorganicP,OrganicP,PlantavailableP

Kendall's coefficient of concordance
====================================

Variates: InorganicP, OrganicP, PlantavailableP.

Coefficient: 0.612
Adjusted for ties: 0.615
Sample size: 17
Number of samples: 3
Sum of squares: 2247.00
Chi-Square: 29.5
Degrees of freedom: 16.0
Probability: 0.021
```

2.8.5 The kappa coefficient

KAPPA procedure

Calculates a kappa coefficient of agreement for nominally scaled data (A.J. Rook).

Option

PRINT = *string token*	Whether to print kappa and its associated information (test); default test

Parameters

DATA = *tables*	Data sets, each consisting of an object × category table whose entries are the number of judges assigning the *i*th object to the *j*th category
STATISTIC = *scalars*	Save the value of kappa for each data table
VARIANCE = *scalars*	Save the corresponding variances

The kappa coefficient (which can be calculated by the Kappa Statistic menu of Genstat *for Windows*) provides a way of assessing the agreement between judges who have rated a set of n objects or subjects using a nominal scale: that is, each judge has allocated each object to one of m different categories.

The data for KAPPA, specified by the DATA parameter, consist of an $n \times m$ table whose entries indicate the number of judges that have assigned the *i*th object to the *j*th category. This must not contain any missing values and all the row totals must be equal.

Kappa takes the value one when there is complete agreement and zero when there is none (except that expected by chance). The printing of the test statistic and its associated information

is controlled by the PRINT option. With the default, test, the procedure prints the actual and expected proportion of times that the judges agree, the resulting value of kappa and its variance. When *N* is large, the sampling distribution of kappa is approximately Normal. The procedure thus also prints the value of kappa divided by the variance, and its probability assuming a Normal distribution. A warning is printed if *N* is less than 20. The STATISTIC and VARIANCE parameters allow kappa and its variance to be saved, in scalars.

Example 2.8.5

```
  2  " Data from Siegel and Castellan (1988) p.287."
  3  FACTOR [LEVELS=29] Object
  4  FACTOR [LEVELS=5] Category
  5  TABLE  [CLASSIFICATION=Object,Category; \
  6    VALUES=(4(0),4, 2,0,2,2(0))2, 3(0),1,3, 2(1),2,2(0), (3,0,1,2(0))2,\
  7    2(0),2(2),0, 3,0,1,2(0), 4(0),4, (4,4(0))3, 2(0),3,1,0, 1,0,2,1,0, \
  8    3(0),2(2), 4(0),4, 2(0),3,0,1, 0,1,3,2(0), 2(0),1,0,3, 2(0),3,1,0, \
  9    (4,4(0))2, 2,0,2,2(0), 1,0,3,2(0), (2,0,2,2(0))2, 0,1,2,0,1] Fish
 10  PRINT  Fish; FIELD=4; DECIMALS=0

             Fish
   Category   1   2   3   4   5
     Object
          1   0   0   0   0   4
          2   2   0   2   0   0
          3   0   0   0   0   4
          4   2   0   2   0   0
          5   0   0   0   1   3
          6   1   1   2   0   0
          7   3   0   1   0   0
          8   3   0   1   0   0
          9   0   0   2   2   0
         10   3   0   1   0   0
         11   0   0   0   0   4
         12   4   0   0   0   0
         13   4   0   0   0   0
         14   4   0   0   0   0
         15   0   0   3   1   0
         16   1   0   2   1   0
         17   0   0   0   2   2
         18   0   0   0   0   4
         19   0   0   3   0   1
         20   0   1   3   0   0
         21   0   0   1   0   3
         22   0   0   3   1   0
         23   4   0   0   0   0
         24   4   0   0   0   0
         25   2   0   2   0   0
         26   1   0   3   0   0
         27   2   0   2   0   0
         28   2   0   2   0   0
         29   0   1   2   0   1

 11  KAPPA  Fish

Measures of agreement for nominally scaled data
===============================================

Proportion of times judges agree
      Actual           Expected    Kappa coefficient    Variance
       0.580              0.288                0.410     0.00271


Test of significance of Kappa
-----------------------------

Kappa / s.e.(Kappa)  Normal probability
              7.887              <0.001
```

2.8.6 The gamma statistic

GSTATISTIC procedure

Calculates the gamma statistic of agreement for ordinal data (A.W. Gordon).

Options

PRINT = *string token*	Whether to print the statistic with its associated information and the resulting test (test); default test
METHOD = *string token*	Type of test required (twosided, positive, negative); default twos

Parameters

DATA = *tables*	Tables of data each classified by the two variables (factors) of interest
STATISTIC = *scalars*	Save the value of gamma for each data table
VARIANCE = *scalars*	Save the corresponding variances

The gamma statistic (Siegel & Castellan 1988, pages 291-298) provides a way of assessing the agreement between two variables measured using ordinal scales. In Genstat these would each be represented as factors whose levels represent a ranking of the individuals according to some measurement.

For example, suppose we have a factor A with r levels and a factor B with k levels. The data for GSTATISTIC, specified by the DATA parameter, consists of an r by k table classified by A and B, whose entries indicate the number of times that the ith level of variable A occurs with the jth level of variable B. The table must not contain any missing values. The statistic has the value 1 when there is no disagreement in the ordering of the variables, −1 if the ordering defined by A has no disagreement with the reverse of the ordering defined by B, and zero if the variables are independent.

The printing of the test statistic and its associated information is controlled by the PRINT option. With the default, test, the procedure prints the number of times that the variables agree and disagree, the resulting value of gamma and its variance. When the number of observations N is large, the sampling distribution of gamma is approximately Normal. The procedure thus also prints the value of gamma divided by the variance, and its probability assuming a Normal distribution. A warning is printed if N is less than 20.

The test is assumed to be two-sided (i.e. no prior knowledge is assumed about the type of association) unless otherwise requested by the METHOD option. Setting METHOD=positive will give a one-sided test of the null hypothesis that there is a positive association. Similarly, METHOD=negative will produce a one-sided test that there is a negative association.

The STATISTIC and VARIANCE parameters allow gamma and its variance to be saved, in scalars.

Example 2.8.6

```
2  " Example from Siegel and Castellan (1988) p.296."
3  FACTOR [LABELS=!t('Successful quitter','In-process quitter',\
4         'Unsuccessful quitter')] Ability
5  &      [LABELS=!t('1','2-4','5-9','10-14','15-19','20-25','>25')] Time
6  TABLE  [CLASSIFICATION=Ability,Time; VALUES=13,29,26, 22,9,8,\
7         8,5,2, 6,2,1, 3,0,1, 9,16,14, 21,16,29] Nurses
8  PRINT  Nurses; FIELD=6; DECIMALS=0

                 Nurses
            Time     1   2-4   5-9 10-14 15-19 20-25   >25
         Ability
```

```
  Successful quitter    13    29    26    22     9     8     8
  In-process quitter     5     2     6     2     1     3     0
Unsuccessful quitter     1     9    16    14    21    16    29

   9  GSTATISTIC Nurses

Measures of association for ordered variables
=============================================

      No agreements   No disagreements   Gamma statistic    Variance
              10580               3690            0.4828     0.01290


Two-tailed test of significance for Gamma non-zero
--------------------------------------------------

   Gamma/s.e.(Gamma)  Normal probability
               4.251              <0.001
```

2.8.7 Lin's concordance correlation coefficient

LCONCORDANCE procedure

Calculates Lin's concordance correlation coefficient (R.W. Payne & M.S. Dhanoa).

Options

PRINT = *string token*	Controls printed output (concordance); default conc
GROUPS = *factor*	Defines the sets of measurements when they are all supplied in a single DATA variate
CONCORDANCE = *scalar* or *variate*	Saves Lin's the concordance coefficient
LOWER = *scalar* or *variate*	Saves the lower confidence limit for the coefficient
UPPER = *scalar* or *variate*	Saves the upper confidence limit for the coefficient
CORRELATION = *scalar* or *variate*	Saves the correlation coefficient
CB = *scalar* or *variate*	Saves the bias correction factor
ZTRANSFORMATION = *scalar* or *variate*	Saves the Z transformation of the coefficient
ZSD = *scalar* or *variate*	Saves the standard deviation of the Z transformation
CIPROBABILITY = *scalar*	Defines the size of the confidence interval; default 0.95 i.e. 95%
REFERENCELEVEL = *scalar* or *text*	Defines the set of measurements to be used as the control if there are more than two variates or groups; default 1

Parameter

DATA = *variates*	List of variates specifying the sets of measurements to be compared, or a single variate containing all the measurements (the GROUPS option must then be set to indicate the set to which each unit belongs)

Lin's concordance correlation coefficient measures how well a new set of observations reproduce an original set. So, for example, it can be used to assess the effectiveness of a new instrument or a new measurement method. The coefficient is formed by multiplying two components. The first is the ordinary Pearson correlation coefficient (2.8.1), which assesses the linearity of the relationship between the two sets of measurements. However, for the second set to reproduce the first, additional requirements are that the slope of the line relating the two sets should be one

and that the line should go through the origin. These other aspects are assessed by the second component, which is known as C_b.

The measurements are supplied using the DATA parameter. You can set this to a list of variates, one for each measurement. Alternatively, you can put them all into a single variate, and set the GROUPS option to a factor to identify which measurement is stored in each unit of the variate. (LCONCORDANCE then assumes that the individuals that were measured are recorded in the same order within each set of measurements.) If there are more than two sets of measurements, LCONCORDANCE takes one of these as the control (i.e. the standard) set, and compares the others with this. By default the control is first variate if DATA has been set to a list of variates, or the set corresponding to the reference level of the GROUPS factor (see the FACTOR directive, 1:2.3.3) if there was a single variate. However, you can define a different control by setting the REFERENCELEVEL option, to a scalar to indicate the number of the variate within the list of DATA variates of the level of the GROUPS factor. Alternatively, if the GROUPS factor has labels, you can set REFERENCELEVEL to a text.

Lin (1989, 2000) has shown that, if the coefficient is given an inverse hyperbolic tangent transformation (i.e. a Z-transformation), the result has an approximate Normal distribution. LCONCORDANCE uses this to produce a confidence interval for the coefficient. The size of the interval is specified by the CIPROBABILITY option; the default is 0.95 (i.e. 95%).

By default, the concordance coefficient, the lower and upper confidence limits, the correlation coefficient and Cb are printed. However, you can set option PRINT=* to suppress this. The CONCORDANCE, LOWER, UPPER, CORRELATION, CB, ZTRANSFORMATION and ZSD parameters allow the coefficient and all the associated information to be saved.

The coefficient is illustrated in Example 2.8.7.

Example 2.8.7

```
  2  " Data from Muller & Buttner (1994), Statistics in Medicine, 13, 2465-76;
 -3    also see Nickerson (1997), Biometrics, 53, 1503-1507."
  4  VARIATE [VALUES=4.8,5.6,6.0,6.4,6.5,6.6,6.8,7.0,7.0,7.2,7.4,7.6,\
  5          7.7,7.7,8.2,8.2,8.3,8.5,9.3,10.2,10.4,10.6,11.4] Trial1
  6  &       [VALUES=5.8,5.1,7.7,7.8,7.6,8.1,8.0,8.1,6.6,8.1,9.5,9.6,\
  7          8.5,9.5,9.1,10.,9.1,10.8,11.5,11.5,11.2,11.5,12.0] Trial2
  8  LCONCORDANCE Trial1,Trial2

Lin's concordance correlation coefficient
=========================================

   Concordance       Lower       Upper    Correlation        Cb
        0.7512      0.5751      0.8608         0.9244    0.8126
```

Lin (1989) derives the coefficient (ρ_c) by considering how well the relationship between the measurements is represented by a line through the origin at an angle of 45 degrees (as would be generated if the two measurements generated identical results):

$$\rho_c = 1 - d_c^2 / d_u^2$$

where d_c^2 is the expected squared perpendicular deviation from the line, and d_u^2 is the expected squared perpendicular deviation from the line when the measurements are uncorrelated.

This can be written as

$$\rho_c = \rho \times C_b$$

The term ρ is the Pearson product-moment correlation coefficient, while C_b is a bias correction factor which is calculated by

$$C_b = 2 / (v + 1/v + u^2$$
$$v = s_1 / s_2$$
$$u = (m_1 - m_2) / \sqrt{(s_1 \times s_2)}$$

where m_i and s_i (i = 1,2) are the mean and standard deviation of the i^{th} set of measurements.

Methods for determining sample sizes for Lin's coefficient are described in 4.12.11.

2.8.8 Bland-Altman plots

BLANDALTMAN procedure

Produces Bland-Altman plots to assess the agreement between two variates (A.R.G. McLachlan).

Options

PRINT = *string tokens*	Controls printed output (summary, estimates); default * i.e. none
PLOT = *string tokens*	What to plot (blandaltman, normal); default blan
DMETHOD = *string token*	Method for calculating differences (differences, ratios, %differences, percentages); default diff
LMETHOD = *string token*	Method for calculating limits of agreement when regression is not used (normaldistribution, percentile); default norm
REGMETHOD = *string tokens*	Whether to use regression to calculate bias (i.e. mean) or limits (bias, mean, limits, auto); default * i.e. none
CIPROBABILITY = *scalar*	Probability level for limits of agreement, confidence intervals and percentiles; default 0.95
LOWERLIMIT = *scalar*	Lower limit of agreement to use instead of a calculated limit
UPPERLIMIT = *scalar*	Upper limit of agreement to use instead of a calculated limit
ALPHALEVEL = *scalar*	Critical probability level used for regression when REGMETHOD=auto; default 0.05
XBLANDALTMAN = *string token*	X-values to use for the Bland-Altman plot (mean, Y1, Y2); default mean
REFERENCELINECHOICE = *string tokens*	Reference lines to plot on a Bland-Altman plot (bias, mean, limits, zero); default bias
GRAPHICS = *string token*	Type of graph (highresolution, lineprinter); default high
WINDOW = *scalar*	Window for the plot; default 3
SCREEN = *string token*	Whether to clear or keep the screen before displaying the plot (keep, clear); default clea
PENZEROLINE = *scalar*	Pen to use for the zero reference line
PENMEANLINE = *scalar*	Pen to use for the mean reference line
PENLIMITLINES = *scalar*	Pen to use for the reference lines showing limits of agreement

Parameters

Y1 = *variates*	First variate
Y2 = *variates*	Second variate
LABELS = *texts*	Labels for individual points on the Bland-Altman plot
MEANS = *variates*	Saves the means
DIFFERENCES = *variates*	Saves the differences, ratios or % differences (according to the DMETHOD option)
TITLE = *texts*	Title for the Bland-Altman plot
YTITLE = *texts*	Title for y-axis of the Bland-Altman plot
XTITLE = *texts*	Title for x-axis of the Bland-Altman plot

PEN = *scalars*, *variates* or *factors*	Pen for plotting points on the Bland-Altman plot; default 1

Bland-Altman plots provide an effective way of assessing two different methods for measuring some quantity (Bland & Altman 1999; see also Altman & Bland 1983 and Bland & Altman 1986). The data are supplied by the Y1 and Y2 parameters, in two variates containing measurements on the same set of samples. The default display plots the differences between the measurements against their mean, so that the sizes of the discrepancies can be assessed while also seeing whether there is any bias or nonlinearity between the methods. Ideally, the points should lie within a rectangle arranged symmetrically around the x-axis i.e. similar amounts of scatter above and below the line of zero difference. The means and differences can be saved, in variates, using the MEANS and DIFFERENCES parameters, respectively.

The DMETHOD option controls the type of difference that is displayed, with settings:

differences	differences Y1 - Y2 (default),
ratios	Y1 / Y2,
%differences	(Y1 - Y2) / ((Y1 + Y2)/2) * 100, and
percentages	synonym of %differences.

The plot can also show "limits of agreement" which are intended to represent boundaries on the acceptable difference between the methods. These can be supplied by the LOWERLIMIT and UPPERLIMIT options, Alternatively, if LOWERLIMIT and UPPERLIMIT are not set, the limits are calculated by the procedure according to the setting of the LMETHOD option:

normaldistribution	uses confidence limits calculated assuming that the differences have a Normal distribution (default), and
percentile	takes percentiles of the differences.

The CIPROBABILITY option specifies the probability for calculating the limits of agreement when LMETHOD=norm, or the percentiles used for the limits when LMETHOD=perc. The default of 0.95 gives 95% limits of agreement, and percentiles of 2.5 and 97.5%.

The REFERENCELINECHOICE option allows reference lines can be included on the Bland-Altman plot:

mean or bias	plots a line at the overall mean of the differences (default),
limits	plots upper and lower limits of agreement, and
zero	plot horizontal line at zero, or one when DMETHOD=ratio.

If there seems to be a trend in the plot (differences becoming larger or smaller as the means increase), it can be useful to fit a linear regression (on the mean) to the bias, or to the variation in the bias, or both. This is controlled by the REGMETHOD option. Setting REGMETHOD to mean or bias fits a line through the Bland-Altman plot to estimate the mean or bias. Limits of agreement are then calculated assuming a constant variance and a Normal distribution so that, if references lines are plotted for the limits are plotted, they will be parallel to the reference line for the mean. Alternatively, if REGMETHOD=limits, linear regression is used to estimate the variation in the differences. The limits then form a 'fan-shape' pattern about the horizontal bias line. These two settings can be combined (REGMETHOD=bias,limits) so that linear regression is used to estimate both the bias and the variation in the differences. Finally, if you set REGMETHOD=auto, the procedure automatically determines whether or not linear regression should be used to estimate either the bias or the variation or both. The ALPHALEVEL option then specifies the critical value for testing the significance of the regressions (default 0.05 i.e. 5%), to decide whether they should be used.

The PLOT option controls the plots that are produced:

blandaltman	produces the Bland-Altman plot (default), and
normal	produces a Normal (q-q) plot of the differences.

The x-values to be used in the Bland-Altman plot are controlled by the XBLANDALTMAN option. The default is to use the averages of the Y1 and Y2 variates (as recommended by Bland &

Altman 1995). Alternatively, the settings `Y1` and `Y2` allow one of the two variates to be used instead; Krouwer (2008) recommended plotting against measurements from a reference method, if this has provided much better precision.

By default high-precision graphics is used, but you can set option `GRAPHICS=lineprinter` to produce character-based graphs in the output window instead. The `WINDOW` option can be used to specify which graphics window to use for a high-resolution graph, and the `SCREEN` option allows you to stop the screen being cleared before plotting the Bland-Altman graph. Note that this does not to apply to the Normal probability plots, as the `DPROBABILITY` procedure (that is used to produce the plot) does not support the `SCREEN` option.

There are several options and parameters that can be used to modify the appearance of the Bland-Altman plot. The `TITLE` parameter can supply an overall title, and the `YTITLE` and `XTITLE` parameters can supply titles for the y- and x-axis. You can specify a text containing labels for the points in the Bland-Altman plot using the `LABELS` parameter. The `PEN` parameter allows you to specify a pen or pens for the points (default 1). The `PENZEROLINE`, `PENMEANLINE` and `PENLIMITSLINES` options specify pens for the reference lines at zero, mean difference and limits of agreement, respectively. If these options are not set, `BLANDALTMAN` uses the line colours, thicknesses and styles (if set) from pens 1, 2 and 3, respectively.

The `PRINT` option controls the printing of the results, with settings:

`estimates`	to print the estimates, and
`summary`	to print a summary showing the number and percentage of values above and below zero, and outside the limits of agreement.

When regression is being used, the estimates consist of the slope of the line, with its standard error and confidence interval , together with the sample size. Otherwise, they consist of the mean difference, limits of agreement, standard error of the differences and the sample size. By default, nothing is printed.

Example 2.8.8 assesses two sets of measurements of peak expiratory flow rate, one made with a Wright peak flow meter, and the other with a mini Wright meter; see Bland & Altman (1986).

Example 2.8.8

```
   2  VARIATE [NVALUES=17] Wright,Mini; VALUES=\
   3          !(494,395,516,434,476,557,413,442,650,433,\
   4            417,656,267,478,178,423,427),\
   5          !(512,430,520,428,500,600,364,380,658,445,\
   6            432,626,260,477,259,350,451)
   7  BLANDALTMAN [PRINT=estimates,summary; REFERENCE=mean,limit]\
   8              Y1=Wright; Y2=Mini

Bland-Altman results for differences between Wright and Mini
============================================================

Difference = Wright - Mini
n = 17 paired values

Numbers
-------

                          Above bias  Below bias       Total
                  Result
 Values above upper limit          0           0           0
    Values between limits          7           9          16
 Values below lower limit          0           1           1
                   Total          7          10          17
```

```
Percentages
-----------

                          Above bias  Below bias       Total
                  Result
 Values above upper limit          0           0           0
    Values between limits      41.18       52.94       94.12
 Values below lower limit          0        5.88        5.88
                   Total      41.18       58.82         100

Estimates
---------

Mean bias assumed constant, with assumed constant variance, and parallel limits
of agreement

                   Estimate     95% C.I.
     Difference       -2.12 (-22.05, 17.81)
Lower 95% limit      -78.10 (-112.7, -43.53)
Upper 95% limit       73.86  (39.29, 108.4)
```

The plot, in Figure 2.8.8, shows no obvious relation between the difference and the mean. The confidence limits for the bias include zero, and there is only one point outside the 95% limits of agreement. However, the limits are rather wide, reflecting the small sample size. In practical terms these may not be acceptable.

Note that the procedure does not cater for repeated measures of subjects. See Bland & Altman (1999, 2007) for information on how different types of repeated measures can be handled.

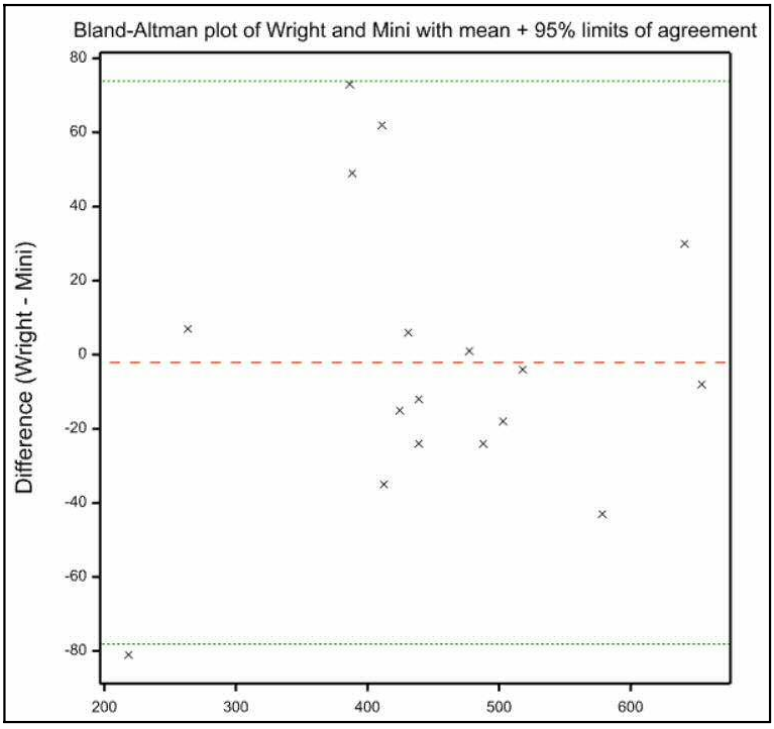


Figure 2.8.8

2.9 Tests for independence and changes in two-way tables

When measurements are qualitative or categorical, a different approach is needed to establish relationships than when they are quantitative. One way is to analyse the counts of individuals with each combination of levels of the categorical variables: a set of counts like this is often presented in a table known as a *contingency table*. In a two-way table you may want to assess whether the factors in the rows and columns are independent, or whether they are associated. In this section we show two ways of doing this: the standard chi-square test (2.9.1) and Fisher's exact test (2.9.2). Both of these are available through the Contingency Tables menu of Genstat *for Windows*.

Another situation involving two-way tables is covered by McNemar's test (2.9.3). This is relevant to "before and after" designs, where subjects are assessed on two occasions (e.g. before

and after a treatment) and the aim is to see whether their responses (selected from one of two possibilities) have changed. Cochran's Q test (2.9.4) extends McNemar's test to three or more occasions. These are also available through menus in Genstat *for Windows*.

You can form the table of counts from the raw data using the TABULATE directive (1:4.11.1). Alternatively you can provide the tabulated data directly, while declaring the table using the TABLE directive (1:2.5), or by declaring the table and then reading in its contents using the READ directive (1:3.1.1).

2.9.1 The chi-square test

CHISQUARE procedure

Calculates chi-square statistics for one- and two-way tables (A.D. Todd & P.K. Leech).

Options

PRINT = *string tokens*	Output required (test, probability, fittedvalues, tchisquare); default test, prob
METHOD = *string token*	Method for calculating chi-square (pearson, maximumlikelihood); default pear
GOODNESSOFFIT = *string token*	Whether to carry out a goodness-of-fit test for the DATA values against a supplied set of FITTEDVALUES (yes, no); default no

Parameters

DATA = *tables*	Table containing observed data
CHISQUARE = *scalars*	Scalar to save the chi-square value
DF = *scalars*	Scalar to supply or save the degrees of freedom
PROBABILITY = *scalars*	Scalar to save the probability value
FITTEDVALUES = *tables*	Table of expected values
RESIDUALS = *tables*	Table of standardized residuals
TCHISQUARE = *tables*	Table whose cells show the individual contributions to the chi-square value

The CHISQUARE procedure calculates chi-square statistics. The DATA parameter supplies the data values. If these are in a two-way table, CHISQUARE produces the usual test of association between the row and column factor of the table; if a one-way table is supplied, the statistic assesses whether the different cells of the table contain different proportions of the data. Alternatively, you can set option GOODNESSOFFIT=yes to request a goodness-of-fit test between the data values and a set of expected values supplied by the FITTEDVALUES parameter; if you provide the degrees of freedom, using the DF parameter, the procedure can also calculate the probability value.

The PRINT option controls the printed output, with the settings: test to print the chi-square value and degrees of freedom; probability for the probability value; fittedvalues data, fitted (expected) values and standardized residuals; and tchisquare to show the contribution of each cell of the table to the chi-square value. By default, the statistic is calculated by the usual Pearson approximation

$$\text{chi-square} = \text{sum}((o-e) \times (o-e) / e),$$

where o = observed, and e = expected. Alternatively, you can set option METHOD=likelihood to calculate the chi-square by maximum likelihood (using the Genstat facilities for generalized linear models). Parameters CHISQUARE, DF, PROBABILITY, FITTEDVALUES, RESIDUALS and TCHISQUARE allow the results to be saved in appropriate Genstat data structures.

Example 2.9.1, analyses a two-way table containing the results from a survey of smoking

habits. The classifying factors both have two levels, so the table has four cells. The chi-square test assesses the independence of the two classifications, `Mortality` and `Smoking`. Essentially it is testing whether the distribution of subjects between the two categories of one factor appears to change according to the categories of the other factor.

Example 2.9.1

```
  2  " Relationship between smoking habits and mortality in Canada.
 -3    Data from Best et al. (1966); also analysed by Snedecor &
 -4    Cochran (1989) p.124."
  5  FACTOR [LABELS=!t(Dead,Alive)] Mortality
  6  & [LABELS=!t(Nonsmoker,'Pipe smoker')] Smoking
  7  TABLE [CLASS=Mortality,Smoking; VALUES=117,54,950,348] Counts
  8  PRINT Counts; DECIMALS=0

                Counts
    Smoking   Nonsmoker Pipe smoker
   Mortality
        Dead         117          54
       Alive         950         348

  9  " Perform Pearson chi-square test of independence of classifications."
 10  CHISQUARE Counts

Chi-square test for association between Mortality and Smoking
=============================================================

Pearson chi-square value is 1.73 with 1 df.
Probability level (under null hypothesis) p = 0.189
```

The test statistic here indicates that the two classifying factors, `Smoking` and `Mortality`, are independent; that is, that there is no evidence that they are associated.

If you set the `METHOD` option of `CHISQUARE` to `maximumlikelihood`, the procedure uses the Genstat facilities for generalized linear models (GLMs). This produces a statistic known as the *deviance*, which is equivalent (although calculated differently) to the Pearson chi-square statistic. The GLM facilities can actually handle much more complicated situations that this (for example three or more classifications), and fit much more sophisticated models (for example involving variates as well as factors). Full details are given in Chapter 4.

2.9.2 Fisher's exact test and permutation tests

The chi-square test is approximate: the test statistics are only approximately distributed as chi-square statistics with one degree of freedom. The approximation improves as the number of observations increases, and in Example 2.9.2 the numbers are large enough for the approximation to be good. However, Genstat also provides an exact method to test for independence in this simple case of a two-by-two table. This method, known as Fisher's exact test, involves evaluating all 2×2 tables with the same margins as the observed table and can be carried out by the `FEXACT2X2` procedure. For larger tables, too many tables are generally possible for it to be feasible to evaluate them all, and so Genstat provides a random permutation test instead. The procedure `CHIPERMTEST` is described at the end of this section.

`FEXACT2X2` procedure

Does Fisher's exact test for 2×2 tables (M.S. Ridout & M.W. Patefield).

Option

`PRINT` = *string tokens*	Controls printed output (`probabilities`, `tables`); default `prob`

Parameters

TABLE = *tables* or *variates*	The numbers in each 2×2 table, ordered row by row or column by column
PROBABILITIES = *variates*	Saves the probabilities for each table in a variate of length 6 (to store in positions 1, 3 and 5 one-tailed, two-tailed calculated as twice the one-tailed probability, and as the sum of the probabilities of all tables with probability less than that of the observed table with the corresponding mid-p values stored in positions 2, 4 and 6)

The TABLE parameter of the PROCEDURE supplies the four numbers that comprise the 2×2 table, either as a 2×2 Genstat table, with no margins, or as a variate consisting of the four numbers ordered either row by row or column by column. The procedure calculates the one-tailed significance level that is produced by the exact test. The mid-p value, which includes only half the probability of the observed table, is also calculated. See Hirji, Tan & Elashoff (1991) for a discussion of mid-p values. Several methods have been proposed for calculating a two-tailed significance level, two of which are implemented in the procedure. The first method simply doubles the one-tailed significance level whereas the second method calculates the cumulative probability of all outcomes that are no more probable than the observed table. See Yates (1984) for discussion of these and other methods. The procedure also calculates mid-p values corresponding to each of the two-tailed significance levels. The various probabilities can be saved, in a variate of length six, using the PROBABILITIES parameter.

The procedure has a single option PRINT to control printed output. By default PRINT=probabilities. There is also another setting tables which causes the procedure to display all 2×2 tables with margins that are the same as the observed table together with their probabilities of occurrence under the null hypothesis of no association and the cumulative probabilities calculated from both tails. This display was proposed by Hill (1984).

Fisher's exact test for the smoking data is shown in Example 2.9.2a. The two-tailed significance values are equivalent to the probability given by the chi-square test, and generate the same conclusions.

Example 2.9.2a

```
  11  "Use Fisher's exact test."
  12  FEXACT2X2 Counts

Fisher's exact test
===================

 One-tailed significance level   0.111
                   Mid-P value   0.096

 Two-tailed significance level
     Two times one-tailed significance level   0.223
                                Mid-P value   0.193
     Sum of all outcomes with Prob<=Observed   0.202
                                Mid-P value   0.186
```

CHIPERMTEST procedure

Performs a random permutation test for a two-dimensional contingency table (L.H. Schmitt, M.C. Hannah & S.J. Welham).

Options

PRINT = *string tokens*	Output required (summary, observed, expected);

	default `summ`
`PLOT` = *string token*	What to plot (`histogram`); default `hist`
`METHOD` = *string token*	Method for calculating chi-square (`pearson`, `maximumlikelihood`); default `pear`
`NTIMES` = *scalar*	Number of permutations to make; default 999
`SEED` = *scalar*	Seed for the random number generator used to make the permutations; default 0 continues from the previous generation or (if none) initializes the seed automatically

Parameters

`DATA` = *tables*	Table containing observed data
`CHISQUARE` = *scalars*	Saves the observed chi-square value
`CHIPERMUTED` = *variates*	Saves the chi-square values from the permuted data sets
`PROBABILITY` = *scalars*	Saves the probability value from the test

`CHIPERMTEST` uses a random permutation test to calculate the significance probability of the chi-square test. The permutations simulate the random distribution of table values that may occur in tables that have the same overall distribution of numbers over the columns, and over the rows, as in the original table. We can assess the significance of the chi-square statistic given by the observed table, by seeing where it lies in the distribution of statistics that we obtain from the permuted data.

The `NTIMES` option specifies how many permutations are done (default 999). The `SEED` option supplies the seed that is used in the `RANDOMIZE` directive to generate the permutations. The default of zero continues the existing sequence of random numbers if `RANDOMIZE` has already been used in the current Genstat job. If `RANDOMIZE` has not yet been used, Genstat picks a seed at random.

The `DATA` parameter supplies the observed data values, in a table with two classifying factors. The `CHISQUARE` can save the chi-square statistic calculated from the `DATA` table (in a scalar). The `CHIPERMUTED` can save the chi-square statistics calculated from the permuted data sets (in a variate), and the `PROBABILITY` parameter can save the significance probability from the permutation test (in a scalar).

The `PRINT` option controls the output, with the following settings:

`summary`	prints a summary, containing the chi-square statistic, the minimum and maximum statistics calculated from the permuted data sets, and the probability (default);
`observed`	prints the `DATA` table; and
`expected`	prints the expected values for tables with the same overall distribution of numbers over rows and over columns, but no interaction between the row and column factors (i.e. in a table where the rows and columns are independent).

By default, `CHIPERMTEST` plots a histogram showing the distribution of statistics obtained from the permuted data sets, with the chi-square statistic from the observed data superimposed as a vertical line. You can suppress this by setting option `PLOT=*`.

As in the `CHISQUARE` procedure (2.9.1), the `METHOD` option controls whether the chi-square statistic is calculated by the usual Pearson approximation or by maximum likelihood.

Example 2.9.2b shows a permutation test for the smoking data. The probability is similar to that given by the chi-square test in Example 2.9.1, and again leads to the same conclusion.

Example 2.9.2b

```
13  " Do a permutation test."
14  CHIPERMTEST [PLOT=*] Counts
```

```
Contingency table permutation test
==================================

* MESSAGE: Default seed for random number generator used with value 510012

2 * 2 contingency table Counts
Pearson chi-square 1.73
Range of values from 999 permutations (0.00, 14.97)
Probability 0.196
```

2.9.3 McNemar's test

MCNEMAR procedure

Performs McNemar's test for the significance of changes (R.W. Payne & D.A. Murray).

Options

PRINT = *string tokens*	Controls printed output (test, table); default test
METHOD = *string token*	Type of test required (twosided, greaterthan, lessthan); default twos

Parameters

Y1 = *factors* or *tables*	Factor containing the responses obtained before the treatment (with 1 indicating a positive response) or two-by-two table (classified by factors representing the two occasions of testing) summarizing the responses before and after treatment
Y2 = *factors*	Factor containing the responses obtained after the treatment (need not be specified if Y1 is a table)
STATISTIC = *scalars*	Saves the test statistic
PROBABILITY = *scalars*	Saves the probability value

McNemar's test is useful for analysing studies where subjects are assessed before and after a treatment. The response on each occasion is assumed to be categorized by a factor with two levels. Usually level 1 represents a *negative* response, and level 2 a *positive* response. The test assesses the consistency of the responses on the two occasions. By default the test is assumed to be two-sided (that is, changes in the overall response from level 1 to level 2 or from level 2 to level 1 are equally of interest). However, you can set the METHOD option to greaterthan for a one-sided test of the null hypothesis that the number of level 2 responses is not increasing (i.e. that the overall response is not becoming more positive), or to lessthan for a test of the null hypothesis that the number of level 2 responses is not decreasing.

The data for the test can be supplied as two variates (one for each occasion) using the Y1 and Y2 parameters. Positive responses are represented by the value one, and other values are taken to indicate negative responses. (So the variates might be formed from logical tests, for example using the .EQ. or .EQS. operators.) If Y1 or Y2 are restricted the test is made on only the units not excluded by the restriction. Alternatively, you can set Y1 to a two-by-two table classified by a factor representing the assessments before the treatment and another representing the assessments after the treatment.

In its original form, the test leads to a chi-square test (see the *Method* Section in the description of MCNEMAR in Part 3 of the *Reference Manual* for details). However, this may be inaccurate when there are small numbers of subjects. Consequently Genstat also provides an exact probability (based on the binomial distribution). The value of the statistic can be saved using the STATISTIC parameter, and the exact probability can be saved using the PROBABILITY

parameter.

Printed output is controlled by the PRINT option, with settings:

test	to print the test statistic and probabilities, and
table	to print the table of responses.

The default is PRINT=test.

Example 2.9.3 analyses an example from Siegel (1956) page 65. This assesses whether the type of person (adult or child) with whom children first initiate contact each day at a nursery school changes between their first and thirteenth day. The table shows that 14 children have changed their "object of initiation" from child to adult, while for 4 children it has changed from adult to child. McNemar's test shows that this represents a significant change.

Example 2.9.3

```
  2  FACTOR  [LABELS=!t('adult','child')] First,Thirtieth
  3  TABLE   [CLASSIFICATION=First,Thirtieth; VALUES=4,14,4,3] Object
  4  PRINT   Object; DECIMALS=0

                 Object
   Thirtieth      adult       child
       First
       adult          4          14
       child          4           3

  5  MCNEMAR [METHOD=greaterthan] Object

McNemar's test
==============

Statistic:               4.500
Chi-square probability:  0.017
Exact probability:       0.015
```

Methods for determining sample sizes for McNemar's test are described in 4.12.8.

2.9.4 Cochran's Q test

QCOCHRAN procedure

Performs Cochran's *Q* test for differences between related samples (D.A. Murray).

Options

PRINT = *string token*	Controls printed output (test); default test
METHOD = *string token*	Form of the test (exact, chisquare); default exac for small samples, otherwise chis
GROUPS = *factor*	Defines the groups if there only one variable supplied for the DATA
STATISTIC = *scalar*	Scalar to save the Q value
PROBABILITY = *scalar*	Scalar to save the probability for the Q Test
MAXTIME = *scalar*	Defines a limit for the maximum time for calculating the exact test; default * i.e. no limit.

Parameter

DATA = *variates*	List of related samples, or variate containing all the samples (the GROUPS option must then be set to indicate the variable recorded in each unit belongs)

Cochran's *Q* test is an extension to the McNemar test for related samples that provides a method for testing for differences between three or more matched sets of frequencies or proportions. The matching samples can be based on *k* characteristics of *N* individuals that are associated with the response. Alternatively *N* individuals may be observed under *k* different treatments or conditions (e.g. different questions or one question at different times).

The data must be supplied as dichotomous variables containing 0 to represent failure (or absence), and 1 to represent success (or presence). The variables can be stored in separate variates and the DATA parameter set to list them all. Alternatively, all the data can be stored in a single variate, and the GROUPS option set to a factor to indicate which variable is recorded in each unit of the variate. (QCOCHRAN then assumes that the individuals are recorded in the same order for each variable.)

In its original form, the test leads to a chi-square test (see the *Method* Section in the description of QCOCHRAN in Part 3 of the *Reference Manual* for details). However, this may be inaccurate when there are small numbers of subjects or samples. Consequently QCOCHRAN also provides an exact probability (based on the exact distribution of *Q* under a permutation model). The form of the test can be set to either chi-square or exact by using the METHOD option. The default is to use the exact test if the number of values in the samples is less than 4 and the product of this value with the number of samples is less than 24, otherwise the chi-square method is used. The time and memory required for the exact calculation can become impractible as the number of samples and values increases. So the chi-square approximation should be used for large problems. The MAXTIME option can be used to set a limit on the time (in seconds) to be used to calculate the exact probabilty; if this is time exceeded, the computation is terminated.

The *Q* statistic can be saved using the STATISTIC parameter, and the probability can be saved using the PROBABILITY parameter. By default QCOCHRAN prints the *Q* value and its probability, but you can set option PRINT=* to suppress these.

Example 2.9.4

```
  2  " Responses by housewives under 3 types of interview. Data from Siegel
 -3    (1956), Nonparametric Statistics for the Behavioural Sciences, p.164."
  4  VARIATE  [VALUES=0,1,0,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1] Response1
  5  &        [VALUES=0,1,1,0,0,1,1,1,0,0,1,1,1,1,1,1,1,1] Response2
  6  &        [VALUES=0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,1,0,0] Response3
  7  QCOCHRAN Response1,Response2,Response3

Cochran's Q test
================

Q statistic 16.667, probability < 0.001
```

2.9.5 The Cochran-Mantel-Haenszel test

CMHTEST procedure

Performs the Cochran-Mantel-Haenszel test (D.A. Murray).

Options

PRINT = *string token*	Controls printed output (test); default test
CLASSIFICATION = *factors*	Classifying factors for a DATA variate or classifying factors for the *R*×*C* tables in a DATA table
CONTINUITY = *string token*	Continuity correction for 2×2×*K* Mantel-Haenszel test (correct, none); default corr
CIPROBABILITY = *scalar*	Size of confidence interval for common odds ratio in 2×2×*K* tables; default 0.95

Parameters

DATA = *tables* or *variates*	Data values
STATISTIC = *scalars*	Save the test statistic
PROBABILITY = *scalars*	Save the probability for the test
ODDSRATIO = *scalars*	Save the common odds ratio for the 2×2×*K* table case
LOWER = *scalars*	Save lower limit of the confidence interval of odds ratio
UPPER = *scalars*	Save upper limit of the confidence interval of odds ratio

Procedure CMHTEST performs the Cochran-Mantel-Haenszel test for average partial association between two nominal variables adjusting for control variables. The data are represented by a series of *K* (*R*×*C*) contingency tables, where *K* represents the strata for the control variables. If there are two or more control variables then these are combined to form a single factor (K) with a level for every combination of the control factors. For the case where there are two dichotomous variables of interest, i.e. a series of *K* (2×2) tables, CMHTEST calculates the Mantel-Haenszel chi-square statistic, and an overall estimate of relative risk as described in Mantel & Haenszel (1959). Otherwise the Generalized Cochran-Mantel-Haenszel test is used, as in Landis *et al*. (1978).

The data can be supplied as a table using the DATA parameter where the first two classifying factors of the table indicate the variables of interest, and the remaining factors are combined to form a factor with a level for every combination of the remaining factors. If the first two classifying factors are not the ones of interest, then the CLASSIFICATION option can be used to supply the names of the classifying factors to use. The data can also be supplied in variates, with the CLASSIFICATION option set to the classifying factors and the first two factors in the list indicating the variables of interest. For a series of *K* (2×2) tables the CONTINUITY option can be used to control whether to apply a continuity correction to the Mantel-Haenszel chi-square test.

The PRINT option controls printed output, with settings:

test	the test statistic and probability, also the common odds ratio and confidence interval when there are *K* (2×2) tables

A 95% confidence interval is calculated for the common odds ratio, but this can be changed by setting the CIPROBABILITY option to the required value (between 0 and 1).

The test statistic can be saved using the STATISTIC parameter, and the probability can be saved using the PROBABILITY parameter. For a series of *K* (2×2) tables the odds ratio, lower and upper odds-ratio confidence interval can be saved with the ODDSRATIO, LOWER and UPPER parameters respectively.

Example 2.9.5

```
  2  " Women with epidermoid and undifferentiated pulmonary carcinoma:
 -3    assess association between carcinoma and smoking, adjusted for
 -4    age and occupation (data from Mantel & Haenszel 1959). "
  5  FACTOR  [LEVELS=2; LABELS=!t('Pulmonary carinoma','Controls')] Cases
  6  FACTOR  [LEVELS=2; LABELS=!t('Smoker','Nonsmoker')] Smoke
  7  FACTOR  [LEVELS=4; LABELS=!t('under 45','45-54','55-64','over 65')] Age
  8  FACTOR  [LEVELS=3; LABELS=!t('Housewives','White-collar','Other')]\
  9          Occupation
 10  TABLE   [CLASS=Cases,Smoke,Age,Occupation] Pulmonary
 11  READ    Pulmonary

   Identifier   Minimum      Mean   Maximum    Values   Missing
    Pulmonary    0.0000     6.521     49.00        48         0    Skew

 14  CMHTEST Pulmonary
```

```
Mantel-Haenszel test
--------------------

Test statistic:    30.66 on 1 d.f. (with continuity)
Probability:       < 0.001
Common odds ratio: 10.68
95% confidence interval for common odds ratio (4.162, 27.42)
```

2.9.6 Cochran-Armitage chi-square test for trend

CATRENDTEST procedure

Calculates the Cochran-Armitage chi-square test for trend (A.I. Glaser).

Option

PRINT = *string token*	Output required (test); default test

Parameters

DATA = *tables*	Table containing observed data
TREND = *factors*	Dimension of the table representing the trend; can default if only one dimension of size greater than 2
CHISQUARE = *scalars*	Saves the chi-square for trend
PROBABILITY = *scalars*	Saves the probability value for trend
DEVCHISQUARE = *scalars*	Saves the chi-square for deviations from a linear trend
DEVDF = *scalars*	Saves the degrees of freedom for the chi-square for deviations
DEVPROBABILITY = *scalars*	Saves the probability value for the chi-square for deviations

The CATRENDTEST procedure calculates the Cochran-Armitage chi-square test for trend. Categorical data can be collected and categorized by explanatory factors (such as dosage or treatment level), and any analysis will try to indicate relationships between the response (binary) factor and explanatory factors. The Cochran-Armitage chi-square test calculates a chi-square statistic on 1 degree of freedom for a linear trend in the responses. The data are represented by a ($2\times K$ or $K\times 2$) contingency table, where K represents the explanatory factor (known as the *trend*).

The DATA parameter supplies the data values in a two-way table. The TREND parameter can be set to a factor to indicate which dimension of the table represents the trend; if this is omitted CATRENDTEST assumes that the trend is in the dimension with more than 2 rows or columns (the other dimension must have exactly 2 rows or columns).

By default CATRENDTEST prints the results of tests for trend and for deviation from a trend (chi-square values, degrees of freedom and probabilities), but you can suppress these by setting option PRINT=*.

Parameters CHISQUARE, PROBABILITY, DEVCHISQUARE, DEVDF and DEVPROBABILITY allow the results to be saved (in scalars).

Example 2.9.6 analyses data from Table 15.1 of Armitage, Berry & Matthews (1994). This records the numbers of patients accepting or declining invitations to attend screening mammography, according to the length of time since their doctor's appointment. The test shows that there genuinely does seem to be a linear trend of acceptance with time, and no significant deviations from a linear relationship.

Example 2.9.6

```
 2  FACTOR [LEVELS=2; LABELS=!T('Yes','No')] Attendence
 3  FACTOR [LABELS=!t('<6 months','6-12 months','1-2 years','>2 years')] Time
 4  TABLE  [CLASS=Time,Attendence; VALUES=59,97,10,31,12,36,5,28] Patient
 5  CATRENDTEST Patient; TREND=Time

Cochran-Armitage test for trend
===============================

Trend: chi-square 8.18 on 1 d.f., probability 0.004
Deviation from trend: chi-square 0.74 on 2 d.f., probability 0.691
```

2.10 Six sigma

Genstat has wide range of facilities to support the six-sigma approach to quality improvement. This section describes procedures for assessing the output of a process, to see if it is operating within its limits of expected variation. These include control charts for means, standard deviations or ranges of a continuous measurement (2.10.1), c or u charts for numbers (2.10.5) of defective items in a sample, p or np charts for proportions of defective items (2.10.4), exponentially weighted moving-average charts (2.10.3) and CUSUM (i.e. cumulative sum) tables (2.10.2). It also describes the calculation of *capability statistics* to assess how well the distribution of the output from a process lies within its specification limits (2.10.6).

There is also full statistical backup for wider-ranging investigations. Useful commands (with section numbers in brackets for those described in the *Guide*) include the following:

NORMTEST	performs tests of univariate and/or multivariate Normality (2.2.11)
WSTATISTIC	calculates the Shapiro-Wilk test for Normality (2.2.11)
TABSORT	sorts tables to put margins are in ascending or descending order for display as a Pareto chart (1:4.11.5)
AFRESPONSESURFACE	uses the BLKL algorithm to construct response-surface designs (4.9.14)
AGBOXBEHNKEN	generates Box-Behnken designs (4.9.12)
AGCENTRALCOMPOSITE	generates central composite designs (4.9.11)
AGFACTORIAL	generates minimum aberration complete and fractional factorial designs (4.9.2)
AGDESIGN	selects from a set of standard designs including factorials with interactions confounded with blocks (4.9.3)
AGFRACTION	generates fractional factorial designs
AGMAINEFFECT	generates designs to estimate main effects of two-level factors i.e. Plackett-Burman designs (4.9.13)
FKEY	forms design keys for balanced designs with several error terms, allowing for confounded and aliased treatments (4.13.6)
ANOVA	analysis of variance for balanced designs (4.1.2)
AUNBALANCED	analysis of variance for unbalanced designs (4.8.1)
FIT	fits a linear, generalized linear, generalized additive, or generalized nonlinear model (3.1.2)
FITCURVE	fits a standard nonlinear regression model (3.7.1)
FITNONLINEAR	fits a nonlinear regression model or optimizes a function (3.8.2)
RQUADRATIC	fits a quadratic surface and estimates its stationary point
REML	fits an unbalanced linear mixed model and estimates

	variance components (5.3.1)
YTRANSFORM	estimates the parameter lambda from various single-parameter transformations, includling power (Box-Cox), modulus, folded power, Guerrero-Johnson, Aranda-Ordaz and power logit

2.10.1 Control charts for mean, standard deviation or range

SPSHEWHART procedure

Plots control charts for mean and standard deviation or range (A.F. Kane & R.W. Payne).

Options

PRINT = *string token*	What to print (warnings); default * i.e. nothing
PLOT = *string token*	Type of chart to plot to accompany the chart of sample means (range, standarddeviation); default stan
METHOD = *string token*	Type of control limits (probability, sigma); default sigm
TOLERANCEMULTIPLIER = *scalar*	Multiplier to use to test whether to use mean sample size for control limits; default 1
PROBABILITY = *scalars*	Probability value(s) to use to calculate control limits when METHOD=probability; default 0.01, 0.025
WINDOWS = *scalar*	Which high-resolution graphics windows to use; if unset SPSHEWHART automatically sets up two windows containing the upper and lower halves of the screen
SCREEN = *string token*	Whether or not to clear the graphics screen before plotting (clear, keep); default clea

Parameters

DATA = *variates* or *pointers*	Data measurements
SAMPLES = *factors* or *scalars*	Factor identifying samples or scalar indicating the size of each sample
MEAN = *scalars*	Sets or saves the sample mean value
SIGMA = *scalars*	Sets or saves the sample standard deviation

SPSHEWHART plots the standard charts devised by Shewhart (1931) for the control of manufacturing processes. The data values consist of samples of measurements made on successive occasions, which are specified by the DATA and SAMPLES parameters. DATA can be set to a variate containing the measurement and SAMPLES to a factor identifying the samples. Alternatively, if the samples are all of the same size and occur in the DATA variate one sample at a time, you can set SAMPLES to a scalar indicating the size of each sample. Finally, if the samples are in separate variates, you can set DATA to a pointer containing the variates (SAMPLES is then unset).

Two charts are produced. The first chart plots the mean of each sample. It also contains a centre line (indicating a target value) and lines representing upper and lower control limits (bounding the zone outside which the process is said to be out of control). The MEAN and SIGMA parameters allow you to supply values for the process mean and standard deviation if these are available either as targets or from previous observations. If they are unset, or if they are set to scalars containing missing values, the values are calculated from the data values, as described at the end of this Subsection. The traditional chart (and the one that is most popular in the USA) sets the centre line at the mean, and the control limits at 3 × SIGMA and −3 × SIGMA from the mean. The alternative (often used in the UK and requested by setting option METHOD to

`probability`) sets control limits according to probability values. Usually the lower control limit is at the equivalent deviate value for a probability of 0.01, and the upper limit is at the value for 0.99 (see the *Methods* Section). There may also be intermediate warning limits, usually at 0.025 and 0.975. These are the default probabilities used by `SPSHEWHART`, but you can set the `PROBABILITY` option to a variate containing one or two values to define other limits. (If the values are p_1 and p_2, the limits are then for probabilities p_1, p_2, 100−p_2, 100−p_1.)

The control limits relevant to each batch will depend on the sample sizes. The `TOLERANCE` option determines whether an average sample size is used if the individual sizes are not exactly equal: this will happen unless either

```
MIN(sample_size) * TOLERANCE < MEAN(sample_size)
```

or

```
MEAN(sample_size) * TOLERANCE < MAX(sample_size)
```

The second chart is either for the standard deviation of values in each sample or for their range, according to the setting of the `PLOT` option (by default `PLOT=standarddeviation`). Traditionally, before computers were available, the range chart was more popular. However, it is less sensitive than the standard deviation, particularly for larger samples, and `SPSHEWHART` does not permit range charts if any sample size is greater than 25.

You can set `PRINT=warnings` to list any batches that are outside the control limits; by default these are suppressed. As usual, the `WINDOWS` option specifies which high-resolution graphics windows to use for the plots. If this is unset, `SPSHEWHART` automatically sets up and uses two windows containing the upper and lower halves of the screen. The `SCREEN` option controls whether or not to clear the graphics screen before plotting the charts.

Example 2.10.1a plots traditional control charts (Figure 2.10.1a) for the mean and standard deviation for some measurements of the diameter of the insides of samples of piston rings (Montgomery 1985, page 207).

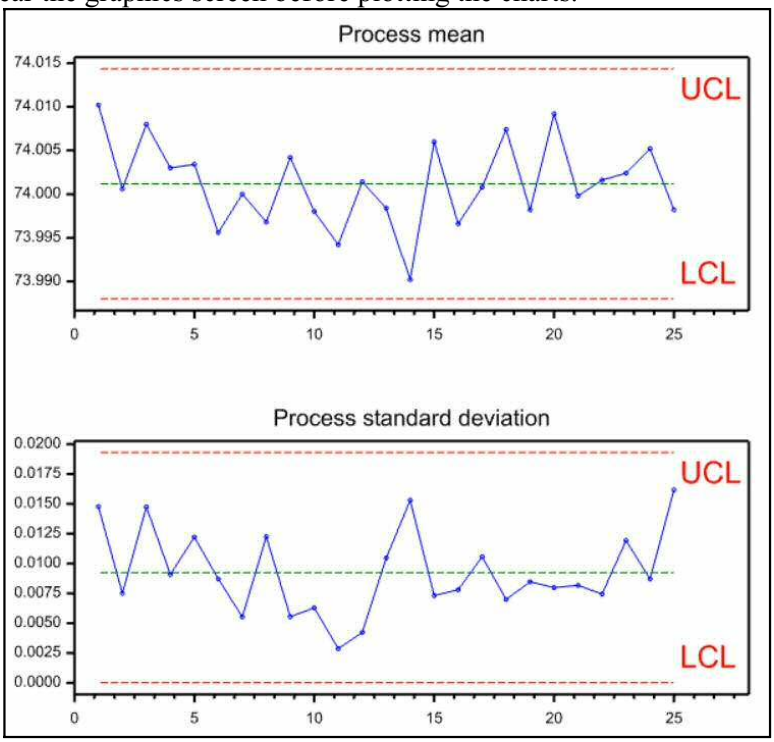


Figure 2.10.1a

Example 2.10.1a

```
 2  VARIATE     Diameter
 3  READ        Diameter

  Identifier   Minimum      Mean   Maximum    Values   Missing
    Diameter     73.97     74.00     74.03       125         0

29  SPSHEWHART [PRINT=warnings] Diameter; SAMPLES=5
```

If the number in each sample is one, the chart of the means is known as an individuals chart. There is now no within-sample replication, so the range chart instead presents a moving range displaying the range between each sample and the previous sample. Similarly, the standard deviations are calculated between each sample and its previous sample.

Example 2.10.1b shows an individuals chart and moving range chart (Figure 2.10.1b) for some measurements of the viscosity of aircraft primer paint (Montgomery 1985, page 242).

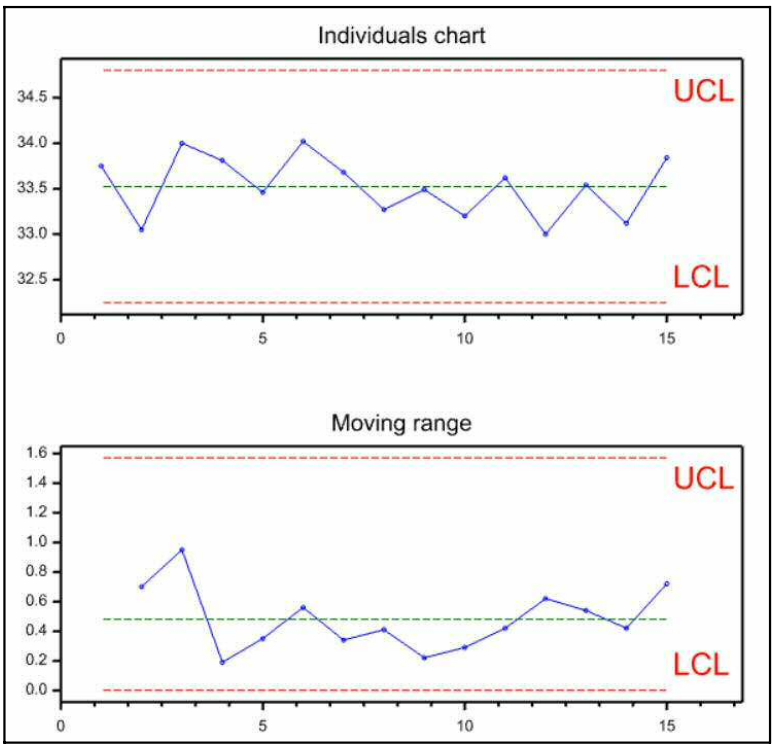


Figure 2.10.1b

Example 2.10.1b

```
30  VARIATE     [VALUES=33.75,33.05,34.00,33.81,33.46,\
31                      34.02,33.68,33.27,33.49,33.20,\
32                      33.62,33.00,33.54,33.12,33.84] Viscosity
33  SPSHEWHART [PRINT=warnings; PLOT=range] Viscosity; SAMPLES=1
```

SPSHEWHART follows the standard methods as described for example by Nelson (1982), Montgomery (1985) or Ryan (1989). If required, the mean is estimated in the usual way by the average of the sample values. Likewise, the standard deviation is estimated by the average of the standard deviations of the samples, divided by a bias correction constant c_4:

$$c_4 = \sqrt{(2/n)} \times \texttt{GAMMA}(n/2) \,/\, \texttt{GAMMA}((n-1)/2)$$

where n is the sample size.

First of all we describe the calculations with METHOD=sigma. In the mean chart, the centre line is at the mean (i.e. MEAN), and the control limits at MEAN + 3 × SIGMA and MEAN − 3 × SIGMA. In the range chart, if the standard deviation has been supplied, the centre line is at d_2 × SIGMA and the control limits at D_1 × SIGMA and D_2 × SIGMA; if the standard deviation has not

been supplied, the centre line is at the mean of the ranges observed in the samples, and the control limits are at $D_3 \times$ SIGMA and $D_4 \times$ SIGMA. (See Appendix VI of Montgomery, or Nelson 1982 Table 1 for values of the constants d_2, and $D1-D_4$.) In the standard-deviation chart, the centre line is at SIGMA $\times c_4$ (so that it exhibits the same bias as the sample standard deviations) and the control limits are at $3 \times$ SIGMA $\times \sqrt{(1 - c_4^2)}$ above and below the centre line.

For METHOD=probability, the centre lines are unaffected. However, the control limits for the means chart are now at

```
EDNORMAL(PROBABILITY) * SIGMA / SQRT(N)
```

above and below the centre line. For the range chart, the control limits are at

```
SIGMA * EDSRANGE(PROBABILITY; 1000; N)
```

and

```
SIGMA * EDSRANGE(1-PROBABILITY; 1000; N)
```

(where the high value 1000 used for the degrees of freedom of the Studentized range is to obtain the value for the Normal range). For the standard-deviation chart, the control limits are at

```
SQRT(EDCHI(PROBABILITY; N-1) / (N-1))
```

and

```
SQRT(EDCHI(1-PROBABILITY; N-1) / (N-1))
```

2.10.2 CUSUM tables

SPCUSUM procedure

Prints CUSUM tables for controlling a process mean (A.F. Kane & R.W. Payne).

Options

REFERENCEVALUE = *scalars*	Specifies the upper and then the lower reference values, or just one of these if they are both the same; default 0.5
THRESHOLD = *scalars*	Detection thresholds, upper and then the lower, or just one of these if they are both the same; default 5
HEADSTART = *scalars*	Headstart values, upper and then the lower, or just one of these if they are both the same; default 0

Parameters

DATA = *variates* or *pointers*	Data measurements
SAMPLES = *factors* or *scalars*	Factor identifying samples or scalar indicating the size of each sample
MEANTARGET = *scalars*	Specifies the target value for the sample means
SIGMA = *scalars*	Specifies or saves the standard deviation of the observations

---

SPCUSUM prints cumulative sum (or *CUSUM*) charts (see for example Section 5.3 of Ryan 1989). These are more sensitive than Shewhart charts (2.10.1) for detecting small shifts in the process. The data values consist of samples of measurements made on successive occasions, which are specified by the DATA and SAMPLES parameters. DATA can be set to a variate containing the measurement and SAMPLES to a factor identifying the samples. Alternatively, if the samples are all of the same size and occur in the DATA variate one sample at a time, you can set SAMPLES to a scalar indicating the size of each sample. Finally, if the samples are in separate variates, you can set DATA to a pointer containing the variates (SAMPLES is then unset).

The chart displays columns containing:

1) the sample number;

2) the sample mean;
3) z, the devation of the mean from a target value, divided by its standard deviation;
4) SH, the upper *CUSUM*;
5) SL, the lower *CUSUM*.

An asterisk is printed alongside any values SH and SL that exceed a threshold value, indicating that the process is out of control.

The *CUSUM* values SH_i and SL_i for each sample i are calculated as

$$SH_i = z_i - k_u + SH_{i-1}$$
or $$= 0 \quad \text{if } z_i - k_u + SH_{i-1} < 0$$
$$SL_i = -z_i - k_l + SL_{i-1}$$
or $$= 0 \quad \text{if } -z_i - k_l + SL_{i-1} < 0$$

The target value is specified by the MEANTARGET parameter. The SIGMA parameter can be used to specify the standard deviation of the individual observations (which is required to calculate the standard deviation of the deviations of the sample means from the target value). It this is not set or if it is set to a missing value, the standard deviation is calculated using the within-sample replication, as the average of the standard deviations of the samples, divided by a bias correction constant c_4:

$$c_4 = \sqrt{(2/n)} \times \text{GAMMA}(n/2) \,/\, \text{GAMMA}((n-1)/2)$$

where n is the sample size. You can thus save the calculated standard deviation by setting SIGMA to a scalar containing a missing value.

The *reference values* k_u and k_l are specified by the REFERENCEVALUE option. If they are both the same, you need specify this only once. Their default is 0.5. Similarly the threshold value, or values, are specified by the THRESHOLD option; by default these take the value 5. The *CUSUM*s usually start at 0, but you can specify another value or values using the HEADSTART option.

Example 2.10.2 shows a CUSUM table based on data in Table 5.4 of Ryan (1989). However, as the values in the table are given to only 2 decimal places the sample means differ slightly from those in the book.

Example 2.10.2

```
  2  VARIATE x
  3  READ x

   Identifier   Minimum      Mean   Maximum    Values   Missing
            x    -2.530    0.2523     2.300        80         0

 24  SPCUSUM [THRESHOLD=4] x; SAMPLES=4; MEANTARGET=0; SIGMA=1

CUSUM table
===========

Sample   Average         z        SH        SL
     1    0.4050     0.810     0.310     0.0000
     2    1.1050     2.210     2.020     0.0000
     3   -0.0350    -0.070     1.450     0.0000
     4   -1.0450    -2.090     0.000     1.5900
     5   -0.0375    -0.075     0.000     1.1650
     6   -0.6850    -1.370     0.000     2.0350
     7   -0.8500    -1.700     0.000     3.2350
     8    0.8125     1.625     1.125     1.1100
     9    0.6250     1.250     1.875     0.0000
    10   -0.3775    -0.755     0.620     0.2550
    11    0.5375     1.075     1.195     0.0000
    12    0.1850     0.370     1.065     0.0000
    13    0.3325     0.665     1.230     0.0000
    14    0.3250     0.650     1.380     0.0000
    15    0.4450     0.890     1.770     0.0000
    16    0.2150     0.430     1.700     0.0000
    17    0.8825     1.765     2.965     0.0000
    18    0.1650     0.330     2.795     0.0000
    19    0.9500     1.900     4.195*    0.0000
```

```
    20    1.0900     2.180     5.875*    0.0000
Values of SH and SL over the threshold are marked by asterisks.
```

2.10.3 Moving-average control charts

SPEWMA procedure

Plots exponentially weighted moving average control charts (A.F. Kane & R.W. Payne).

Options

PRINT = *string token*	What to print (warnings); default * i.e. nothing
TOLERANCEMULTIPLIER = *scalar*	Multiplier to use to test whether to use mean sample size for control limits; default 1
WEIGHT = *scalar*	Weight parameter used in the calculation of the exponentially weighted moving-average statistic; default 0.25
NSIGMA = *scalar*	Number of multiples of sigma to use for control limits; default 3
WINDOW = *scalar*	Which high-resolution graphics window to use; default 3
SCREEN = *string token*	Whether or not to clear the graphics screen before plotting (clear, keep); default clea

Parameters

DATA = *variates* or *pointers*	Data measurements
SAMPLES = *factors* or *scalars*	Factor identifying samples or scalar indicating the size of each sample
MEAN = *scalars*	Sets or saves the sample mean value
SIGMA = *scalars*	Sets or saves the sample standard deviation

Exponentially weighted moving-average control charts provide another effective means of detecting small shifts in a process (see Ryan 1989, Section 5.5). The data values consist of samples of measurements made on successive occasions, which are specified by the DATA and SAMPLES parameters. DATA can be set to a variate containing the measurement and SAMPLES to a factor identifying the samples. Alternatively, if the samples are all of the same size and occur in the DATA variate one sample at a time, you can set SAMPLES to a scalar indicating the size of each sample. Finally, if the samples are in separate variates, you can set DATA to a pointer containing the variates (SAMPLES is then unset).

The chart plots a statistic w whose value for sample t is a weighted average of the mean of sample t, and the value of the statistic for sample $t-1$:

$$w_t = r_t \times xbar_t + (1 - r) \times w_{t-1}$$

where *xbar* is the variate of sample means, and r is the weighting parameter specified by the WEIGHT option of the procedure with default 0.25. (Notice that the statistic involves all the previous means, but with exponentially decreasing weights.)

The position of the central line for the chart is specified, in a scalar, by the MEAN parameter. If this is not set, or if it is set to a scalar containing a missing value, the overall mean of the samples is used. (So you can save the calculated mean by setting MEAN to a scalar containing a missing value.) There are also control lines $-nsigma \times \text{var}(w)$ and $+nsigma \times \text{var}(w)$, where *nsigma* is specified by the NSIGMA option (default 3) and var(w) is the variance of the statistic w. For sample t, this is

$$(3 \times sigma / \sqrt{(\text{REP}_t)}) \times \sqrt{((r/(2 - r)) \times (1 - (1 - r)^{2t}))}$$

where REP is a variate containing the number of observations in each sample, and *sigma* is the standard deviation of a single observation. The SIGMA parameter can be used to supply a value for *sigma*. It this is not set or if it is set to a missing value, *sigma* is calculated using the within-sample replication as the average of the standard deviations of the samples, divided by a bias correction constant c_4:

$c_4 = \sqrt{(2/n)} \times \texttt{GAMMA}(n/2) / \texttt{GAMMA}((n-1)/2)$

The TOLERANCE option determines whether an average replication is used if the replication of the individual samples is no exactly equal: this will happen unless either

```
MIN(REP) * TOLERANCE < MEAN(rep)
```

or

```
MEAN(rep) * TOLERANCE < MAX(rep)
```

You can set PRINT=warnings to list any batches that are outside the control limits; by default these are suppressed. As usual, the WINDOWS option specifies which high-resolution graphics window to use for the plot (default 3), and the SCREEN option controls whether or not to clear the graphics screen before plotting the charts.

Example 2.10.3 illustrates the use of the procedure using data in Table 7.6 and Figure 7-8b of Montgomery (1985). The resulting chart is in Figure 2.10.3.

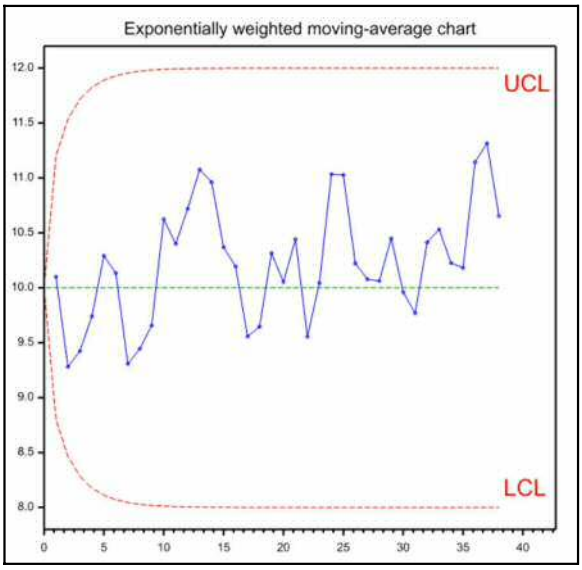


Figure 2.10.3

Example 2.10.3

```
2  VARIATE [VALUES=10.5,6.0,10.0,11.0,12.5,9.5,6.0,10.0,10.5,14.5,\
3          9.5,12.0,12.5,10.5,8.0,9.5,7.0,10.0,13.0,9.0,\
4          12.0,6.0,12.0,15.0,11.0,7.0,9.5,10.0,12.0,8.0,\
5          9.0,13.0,11.0,9.0,10.0,15.0,12.0,8.0] xbar_t
6  SPEWMA  [WEIGHT=0.2] xbar_t; SAMPLES=1; MEAN=10; SIGMA=2
```

2.10.4 Control charts for proportions of defective items

SPPCHART procedure

Plots *p* or *np* charts for binomial testing for defective items (A.F. Kane & R.W. Payne).

Options

PRINT = *string token*	What to print (warnings); default * i.e. nothing
PLOT = *string token*	Type of chart to plot (p, np); default p
METHOD = *string token*	Method to use to obtain the control limits (complementaryloglog, given, logit, probit, untransformed); default untr
TOLERANCEMULTIPLIER = *scalar*	Multiplier to use to test whether to use mean sample size

	for control limits; default 1
WINDOW = *scalar*	Which high-resolution graphics window to use; default 3
SCREEN = *string token*	Whether or not to clear the graphics screen before plotting (clear, keep); default clea

Parameters

NDEFECTIVE = *variates*	Number of defective items
NTESTED = *scalars* or *variates*	Number of items tested
CENTRELINE = *scalars*	Sets or saves centre line
LOWERCONTROLLIMIT = *scalars* or *variates*	Sets or saves lower control limit
UPPERCONTROLLIMIT = *scalars* or *variates*	Sets or saves upper control limit

The *p* and *np* charts evaluate testing schemes in which items in successive batches are classified as either good or defective. The number of defective items in each batch is specified, in a variate, by the NDEFECTIVE parameter. The NTESTED parameter supplies the number of items in each batch – this can be a scalar if the batches are all of the same size, otherwise it is a variate.

The PLOT option controls the type of chart: the *p* chart plots the proportion of defective items while the *np* chart (which is most useful each batch of items has the same total size) plots the number of defective items.

The charts contain not only the observed numbers or proportions but also a centre line (indicating a target value) and lines showing upper and lower control limits (bounding the zone outside which the process is said to be out of control). The control limits relevant to each batch will depend on the batch sizes. The TOLERANCE option determines whether an average total size is used if the individual totals are not exactly equal: this will happen unless either

```
MIN(NTESTED) * TOLERANCE < MEAN(TESTED)
```

or

```
MEAN(TESTED) * TOLERANCE < MAX(NTESTED)
```

The METHOD option specifies how the various lines are to be defined, with the following settings. They are defined below for a *p* chart. For an *np* chart, the values are simple multiplied by the batch size(s).

untransformed	this is the default setting, and requests the method conventionally used in SPC. The centre line is at p = (total number defective) / (total number tested) and the limits are at $p \pm 3 \times \sqrt{(p / (1-p))}$
given	specifies that the values are supplied by the CENTRELINE, LOWERCONTROLLIMIT and UPPERCONTROLLIMIT parameters.
logit	obtains the values as the batch mean +/– three times its standard error as estimated on the logit scale of a generalized linear model (with binomial distribution).
probit	obtains the values as the batch mean +/– three times its standard error as estimated on the probit scale of a generalized linear model
complementaryloglog	obtains the values as the batch mean +/– three times its standard error as estimated on the complementary-log-log scale of a generalized linear model.

For settings of METHOD other than given, the CENTRELINE, LOWERCONTROLLIMIT and UPPERCONTROLLIMIT parameters can be used to save the centre line and limits.

You can set PRINT=warnings to list any batches that are outside the control limits; by default these are suppressed. As usual, the WINDOW option specifies which high-resolution graphics window to use for the plot, and the SCREEN option controls whether or not to clear the graphics screen before plotting.

Example 2.10.4 produces a p chart (Figure 2.10.4) for the proportions of defective cans in successive samples of size 50 (see Montgomery 1985, page 152). Notice that two of the samples contain unacceptably high proportions of defects.

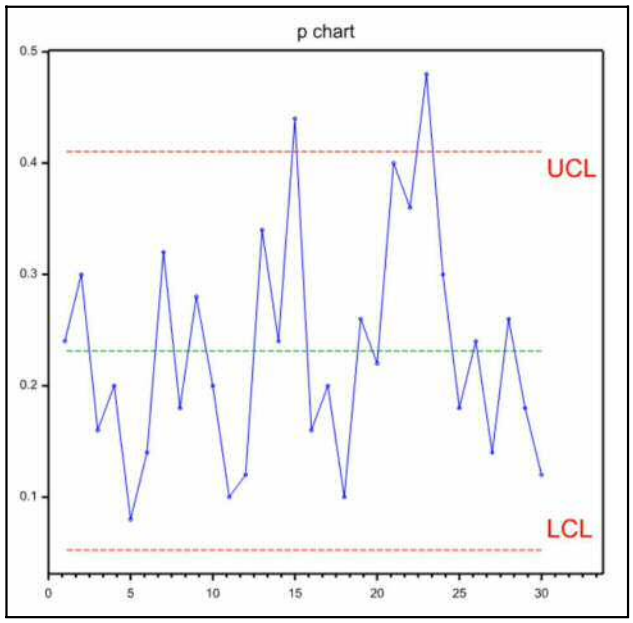


Figure 2.10.4

Example 2.10.4

```
  2  VARIATE  [VALUES=12,15,8,10,4,7,16,9,14,10,5,6,17,12,22,\
  3           8,10,5,13,11,20,18,24,15,9,12,7,13,9,6] Cans
  4  SPPCHART [PRINT=warnings] Cans; NTESTED=50

***** Warning: the process is out of control.

Samples 15 and 23 are outside the control limits.
```

2.10.5 Control charts for numbers of defects

SPCCHART procedure

Plots *c* or *u* charts representing numbers of defective items (A.F. Kane & R.W. Payne).

Options

PRINT = *string token*	What to print (warnings); default * i.e. nothing
PLOT = *string token*	Type of chart to plot (c, u); default c
METHOD = *string token*	Method to use to obtain the control limits (given, loglinear, untransformed); default untr
TOLERANCEMULTIPLIER = *scalar*	Multiplier to use to test whether to use mean sample size for control limits; default 1
WINDOW = *scalar*	Which high-resolution graphics window to use; default 3
SCREEN = *string token*	Whether or not to clear the graphics screen before plotting (clear, keep); default clea

Parameters

NDEFECTIVE = *variates*	Number of defective items
NTESTED = *scalars* or *variates*	Number of items tested

CENTRELINE = *scalars*	Sets or saves centre line
LOWERCONTROLLIMIT = *scalars* or *variates*	Sets or saves lower control limit
UPPERCONTROLLIMIT = *scalars* or *variates*	Sets or saves upper control limit

The *c* and *u* charts evaluate testing schemes in which numbers of defects are measured in successive batches of items. The number of defects per batch is specified, in a variate, by the NDEFECTIVE parameter. The NTESTED parameter supplies the number of items in each batch – this can be a scalar if the batches are all of the same size, otherwise it is a variate.

The PLOT option controls the type of chart: the *c* chart plots number of defects per batch, while the *u* chart plots the number of defects per item.

The charts contain not only the observed numbers of defects but also a centre line (indicating a target value) and lines showing upper and lower control limits (bounding the zone outside which the process is said to be out of control). The control limits relevant to each batch in a *u* chart will depend on the batch sizes. The TOLERANCE option determines whether an average sample size is used if the individual sizes are not exactly equal: this will happen unless either

```
MIN(NTESTED) * TOLERANCE < MEAN(TESTED)
```

or

```
MEAN(TESTED) * TOLERANCE < MAX(NTESTED)
```

The METHOD option specifies how the various lines are to be defined, with the following settings.

untransformed	this is the default setting, and requests the method conventionally used in SPC. For a *c* chart, the centre line is at *c* = (total number defects) / (number of batches) and the limits are at $c \pm 3 \times \sqrt{(c)}$. For a *u* chart, the centre line is at *u* = (total number defects) / (total number of items) and the limits are at $u \pm 3 \times \sqrt{(u/n)}$.
given	specifies that the values are supplied by the CENTRELINE, LOWERCONTROLLIMIT and UPPERCONTROLLIMIT parameters.
loglinear	obtains the values by fitting a generalized linear model with Poisson distribution and log link.

For settings of METHOD other than given, the CENTRELINE, LOWERCONTROLLIMIT and UPPERCONTROLLIMIT parameters can be used to save the centre line and limits.

You can set PRINT=warnings to list any batches that are outside the control limits; by default these are suppressed. As usual, the WINDOW option specifies which high-resolution graphics window to use for the plot, and the SCREEN option controls whether or not to clear the graphics screen before plotting.

The two types of chart are illustrated in Example 2.10.5 and Figures 2.10.5a and 2.10.5b.

Example 2.10.5

```
  2  " c chart: data from Montgomery (1985) page 174."
  3  VARIATE  [VALUES=21,24,16,12,15,5,28,20,31,25,20,24,16,\
  4           19,10,17,13,22,18,39,30,24,16,19,17,15] Nonconformities
  5  SPCCHART [PRINT=warnings] Nonconformities; NTESTED=100

Warnings
========

Sample 6 is below the lower control limit.
```

```
Sample 20 is above the upper control limit.
```

```
  6  " u chart: data from  Montgomery (1985) page 181."
  7  VARIATE  [VALUES=10,12,8,14,10,16,11,7,10,15,9,5,7,11,12,6,8,10,7,5]\
  8           Nonconformities
  9  SPCCHART [PRINT=warnings; PLOT=u] Nonconformities; NTESTED=5
```

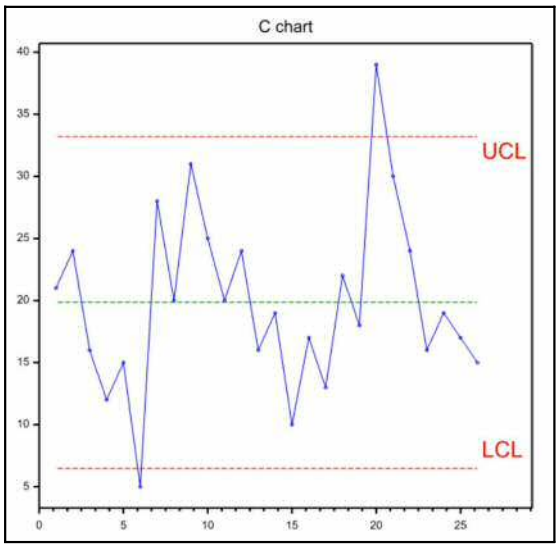


Figure 2.10.5a

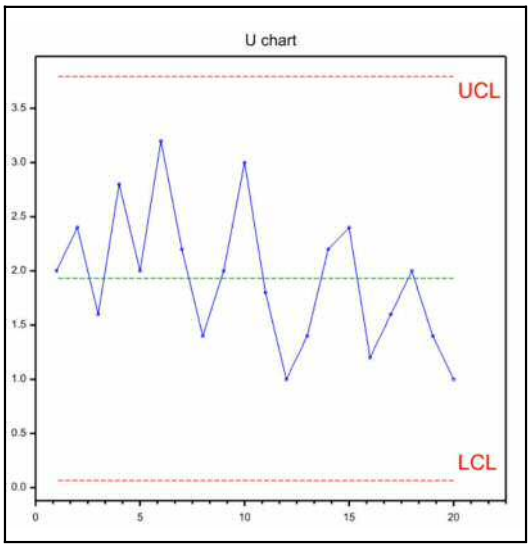


Figure 2.10.5b

2.10.6 Capability statistics

SPCAPABILITY procedure

Calculates capability statistics (R.W. Payne).

Option

PRINT = *string tokens*	Controls output (cpk, ppk, histogram); default cpk, ppk

Parameters

DATA = *variates* or *pointers*	Data measurements
SAMPLES = *factors* or *scalars*	Factor identifying samples or scalar indicating the size of each sample
LOWERLIMIT = *scalars*	Specifies the lower specification limit for each set of data
UPPERLIMIT = *scalars*	Specifies the upper specification limit for each set of data
CPK = *scalars*	Saves the index C_{pk}
PPK = *scalars*	Saves the index P_{pk}

Capability statistics assess the extent to which the output of a process lies within its specification limits. The data values consist of samples of measurements made on successive occasions, which are specified by the DATA and SAMPLES parameters. DATA can be set to a variate containing the measurement and SAMPLES to a factor identifying the samples. Alternatively, if the samples are all of the same size and occur in the DATA variate one sample at a time, you can set SAMPLES to a scalar indicating the size of each sample. Finally, if the samples are in separate variates, you

can set DATA to a pointer containing the variates (SAMPLES is then unset). The LOWERLIMIT parameter supplies the lower specification limit of the process, and the UPPERLIMIT parameter supplies the upper limit.

There are two indexes that can be calculated. The index C_{pk} is the minimum of the two quantities C_{pl} and C_{pu}. These are defined as

C_{pl} = (LOWERLIMIT - *mean*) / (3 × *sigma*)

C_{pu} = (UPPERLIMIT - *mean*) / (3 × *sigma*)

where *sigma* is the within-sample standard deviation (see for example Ryan 1989, Chapter 7). The alternative index, P_{pk}, is the minimum of the two quantities P_{pl} and P_{pu}. These have similar definitions to C_{pl} and C_{pu}, except that *sigma* now also includes the between-sample variation.

The PRINT option controls which of these are printed, with settings cpk and ppk. There is also a setting histogram, which plots a histogram of the data together with vertical lines indicating the lower and upper limits. By default PRINT=cpk,ppk. The indexes can also be saved, in scalars, using the parameters CPK and PPK.

Example 2.10.6 produces capability statistics for the samples of piston rings examined in Example 2.10.1a

Example 2.10.6

```
  34  SPCAPABILITY Diameter; SAMPLES=5; LOWERLIMIT=73.95; UPPERLIMIT=74.05

Process capability
==================

Data variate: Diameter

Index Cpk        1.656
Lower index Cpl  1.735
Upper index Cpu  1.656

Index Ppk        1.613
Lower index Ppl  1.691
Upper index Ppu  1.613
```

2.11 Ecological data

This section describes the facilities in Genstat for displaying, summarizing and modelling ecological data. ECDIVERSITY calculates diversity indices, which provide a summary statistic for the diversity in a community (2.11.1). ECABUNDANCE allows the distribution of species abundance data to be visualized using rank-abundance or k-dominance plots (2.11.2). A range of distributions and models to describe species abundance data can be fitted using ECFIT (2.11.3). An alternative approach to describing species abundance data is to try to predict how available niche space might be divided amongst species and then evaluate whether the observed species abundances match these expected abundances. ECNICHE can be used to generate relative abundances for different niche-based models (2.11.4). ECRAREFACTION compares the species richness of communities can be compared by rarefaction (2.11.5). This method estimates the number of species that would be found if sampling effort was reduced, i.e. to "rarefy" sample data to the same number of individuals as in another sample to provide a direct comparison. ECACCUMULATION plots species accumulation curves. These show the rate at which new species are found within a community, and can be extrapolated to provide an estimate of species richness (2.11.6). This does a permutation test based on the ranks of similarities between sampling units. LORENZ plots the Lorenz curve, which provides a graphical representation of the inequality of a sample of numbers, and calculates the Gini and asymmetry coefficients (2.11.8). Also, described elsewhere, ECANOSIM compares communities between sites by a nonparametric analysis of similarities known as ANOSIM (6.1.6).

2.11.1 Diversity measures

A diversity index is a measure of the diversity of a population of individuals within a community or area that is used in the analysis of data such as multi-species ecological data. There are two components to diversity: richness and evenness. Richness is the measure of the number of species or items within a sample where the more species or items in a community or area the higher the diversity (or greater richness). Evenness is a measure of the relative abundance of the different species or items within a community or area. The more nearly equal the species relative abundances the higher the diversity. Many indices have been proposed as measures of diversity. The ECDIVERSITY procedure can calculate some of the best known indices.

ECDIVERSITY procedure

Calculates measures of diversity with jackknife or bootstrap estimates (D.A. Murray).

Options

PRINT = *string tokens*	Controls printed output (index, estimate); default inde
INDEX = *string token*	Controls the type of measurement to be calculated (hshannon, qstatistic, simpsonyule, bergerparker, ibrillouin, ebrillouin, dmcintosh, emcintosh, evar, logseriesalpha, lognormallambda, jshannon, margalef, isimpson, richness); default hsha
GROUPS = *factor*	Defines the groups if there is more than one sample
BMETHOD = *string token*	Controls whether to use the bootstrap or jackknife method (jackknife, bootstrap); default jack for multiple samples and boot for individual samples
NBOOT = *scalar*	Number of times to resample in bootstrap; default 100
SEED = *scalar*	Seed for random number generator for bootstrap; default 0
CIPROBABILITY = *scalar*	Probability for the confidence interval produced by either jackknife or bootstrap method; default 0.95

Parameters

INDIVIDUALS = *variates*	Number of individuals per species
SPECIES = *variates*	Number of species
SAVE = *variate* or *pointer*	Saves the diversity indices

The numbers of individuals per species are specified using the INDIVIDUALS parameter. The SPECIES parameter specifies a variate containing the number of species for the associated number of individuals denoted in the corresponding element of INIDIVIDUALS. SPECIES can be omitted if each of the values in INDIVIDUALS corresponds to one species. The GROUPS option can be used to calculate measures of diversity for different samples. The SAVE parameter allows the diversity indices to be saved in a variate or in a pointer to a set of variates for each group.

The INDEX option can be used to calculate one or more of the diversity measures, as follows. The log series α index is estimated by fitting a log series model using the ECFIT procedure. The log-Normal λ is the ratio of the S^* and σ parameters estimated by fitting a Poisson-log-Normal distribution using the ECFIT procedure.

The Q statistic is calculated by:

$$Q = (0.5 \times n_{R1} + \sum_{r=R1+1 \ldots R2-1} \{ n_r \} + 0.5 \times n_{R2}) / \log(R2 / R1),$$

where n_r is the total number of species with abundance r, $R1$ and $R2$ are the 25% and 75%

quartiles, n_{R1} is the number of species where R1 lies, and n_{R2} is the number of species where R2 lies.

The Shannon-Weiner index is evaluated by:

$$H' = -\sum_i (n_i / N) \times \log(n_i / N)$$

where n_i are the individuals, N is total number of individuals.

The Shannon-Weiner evenness (Pielou J) is given by

$$J' = H' / \log(S)$$

where H' is the Shannon index and S is the total number of species.

The Brillouin index is given by

$$HB = (\log(N!) - \sum_i \{\log(n_i!)\}) / N$$

where n_i is the individual in species i and N is total number of individuals.

The Brillouin evenness index is then calculated by

$$E = HB / HBmax$$

and

$$HBmax = 1 / N \times \log(N! / ((N/S)!^{S-r} \times ((N/S)+1)!^{r})$$

where N/S is the integer of N/S and $r = N - S(N/S)$

Simpsons index D is calculated by

$$D = \sum_i \{n_i \times (n_i - 1)\} / (N \times (N - 1))$$

and is expressed in the output as both $1-D$ and $1/D$

The Margalef index is:

$$Dmn = (S - 1) / \log(N)$$

where S is total number of species and N is total number of individuals.

McIntosh's measure of diversity is expressed as

$$D = (N - \sqrt{(\sum_i \{n_i^2\} } / (N - \sqrt{(N)})$$

and the evenness measure is given by

$$E = (N - \sqrt{(\sum_i \{n_i^2\}) } / (N - N / \sqrt{(S)})$$

where n_i is the individual in species i and N is total number of individuals.

The Berger-Parker index is

$$d = Nmax / N$$

where $Nmax$ is the number of individuals in the most abundant species.

The Evar (Smith and Wilson's evenness) index is evaluated by

$$Evar = 1 - 2 / (\pi \times \arctan(\sum_i \{ \log(n_i) - \sum_j \{ \log(n_j) \} \}^2 / S))$$

where n_i and n_j are the number of individuals in species i and j respectively, and S is the total number of species

The `PRINT` option controls printed output, with settings:

`index`	the index of diversity or evenness,
`estimate`	bootstrap or jackknife estimate with confidence limits for the statistic.

The `BMETHOD` option can be used to select either the bootstrap or jackknife (for multiple samples) method to produce an estimate of the diversity measure with an associated confidence interval. To produce a bootstrap or jackknife estimate for multiple samples, each sample must contain the same number of values where each element corresponds to the same species within each sample. For the calculation of the bootstrap confidence intervals of the diversity measures, the `NBOOT` option specifies how many bootstrap samples to take (default 100). The probability level for the confidence interval can be set by the `CIPROBABILITY` option; by default 0.95. The `SEED` option specifies the seed to use in the random number generator used to construct the bootstrap samples. The default value of zero continues an existing sequence of random numbers or, if the generator has not yet been used in this run of Genstat, it initializes the generator automatically.

Example 2.11.1 uses `ECDIVERSITY` to calculate the Shannon and Simpson indices.

Example 2.11.1

```
  2  " Data from censuses of bird territories in woodlands in Killarney,
 -3    Ireland. (see Maguarran, A.E., 2004, Measuring Biological Diversity,
 -4    Blackwell, pages 237-240)"
  5  FACTOR  [NVALUES=69; LEVELS=3; VALUES=23(1...3);\
  6          LABELS=!t('Derrycunnihy oakwood','Muckross yew wood',\
  7                    'Sitka spruce plot')] Location
  8  VARIATE [VALUES=35,26,25,21,16,11,6,5,3,3,3,3,3,2,2,2,1,1,1,1,0,0,0,\
  9                   9,20,10,21, 5,14,0,3,2,6,9,2,0,0,0,6,0,0,0,1,1,1,0,\
 10                  14,10, 0,30, 4, 6,0,0,7,3,0,0,0,0,0,0,0,0,0,0,0,0,1]\
 11          Territories
 12  ECDIVERSITY [INDEX=hshannon,isimpson; GROUPS=Location] Territories

Diversity indices for Group Derrycunnihy oakwood
------------------------------------------------

                  Diversity Index
Shannon-Weiner H            2.408
Simpson 1/D                 8.717


Diversity indices for Group Muckross yew wood
---------------------------------------------

                  Diversity Index
Shannon-Weiner H            2.346
Simpson 1/D                 9.181


Diversity indices for Group Sitka spruce plot
---------------------------------------------

                  Diversity Index
Shannon-Weiner H            1.715
Simpson 1/D                 4.505


Diversity indices for Total
---------------------------

                  Diversity Index
Shannon-Weiner H            2.410
Simpson 1/D                 8.498
```

2.11.2 Plotting species abundance data

ECABUNDANCEPLOT procedure

Produces rank/abundance, *ABC* and *k*-dominance plots (D.A. Murray).

Options

PRINT = *string token*	Controls printed output (summary); default summ
PLOT = *string token*	Controls the type of plot (rankabundance, kdominance, abc); default rank, kdom
GROUPS = *factor*	Defines the groups if there is more than one sample

Parameters

INDIVIDUALS = *variates*	Number of individuals per species
SPECIES = *variates*	Number of species
BIOMASS = *variates*	Biomass data for each species for an ABC plot

A rank/abundance plot (or Whittaker plot) can be used to visualize species abundance

distributions. In this plot, the number of individuals of each species are sorted in descending order, and the proportion of the total number of individuals for each species is then plotted on the log scale against the species rank. The shape of the rank/abundance plot can provide an indication of dominance or evenness, for example, steep plots signify assemblages with high dominance and shallower slopes indicate higher evenness.

A *k*-dominance plot displays the cumulative proportion abundance against the log species rank. For this type of plot, more elevated curves represent less diverse assemblages.

An abundance/biomass comparison (or *ABC* curve) is an adaption of the *k*-dominance curve where two measures of abundance are plotted: the number of individuals and biomass data. This plot is useful to explore the level of disturbance affecting assemblage.

The numbers of individuals per species are specified using the `INDIVIDUALS` parameter. The `SPECIES` parameter specifies a variate containing the number of species for the associated number of individuals specified in the corresponding element of `INDIVIDUALS`. `SPECIES` can be omitted if each of the values in `INDIVIDUALS` corresponds to one species. The `GROUPS` option can be used to plot the relative abundance for different samples.

The `PLOT` option can be used to produce a rank/abundance plot, *k*-dominance curve and an *ABC* curve. You can display a summary of the number of individuals and species by setting the option `PRINT=summary`. Selecting this option will also display the *W* statistic for an *ABC* curve. The *W* statistic for an *ABC* curve is defined by

$$W = \sum_i (B_i - A_i) / (50 \times (S - 1))$$

where S is the total number of Species, B_i is the biomass value of each species rank i, and A_i is the abundance value of each species rank i.

Example 2.11.2 uses `ECABUNDANCE` to produce the rank abundance and k-dominance plots shown in Figures 2.11.2a and 2.11.2b respectively for the data in Example 2.11.1.

Example 2.11.2

```
  13  ECABUNDANCEPLOT [GROUPS=Location] Territories

Summary of the number of individuals and species
------------------------------------------------

                              Number of individuals  Number of species
Group Derrycunnihy oakwood                      170                 20
   Group Muckross yew wood                      110                 15
```

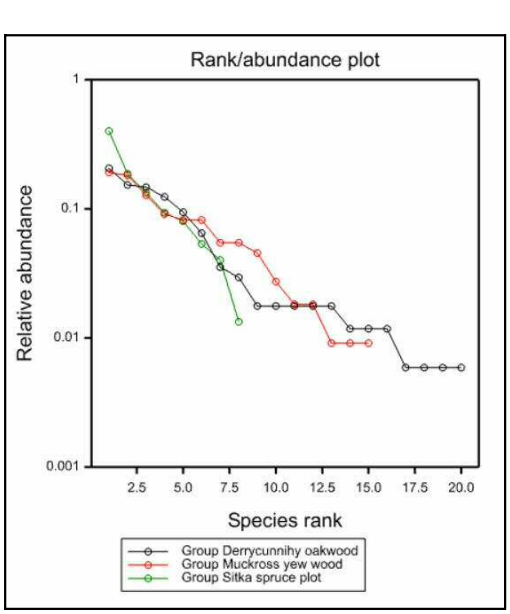


Figure 2.11.2a

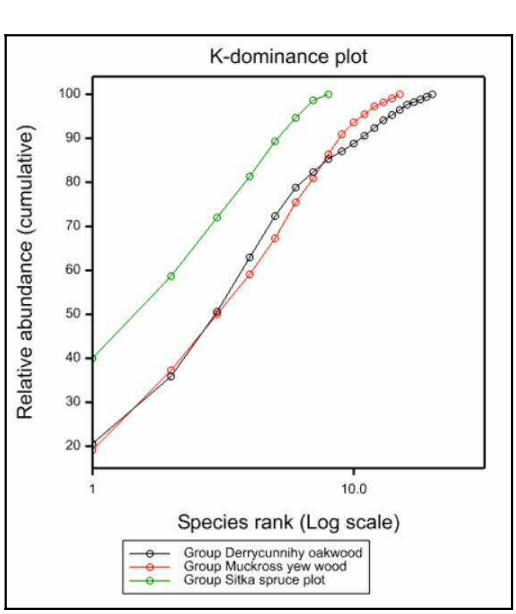


Figure 2.11.2b

2.11.3 Species abundance models

ECFIT procedure

Fits models to species abundance data (D.A. Murray).

Options

PRINT = *string tokens*	Controls printed output (summary, estimates, fittedvalues); default summ, esti
MODELTYPE = *string token*	The model or distribution fitted to the data (logseries, plognormal, negativebinomial, geometric, zipf, mandelbrotzipf); default logs
GROUPS = *factor*	Defines the groups if there is more than one sample
LOGBASE = *string token*	Log base to use to form the octaves for the logseries, Poisson log-Normal and negative binomial distributions (two, ten); default two
PLOT = *string token*	Plots the fitted values (fittedabundance, rankabundance); default fitt

Parameters

INDIVIDUALS = *variates*	Number of individuals per species
SPECIES = *variates*	Number of species
ESTIMATES = *variates*	Saves the model estimates
EGROUPS = *factors*	Saves the grouping of the estimates

ECFIT provides a range of distributions and models that can be used to describe species abundance data. The numbers of individuals per species are specified using the INDIVIDUALS parameter. The SPECIES parameter specifies a variate containing the number of species for the associated number of individuals specified in the corresponding element of INDIVIDUALS. SPECIES can be omitted if each of the values in INDIVIDUALS corresponds to one species. The GROUPS option can be used to fit models for different samples.

The distribution or model to be fitted to the data is specified by the MODELTYPE option. For the log series, Poisson log-Normal and negative binomial distributions the species abundance data are grouped into "octaves" using a logarithmic scale. These distributions are then fitted using the DISTRIBUTION directive using the octave classes. The log base for forming the octaves for the log series, Poisson log-normal and negative binomial distributions can be supplied using the LOGBASE option. The default is to use log base 2, i.e. representing doubling in species abundance.

For the geometric series the abundances are ranked from the most to least abundant, and fitted using FITNONLINEAR where the series is given by

$$a_i = N / (1 - (1 - k)^S) \times k \times (1 - k)^{i-1}$$

where a_i is the total number of individuals in the ith species, N is the total number of individuals, k is the proportion of remaining niche space, and $1 / (1 - (1 - k)^S)$ is a constant that ensures $\sum_i a_i = N$.

The Zipf and Zipf-Mandelbrot models are also fitted using FITNONLINEAR. The Zipf model is given by

$$A_i = A_1 \times i^{-\gamma}$$

where A_1 is the fitted abundance of the most abundant species, and γ is a constant representing the average probability of the appearance of a species.

The Zipf-Mandelbrot is an extension of the Zipf model and is expressed as

$$A_i = A_1 \times (i + \beta)^{-\gamma}$$

where A_1 and gamma are as before, and beta is a constant.

The parameter estimates from the fitted model can be saved using the ESTIMATES parameter. The EGROUPS factor saves a factor indicating the group strucure of the estimates.

The PRINT option controls printed output, with settings:

summary	summary of the analysis,
estimates	the parameter estimates,
fittedvalues	the fitted values.

The PLOT option can be used to produce a plot of the fitted model or distribution. For the geometric series, Zipf and Zipf-Mandelbrot models, the fitted model can also be displayed on a rank/abundance plot on the log-scale.

Example 2.11.3 uses ECFIT to fit the log series model.

Example 2.11.3

```
  2  " Frequency distribution of individuals per species in a light trap
 -3    sample of Macrolepidoptera collected at Rothamsted Research.
 -4    (see Lewis & Taylor 1967, Introduction to Experimental Ecology,
 -5    Academic Press, page 244) "
  6  VARIATE [VALUES=1...18,20...23,25,28,29,33,34,38,39,40,42,48,\
  7          51,52,53,58,61,64,69,73,75,83,87,88,105,115,131,139,\
  8          173,200,223,232,294,323,603,1799] Individuals
  9  VARIATE [VALUES=37,22,12,12,11,11,6,4,3,5,2,4,2,3,2,2,4,2,4,4,\
 10          1,1,1,2,2,2,2,1,1,3,2,2,1,1,1,1,1,2,1,1,1,1,1,1,1,1,\
 11          1,1,1,1,1,1,1,1,1] NumSpecies
 12  ECFIT  [PRINT=summary,estimates,fitted; PLOT=*; MODELTYPE=logseries]\
 13          Individuals; SPECIES=NumSpecies

Summary of analysis
-------------------

Model: Logseries
Pr(X=r) = alpha*(x**r)/r, r>0, 0<x<1
Deviance: 14.15 on 10 d.f.
Number of individuals:         6815
Number of species:              197


Estimates of parameters
-----------------------

Parameter    Estimate        s.e.
alpha          34.741       0.867
x               0.997


Fitted values
-------------

Octave    Observed      Fitted
    1        37.00       34.62
    2+       22.00       28.71
    4+       24.00       25.91
    8+       32.00       24.26
   16+       23.00       22.79
   32+       21.00       20.81
   64+       20.00       17.67
  128+        8.00       12.89
  256+        6.00        6.96
  512+        2.00        2.13
 1024+        1.00        0.23
 2048+        1.00        0.02
```

2.11.4 Niche-based models

ECNICHE procedure

Generates relative abundance of species for niche-based models (D.A. Murray).

Options

PRINT = *string token*	Controls printed output (model, expected, replications); default mode, expe
MODELTYPE = *string token*	The niche model (powerfraction, fixedratio, preemption, randomfraction, macarthurfraction); default powe
METHOD = *string token*	Whether to use the Fortran DLL to calculate the relative abundance (dll, commands); default * uses the DLL in Windows implementations, and commands for other platforms
POWER = *scalar*	Power for the Power fraction model, must be in the range 0 to 1
URATIO = *scalar*	Ratio for the fixed ratio model
SEED = *scalar*	Seed for random number generator for the random division of the niche space; default 0
PLOT = *string token*	Plots the average relative abundance (relativeabundance); default rela

Parameters

NREPLICATES = *scalars*	Number of replications
NSPECIES = *scalars*	Number of species
EXPECTED = *variates*	Saves the expected average relative abundance
SDEXPECTED = *variates*	Saves the standard deviation for the expected mean relative abundance

The relative abundance of species can be modelled using deterministic models, such as the log series, or by stochastic models based on assumed patterns of resource use, such as niche-based models. ECNICHE can be used to simulate relative abundances (proportional abundance of species) for niche-apportionment, where species are considered to be associated with different processes of niche division, and sequential breakage models. Niche apportionment and sequential breakage models generate relative abundances using a two step process. In the first step the target niche (the total niche space in the very first step) is divided using a given probability distribution, for example, a random selection using the uniform distribution. In the second step a new target niche space is selected using a probabilistic weighting. The process is then repeated by dividing a selected target niche and selecting a new niche for division. ECNICHE includes Tokeshi's (1993, 1996) niche apportionment models for the dominance preemption, random fraction, power fraction and MacArthur fraction. The dominance preemption model assumes that each species in turn preempts over half the remaining niche space and is dominant over all remaining species combined. The random faction model represents the situation where new species compete for the niche space of existing species, and takes a random proportion of the previously existing niche. Therefore, species with different niche sizes or abundances have the same chance of being selected for a subsequent niche division. In the power fraction model, the probability of selection is proportional to niche size (or abundance) raised to a power exponent k ($0 \leq k \leq 1$). In the MacArthur fraction model (broken-stick model) the probability of a niche being selected for division is related to its size. So, larger niches are more likely to be invaded by species. ECNICHE also provides the sequential

breakage model where the target niche is selected at random and then divided to produce two segments relative to a ratio such as 0.75:0.25.

The number of replications for the model are specified using the NREPLICATES parameter. The NSPECIES parameter specifies the number of species within the assemblage. The mean relative abundance of species and associated standard deviations can be saved using the EXPECTED and SDEXPECTED parameters respectively.

The model to use to generate the relative abundances for the species is specified by the MODELTYPE option. The power for the Power fraction model is specified using the POWER option, and must range between 0 and 1. For the sequential breakage model, the largest value of the ratio of division is specified using the URATIO option, and must range between 0.5 and 1. The SEED option specifies the seed to use in the random division of the niche space. The default value of zero continues an existing sequence of random numbers or, if the generator has not yet been used in this run of Genstat, initializes the generator automatically.

For a large number of replications the calculation of the relative abundance of species can be slow. For the PC Windows implementation, a Fortran DLL is available that uses the OWN calculate function. By default the procedures uses the DLL, however, you can choose to use the Genstat commands by setting option METHOD=commands.

The PRINT option controls printed output, with settings:

model	the niche model,
expected	the expected mean relative abundance,
replications	the relative abundances for each replication; this can produce a lot of output, so it is recommended that this be used only for monitoring.

By default PRINT=model,expected.

The PLOT option controls whether ECNICHE produces a plot of the average relative abundance on the log scale, as shown in Figure 2.11.4; the default PLOT=relativeabundance gives the plot.

Example 2.11.4 shows how to generate relative abundances for the data in Example 2.11.3 using Tokehi's power fraction model.

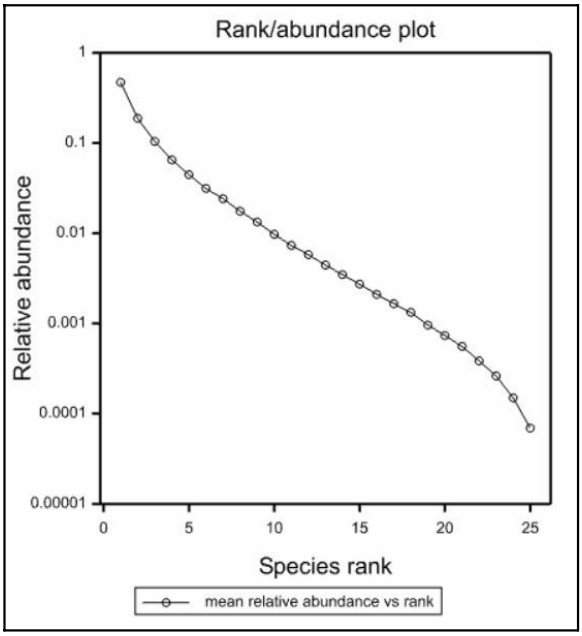


Figure 2.11.4

Example 2.11.4

```
  14  ECNICHE [MODELTYPE=power; POWER=0.2; SEED=2635] 250;25


Niche apportionment model
-------------------------

Model:  Power Fraction
Power:  0.2


Expected mean relative abundance
--------------------------------

Species rank Mean relative abundance Standard deviation
           1                 0.47096            0.20305
           2                 0.18762            0.08069
           3                 0.10393            0.05366
           4                 0.06503            0.03780
           5                 0.04456            0.03050
           6                 0.03130            0.02320
           7                 0.02413            0.01955
           8                 0.01750            0.01612
           9                 0.01331            0.01330
          10                 0.00971            0.01002
          11                 0.00731            0.00786
          12                 0.00579            0.00652
          13                 0.00443            0.00524
          14                 0.00348            0.00423
          15                 0.00273            0.00357
          16                 0.00210            0.00290
          17                 0.00166            0.00232
          18                 0.00133            0.00198
          19                 0.00096            0.00144
          20                 0.00073            0.00117
          21                 0.00056            0.00093
          22                 0.00038            0.00067
          23                 0.00026            0.00051
          24                 0.00015            0.00031
          25                 0.00007            0.00019
```

2.11.5 Rarefaction

ECRAREFACTION procedure

Calculates individual or sample-based rarefaction (D.A. Murray).

Options

PRINT = *string token*	Controls printed output (summary); default summ
METHOD = *string token*	Controls the type of rarefaction (individual, sample); default indi
PLOT = *string token*	Controls plot type (expected); default expe
SAMPLESIZES = *scalar* or *variate*	A scalar defining a step between sample sizes or number of samples to estimate the number of species; alternatively, a variate specifing the actual sample size values or number of samples
CIPROBABILITY = *scalar*	Probability for the confidence interval; default 0.95

Parameters

DATA = *variates*, *matrices* or *pointers*

For individual-based rarefaction, a variate containing the

	number of individuals for each species; for sample-based rarefaction, a pointer or matrix specifying the number of individuals for each species for different sites/samples
EXPECTED = *variates*	Saves the expected number of species at each sample size
VARIANCE = *variates*	Saves the variance for the expected number of species
LOWER = *variates*	Saves the lower confidence limit at each sample size
UPPER = *variates*	Saves the upper confidence limit at each sample size

Rarefaction is a method that can be used to estimate the number of species that would be found if sampling effort was reduced to a specified level. This then allows comparisons amongst communities where sampling effort is unequal. For individuals in a sample, individual-based rarefaction can be used to estimate the number of species that would be observed given a smaller number of individuals (Heck *et al*. 1975). Sample-based rarefaction can be used to estimate the expected number of species that would be observed given a smaller number of samples (Colwell *et al*. 2004). Rarefaction assumes that individuals have been sampled randomly and sample-based rarefaction assumes a random sample ordering. The method also assumes that the samples that are to be compared are not obtained by different collecting techniques or from communities that are intrinsically different.

For individual-based rarefaction, the number of individuals for each species are specified in a variate using the DATA parameter. For sample-based rarefaction, the data can be supplied using the DATA parameter either as a matrix where the rows contain the number of individuals for each species and the columns specify the different samples, or as a pointer to variates containing samples for the individuals for each species. The expected number of species and associated variance can be saved using the EXPECTED and VARIANCE parameters respectively. The LOWER and UPPER parameters can be used to save the lower and upper bounds for the confidence interval. The type of rarefaction (individual or sample-based) is specified using the METHOD option. For individual-based rarefaction the expected number of species in a sample of size n is calculated by:

$$E(S_n) = S - (1 / C(n, N)) \times \sum_i \{ C(n, N-N_i) \}$$

where N_i is the number of individuals in species i of the unrarefied sample, $C(n, N)$ is the number of combinations of n from N and $C(n, N-N_i)$ is the number of combinations of n from $N-N_i$. The variance, $var(S_n)$, is outlined in Heck *et al*. (1975).

Sample-based rarefaction is calculated by

$$t(h) = S_{obs} - \sum_{j=1...H} \{ a_{jh} \times s_j \} \quad \text{for } h = 1 \ldots H$$

where s_j is the number of species found in exactly j samples of a total of H samples, S_{obs} is defined by

$$S_{obs} = \sum_{j=1...H} \{ s_j \}$$

and the combinational coefficients a_{jh} are estimated by

$$a_{jh} = ((H - h)! \times (H - j)!) / ((H - h - j)! \times H!) \quad \text{for } j + h \le H$$

$$a_{jh} = 0 \quad \text{otherwise}$$

The variance is estimated by

$$var(h) = \sum \{ (1 - a_{jh})^2 \times s_j - t(h)^2 / S^{\sim}$$

where

$$S^{\sim} = S_{obs} + (H - 1) \times s_1^2 / (2 \times H \times s_2)$$

The SAMPLESIZES option specifies the sample sizes or number of samples for which the expected number of species is calculated. A scalar can be supplied to specify a step between each sample size, or a variate can be provided containing the actual sample sizes. By default the expected values are calculated for all possible sample sizes.

By default a summary is printed, giving the expected species richness, variance and

confidence limits, but you can set option `PRINT=*` to suppress this.

A plot of the expected number of species and confidence limits can be specified using the `expected` setting of the `PLOT` option. The probability level for the confidence intervals can be set by the `CIPROBABILITY` option; by default 0.95.

Example 2.11.5 shows an example of individual-based rarefaction. The results are saved, to produce the plot in Figure 2.11.5.

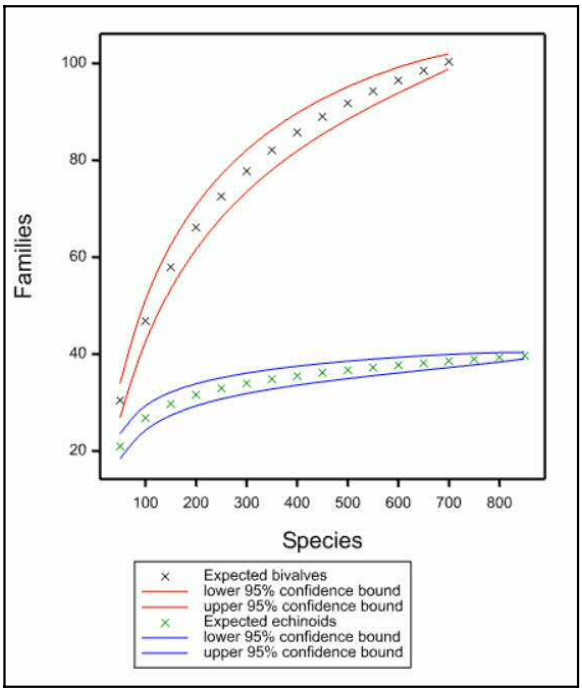


Figure 2.11.5

Example 2.11.5

```
  2  " Data from Siegel & German (1982). Biometrics, 38, 235-242.
 -3    Distribution of species within families of echinoids and bivalves."
  4  VARIATE       [VALUES=24(1),16(2),9(3),9(4),6(5),6(6),6(7),5(8),2(9),\
  5                12,4(13),2(14),15,16,3(17),20,22,2(29),35,55,99] Bivalves
  6  VARIATE       [VALUES=6(1),2,2,3,3,5,5,6,6,7,9,10,11,11,13,13,14,15,\
  7                23,24,25,25,26,29,32,33,33,36,36,42,49,58,61,86,134]\
  8                Echinoids
  9  ECRAREFACTION [PLOT=*; SAMPLESIZE=50] Bivalves; EXP=bexp; \
 10                LOWER=blow; UPPER=bupp

Individual-based rarefaction
----------------------------

 Sample size Expected Species    Variance    Lower CI    Upper CI
          50            30.51       7.570       26.98       34.03
         100            46.89      11.371       42.57       51.21
         150            57.97      12.620       53.42       62.53
         200            66.18      12.675       61.62       70.75
         250            72.60      12.151       68.14       77.07
         300            77.81      11.331       73.49       82.12
         350            82.14      10.347       78.02       86.26
         400            85.83       9.261       81.93       89.73
         450            89.02       8.102       85.37       92.67
         500            91.82       6.883       88.46       95.18
         550            94.31       5.610       91.28       97.35
         600            96.54       4.284       93.89       99.20
         650            98.56       2.901       96.38      100.74
         700           100.39       1.455       98.85      101.94

 11  ECRAREFACTION [PLOT=*; SAMPLESIZE=50] Echinoids; EXP=eexp; \
 12                LOWER=elow; UPPER=eupp

Individual-based rarefaction
----------------------------
```

```
 Sample size Expected Species    Variance    Lower CI    Upper CI
          50             21.02       4.107       18.42       23.62
         100             26.84       3.909       24.30       29.37
         150             29.75       3.502       27.35       32.15
         200             31.61       3.166       29.33       33.89
         250             32.95       2.903       30.77       35.13
         300             33.99       2.689       31.89       36.09
         350             34.84       2.502       32.81       36.87
         400             35.56       2.327       33.60       37.51
         450             36.19       2.153       34.31       38.07
         500             36.75       1.973       34.95       38.55
         550             37.26       1.783       35.55       38.98
         600             37.74       1.578       36.13       39.35
         650             38.18       1.357       36.68       39.67
         700             38.59       1.118       37.23       39.94
         750             38.98       0.859       37.79       40.17
         800             39.35       0.580       38.38       40.33
         850             39.71       0.278       39.03       40.38
```

```
 13  PEN    2,4; SYMBOL=0; METHOD=mono; LINE=1
 14  XAXIS  [RESET=yes] 1; TITLE='Species'
 15  YAXIS  [RESET=yes] 1; TITLE='Families'
 16  DGRAPH [WINDOW=1] bexp,blow,bupp,eexp,elow,eupp; \
 17         3(!(50,100...700)),3(!(50,100...850)); PEN=1,2,2,3,4,4;\
 18         DESCRIPTION='Expected bivalves','lower 95% confidence bound',\
 19         'upper 95% confidence bound','Expected echinoids',\
 20         'lower 95% confidence bound','upper 95% confidence bound'
```

2.11.6 Species accumulation curves

ECACCUMULATION procedure

Plots species accumulation curves for samples or individuals (D.A. Murray).

Options

PRINT = *string token*	Controls printed output (summary); default summ
CURVE = *string token*	Controls the type of species accumulation curve (collector, random, coleman); default coll
PLOT = *string token*	Controls plot type (sac); default sac
METHOD = *string token*	Controls collector curve when data supplied in variate or factor with groups (individual, sample); default samp
GROUPS = *factor*	Grouping factor for samples when data are supplied in variate of factor of individuals
NPERMUTATIONS = *scalar*	A scalar defining the number of permutations to be performed for the random method; default 100
SEED = *scalar*	Seed for random number generator; default 0
SCREEN = *string token*	Whether to clear screen before displaying the graph (keep, clear); defaul clea
WINDOW = *scalar*	Window for the graph; default 1
KEYWINDOW = *scalar*	Window number for the key (zero for no key); default 2
PEN = *scalar*	Pen number to draw the curve; default 1

Parameters

DATA = *variates*, *factors*, *matrices* or *pointers*
: For individual-based collector curves, a variate or factor containing the individuals in the order they were collected; for sample-based species accumulation curves, a pointer or matrix specifying the number of

	individuals for each species for different sites/samples
RICHNESS = *variates*	Saves the observed number of species for the collector method and the average or expected number of species at each sample size for the Coleman and random methods
VARIANCE = *variates*	Saves the variance for the richness (Coleman and random methods only)

Species accumulation curves show the rate at which new species are found within a community, and can be extrapolated to provide an estimate of species richness. The simplest curve is the *collectors* curve. This plots the cumulative number of species recorded as a function of sampling effort (i.e. number of individuals collected or cumulative number of samples). The order in which samples are included in a species accumulation curve will influence the overall shape. A smooth accumulation curve can be produced by repeating a process of randomly adding the samples to the accumulation curve and then plotting the mean of these permutations. ECACCUMULATION can also plot a *Coleman* curve (see Coleman *et al.* 1982). Here the expected number of species is calculated by

$$s_\alpha = S - \sum_{i=1\ldots S} (1 - \alpha)^{ni}$$

where S is the number of species, n^i is the number of individuals belonging to ith species and α is the relative area

$$\alpha = a / \sum a_k$$

The variance for the Coleman curve is estimated by

$$v_\alpha = \sum_{i=1\ldots S} (1 - \alpha)^{ni} - \sum_{i=1\ldots S} (1 - \alpha)^{2 \times ni}$$

For sample-based species accumulation curves, the data can be supplied using the DATA parameter, either as a matrix where the rows contain the number of individuals for each species and the columns specify the different samples or sites, or as a pointer to variates containing samples for the individuals for each species. Alternatively, the individual species numbers or labels can be supplied in either a variate or factor using the DATA parameter while the samples are identified by supplying a grouping factor using the GROUPS option. Individual-based species accumulation curves can be formed using the collector method, where the individual species numbers or labels are specified in either a variate or factor using the DATA parameter. The species numbers or labels must be specified in the order in which they were collected within the variate or factor. Different samples of individuals can be plotted on the same graph by supplying a grouping factor using the GROUPS option and specifying the individual setting of the METHOD option. For the collector curve the observed number of species can be saved using the RICHNESS parameter. For the random and Coleman curves the average and expected number of species and associated variance can be saved using the RICHNESS and VARIANCE parameters respectively. The type of species accumulation curve (collector, random or Coleman) is specified using the CURVE option. If the collector curve is chosen and the data have been supplied using the individual values with a grouping factor, the METHOD option can be used to choose whether to produce a sample-based plot or a plot of the individual-based curves. The number of permutations used for the random method can be supplied using the NPERMUTATIONS option, by default 100 permutations are used. The SEED option specifies the seed to use for the sub-sampling without replacements. The default value of zero continues an existing sequence of random numbers or, if the generator has not yet been used in this run of Genstat, initializes the generator automatically.

The PRINT option controls printed output, with settings:

summary	the species richness and variance (for Coleman and random methods); this is the default.

A plot of the species accumulation curve can be specified using the `sac` setting of the `PLOT` option. The graphical display can be controlled using the `SCREEN`, `WINDOW`, `KEYWINDOW` and `PEN` options. By default the curves are produced in window 1 using pen 1 and drawn on a new screen.

Example 2.11.6 plots species accumulation curve for some data on beetles from Magurran (2003); see Figure 2.11.6.

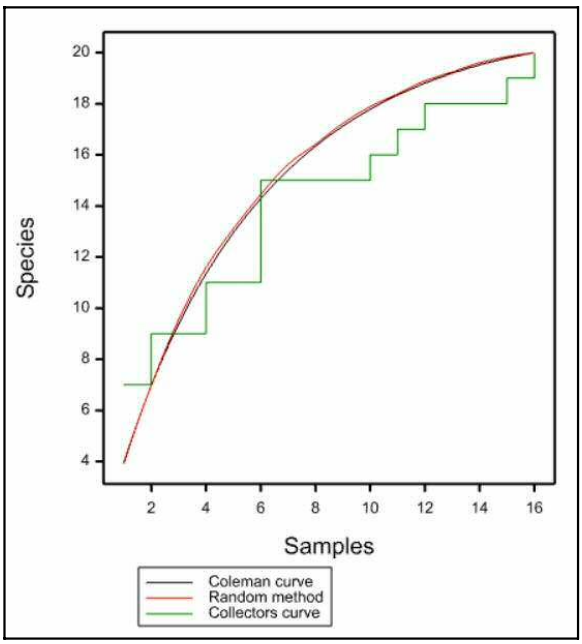


Figure 2.11.6

Example 2.11.6

```
  2  " Abundance of carabid beetles sampled in hedgerows (Magurran 2003)."
  3  VARIATE         [VALUES=0,0,1,0,2,0,6,1,0,0,0,1,1,0,0,1,0,0,0,0] S1
  4  &               [VALUES=0,0,0,1,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0] S2
  5  &               [VALUES=6(0),4,13(0)] S3
  6  &               [VALUES=5(0),2,3,3,6(0),2,5(0)] S4
  7  &               [VALUES=6(0),4,4(0),4,0,0,1,5(0)] S5
  8  &               [VALUES=0,0,2,0,1,0,3,2,1,1,4,0,0,1,1,0,1,0,0,0] S6
  9  &               [VALUES=6(0),2,0,0,0,1,9(0)] S7
 10  &               [VALUES=6(0),1,0,1,11(0)] S8
 11  &               [VALUES=16(0),1,0,0,0] S9
 12  &               [VALUES=0,0,2,0,2,0,1,1,0,0,0,1,0,0,0,1,0,0,2,0] S10
 13  &               [VALUES=12,5(0),5,13(0)] S11
 14  &               [VALUES=0,1,1,1,0,0,11,5,0,1,2,9,6(0),1,0] S12
 15  &               [VALUES=32,0,0,1,9(0),1,0,0,0,0,1,0] S13
 16  &               [VALUES=2,0,2,0,0,1,3,0,0,0,1,9(0)] S14
 17  &               [VALUES=4(0),1,0,9,3,0,0,0,1,0,0,0,0,0,1,1,0] S15
 18  &               [VALUES=0,0,0,0,2,1,2,5(0),1,5(0),1,1] S16
 19  POINTER         [VALUES=S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,\
 20                  S11,S12,S13,S14,S15,S16] Beetle
 21  ECACCUMULATION [PRINT=*; CURVE=coleman] Beetle
 22  ECACCUMULATION [PRINT=*; CURVE=random; SCREEN=keep; PEN=2] Beetle
 23  ECACCUMULATION [PRINT=*; CURVE=collector; SCREEN=keep; PEN=3] Beetle
```

2.11.7 Nonparametric estimation of species richness

`ECNPESTIMATE` procedure

Calculates nonparametric estimates of species richness (D.A. Murray).

Options

`PRINT` = *string token*	Controls printed output (`summary`, `estimates`); default `summ`, `esti`
`GROUPS` = *factor*	Grouping factor for different samples
`NBOOT` = *scalar*	A scalar defining the number of bootstrap samples to be

	performed; default 100
SEED = *scalar*	Seed for random number generator; default 0

Parameters

DATA = *variates*, *matrices* or *pointers*	
	A variate containing abundances of species or a pointer or matrix specifying the individuals for each species for different sites/samples
ESTIMATES = *variates* or *pointer*	Saves the estimated species richness in a variate, or in a pointer if GROUPS are specified
SE = *variates* or *pointers*	Saves the analytic standard errors in a variate, or in a pointer if groups are specified
BSE = *variates* or *pointers*	Saves the bootstrap standard errors in a variate, or in a pointer if groups are specified

Richness is the measure of the number of species within a sample. ECNPESTIMATE provides a number of nonparametric estimators for measuring true species richness. These estimators include the Chao 1, Chao 2, ACE, ICE, first-order jackknife, second-order jackknife and bootstrap. The Chao 1 and ACE are based on the abundances within the samples, whereas the other estimators are incidence-based using frequencies of species in a set of samples. Standard errors are calculated using analytical results where possible. In addition, for multiple samples, standard errors are calculated by resampling with replacement.

The data can be supplied using the DATA parameter either as a matrix where the rows contain the number of individuals for each species and the columns specify the different samples or sites, or as a pointer to variates containing samples for the individuals for each species. Alternatively, the individual species numbers can be supplied in a variate for a single sample/site. The GROUPS option can supply a grouping factor to produce estimates for different groups. The estimates and standard errors can be saved using the ESTIMATES, SE (analytic standard errors) and BSE (bootstrap standard errors) parameters. If a grouping factor is supplied then they will be saved in a pointer to variates, otherwise they are saved in a variate.

The PRINT option controls printed output, with settings:

summary	a summary of the data,
estimates	the species richness estimates and standard errors.

The NBOOT option specifies how many bootstrap samples to take to calculate the bootstrap standard errors and confidence intervals (default 100). The probability level for the confidence interval can be set by the CIPROBABILITY option; by default 0.95. The SEED option specifies the seed to use in the random number generator used to construct the bootstrap samples. The default value of zero continues an existing sequence of random numbers or, if the generator has not yet been used in this run of Genstat, it initializes the generator automatically.

Example 2.11.7 illustrates the use of ECNPESTIMATE using data from Table 5 of Helshe & Forrester (1983), which contains a benthic infaunal sample of a subtidal marsh creek in the Pettquamscutt River in Southern Rhode Island collected in April 1978 by Jeffrey Hyland of the Graduate School of Oceanography of the University of Rhode Island.

Example 2.11.7

```
 2  POINTER      [NVALUES=10] quad
 3  VARIATE      [VALUES=0,2,0,1,0,1,1,2,0,0,0,0,0,8] quad[1]
 4  VARIATE      [VALUES=13,2,1,0,0,1,0,0,1,0,0,0,0,36] quad[2]
 5  VARIATE      [VALUES=21,4,0,1,1,2,0,0,0,1,3,5,0,14] quad[3]
 6  VARIATE      [VALUES=14,4,0,2,2,1,0,0,0,0,0,1,0,19] quad[4]
 7  VARIATE      [VALUES=5,1,0,0,0,0,0,0,0,0,0,0,0,3] quad[5]
 8  VARIATE      [VALUES=22,1,0,6,0,1,0,0,0,0,0,2,0,22] quad[6]
 9  VARIATE      [VALUES=13,1,0,0,1,0,0,0,0,0,0,0,0,6] quad[7]
```

```
  10  VARIATE      [VALUES=4,0,1,0,0,0,0,0,0,0,0,0,1,8] quad[8]
  11  VARIATE      [VALUES=4,1,0,1,0,1,0,0,0,0,0,0,0,5] quad[9]
  12  VARIATE      [VALUES=27,6,0,2,1,5,0,0,0,0,2,3,0,41] quad[10]
  13  ECNPESTIMATE [SEED=204029] quad

Nonparametric estimation of species richness
============================================

Total number of species observed in all samples pooled 14
Number of rare species (<= 10 indivduals)              8
Number of abundant species (> 10 individuals)          6
Number of infrequent species (in <= 10 samples)        14
Number of frequent species (in > 10 samples)           0
Total number of species                                10
Singletons                                             4
Doubletons                                             2
Uniques                                                5
Duplicates                                             2
Number of individuals in rare species                  18
Number of occurences of infrequent species             58


Estimates for species richness
==============================

Abundance-based estimators
--------------------------

Estimator    Estimate        s.e.
   Chao 1       18.00       5.292
      ACE       18.75

Presence/Absence-based estimators
---------------------------------

              Estimate        s.e.
     Chao 2       20.25       7.552
Jackknife 1       18.50       2.012
Jackknife 2       21.08
  Bootstrap       15.97       1.356
        ICE       18.81

Resampling with replacement estimate for species richness
---------------------------------------------------------

              Estimate        s.e.
     Chao 1       16.99       3.055
     Chao 2       18.71       5.148
Jackknife 1       18.48       1.687
Jackknife 2       21.06       3.481
  Bootstrap       15.96       0.637
        ICE       20.06       5.680
        ACE       20.10       6.091

Warning: bootstrap of ACE estimate includes samples where all rare species are equal
to singletons; these samples have been excluded and the bootstrap estimate is based
on 99 samples.
```

The Chao 1 estimator of the absolute number of species in an assemblage is calculated by:

$$\text{s(Chao 1)} = S_{obs} + F_1^2 / (2 \times F_2)$$

where S_{obs} is the number of species in the sample, F_1 is the number of observed species represented by a single individual (frequency of singletons), and F_2 is the number of species that have exactly two individuals (frequency of doubletons). The variance for the estimate is given by:

$$\text{var(Chao 1)} = F_2 \times \{ 0.5 \times (F_1 / F_2)^2 + (F_1 / F_2)^3 + 0.25 \times (F_1 / F_2)^4 \}$$

When F_2 equals 0 the modified bias-corrected estimate is used:

$$\text{s(Chao 1)} = S_{obs} + F_1 \times (F_1 - 1) / 2$$

and

$$\text{var(Chao 1)} = \{F_1 \times (F_1 - 1) / 2\} + \{F_1 \times (2 \times F_1 - 1)^2 / 4\} - F_1^4 / (4 \times \text{s(Chao 1)})$$

The Chao 2 estimator is calculated by:

$$\text{s(Chao 2)} = S_{obs} + Q_1^2 / (2 \times Q_2)$$

where S_{obs} is the number of species in sample, Q_1 is the number of species that occur in exactly one sample (uniques), and Q_2 is the number of species that occur in exactly two samples (duplicates). The variance for the estimate is given by:

$$\text{var(Chao 2)} = Q_2 \times \{ 0.5 \times (Q_1 / Q_2)^2 + (Q_1 / Q_2)^3 + 0.25 \times (Q_1 / Q_2)^4 \}$$

When Q_2 equals 0 the modified bias-corrected estimate is used:

$$\text{s(Chao 2)} = S_{obs} + Q_1 \times (Q_1 - 1) / 2$$

and

$$\begin{aligned}\text{var(Chao 2)} = {} & \{(H - 1) / H\} \times Q_1 \times (Q_1 - 1) / 2 \\ & + \{(H - 1) / H\}^2 \times Q_1 \times \{2 \times Q_1 - 1)^2\} / 4 \\ & + \{(H - 1) / H\}^2 \times Q_1^4 / (4 \times \text{Chao2})\end{aligned}$$

where H is the total number of samples.

The first-order jackknife estimate is evaluated by:

$$\text{s(jack1)} = S_{obs} + Q_1 \times (H - 1) / H$$

with variance

$$\text{var(jack1)} = \{(H - 1) / H\} \times \{ \textstyle\sum_{j=1\ldots S} (j^2 \times f_j) - (Q_1^2 / H) \}$$

where S is the number of species, Q_1 is the number of species that occur in exactly one sample and f_j is the number of samples with j unique species.

The second-order jackknife estimate is given by:

$$\text{s(jack2)} = S_{obs} + Q_1 \times (2 \times H - 3) / H - Q_2 \times (H - 2)^2 / \{H \times (H - 1)\}$$

where Q_1 is the number of species that occur in exactly one sample, and Q_2 is the number of species that occur in exactly two samples.

The bootstrap estimate is calculated by:

$$\text{s(boot)} = S_{obs} + \textstyle\sum_{j=1\ldots S} (1 - p_j)^H$$

where p_j is the proportion of species j. The variance is calculated using the method given in Smith & van Belle (1984).

The abundance-based coverage estimator (ACE) is given by:

$$\text{s(ACE)} = S_{abund} + S_{rare} / C_{ACE} + (F_1 / C_{ACE}) \times \gamma^2$$

where S_{abund} is the number of abundant species (>10), S_{rare} is the number of rare species ($\leq$10), F_1 is the number of singletons,

$$C_{ACE} = 1 - F_1 / N_{rare}$$

where N_{rare} is the total number of individuals in rare species, and

$$\gamma = \max \{(S_{\text{rare}}/C_{ACE}) \times \textstyle\sum_{i=1\ldots 10} \{i \times (i-1) \times F_i\} / (N_{rare} \times (N_{rare} - 1)) - 1, 0\}$$

The incidence-based coverage estimator (ICE) is given by:

$$\text{s(ICE)} = S_{freq} + S_{infr} / C_{ICE} + (Q_1 / C_{ICE}) \times \gamma^2$$

where S_{freq} is the number of frequent species (>10), S_{infr} is the number of infrequent species (<=10), Q_1 is the number of uniques, $C_{ICE} = 1 - Q_1 / N_{infr}$ where N_{infr} is the total number of occurrences of infrequent species, and

$$\gamma = \max\{(S_{infr}/C_{ICE}) \times (M_{infr}/(M_{infr}-1)) \times (\textstyle\sum_{i=1\ldots 10} \{i \times (i-1) \times Q_i\} / N_{infr}^2) - 1, 0\}$$

where M_{infr} is the number of samples with at least one infrequent species.

The bootstrap standard errors are generated using the `BOOTSTRAP` procedure sampling with replacement, and the species richness estimates are calculated from these samples.

2.11.8 Lorenz curve and Gini coefficient

LORENZ procedure

Plots the Lorenz curve and calculates the Gini and asymmetry coefficients (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (gini, lorenz, asymmetry); default gini, lore, asym
PLOT = *string token*	Controls graphical output (curve); default curv
TITLE = *string*	Title for the graph; default uses the identifier of the DATA variate
NBOOT = *scalar*	Number of samples to make to construct the bootstrap confidence intervals; default 100
SEED = *scalar*	Seed for the random number generator used to construct the bootstrap samples; default 0 i.e. continue an existing sequence of random numbers or, if none, initialize the generator automatically
CIPROBABILITY = *scalar*	Probability for the bootstrap confidence interval; default 0.95

Parameters

DATA = *variates*	Specifies sets of data values
GINI = *scalars*	Saves the Gini coefficient for each DATA variate
ASYMMETRY = *scalars*	Saves the asymmetry coefficient for each DATA variate

The Lorenz curve provides a graphical representation of the inequality of a sample of numbers. In economics the numbers could be the annual incomes of a group of people, or in ecology they could be population sizes of a set of species of animal or plant. The y-coefficients for the curve are formed by sorting the numbers, calculating their cumulative totals, and then dividing these by the grand total. The x-coefficients are simply the numbers 0, 1, ... *n*, where *n* is the size of the sample. If the numbers are all equal, the curve will form a straight line, known as the line of equality, running from the origin to the point (1, 1). Inequalities amongst the numbers cause the curve to lie below the line of equality.

The Gini coefficient is the area between the line of equality and the Lorenz curve area, divided by area under the line of equality. So, a value close to zero indicates near equality, while a value near to one shows a high amount of inequality. The asymmetry coefficient assesses the amount of asymmetry of the Lorenz curve. The axis of symmetry for the curve is the line from (1, 0) to (0, 1). The coefficient is less than one if the point where the Lorenz curve is parallel to the line of equality lies below the axis of symmetry, and greater than one if it lies above the axis.

The numbers whose equality is to be studied are specified, in a variate, by the DATA parameter. Their Gini and asymmetry coefficients can be saved, in scalars, using the GINI and ASYMMETRY parameters respectively.

Printed output is controlled by the PRINT option, with settings:

asymmetry	prints the coefficient of asymmetry,
gini	prints the Gini,
lorenz	prints the coordinates of the Lorenz curve.

By default, these are all printed.

The procedure can also print bootstrap confidence intervals for the Gini and asymmetry coefficients. The probability level for the interval is specified by the CIPROBABILITY option; the default of 0.95 gives 95% intervals. The NBOOT option specifies how many bootstrap samples to take (default 100). If you do not want the confidence intervals, you should set NBOOT=0. The SEED option specifies the seed to use in the random number generator used to construct the bootstrap samples. The default value of zero continues an existing sequence of random numbers or, if the generator has not yet been used in this run of Genstat, it initializes the generator automatically.

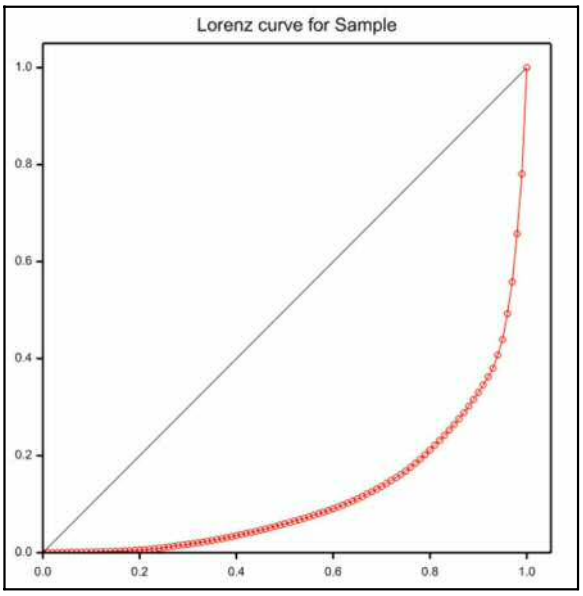


Figure 2.11.8

By default curve is plotted, but you can set PLOT=* to suppress the plot. The TITLE option can supply a title for the graph.

Example 2.11.8 and Figure 2.11.8 illustrates LORENZ using some (rather non-uniform) random numbers from a log-Normal distribution.

Example 2.11.8

```
  2  CALCULATE [SEED=490317] Sample = GRLOGNORMAL(100; 10; 2)
  3  LORENZ    [SEED=846064] Sample

Lorenz curve for Sample
=======================

Gini coefficient 0.7562
95% Bootstrap confidence interval (0.609, 0.814)

Coefficient of asymmetry 1.057
95% Bootstrap confidence interval (0.934, 1.137)
```

3 Regression analysis

This chapter describes the Genstat commands for regression, generalized linear models, generalized additive models and nonlinear curve fitting. The contents thus correspond to the Regression Analysis menus in Genstat *for Windows*.

The simplest meaning of the word *regression* is the technique for fitting a straight line that relates one quantitative variable to another. The *response variable* is supposed to be dependent on the *explanatory variable*. We describe how to do this simple linear regression with Genstat in Section 3.1.

In later sections we use the word regression to cover a much wider class of relationships. We look at more than two variables, at qualitative variables, and at nonparametric and nonlinear relationships, including regression trees. But the common feature is that we shall always be modelling the dependence of one variable on others.

The word linear here does not mean linear in terms of the explanatory variables, but rather linear in terms of the parameters or coefficients that have to be estimated. Thus the regression

$$y_i = \alpha + \beta x_i + \gamma x_i^2 + \varepsilon_i$$

is in fact linear: it is linear in terms of the parameters α, β and γ, even though it is not linear in terms of the explanatory variable *X*.

In the model for simple linear regression, it is usually assumed that the response variable has a Normal distribution with constant variance. But other distributions can be used, and the variance need not be constant. For example, the distribution could be Poisson in which the variance is equal to the mean. These extensions are provided by *generalized linear models*, as described in Section 3.5.

In most of the models in this chapter, we assume that there is only one component of variation: that is, they contain only one error term like ε in the equation above. When there are more components with Normally distributed data, some results can be obtained by the methods described here: for example, you could analyse the effects of treatment factors after eliminating some grouping of the units into blocks, by treating the blocking factor as if it were another treatment factor. But it is usually more convenient, and more efficient, to use the methods of Chapter 4 if the design is balanced, or those of Chapter 5 otherwise. However, Section 3.5 does cover generalized linear mixed models and hierarchical generalized linear models, which extend the generalized linear models theory to handle more than one error term.

We assume in this chapter that you know which is the response variable and which are explanatory variables. There are more general methods of investigating relationships between variables, in which no single variable is treated as a response; see Chapter 6. We also assume that the relationship between the response variable and explanatory variables relates the mean of the response to given explanatory values. The methods of regression analysis are not applicable to law-like relationships, with values of both the response and the explanatory variables subject to error; for more details, see Sprent (1969). Finally, we assume that the errors in the regression models are uncorrelated. For example, the quantities ε_i in the equation above are assumed to be independently distributed. When there is some correlation between the errors, the methods of Chapter 4 may be suitable, particularly if the correlation is constant within some groups of the data and zero between the groups. Alternatively, if there is a serial pattern of correlation, where the order of the observations is important, the methods of Chapters 5 or 7 may be used.

The information in this chapter is grouped mainly by type of analysis, rather than by command. So first we summarize the commands, giving references to the sections below where they are described. Details of those not covered here can be found in the *Genstat Reference Manual*. There are three preliminary directives for defining the form of model to be fitted, of which the `MODEL` directive must always be given first:

MODEL	defines the response variate(s) and the type of model to be fitted (3.1.1)
TERMS	specifies a maximal model, containing all terms to be used in subsequent regression models (3.2.3)
RCYCLE	controls iterative fitting of generalized linear models, generalized additive models and nonlinear models, and specifies parameters and bounds for nonlinear models (3.5.4)

Separate directives carry out the fitting of the various types of model:

FIT	fits a linear model, a generalized linear model, a generalized additive model, or a generalized nonlinear model (3.1.2)
FITCURVE	fits a standard nonlinear regression model (3.7.1)
FITNONLINEAR	fits a user-defined nonlinear regression model or optimizes a scalar function (3.8.2)

Further directives are provided to allow sequential modification of the set of explanatory variables:

ADD	adds extra terms to any type of regression model (3.2.4)
DROP	drops terms from any type of regression model (3.2.4)
SWITCH	adds terms to, or drops them from, any type of regression model (3.2.4)
TRY	displays results of single-term changes to a linear or generalized linear model (3.2.5)
STEP	selects terms to include in or exclude from a linear or generalized linear model (3.2.7)

Once you have fitted the model, you can display further results, form and compare predictions, plot the fitted model, produce diagnostic plots, store the results in data structures for use elsewhere in Genstat, do permutation (or exact) texts, or calculate power information about the model:

RDISPLAY	displays the fit of any type of regression model (3.1.3, 3.5.3, 3.7.4)
PREDICT	forms predictions from a linear or generalized linear model (3.3.4, 3.5.3)
RCOMPARISONS	calculates comparison contrasts amongst the levels of one of the classifying factors of a table of predicted means (3.3.5)
RTCOMPARISONS	calculates comparison contrasts amongst a multi-way table of predicted means (3.3.5)
RCURVECOMMONNONLINEAR	refits a standard curve with common nonlinear parameters across groups to provide s.e.'s for linear parameters (3.7.7)
RFUNCTION	estimates functions of parameters of any type of regression model (3.7.5)
RGRAPH	draws a graph to display the fit of any type of regression model (3.1.6)
RCHECK	provides diagnostic plots and other information for checking the fit of any type of regression model (3.1.7
RDESTIMATES	plots one- or two-way tables of regression estimates (3.3.6)
RKEEP	stores the results from any type of regression model (3.1.4,

3.5.3, 3.7.4)

RSPREADSHEET	puts results from a regression, generalized linear or nonlinear model into spreadsheets (3.1.5)
RKESTIMATES	saves estimates and other information about individual terms in a regression analysis (3.2.2)
RWALD	calculates Wald and F tests for dropping terms from a regression (3.2.6)
RPERMTEST	does random permutation tests for regression models (3.1.9)
RPOWER	calculates the power (probability of detection) for regression models (3.1.8)

There are also many specialized procedures in the Procedure Library; see Part 3 of the *Genstat Reference Manual*.

BREGRESSION	constructs a regression tree (3.9.1)
BRDISPLAY	displays a regression key (3.9.2)
BRVALUES	forms values for nodes of a regression tree (3.9.3)
BPRUNE	prunes a tree using minimal cost complexity (3.9.3)
BRPREDICT	makes predictions using a regression tree (3.9.4)
BRKEEP	saves information from a regression tree (3.9.5)
BRFOREST	constructs a random regression forest
BRFDISPLAY	displays information about a random regression forest
BRFPREDICT	makes predictions using a random regression forest
FITINDIVIDUALLY	fits regression and generalized linear models one term at a time (3.5.3)
GEE	fits models to longitudinal data by generalized estimating equations (3.5.12)
GLM	analyses non-standard generalized linear models
GLMM	fits a generalized linear mixed model (3.5.10)
GLDISPLAY	displays further output from a GLMM analysis (3.5.10)
GLKEEP	saves results from a GLMM analysis (3.5.10)
GLPERMTEST	does random permutation tests for generalized linear mixed models (3.5.10)
GLPLOT	plots residuals from a GLMM analysis (3.5.10)
GLPREDICT	forms predictions from a GLMM analysis (3.5.10)
GLRTEST	calculates likelihood tests to assess random terms in a generalized linear mixed model (3.5.10)
GLTOBITPOISSON	uses the Tobit method to fit a generalized linear mixed model with censored Poisson data
HGANALYSE	analyses data using hierarchical generalized linear models (3.5.11)
HGDISPLAY	displays a hierarchical generalized linear model analysis (3.5.11)
HGFIXEDMODEL	defines the fixed model for a hierarchical generalized linear model (3.5.11)
HGFTEST	calculates likelihood tests for fixed terms in a hierarchical generalized linear model (3.5.11)
HGKEEP	saves information from a hierarchical generalized linear model analysis (3.5.11)
HGNONLINEAR	defines nonlinear parameters for the fixed model of a hierarchical generalized linear model (3.5.11)
HGPLOT	produces model-checking plots for a hierarchical

generalized linear model analysis (3.5.11)

HGGRAPH	draws a graph to display the fit of hierarchical generalized linear model analysis (3.5.11)
HGPREDICT	forms predictions from hierarchical hierarchical generalized linear model analysis (3.5.11)
HGRANDOMMODEL	defines the random model for a hierarchical generalized linear model (3.5.11)
HGDRANDOMMODEL	extends a hierarchical generalized linear model to become a double hierarchical generalized linear model (3.5.11)
HGRTEST	calculates likelihood tests for random terms in a hierarchical generalized linear model (3.5.11)
HGSTATUS	displays the current HGLM model definitions (3.5.11)
HGTOBITPOISSON	uses the Tobit method to fit a hierarchical generalized linear model with censored Poisson data
HGWALD	Prints or saves Wald tests for fixed terms in an HGLM (3.5.11)
PROBITANALYSIS	fits probit models allowing for natural mortality and immunity (3.5.9)
FIELLER	calculates effective doses and relative potencies (3.5)
MICHAELISMENTEN	fits the Michaelis-Menten equation for substrate concentration versus time data
MMPREDICT	predicts the Michaelis-Menten curve for a particular set of parameter values
NLAR1	fits curves with an AR1 or a power-distance correlation model (8.1.6)
RAR1	fits regressions with an AR1 or a power-distance correlation model (8.1.6)
RMPLCONFIDENCE	estimates profile likelihood confidence intervals of predicted group means from a linear or generalized linear model analysis
RPLCONFIDENCE	estimates profile likelihood confidence intervals of parameters in a linear or generalized linear model
RQLINEAR	fits and plots quantile regressions for linear models (3.10.1)
RQNONLINEAR	fits and plots quantile regressions for nonlinear models
RQSMOOTH	fits and plots quantile regressions for loess or spline models
RSCREEN	performs screening tests for generalized or multivariate linear models (3.2.9)
RSEARCH	helps search through models for a regression or generalized linear model (3.2.8)
R0INFLATED	fits zero-inflated regression models to count data with excess zeros (3.5.13)
R0KEEP	saves information from models fitted by R0INFLATED (3.5.13)
RBRADLEYTERRY	fits the Bradley-Terry model for paired-comparison preference tests
RCATENELSON	performs a Cate-Nelson graphical analysis of bivariate data
RCIRCULAR	does circular regression of mean direction for an angular response

RFINLAYWILKINSON	performs Finlay and Wilkinson's joint regression analysis of genotype-by-environment data
RIDGE	does ridge regression and principal component regression analyses
LRIDGE	does logistic ridge regression
RLASSO	performs lasso using iteratively reweighted least-squares
RLFUNCTIONAL	fits a linear functional relationship model
RMGLM	fits a model where different units follow different generalized linear models
RNEGBINOMIAL	fits a negative binomial generalized linear model, estimating the aggregation parameter
RNONNEGATIVE	fits a generalized linear model with nonnegativity constraints
RPAIR	gives t-tests for all pairwise differences of means from linear or generalized linear models
RPARALLEL	carries out analysis of parallelism for nonlinear functions
RQUADRATIC	fits a quadratic surface and estimates its stationary point
RRETRIEVE	retrieves a regression save structure from an external file
RSTORE	stores a regression save structure in an external file
RSCHNUTE	fits a general four-parameter growth model to a non-decreasing response variate
RTOBITPOISSON	uses the Tobit method to fit models to censored Poisson data
RVALIDATE	fits regression models to validate predictions, for example from a deterministic model, against observed data
RYPARALLEL	fits the same regression model to several response variates, and collates the output
R2LINES	fits two-straight-line (broken-stick) models
IFUNCTION	estimates implicit and/or explicit functions of parameters
MINIMIZE	finds the minimum of a function calculated by a procedure
MIN1DIMENSION	finds the minimum of a function in one dimension
SIMPLEX	searches for the minimum of a function using the Nelder-Mead simplex algorithm
SVGLM	fits generalized linear models to survey data
YTRANSFORM	estimates the parameter lambda of a single parameter transformation
XOCATEGORIES	performs analyses of categorical data from cross-over trials
EXTRABINOMIAL	fits models to overdispersed proportions
DILUTION	calculates most probable numbers from dilution series data
DSEPARATIONPLOT	creates a separation plot for visualising the fit of a model with a dichotomous (i.e. binary) or polytomous (i.e. multi-categorical) outcome
WADLEY	fits models for Wadley's problem, allowing alternative links and errors

3.1 Simple linear regression

The word *simple* here refers to the fact that there is only one explanatory variable. Suppose you have observations $\{y_i: i = 1...N\}$ of a response variable Y, and $\{x_i: i = 1...N\}$ of an explanatory variable X. Then the model for simple linear regression is:

$$y_i = \alpha + \beta x_i + \varepsilon_i$$

where α and β are unknown *parameters*: that is, they are numerical characteristics of the model that determine the precise nature of the relationship. The values $\{\varepsilon_i: i = 1...N\}$ are *errors* which are random variables, assumed to be identically and independently distributed with a Normal distribution. The model can also be written as

$$y_i = f_i + \varepsilon_i$$

where the values $\{f_i: i = 1...N\}$ are the *fitted values* generated by the model. So

$$f_i = \alpha + \beta x_i$$

For further details, see the books by Seber (1977), Draper & Smith (1981) or Weisberg (1985), or indeed any other standard statistical text.

The model can alternatively be written in matrix form:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

where the vector $\boldsymbol{\beta} = (\alpha,\beta)'$, and $\boldsymbol{X}$ is an $N \times 2$ matrix whose first column consists just of 1's, called the *design matrix*. (This is standard terminology although, of course, regression is often used when it has not been possible to use any special design.)

Example 3.1 shows the commands to fit a simple linear regression. (In Genstat *for Windows* this type of regression analysis can be obtained by selecting Simple Linear Regression in the Regression list box of the Linear Regression menu.) The model here is a linear relationship between the logarithm of barometric pressure and the boiling point of water. Forbes (1857) collected these measurements at the tops of mountains with the intention that, on any other mountain, he would be able to predict barometric pressure (and hence the height of the mountain) by boiling water at the summit.

Example 3.1

```
  2  " Simple linear relationship between boiling point and barometric
 -3    pressure. Data from Forbes (1857); analysed by Weisberg (1985) p.3."
  4  READ [PRINT=data] Boiltemp,Pressure

  5  194.50 20.79  194.25 20.79  197.90 22.40  198.43 22.67  199.45 23.15
  6  199.95 23.35  200.93 23.89  201.15 23.99  201.35 24.02  201.30 24.105
  7  203.55 25.14  204.60 26.57  209.47 28.49  208.57 27.760 210.72 29.040
  8  211.95 29.879 212.18 30.064 :
  9  CALCULATE Logpress = 100*LOG10(Pressure)
 10  "DGRAPH [TITLE='Forbes data'] Logpress; Boiltemp"
 11  MODEL Logpress
 12  FIT Boiltemp

Regression analysis
===================

 Response variate: Logpress
     Fitted terms: Constant, Boiltemp


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression       1      425.349     425.3493   3000.08
Residual        15        2.127       0.1418
Total           16      427.476      26.7173

Percentage variance accounted for 99.5
Standard error of observations is estimated to be 0.377.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           12      142.439        3.71
```

```
Estimates of parameters
-----------------------

Parameter      estimate         s.e.      t(15)
Constant        -42.10         3.32     -12.68
Boiltemp         0.8953       0.0163      54.77
```

The first two statements set up variates storing the values of the two variables to be analysed and the DGRAPH statement displays the scatterplot in Figure 3.1; the next two statements fit the regression.

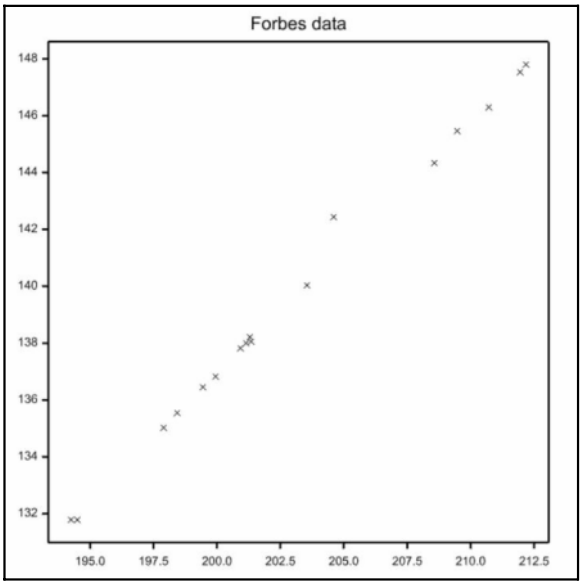


Figure 3.1

It is often necessary to give CALCULATE statements before the regression statements. Though the model is linear, it can be fitted to a transformation of the response variable (as here), or of the explanatory variable, or both. This can be done to get variables that are expected to be linearly related, or to get a response variable with an approximately Normal distribution with constant variance. Unfortunately, both of these conditions are needed for the regression analysis to be valid; when one set of transformations does not achieve both – as is usually the case with a response variable of counts or proportions, for example – then it is best to fit a generalized linear model (3.5) or a nonlinear model (3.7 and 3.8). Additive models (3.4) can be used when there is no predetermined form of a relationship.

You can fit models to subsets of the data by using the RESTRICT directive (1:4.4.1). The regression directives also automatically exclude any unit that contains a missing value for either variate. However, if only the response is missing, Genstat does give you some information about the unit (3.1.2).

Most of the directives in this section are relevant also to multiple regression and to nonlinear regression. But you can understand their main features most readily by seeing them in the simplest case.

3.1.1 The MODEL directive

MODEL directive

Defines the response variate(s) and the type of model to be fitted for linear, generalized linear, generalized additive and nonlinear models.

Options

DISTRIBUTION = *string token*	Distribution of the response variable (normal, poisson, binomial, gamma, inversenormal, multinomial, calculated, negativebinomial, geometric, exponential, bernoulli); default norm
LINK = *string token*	Link function (canonical, identity, logarithm, logit, reciprocal, power, squareroot, probit, complementaryloglog, calculated, logratio);

	default `cano` (i.e. `iden` for `DIST=norm` or `calc`; `loga` for `DIST=pois`; `logi` for `DIST=bino`, `bern`, or `mult`; `reci` for `DIST=gamm` or `expo`; `powe` for `DIST=inve`; `logr` for `DIST=nega` or `geom`)
`EXPONENT` = *scalar*	Exponent for power link; default -2
`AGGREGATION` = *scalar*	Fixed parameter for negative binomial distribution (parameter *k* as in variance function Var = mean + mean^2/*k*); default 1
`KLOGRATIO` = *scalar*	Parameter for logratio link, in form log(mean/(mean+*k*)); default as set in `AGGREGATION` option
`DISPERSION` = *scalar*	Value of dispersion parameter in calculation of s.e.s etc; default `*` for `DIST=norm`, `gamm`, `inve`, or `calc`, and 1 for `DIST=pois`, `bino`, `mult`, `nega`, `geom`, `expo` or `bern`
`WEIGHTS` = *variate* or *symmetric matrix*	Variate of weights for weighted regression, or symmetric matrix of weights (one row and column for each unit of data) for generalized least squares; default `*`
`OFFSET` = *variate*	Offset variate to be included in model; default `*`
`GROUPS` = *factor*	Absorbing factor defining the groups for within-groups linear or generalized linear regression; default `*`
`RMETHOD` = *string token*	Type of residuals to form, if any, after each model is fitted (`deviance`, `Pearson`, `simple`); default `devi`
`DMETHOD` = *string token*	Basis of estimate of dispersion, if not fixed by `DISPERSION` option (`deviance, Pearson`); default `devi`
`FUNCTIONVALUE` = *scalar*	Scalar whose value is to be minimized by calculation; default `*`
`YRELATION` = *string token*	Whether to analyse the y-variates separately, as in ordinary regression, or to analyse them cumulatively as counts in successive categories of a multinomial distribution (`separate, cumulative`); default `sepa`
`DCALCULATION` = *expression structures*	Calculations to define the deviance contributions and variance function for a non-standard distribution; must be specified when `DIST=calc`
`LCALCULATION` = *expression structures*	Calculations to define the fitted values and link derivative for a non-standard link; must be specified when `LINK=calc`
`DFDISPERSION` = *scalar*	Allows you to specify the number of degrees of freedom for a dispersion parameter specified by the `DISPERSION` option; if this is not set, the supplied dispersion is assumed to be known exactly
`SAVE` = *identifier*	To name regression save structure; default `*`

Parameters

`Y` = *variates*	Response variates; only the first is used in nonlinear models and in generalized linear models except when `DIST=mult`, when they specify the numbers in each

	category of an ordinal response model
NBINOMIAL = *variate* or *scalar*	Total numbers for DIST=bino
RESIDUALS = *variates*	To save residuals for each y variate after fitting a model
FITTEDVALUES = *variates*	To save fitted values, and provide fitted values if no terms are given in FITNONLINEAR
LINEARPREDICTOR = *variate*	Specifies the identifier of the variate to hold the linear predictor
DERIVATIVE = *variate*	Specifies the identifier of the variate to hold the derivative of the link function at each unit
DEVIANCE = *variate*	Specifies the identifier of the variate to hold the contribution to the deviance from each unit
VFUNCTION = *variate*	Specifies the identifier of the variate to hold the value of the variance function at each unit

In most applications, you will need only a simple form of the directive:

```
MODEL identifier_of_response_variate
```

Notice that MODEL does not actually fit anything: it simply sets up some structures inside Genstat that are used when you give a FIT statement later on (3.1.2). So when you are doing regression, MODEL will always be accompanied by at least one other regression statement to fit a model, like FIT.

The Y parameter allows a list of variates; if you put more than one for linear regression, then you will get an analysis for each. This is a more efficient way of doing many linear regressions with the same explanatory variables, than separate pairs of MODEL and FIT statements. However, with additive models, generalized linear models and nonlinear models (3.4, 3.5, 3.7 and 3.8), only the first variate will be analysed (with the exception of multinomial response models, 3.5.5); the others will be ignored.

The NBINOMIAL parameter is relevant only for the binomial setting of the DISTRIBUTION option (3.5.1).

The RESIDUALS and FITTEDVALUES parameters allow you to specify variates to contain the residuals and fitted values for each response variable. For example, you could change the MODEL statement above to ensure that each subsequent FIT statement will put the residuals into a variate R and fitted values into a variate F:

```
MODEL Logpress; RESIDUALS=R; FITTEDVALUES=F
```

The residuals are the "unexplained" component of the response variable, standardized as requested by the RMETHOD option (see below). The fitted values are the "explained" component: that is, the combination of parameters and explanatory variables fitted in the model. You can access these sets of values in a different way using the RKEEP directive (3.1.4).

The remaining parameters and the DISTRIBUTION, LINK, EXPONENT, AGGREGATION and KLOGRATIO options are used for generalized linear models, which are described in Section 3.5.1.

The DISPERSION option controls how the variance of the distribution of the response values is calculated. By default, for the Normal distribution, the variance is estimated from the residual mean square (3.1.2), and standard errors and standardized residuals are calculated from the estimate. If you use DISPERSION to supply a value for the variance of the Normal distribution, the standard errors and residuals will be based on this given value instead. The DFDISPERSION option allows you to specify the number of degrees of freedom for a variance specified by the DISPERSION option. You might want to use this, for example, if you had estimated the variance from some other data set. If DFDISPERSION is not set, the supplied variance is assumed to be known exactly. The use of DISPERSION and the associated DMETHOD option with other distributions is described in 3.5.1.

The WEIGHTS option allows you to specify a variate holding weights for each unit, so that you

can perform a *weighted linear regression.* Suppose, for example, you have assigned values to a weights variate W earlier in the program; then the option takes the form: WEIGHTS=W. If the weight for unit *i* is w_i, the regression directives will weight by w_i the contribution to the estimate of dispersion from the *i*th unit. In simple linear regression, the estimate of dispersion is then the weighted residual mean square:

$$\Sigma\{w_i \varepsilon_i^2\}/(N-2)$$

Thus, if the variance of the response variable is not constant, and you know the relative size of the variance for each observation, you can set the weight to be proportional to the inverse of the variance of an observation. Alternatively, if the variance is related in a simple way to the mean, you may just need to specify a different distribution for the response (3.5). You can also supply a symmetric matrix of weights for *generalized least squares* (see 3.6).

The OFFSET option allows you to include in the regression a variable with no corresponding parameter:

$$y_i = \alpha + o_i + \beta x_i + \varepsilon_i$$

where o_i is the *i*th value of the offset variable, *O* say. Linear regression analysis of *Y* with offset *O* is just the same as analysis of *Y*–*O*, but the offset has non-trivial applications in generalized linear models (3.5.1).

The GROUPS option specifies a factor whose effects you want to eliminate before any regression is fitted. The factor must already have been defined. (The effects of factors on regression are discussed in 3.3.) This method of elimination is sometimes called *absorption*; you might want to use it when data from many different groups are to be modelled. Use of GROUPS gives less information than you would get if you included the factor explicitly in the model (leverages, predictions and some parameter correlations cannot be formed), but it saves space and time in fitting the model. You can use GROUPS only with linear and generalized linear models.

The RMETHOD option controls how residuals are formed. By default, residuals are *deviance residuals* standardized by their estimated variance: i.e. the residuals are scaled so that they have equal variances, making it easier for you to assess whether any are especially large. For linear regression, the standardized residuals are:

$$r_i = (y_i - f_i)\sqrt{(w_i / v_i)}$$

In this equation, f_i is the *i*th fitted value, and v_i is the variance of an unstandardized residual:

$$v_i = (1 - l_i)\, s^2$$

Here, s^2 is the estimate of dispersion and l_i is the *leverage* (diagonal of the projection matrix), defined in terms of the design matrix $\boldsymbol{X}$ and the diagonal matrix of weights $\boldsymbol{W}$ by

$$l_i = w_i \{\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}'\}_{ii}$$

Pearson residuals (RMETHOD=Pearson) are relevant to regression models with distributions other than Normal (see 3.5.1); they are identical to the ordinary standardized deviance residuals when the distribution is Normal. If you do not want the residuals to be standardized, you can set RMETHOD=simple. The residual is then simply the difference between the response and the fitted value:

$$r_i = (y_i - f_i)$$

Finally, if you do not want any residuals, you can set the option to a missing value (*) to save space within Genstat. However, you will not then be able to get residuals, fitted values or leverages, and the automatic checks on the fit of a model will not be done (3.1.2).

The FUNCTIONVALUE option is relevant only when you want to optimize a general function (3.8.4). It is ignored unless no response variates are specified by the Y parameter.

The YRELATION option is relevant only for ordinal response models (3.5.5), and the DCALCULATION and LCALCULATION options only for generalized linear models that you define yourself (3.5.6).

The SAVE option allows you to specify an identifier for the regression save structure. This structure stores the current state of the regression model, and can be used explicitly in the

directives RDISPLAY (3.1.3), RKEEP (3.1.4), PREDICT (3.3.4) and RFUNCTION (3.7.5). If the identifier in SAVE is of a regression save structure that already has values, those values are deleted. You can reset the current regression save structure at any point in a program by using the SET directive (1:5.6.1). Then, later regression statements would use the model stored in this save structure.

3.1.2 The FIT directive

FIT directive

Fits a linear, generalized linear, generalized additive or generalized nonlinear model.

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, grid, confidence); default mode, summ, esti or grid if NGRIDLINES is set
CALCULATION = *expression structures*	Calculation of explanatory variates involving nonlinear parameters
OWN = *scalar*	Option setting for OWN directive if this is to be used rather than CALCULATE to calculate explanatory variates
CONSTANT = *string token*	How to treat the constant (estimate, omit, ignore); default esti
FACTORIAL = *scalar*	Limit for expansion of model terms; default as in previous TERMS statement, or 3 if no TERMS given
POOL = *string token*	Whether to pool ss in accumulated summary between all terms fitted in a linear model (yes, no); default no
DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality, vertical, df, inflation); default *
FPROBABILITY = *string token*	Printing of probabilities for variance and deviance ratios (yes, no); default no
TPROBABILITY = *string token*	Printing of probabilities for t-statistics (yes, no); default no
SELECTION = *string tokens*	Statistics to be displayed in the summary of analysis produced by PRINT=summary, seobservations is relevant only for a Normally distributed response, and %cv only for a gamma-distributed response (%variance, %ss, adjustedr2, r2, seobservations, dispersion, %cv, %meandeviance, %deviance, aic, bic, sic); default %var, seob if DIST=normal, %cv if DIST=gamma, and disp for other distributions
PROBABILITY = *scalar*	Probability level for confidence intervals for parameter estimates; default 0.95
NGRIDLINES = *scalar*	Number of values of each nonlinear parameter for a grid of function evaluations

SELINEAR = *string token*	Whether to calculate s.e.s for linear parameters when nonlinear parameters are also estimated (yes, no); default no
INOWN = *identifiers*	Setting to be used for the IN parameter of OWN if used to calculate explanatory variates
OUTOWN = *identifiers*	Setting to be used for the OUT parameter of OWN if used to calculate explanatory variates
AOVDESCRIPTION = *text*	Description for line in accumulated analysis of variance (or deviance) table when POOL=yes

Parameter

formula	List of explanatory variates and factors, or model formula

A FIT statement must always be preceded by a MODEL statement, though not necessarily immediately. You can give several FIT statements after a single MODEL statement: for example, you might want to try out different explanatory variables.

The parameter of the FIT directive specifies the explanatory variables in the model. In the simple linear regression above, it consists of the identifier of the explanatory variate alone:

```
FIT Boiltemp
```

If you omit the parameter, Genstat fits a *null model*; that is, a model consisting of just one parameter, the overall mean:

$y_i = \alpha + \varepsilon_i$

The PRINT option controls output. You can give several settings at the same time, to provide reports on several aspects of the analysis.

The model setting gives a description of the model, including response and explanatory variates. Here is a repeat of this aspect of the analysis in Example 3.1; model gives the first lines in this output.

Example 3.1.2a

```
  13  FIT [PRINT=model,summary; FPROBABILITY=yes] Boiltemp

Regression analysis
===================

 Response variate: Logpress
     Fitted terms: Constant, Boiltemp


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       1      425.349     425.3493   3000.08  <.001
Residual        15        2.127       0.1418
Total           16      427.476      26.7173

Percentage variance accounted for 99.5
Standard error of observations is estimated to be 0.377.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           12      142.439        3.71
```

The output from the summary setting is also reproduced here: this starts by giving a summary analysis of variance, which subdivides the total sum of squares, corrected for the mean, between

that explained by the regression (Regression), and that which is not explained (Residual). The table has the standard form with columns for the degrees of freedom (d.f.), the sums of squares (s.s.), the mean squares (m.s.), and for the variance ratio (v.r.). In addition, because we have set the `FPROBABILITY` option, there is a column giving the probability that the variance ratio would be as large as this under the null hypothesis of no relationship; this probability is based on the F-distribution, which is valid only if the distribution of the response is indeed Normal. By default, as seen in Example 3.1.2a, this probability does not appear.

The summary analysis of variance is accompanied by various statistics, determined by the settings of the `SELECTION` option. Example 3.1.2a shows the default settings for a linear regression model `%variance` (percentage variance accounted for) and `seobservations` (standard error of the observations – estimated by the square root of the residual mean square). The percentage variance accounted for is the *adjusted R^2 statistic*, expressed as a percentage:

Percentage variance accounted for = 100 × (1 – (Residual m.s.)/(Total m.s.))

Alternatively, the `adjustedr2` setting gives the adjusted R^2 statistic expressed as a proportion rather than as a percentage. The `r2` setting gives the *unadjusted R^2 statistic*, which is the square of the linear correlation between the response variate and the explanatory variate, and `%ss` gives this value as a percentage which can be interpreted as percentage sum of squares accounted for. The percentage variance accounted for is usually a better guide to the fit of a model than the unadjusted version, but you should remember that neither version is an absolute measure of fit, and both depend on the range of response and explanatory values as well as on the goodness of fit (Seber 1977). If percentage variance accounted for has a negative value, indicating a very poorly fitting model, the message `Residual variance exceeds variance of Y variate` is printed instead. The use of `SELECTION` with generalized linear models is described in Section 3.5.3.

The message below the standard error of observations is produced as a result of several checks made by Genstat on the adequacy of the model. Here, the only report concerns an apparently extreme observation in the data. This report appears for any standardized residuals whose values are particularly large: the criterion is to list residuals greater than that value c corresponding to probability $1/d$ of being exceeded in magnitude by a standard Normal deviate, where d is the number of residual degrees of freedom. However, the value c=2.0 is used instead of any smaller value when there are less than 20 residual degrees of freedom, and the value 4.0 is used instead of any larger value when there are more than 15,773 degrees of freedom. Thus, a message should appear for any extreme outlier, but messages should not appear too often just as a result of random variation.

Genstat makes five other checks on the model that can generate messages in the summary of the analysis. Examples of these can be seen in the other examples of this chapter. One check is for particularly large values of the leverage, using the criterion ck/N, where k and N are the number of parameters and number of units used in the regression model, and c is as used in the check on residuals. The sum of the leverages is always k, so this criterion brings to your attention those observations with more than about twice the average influence. Unlike the other checks, this one does not indicate a potential violation of assumptions, but rather that the analysis may be greatly affected by some observations.

If there are at least 20 observations, two checks are made on the constancy of the variance of the response variable. The fitted values are ordered into three roughly equal-sized groups; Levene tests (Snedecor & Cochran 1989) are carried out to compare the variance of the standardized residuals in the bottom group with those in the top group, and then the middle group is compared with the other two groups combined. Each test will generate a message if the test statistic is significant at the 2.5% level, indicating that the assumption of constant variance may not be tenable. Finally, a "runs" test is carried out on the standardized residuals, ordered according to the fitted values. A message is generated if the sign of successive residuals does not change often enough (again using a 2.5% significance level), indicating that there is still some

systematic pattern in the residuals.

Also, with linear and generalized linear models, whenever parameter estimates are printed the *variance inflation factor* is calculated for each parameter and a message is generated if this is greater than 100 (see Example 3.2). This is to warn that some explanatory terms are nearly aliased and that the standard errors of their parameters are consequently inflated. The parameters involved in the relationship are listed with the inflation factors. The variance inflation factor is defined to be the current diagonal value of the inverse matrix $(\boldsymbol{X}'\boldsymbol{X})^{-1}$ corresponding to the parameter, multiplied by the corrected sum of squares of the variate or dummy variate corresponding to the parameter. ($\boldsymbol{X}$ is the design matrix.) This can be interpreted as the ratio of the variance of the parameter estimate in the current model compared with that of the estimate in a model containing just that parameter and the constant. However, the check is not made if the current model contains any `POL` submodel (3.4.1), or any term involving interaction between a variate and a factor (3.3), because the dummy variates generated to represent these effects are very likely to be nearly aliased with each other. The check is also omitted if the constant term is excluded from the model.

These messages are intended to warn you about potential problems in interpreting the analysis, but cannot be relied on to detect all problems. See Cook & Weisberg (1982) for more information about these and other model-checking techniques; the `RCHECK` procedure (3.1.7 provides some further techniques.

You can prevent these messages appearing by using the `NOMESSAGE` option. They will not appear in any case if you have set option `RMETHOD=*` in the `MODEL` statement.

The `estimates` setting produced the last section of output in Example 3.1:

Example 3.1.2b

```
  14  FIT [PRINT=estimates; TPROBABILITY=yes] Boiltemp

Regression analysis
===================

Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(15)  t pr.
Constant         -42.10         3.32   -12.68  <.001
Boiltemp          0.8953       0.0163    54.77  <.001
```

The standard errors of the estimates are based here on the residual mean square. Alternatively, you can supply an estimate of variance by using the `DISPERSION` option of `MODEL`; if you do this, Genstat will print a reminder about the basis of the standard errors. You can prevent this reminder appearing by setting the `NOMESSAGE` option. The t-statistics allow you to test whether each parameter differs significantly from zero, keeping the other parameters fixed. The number of degrees of freedom for such a test is the number of residual degrees of freedom reported in the summary analysis of variance, and this number appears in the column heading. If the estimate of variance is supplied (and taken as known exactly), the "t-statistics" actually have a standard Normal distribution, indicated by the column heading "t(*)". By default, as in Example 3.1, probabilities are not printed because the distributional results depend on the assumptions underlying regression, which you need to check and confirm; but if the `TPROBABILITY` option is set (as in Example 3.1.2b), the corresponding probabilities are displayed. You can also display confidence intervals for the parameters by including the `confidence` setting. The probability value for the intervals is set by the `PROBABILITY` option; default 0.95.

You can use the `deviance` setting if you want only an abbreviated output.

Example 3.1.2c

```
  15  FIT [PRINT=deviance] Boiltemp

Residual d.f. 15, s.s. 2.127
```

The other available settings for the PRINT option are correlations, fitted, accumulated, monitoring and grid. The first two of these are illustrated in Example 3.1.2d. There is a correlation matrix of the parameter estimates, followed by a table of unit labels, values of response variate, fitted values, standardized residuals and leverages. For the unit labels, Genstat will take those associated with the response variate using the NVALUES option of the VARIATE directive (1:2.3.1), if available, or the values of the units structure (1:2.3.4). If neither is available, the integers 1...*N* are printed. If you have weighted the regression by setting the WEIGHTS option of the MODEL directive, the weights are also listed.

Example 3.1.2d

```
  16  FIT [PRINT=correlations,fitted] Boiltemp

Regression analysis
===================

Correlations between parameter estimates
----------------------------------------

Parameter     ref    correlations

Constant        1    1.000
Boiltemp        2   -1.000  1.000
                        1      2


Fitted values and residuals
---------------------------

                                  Standardized
         Unit     Response Fitted value residual Leverage
            1      131.785      132.044    -0.76     0.19
            2      131.785      131.820    -0.10     0.20
            3      135.025      135.088    -0.18     0.11
            4      135.545      135.562    -0.05     0.10
            5      136.455      136.476    -0.06     0.08
            6      136.829      136.923    -0.26     0.08
            7      137.822      137.801     0.06     0.07
            8      138.003      137.998     0.01     0.06
            9      138.057      138.177    -0.33     0.06
           10      138.211      138.132     0.22     0.06
           11      140.037      140.146    -0.30     0.06
           12      142.439      141.086     3.71     0.06
           13      145.469      145.447     0.06     0.14
           14      144.342      144.641    -0.85     0.12
           15      146.300      146.566    -0.78     0.17
           16      147.537      147.667    -0.39     0.21
           17      147.805      147.873    -0.21     0.22

Mean               139.614      139.614    -0.01     0.12
```

In the table, units are omitted according to any restriction in force or to any missing values of explanatory variates (3.1). Fitted values are shown, however, for units with zero weight or in which only the response variate is missing. Residuals are standardized as described in 3.1.1. The accumulated, monitoring and grid settings are discussed later, in 3.2.1, 3.5.3 and 3.8.2 respectively.

The CONSTANT option controls whether the constant parameter is included in the model. In simple linear regression, this parameter is the intercept, in other words the estimate of the response variable when the explanatory variable is zero. By setting CONSTANT=omit, you can prevent the constant parameter being estimated, so that the simple linear regression becomes

$$y_i = \beta x_i + \varepsilon_i$$

This model is particularly useful when y_i and x_i are measurements of the same attribute of a unit, as in calibration, and when you know that they are zero together. However, you need to be careful here: you must be sure that the relationship remains linear right down to zero.

When you omit the constant, the analysis of variance produced by PRINT=summary will not be corrected for the mean, so that the model will be compared with the null model y_i=0. (However, if the effects of factors are present in the model (3.3), setting CONSTANT=omit merely affects how the model is parameterized, and so the analysis will still be corrected for the mean.) The percentage variance accounted for will still be expressed as a percentage of the variance of the response variable about the mean. If you set CONSTANT=omit for a model containing factors without setting FULL=yes in TERMS (see 3.2.3 and 3.3.2), Genstat gives a failure diagnostic. The diagnostic can be suppressed by setting CONSTANT=ignore instead, but this should be done only in special circumstances.

The FACTORIAL option is described in 3.3.1, and the POOL, DENOMINATOR and AOVDESCRIPTION options in 3.2.1.

The NOMESSAGE option controls printing of messages. The aliasing setting is discussed in 3.2.1 and 3.2.3, and the marginality setting in 3.3.3. The leverage setting prevents messages about large leverages, and residual prevents messages about large residuals or non-constant variance or systematic pattern in the residuals. (These messages are those that are associated with the summary setting of the PRINT option.) You use the dispersion setting to prevent reminders appearing about the basis of the standard errors (as would be produced by the estimates setting of the PRINT option).

The FPROBABILITY, SELECTION and TPROBABILITY options are described above with PRINT=summary and PRINT=estimates. The NGRIDLINES, SELINEAR, INOWN and OUTOWN options are for use in the fitting of generalized non-linear models, described in Section 3.5.8.

3.1.3 Further output: the RDISPLAY directive

RDISPLAY directive

Displays the fit of a linear, generalized linear, generalized additive or nonlinear model.

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, confidence); default mode,summ,esti
CHANNEL = *identifier*	Channel number of file, or identifier of a text to store output; default current output file
DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, vertical, df, inflation); default *
FPROBABILITY = *string token*	Printing of probabilities for variance and deviance ratios (yes, no); default no
TPROBABILITY = *string token*	Printing of probabilities for t-statistics (yes, no); default

	no
SELECTION = *string tokens*	Statistics to be displayed in the summary of analysis produced by PRINT=summary, seobservations is relevant only for a Normally distributed response, and %cv only for a gamma-distributed response (%variance, %ss, adjustedr2, r2, seobservations, dispersion, %cv, %meandeviance, %deviance, aic, bic, sic); default %var, seob if DIST=normal, %cv if DIST=gamma, and disp for other distributions
DISPERSION = *scalar*	Dispersion parameter to be used as estimate for variability in s.e.s; default is as set in the MODEL statement
RMETHOD = *string token*	Type of residuals to display (deviance, Pearson, simple); default is as set in the MODEL statement
DMETHOD = *string token*	Basis of estimate of dispersion, if not fixed by DISPERSION option (deviance, Pearson); default is as set in the MODEL statement
PROBABILITY = *scalar*	Probability level for confidence intervals for parameter estimates; default 0.95
DFDISPERSION = *scalar*	Allows you to specify the number of degrees of freedom for a dispersion parameter specified by the DISPERSION option; default is as set in the MODEL statement
SAVE = *identifier*	Specifies save structure of model to display; default * i.e. that from latest model fitted

No parameters

The PRINT option has the same settings as in the FIT directive, except that no monitoring is available. The CHANNEL option selects the output channel to which the results are output, as in the PRINT directive (1:3.2); this may be a text structure, allowing output to be stored prior to display. The DENOMINATOR (3.2.1) and NOMESSAGE, FPROBABILITY, TPROBABILITY, SELECTION and PROBABILITY options are also as in the FIT directive. The DISPERSION DFDISPERSION, RMETHOD and DMETHOD options operate similarly to the options with these names in the MODEL directive, allowing you to change (temporarily – for the output produced by RDISPLAY) the way in which the dispersion parameter and residuals are calculated.

The SAVE option lets you specify the identifier of a regression save structure; the output will then relate to the most recent regression model fitted with that structure.

3.1.4 Storing the results: the RKEEP directive

RKEEP directive

Stores results from a linear, generalized linear, generalized additive or nonlinear model.

Options

EXPAND = *string token*	Whether to put estimates in the order defined by the maximal model for linear or generalized linear models (yes, no); default no
DISPERSION = *scalar*	Dispersion parameter to be used as estimate for variability in s.e.s; default as set in the MODEL directive
RMETHOD = *string token*	Type of residuals to form if parameter RESIDUALS is set

	(deviance, Pearson, simple); default as set in MODEL
DMETHOD = *string token*	Basis of estimate of dispersion, if not fixed by DISPERSION option (deviance, Pearson); default as set in MODEL
PROBABILITY = *scalar*	Probability level for confidence limits; default 0.95
OMODEL = *pointer*	Pointer to settings of options of the current MODEL statement, given unit labels corresponding to the option names of MODEL (starting with 'distribution')
PMODEL = *pointer*	Pointer to settings of parameters of the current MODEL statement, given unit labels corresponding to the parameter names of MODEL (starting with 'y'), only refers to the first setting of Y, FITTEDVALUES and RESIDUAL
STATISTICS = *variates*	Saves all the statistics that could be displayed for the first Y variate by the 'summary' setting of the PRINT option of the fitting directives FIT, ADD etc
CIMETHOD = *string token*	Method to use to calculate confidence intervals for nonlinear models (exact, quadratic); default quad
IGNOREFAILURE = *string token*	Whether to ignore failure to fit a generalized linear model (yes, no); default no
MAXIMALMODEL = *formula structure*	Saves the maximal model (as defined by TERMS)
FITMODEL = *formula structure*	Saves the currently-fitted model (including any contrast functions)
FITCONSTANT = *scalar*	Saves a scalar containing the value one if the constant is included in the fitted model, or zero otherwise
FITTYPE = *scalar*	Saves a scalar to indicate the type of model that has been fitted: 1 for an ordinary regression or generalized linear model (Sections 3.1 - 3.5), 2 for a generalized nonlinear model (Section 3.5.8), 3 for a standard curve (Section 3.7) and 4 for a nonlinear model (Section 3.8)
SAVE = *identifier*	Specifies save structure of model; default * i.e. that from latest model fitted

Parameters

Y = *variates*	Response variates for which results are to be saved; default is the list of response variates in the most recent MODEL statement
RESIDUALS = *variates*	Residuals for each Y variate, as specified by the RMETHOD option
FITTEDVALUES = *variates*	Fitted values for each Y variate
LEVERAGES = *variate*	Leverages of the units for each Y variate
ESTIMATES = *variates*	Estimates of parameters for each Y variate
SE = *variates*	Standard errors of the estimates
INVERSE = *symmetric matrix*	Inverse matrix from a linear or generalized linear model, inverse of second derivative matrix from a nonlinear model
VCOVARIANCE = *symmetric matrix*	Variance-covariance matrix of the estimates
DEVIANCE = *scalars*	Residual ss or deviance
DF = *scalar*	Residual degrees of freedom

TERMS = *pointer* or *formula structure*	Fitted terms (excluding constant)
ITERATIVEWEIGHTS = *variate*	Iterative weights from a generalized linear model
LINEARPREDICTOR = *variate*	Linear predictor from a generalized linear model
YADJUSTED = *variate*	Adjusted response of a generalized linear model
EXIT = *scalar*	Exit status from a generalized linear or nonlinear model
GRADIENTS = *pointer*	Derivatives of fitted values with respect to parameters in a nonlinear model
GRID = *variate*	Grid of function or deviance values from a nonlinear model
DESIGNMATRIX = *matrix*	Design matrix whose columns are explanatory variates and dummy variates
PEARSONCHISQUARE = *scalar*	Pearson chi-square statistic from a generalized linear model
STERMS = *pointer*	Saves the identifiers of the variates that have been smoothed in the current model
SCOMPONENTS = *pointer*	Saves a pointer to variates holding the nonlinear components of the variates that have been smoothed
NOBSERVATIONS = *scalar*	Number of units used in regression, excluding missing data and zero weights and taking account of restrictions
SEFITTEDVALUES = *variate*	Saves standard errors of the fitted values
SELINEARPREDICTOR = *variate*	Saves standard errors of the linear predictor
INFLATION = *variate*	Saves the variance inflation factors of the parameter estimates
UPPER = *variates*	Saves upper confidence limits for the parameter estimates
LOWER = *variates*	Saves lower confidence limits for the parameter estimates
MEANDEVIANCE = *scalars*	Saves the residual mean deviance (or mean square)
TDEVIANCE = *scalars*	Saves the total deviance (or sum of squares)
TDF = *scalars*	Saves the total degrees of freedom (corrected for the mean or uncorrected as displayed by the fitting directives)
TMEANDEVIANCE = *scalars*	Saves the total mean deviance (or mean square)
SUMMARY = *pointer*	Saves the summary analysis-of-variance (or deviance) table as a pointer with a variate or text for each column (source, d.f. etc)
ACCUMULATED = *pointer*	Saves the accumulated analysis-of-variance (or deviance) table as a pointer with a variate or text for each column (source, d.f. etc)
STATISTICS = *variates*	Saves all the statistics that could be displayed for the Y variate by the 'summary' setting of the PRINT option of the fitting directives FIT, ADD etc

RKEEP allows you to copy information from a regression analysis into Genstat data structures. You do not need to declare the structures in advance; Genstat will declare them automatically to be of the correct type and length. By default the information is saved from the most recently fitted model, but you can set the SAVE option to a regression save structure from another fit (saved using the SAVE option of MODEL).

The Y parameter specifies the response variates for which the results are to be saved. Unusually for the first parameter of a directive, this has a default: if you leave it out, Genstat

assumes that results are to be saved for all the response variates, as given in the previous MODEL statement.

The RESIDUALS, FITTEDVALUES, LEVERAGES and SEFITTEDVALUES parameters allow you to save the standardized residuals, the fitted values and the standard errors of the fitted values. For example, RESIDUALS=R puts the residuals in a variate R. The RMETHOD option controls the type of residuals that are formed. You cannot save these values if you had set RMETHOD=* in the MODEL statement. The standard errors of fitted values are defined by:

s.e. = √(leverage × variance function × dispersion / weight)

where the variance function is calculated from the fitted value according to the setting of the DISTRIBUTION option of the current MODEL statement, and the dispersion is the fixed or estimated value of dispersion, as controlled by the DISPERSION and DMETHOD options of the MODEL and RKEEP directives.

The ESTIMATES and SE parameters save the parameter estimates and their standard errors; RKEEP puts them in variates, using the same order as in the display produced by the PRINT option of the directive used to fit the model. Alternatively, if you have used TERMS to define a maximal model, you can set option EXPAND=yes to reorder the estimates to their order in the maximal model (missing values are inserted for the parameters not currently in the model). The variates saving these values are set up with labels (1:2.3); thus, you can refer to individual values in expressions using the labels as displayed when the estimates are fitted. For example, to get the estimate of the constant into a scalar, you could use:

```
RKEEP ESTIMATES=Esti
SCALAR Const
CALCULATE Const = Esti$['Constant']
```

The UPPER and LOWER parameters allow you to save upper and lower confidence limits for the parameter estimates. The probability for the confidence interval is specifed by the PROBABILITY option, with default 0.95. The CIMETHOD option controls the method used with nonlinear models. The default setting, quadratic, uses the same method as for other types of regression, basing the limits on a quadratic surface fitted to the likelihood surface around the optimum. These may be poor approximations if the surface is very non symmetric. The alternative setting, exact, caqlculates the limits directly from the likelihood surface.

The INFLATION parameter allows the variance inflation factors of the parameters to be saved.

The INVERSE parameter allows you to save the inverse matrix as a symmetric matrix: that is, $(\boldsymbol{X}'\boldsymbol{X})^{-1}$ where $\boldsymbol{X}$ is the design matrix. This matrix is the same for all response variates.

The VCOVARIANCE parameter saves the variance-covariance matrix of the estimates for each response variate: these are formed by multiplying the inverse matrix by the relevant variance estimate based on the estimated dispersion, or on the dispersion that you have supplied.

The DEVIANCE parameter lets you save the residual sum of squares, or the *deviance* for distributions other than Normal (3.5). The DF parameter saves the residual degrees of freedom, and the MEANDEVIANCE parameter saves the residual mean deviance. The TDEVIANCE parameter saves the total deviance, the TDF parameter saves the total degrees of freedom (corrected for the mean or uncorrected as displayed by the fitting directives), and the TMEANDEVIANCE parameter saves the total mean deviance.

The ITERATIVEWEIGHTS, LINEARPREDICTOR and YADJUSTED parameters are discussed in 3.5.6, the EXIT and GRADIENTS parameters in 3.7.4, and the GRID parameter in 3.8.1.

The DESIGNMATRIX parameter allows you to save the matrix $\boldsymbol{X}$. The columns correspond to the parameters of the model, ordered as for the ESTIMATES parameter. For simple linear regression with a constant this has only two columns, the first containing ones and the second containing the values of the explanatory variate.

The PEARSONCHI parameter provides the Pearson chi-square statistic for dispersion, which is the same as the residual sum of squares for the Normal distribution, but is different to the deviance for other distributions (3.5.1). The STERMS and SCOMPONENTS parameters are

discussed in 3.4.3.

The NOBSERVATIONS parameter allows you to save the number of units used in the analysis, omitting units with missing values or excluded by restrictions. This will be the same as the total number of degrees of freedom plus one, except in a regression with no constant term and no explanatory factors when it will equal the total number of degrees of freedom.

The DISPERSION option allows you to define the value to be used for the dispersion parameter when calculating the standard errors. The DMETHOD option indicates how this should be calculated if DISPERSION is not set. By default the deviance is used but you can set DMETHOD=Pearson to request the Pearson chi-square statistic to be used instead.

The SUMMARY parameter can be used to save the summary analysis-of-variance (or deviance) table for each response variate. The summary table is saved as a pointer with a variate or text for each of its columns (source, d.f. etc). Similarly, the ACCUMULATED parameter can save the accumulated analysis-of-variance (or deviance) tables.

The STATISTICS parameter saves all the statistics that could be displayed for each response variate by the 'summary' setting of the PRINT option of the fitting directives FIT, ADD etc. Alternatively, the STATISTICS option can be used to save the statistics for the first response variate specified by the MODEL statement.

Options OMODEL and PMODEL allow you to save pointers containing information about the current model. The labels of the pointers can be specified in either lower or upper case, or any mixture. OMODEL can be set to a pointer to store information about each of the options set in the previous MODEL statement. For example, the statement

```
RKEEP [OMODEL=Om]
```

will allow you to refer to the current variate of weights (if one was set in the WEIGHTS option of MODEL) as Om['weights']. Whether or not a variate was set, the statement

```
MODEL [WEIGHTS=Om['weights']] Newobs
```

will allow a new analysis with the same weighting as the old.

The pointer Om has 16 values, with suffixes (in lower case) corresponding to the options of MODEL in the defined order. Similarly, the statement

```
RKEEP [PMODEL=Pm]
```

will set up a pointer storing the (eight) current parameter settings of the previous MODEL statement. However, if there was more than one response variate, the first value of the pointer will be the identifier of the first response variate only: the others are not stored. Similarly, only the fitted-values and residuals variates for the first response will be pointed at. For example, the identifier Pm[1] or Pm['y'] can be used to refer to the current response variate after the RKEEP statement above.

3.1.5 Saving the results to a spreadsheet the RSPREADSHEET procedure

---

RSPREADSHEET procedure

Puts results from a regression, generalized linear or nonlinear model into a spreadsheet (R.W. Payne).

Options

DISPERSION = *scalar*	Dispersion parameter to be used as estimate for variability in s.e.s; default as set in MODEL
RMETHOD = *string token*	Type of residual to use (deviance, Pearson, simple, deletion); default * i.e. as set in MODEL
DMETHOD = *string token*	basis of estimate of dispersion, if not fixed by DISPERSION option (deviance, Pearson); default * i.e. as set in MODEL

SPREADSHEET = *string tokens*	Which spreadsheets to form (summary, estimates, fittedvalues, accumulated); default summary, estimates, fittedvalues
SPESTIMATES = *string tokens*	What to include in the estimates spreadsheet (estimates, se, testimates, prestimates); default esti, se, test, pres
SPFITTEDVALUES = *string tokens*	What to include in the fitted-values spreadsheet (y, fittedvalues, residuals, leverages, sefittedvalues); default y, fitt, resi, leve
SAVE = *regression save structure*	Specifies which analysis to save; default * i.e. most recent regression

Parameters

Y = *variates*	Y-variate of the analysis to be saved
RESIDUALS = *variates*	Identifier of variate to save the residuals from each analysis; default residuals
FITTEDVALUES = *variates*	Identifier of variate to save the fitted values from each analysis; default fittedvalues
LEVERAGES = *variates*	Identifier of variate to save the leverages from each analysis; default leverages
ESTIMATES = *variates*	Identifier of variate to save the estimates from each analysis; default estimates
SE = *variates*	Identifier of variate to save s.e.'s of the estimates from each analysis; default se
TESTIMATES = *variates*	Identifier of variate to save the t-statistics of the estimates from each analysis; default t_statistics
PRESTIMATES = *variates*	Identifier of variate to save the t-probabilities of the estimates from each analysis; default t_probabilities
SEFITTEDVALUES = *variates*	Identifier of variate to save s.e.'s of the fitted values from each analysis; default sefittedvalues
SUMMARY = *pointers*	Identifier of pointer to save the summary analysis-of-variance (or deviance) from each analysis; default summary
ACCUMULATED = *pointers*	Identifier of pointer to save the accumulated analysis-of-variance (or deviance) from each analysis; default accumulated
OUTFILENAME = *texts*	Name of Genstat workbook file (.gwb) or Excel (.xls or .xlsx) file to create

RSPREADSHEET puts results from a regression, generalized linear or nonlinear model into a spreadsheet. By default the results are from the most recent regression, but you use the SAVE option to specify the save structure (from a MODEL statement) from some other analysis. You can use the Y parameter to indicate the y-variate, if the SAVE structure contains results from more than one.

The SPREADSHEET option specifies which pages of the spreadsheet to form, with settings:

summary	summary analysis of variance (or deviance for a generalized linear model),
estimates	estimates with the standard errors etc.,
fittedvalues	fitted values, y-variate, residuals etc., and
accumulated	summary analysis of variance (or deviance for a generalized linear model).

By default, SPREADSHEET=summ,esti,fitt.

The SPESTIMATES option specifies which columns to include in the estimates spreadsheet, with settings:

estimates	estimates,
se	standard errors of estimates,
testimates	t-statistics of of estimates, and
prestimates	t-probabilities of estimates.

By default they are all included.

The SPFITTEDVALUES option specifies which columns to include in the estimates spreadsheet, with settings:

y	y-variate,
fittedvalues	fitted values,
residuals	residuals,
leverages	leverages, and
sefittedvalues	standard errors of fitted values.

By default SPFITTEDVALUES=y,fitt,resi,leve.

To help avoid clashes between the columns of the spreadsheets if you want to save results from more than one analysis, the parameters RESIDUALS, FITTEDVALUES, LEVERAGES, ESTIMATES, SE, TESTIMATES, PRESTIMATES, SEFITTEDVALUES, SUMMARY, ACCUMULATED allow you to specify identifiers for the columns (or sets of columns) that will store the corresponding results in the current spreadsheets. Their defaults are mainly the same as the parameter names, but in lower case letters. The exceptions are that TESTIMATES and PRESTIMATES have defaults t_statistics and t_probabilities, respectively.

You can save the data in either a Genstat workbook (.gwb) or an Excel spreadsheet (.xls or .xlsx), by setting the OUTFILENAME option to the name of the file to create. If the name is specified without a suffix, '.gwb' is added (so that a Genstat workbook is saved). If OUTFILENAME is not specified, the data are put into a spreadsheet opened inside Genstat.

So, you could save the summary table, estimates and fitted values etc. in an Excel spreadsheet called Boilresults.xlsx by giving the command

```
RSPREADSHEET [SPREADSHEET=summary,estimates,fittedvalues;\
              OUTFILE='Boilresults.xlsx]
```

3.1.6 Displaying the model: the RGRAPH procedure

RGRAPH procedure

Draws a graph to display the fit of a regression model (P.W. Lane).

Options

GRAPHICS = *string token*	Type of graphics to produce (lineprinter, highresolution); default high
TITLE = *text*	Title for the graph; default 'Fitted and observed relationship'
WINDOW = *number*	Which high-resolution graphics window to use; default 4 (redefined if necessary to fill the frame)
SCREEN = *string token*	Whether to clear the graphics screen before plotting (clear, keep); default clea
CIPLOT = *string token*	Whether to plot confidence intervals (no, yes); default no
CIPROBABILITY = *scalar*	Probability for confidence interval; default 0.95
BACKTRANSFORM = *string token*	What back-transformation to make (link, none, axis); default link

SAVE = *regression save structure*	Save structure of the model to display; default * uses the most recently fitted regression model

Parameters

INDEX = *variate*	Which explanatory variate to display; default * if GROUPS is set, otherwise INDEX is set to the first variate in the fitted model (must be set for nonlinear models other than standard curves)
GROUPS = *factor*	Which explanatory factor to display; default * if INDEX is set, otherwise GROUPS is set to the first factor in the fitted model (ignored for nonlinear models)

Procedure RGRAPH displays the fit of either a linear regression, a generalized linear model, a generalized additive model, a standard curve or a nonlinear model. If you have fitted several explanatory variates (as, for example in multiple linear regression, Section 3.2), you can use the INDEX parameter to specify which one is to form the x-axis. Likewise, the GROUPS parameter is relevant if you are also fitting explanatory factors, as for example in parallel regression models (3.3). With simple linear regression, there is only one explanatory variate, and so you simply need to type

```
RGRAPH
```

If you are plotting a single regression line, you can set option CIPLOT=yes to include confidence intervals for the fitted relationship. The CIPROBABILITY option sets the size of the interval; the default is 0.95 (i.e. 95%). Figure 3.1.5 shows the resulting plot (with confidence interval) for Example 3.1.

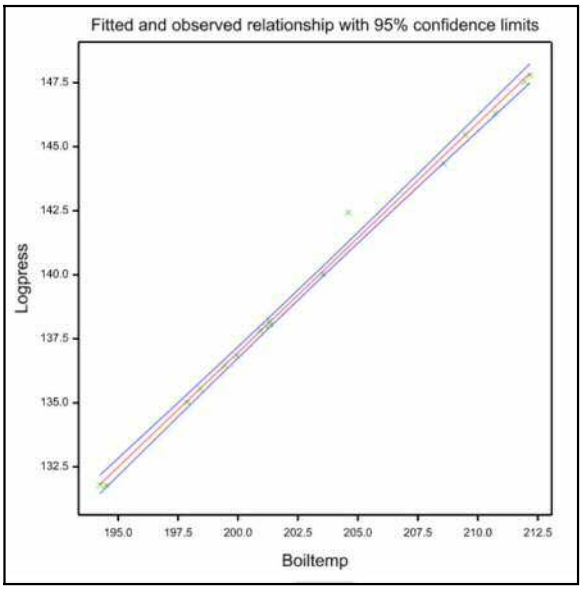


Figure 3.1.5

By default the graph is plotted on the current high-resolution device, but the GRAPHICS option can be set to line for a line-printer plot. The TITLE option allows you to supply a title for the graph. The WINDOW option can be used to select a pre-defined window for high-resolution plots; otherwise window 4 is used, and is redefined if necessary to fill the frame. The SCREEN option allows the graph to be added to an existing high-resolution plot. The colours and symbols used in the displays can be controlled by setting the attributes of the following pens with the PEN directive before calling the procedure:

pen 1	labels for lines when drawn for each level of a factor,
pen 2	fitted lines and means,
pen 3	points, and
pen 4	back-transformed axis marks and labels (see 3.5).

By default the current regression model is displayed, but option SAVE can be set to specify the save structure (from a MODEL statement) of some other model.

For models other than the nonlinear models fitted by FITNONLINEAR or FIT with the CALCULATION option set, RGRAPH plots the relationship between the response variate and either one explanatory variate or one explanatory factor or one of each. If no parameters are set,

RGRAPH takes the first explanatory variate and the first factor in the model, and the predicted relationship is represented by a line for each level of the factor. The display represents the observed relationship as points, plotting the response (adjusted for further explanatory terms in the model, if any) against the chosen explanatory variate, with each point labelled according to the corresponding factor level. If no factor has been fitted a single line is drawn, while if no variate has been fitted the graph simply shows the predicted mean for each level of the factor.

If a linear, generalized linear or generalized additive model has been fitted, the INDEX and GROUPS parameters can be used to specify which explanatory variate and factor, respectively, should be used. If INDEX is set and GROUPS is not, a single line is drawn even if there are factors in the model; similarly if GROUPS is set and INDEX is not, the effect of the factor alone is shown.

For nonlinear models fitted by the FITNONLINEAR directive, a single line is drawn by joining the fitted values, and the response values are shown as points. Any setting of the GROUPS parameter is ignored. For curves fitted by the FITCURVE directive, settings of the INDEX and GROUPS parameters are ignored and the explanatory variate and factor, if any, are determined automatically.

No graph can be drawn if the REG or COMPARISON function have been used in the model. If the SSPLINE function has been used for any variate whose relationship with the response is not actually displayed, then the only adjustment for its effect will be the linear component of the fitted smooth curve. If the displayed variate itself is smoothed, then the curve is formed by interpolation between adjusted fitted values. The POL function is dealt with correctly.

3.1.7 Diagnostics: the RCHECK procedure

RCHECK procedure

Checks the fit of a linear, generalized linear or nonlinear regression (P.W. Lane, R. Cunningham & C. Donnelly).

Options

PRINT = *string tokens*	What to print (index, y, residuals, leverages, Cook); default *
RMETHOD = *string token*	Type of residual to use (deviance, Pearson, simple, deletion); default * i.e. as set in MODEL
INDEX = *variate*	Which variate to use as index; default !(1...n)
ENVELOPE = *string token*	Type of envelope with Normal and half-Normal plots (none, rough, smooth, asymptotic); default none
PROBABILITY = *scalar*	Approximate probability level for envelope; default 0.95
NSIMULATIONS = *scalar*	How many simulations to generate for rough or smooth envelopes; default (1+PROB)/(1-PROB)
SHADE = *string token*	Whether to show shaded envelope rather than boundaries (no, yes); default no
RESIDUALS = *variate*	To store chosen type of residuals; default *
LEVERAGES = *variate*	To store leverages; default *
COOK = *variate*	To store modified Cook's statistics; default *
GRAPHICS = *string token*	Type of graphics to use (lineprinter, highresolution); default high
TITLE = *text*	Title for graph; default identifier of response
WINDOW = *numbers*	Window or series of windows in which to display graphs; default 4, or 5...8 for composite
SCREEN = *string token*	Treatment of previous graphics screen (clear, keep); default clea
SAVE = *regression save structure*	Specifies which model to check; default *

Parameters

YSTATISTIC = *string tokens*	What to display in the graph (residuals, Cook, leverages, absresiduals); default resi
XMETHOD = *string tokens*	What type of graph (fittedvalues, index, normal, halfnormal, histogram, composite); default comp

Diagnostic plots provide powerful ways of checking the assumptions underlying a regression model. If the assumptions are not satisfied, you might need to transform the y-variate or use a generalized linear model (3.5), or you could assess the model by using a permutation test (3.1.9).

The types of graph provided by RCHECK are controlled by the YSTATISTIC and XMETHOD parameters. These can be set to display various types of residuals, the leverages or the modified Cook's statistics as simple plots against fitted values or against an index variate, or as Normal or half-Normal plots, or as a histogram. The most convenient (and default) setting is composite, which displays four plots as a composite picture: histogram, plot against fitted values, Normal plot and half-Normal plot. The YSTATISTIC parameter defaults to residual, so we can obtain the standard set of diagnostic plots simply by typing

```
RCHECK
```

Figure 3.1.7 hows the resulting plot for Example 3.1.

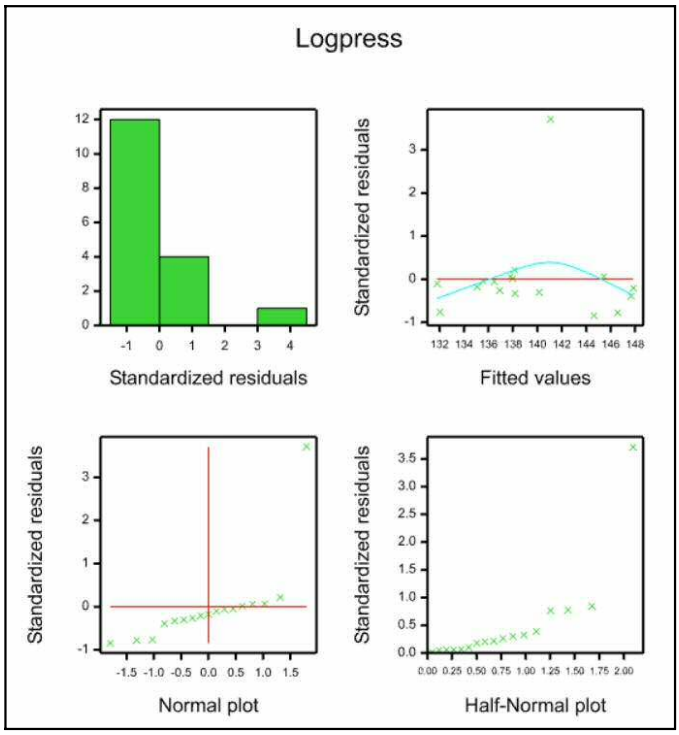


Figure 3.1.7

By default the plots are for the current regression model, but option SAVE can be set to specify the save structure (from a MODEL statement) of some other model.

The graphical displays can be controlled as usual using the GRAPHICS, TITLE, WINDOWS and SCREEN options. The colours and symbols used in the displays can be controlled by setting the attributes of the following pens with the PEN directive before calling the procedure:

pen 2	zero lines in fitted-value, Normal and index plots,
pen 3	points and histogram bars,
pen 4	smooth line in fitted-value and index plots of residuals.

The type of residual that is formed is controlled by the RMETHOD option. Most of the settings are as in MODEL (3.1.1) and RKEEP (3.1.4). Deletion residuals d_i are calculated as follows:

$$d_i = r_i / \sqrt{((n-p-r_i^2)/(n-p-1))}$$

where r_i are the standardized residuals, n is the number of observations, and p is the number of parameters in the model. For generalized linear models other than linear regression,

$$d_i = \text{SIGN}(rd_i) \times \sqrt{((1-l_i) \times rd_i^2 + l_i) \times rp_i^2)}$$

where rd_i and rp_i are the standardized deviance and Pearson residuals respectively.

The equation for the modified Cook's statistics c_i is

$$c_i = \text{ABS}(d_i) \times \sqrt{\{ (n-p) \times l_i / (p \times (1-l_i)) \}}$$

where l_i are the leverages.

In Normal plots, the Normal quantiles are calculated using the equation

q_i = NED((i−0.375) / (n+0.25))

while for a half-Normal plot they are given by

q_i = NED(0.5 + 0.5 × (i−0.375) / (n+0.25))

For generalized linear models, fitted values are transformed by an approximate variance-stabilizing transformation before use in graphs:

Poisson, multinomial, negative binomial and geometric	2 × SQRT(*fitted*)
binomial, Bernoulli	2 × ANG(100 × *fitted* / *nbinomial*)
gamma, exponential	LOG(*fitted*)
inverse Normal	1 / *fitted*

The plots of the residuals against fitted values or an index variate are displayed with a smoothed line fitted through the points, to indicate any potential trend.

Normal and half-Normal plots can be enhanced with an "envelope" by setting the `ENVELOPE` option. The `rough` setting produces an upper and lower bound for the values, and a median line, produced by simulation. The bounds correspond approximately to individual confidence intervals for each value, with probability as set by the `PROBABILITY` option (default 95%). The number of simulations by default is the minimum to allow estimation of the required limits: this is (1+`PROBABILITY`) / (1−`PROBABILITY`). A larger number of simulations can be requested with the `NSIMULATIONS` option, to give better estimates at the expense of more computing time. The `smooth` setting requests that the bounds are smoothed, using a cubic smooting spline with 4 d.f. The `asymptotic` setting produces bounds calculated from the asymptotic distribution of Normal order statistics. The envelope for all these settings can be displayed as a shaded region rather than as a set of three lines by setting the `SHADE` option to `yes`. Envelopes cannot be calculated for nonlinear models or curves, nor for generalized linear models with inverse Normal, negative binomial, geometric, multinomial or calculated distributions. Nor can they be produced for deletion residuals or Cook's statistics; they are not appropriate for leverages, which have no associated distributional assumption.

In addition to the plots, the chosen type of residuals, the leverages and Cook's statistics can be stored in variates (using options `RESIDUALS`, `LEVERAGES` and `COOK`), and any calculated quantities can be printed (using the `PRINT` option). If you do not want any plots, you can set option `GRAPHICS=*`.

The procedure exits if there are fewer than four observations, or fewer than two non-missing standardized residuals.

3.1.8 Power calculations: the `RPOWER` procedure

`RPOWER` procedure

Calculates the power (probability of detection) for regression models (R.W. Payne).

Options

`PRINT` = *string token*	Prints the power (`power`); default `powe`
`TERMS` = *formula*	Specifies the terms (x-variates, factors or model terms) to be fitted in the analysis when the responses to be detected are specified by the `RESPONSE` parameter
`FACTORIAL` = *scalar*	Limit on the number of factors or variates in a model term generated from `TERMS`; default 3
`PROBABILITY` = *scalar*	Significance level at which the response is required to be detected (assuming a one-sided test); default 0.05
`TMETHOD` = *string token*	Type of test to be made (`onesided`, `twosided`, `equivalence`, `noninferiority`, `fratio`,

	chisquare); default ones
SAVE = *rsave*	Regression save structure to provide the information about the regression model

Parameters

RESPONSE = *variates*	Variate of fitted values calculated using regression parameters of the size to be detected; default * implies that the information is to be taken from a regression save structure
RDF = *scalars*	Number of residual degrees of freedom; if unset, this is obtained from the analysis of RESPONSE or from the regression save structure
RSS = *scalars*	Anticipated residual sum of squares; if unset, this is obtained from the analysis of RESPONSE or from the regression save structure
POWER = *scalars* or *variates*	Saves the power

When planning a regression study, it can be useful to know how likely a response is to be detected. This probability of detection, known as the *power* of the study with respect to the response of interest, helps to determine whether the study is sufficiently large or accurate to achieve its purpose. RPOWER can consider any of the regression models that Genstat can analyse, and can calculate the power either for the assessment of the whole model (as represented by the regression sum of squares), or the assessment of individual parameters in the regression model.

To determine the power, you need to define the terms (x-variates, factors or model terms) to be fitted in the regression, and specify the anticipated amount of residual variability. This is most easily done by taking the analysis of a data set similar to the one to be used in the new study. To do this, you should analyse the earlier set of data with the regression directives in the usual way. Provided you do not fit any other regressions in the interim, RPOWER will pick up the information automatically from the save information held within Genstat about the most recent regression analysis. Alternatively, you can save the information explicitly in a regression save structure, by setting the SAVE option of MODEL, and then use this same save structure as the setting of the SAVE option of RPOWER.

Using a save structure allows you to specify any regression model, including any nonlinear or generalized linear model. If you merely have an ordinary linear regression model, you can set up the whole process within RPOWER if you prefer. The terms to be fitted in the model can be specified using the TERMS option of RPOWER. The setting can be a list of x-variates or a model formula, as in the setting of the parameter of the FIT directive. The FACTORIAL option, as in FIT, sets a limit on the number of factors or variates in each of the terms generated from a model formula. The constant is included automatically. (So, if you want to omit the constant and fit a regression through the origin, you should specify a save structure instead.) The RESPONSE parameter then supplies a y-variate calculated with regression parameters set to the sizes of responses to be detected.

In Example 3.1.8 we wish to check the effectiveness of an x-variate containing the values 1, 2, 5, 8 and 9. If we want to detect a regression coefficient of size at least 2.5, we would calculate the response as

```
response = 2.5 * X
```

If we also wanted to check that we can detect a constant (or intercept) of size at least 3, the calculation should become

```
response = 2.5 * X + 3
```

RPOWER analyses the RESPONSE variate using the model specified by TERMS in order to obtain

the values required to be detected for the various regression parameters.

The anticipated residual sum of squares can be specified by the RSS parameter, and the residual degrees of freedom by the RDF parameter. In Example 3.1.8 this is set to 25. The power for detecting the constant is only 0.252, but the power for detecting the regression coefficient is 0.997.

Example 3.1.8

```
  2  " define the suggested x-values "
  3  VARIATE  [VALUES=1,2,5,8,9] X
  4  " calculate the response from the fitted values
 -5    for the parameter values to be detected "
  6  CALCULATE response = 2.5 * X + 3
  7  " calculate the power, assuming a residual sum of squares of 25 "
  8  RPOWER   [TERM=X] response; RSS=25

Probability for a regression analysis
=====================================

For testing with a significance level of 0.050 using a one-sided test.

               Estimates          se       power

    Constant       3.000       2.415       0.252
           X       2.500       0.408       0.997
```

If TERMS and RSS are not set, RPOWER takes the values from the regression save structure (if this is how the model has been specified) or from the analysis of the RESPONSE variate.

The PROBABILITY option specifies the significance level that you intent to use in the analysis to detect a response; the default is 0.05 (i.e. 5%). By default, RPOWER assumes that individual regression parameters are to be assessed by a one-sided t-test, but you can set option TMETHOD=twosided to assess them by a two-sided t-test instead.

Other settings of TMETHOD enable you to test individual parameters for equivalence or for non-inferiority. With equivalence (TMETHOD=equivalence), RESPONSE defines a threshold below which the parameter can be assumed to be equivalent to no response. If the future estimate of the parameter is b and the threshold is b_{lim}, the null hypothesis for equivalence is that either

$$b \leq -b_{lim}$$

or

$$b \geq b_{lim}$$

with the alternative hypothesis that they are equivalent, i.e.

$$-b_{lim} < b < b_{lim}$$

With non-inferiority (TMETHOD=noninferiority), the null hypothesis becomes

$$b \geq -b_{lim}$$

(which represents a simple one-sided t-test).

You can also set TMETHOD=fratio, to assess the power of the F test for the regression in the summary analysis of variance (or deviance); this is an overall test for the whole regression model. Alternatively, if RPOWER is using a save structure from the analysis of a generalized linear model with a non-Normal distribution, you can set TMETHOD=chisquare to assess the power of a chi-square test on the deviance due to the regression model (see 3.5).

The POWER parameter can save the power(s), in a scalar if TMETHOD is set to fratio or chisquare; otherwise in a variate. They are printed by default, but you can set option PRINT=* to stop this.

3.1.9 Permutation and exact tests: the RPERMTEST procedure

RPERMTEST procedure

Does random permutation tests for regression or generalized linear model analyses (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (probability, accumulated, summary, critical); default prob
CONSTANT = *string token*	How to treat the constant (estimate, omit); default esti
FACTORIAL = *scalar*	Limit on the number of variates and/or factors in the terms to be fitted; default 3
NTIMES = *scalar*	Number of permutations to make; default 999
BLOCKSTRUCTURE = *formula*	Model formula defining any blocking to consider during the randomization; default none
EXCLUDE = *factors*	Factors in the block formula whose levels are not to be randomized
SEED = *scalar*	Seed for the random number generator used to make the permutations; default 0 continues from the previous generation or (if none) initializes the seed automatically
SUMMARY = *pointer*	Saves the summary analysis-of-variance (or deviance) table with permutation probabilities and critical values
ACCUMULATED = *pointer*	Saves the accumulated analysis-of-variance (or deviance) table with permutation probabilities and critical values
BINMETHOD = *string token*	How to permute binomial data (individuals, units; default indi

Parameter

TERMS = *formula*	List of explanatory variates and factors, or model formula, defining the model to fit

In regression analyses, random permutation tests provide an alternative to using the F probabilities, printed for variance ratios in summary or accumulated analysis of variance tables, when the assumptions of the analysis are not satisfied. These assumptions can be assessed by studying the residual plots produced by RCHECK (3.1.7. In particular, the use of the F distribution to calculate the probabilities is based on the assumption that the residuals from each stratum have Normal distributions with equal variances, and so the histogram of residuals produced by RCHECK should look reasonably close to the Normal, bell-shaped curve. Experience shows the analysis is robust to small departures from Normality. RPERMTEST can be useful if the histogram looks very non-Normal. You can also use RPERMTEST to generate probabilities for deviances or deviance ratios in generalized linear models, instead of using the customary chi-square or F distributions (which are justified by asymptotic theory).

Before using RPERMTEST, you need to give a MODEL statement to define the y-variate and so on, as usual for a regression or generalized model. The terms to fit in the regression model are specified by the TERMS parameter of RPERMTEST. As in the FIT directive, this can supply a list of variates for a simple or multiple linear regression, or a model formula with variates and/or factors for more complicated models. As usual, the CONSTANT option indicates whether or not to fit the constant, and the FACTORIAL option sets a limit as usual on the number of variates and/or factors in each of the terms generated from a TERMS formula.

The NTIMES option defines how many random permutations to perform; by default there are 999 (as well as the "null" permutation where the data keep their original order). The SEED option allows you to specify the seed to use for the random-number generator that is used to construct them. The default, SEED=0, continues the sequence of random numbers from a previous generation or, if this is the first use of the generator in this run of Genstat, it initializes the seed automatically (see Example 3.1.9. If NTIMES exceed the maximum possible number of permutations for the data, an"exact" test is performed in which the SETALLOCATIONS directive (1:4.3.4) is used to make every permutation once. This is feasible only for small datasets. There are *n*! (n factorial) permutations of *n* units: 3!=6, 4!=24, 5!=120, 6!=720, 7!=5040, 8!=40320, and so on.

If the regression is being used to analyse a designed experiment, you may need to use the BLOCKSTRUCTURE option to specify a block model (see 4.2) to define how to do the randomization. The EXCLUDE option can then restrict the randomization so that one or more of the factors in the block model is not randomized (see 4.11.1).

The BINMETHOD option controls how the permutations are done for binomial data. The original data set will have contained a set of units, each recording a number of "successes" obtained from an observed number of individuals. The default, and recommended, method is to expand the data set to contain individuals themselves, and permute these. Alternatively, you can set BINMETHOD=units if you prefer to permute the units as a whole instead.

The probabilities are determined from the distribution of the statistics of interest, over the permuted datasets. In an ordinary regression, the statistics are the variance ratios from the summary-of-analysis or accumulated-analysis-of-variance tables. In generalized linear models they will be deviances when the dispersion is fixed, or deviance ratios when it is estimated (as defined by the DISPERSION option of the MODEL directive; see 3.1.1 and 3.5.1).

Output is controlled by the PRINT option, with settings:

probability	to print the probability for the whole regression model;
summary	to print the summary-of-analysis table with the usual probability for the regression model replaced by the probability from the permutation test;
accumulated	to print the accumulated analysis of variance or deviance table with the usual probabilities replaced by those from the permutation test;
critical	to accompany the summary or accumulated tables by a table giving estimated critical values for each of the statistics.

The SUMMARY and ACCUMULATED options can save the summary and accumulated table, respectively. They are saved in pointers with a variate or text for each of its columns (source, d.f. etc). The probability variate contains the probabilities from the permutation test, and there are three additional variates to save the critical values.

Example 3.1.9 shows a random permutation test for relationship between the logarithm of barometric pressure and the boiling point of water, which confirms the findings in Example 3.1.

Example 3.1.9

```
  19  RPERMTEST Boiltemp
* MESSAGE: Default seed for random number generator used with value 984953

Probability for model 0.001 (determined from 999 random permutations)
```

3.2 Multiple linear regression

The model for simple linear regression can be extended by adding the effects of further explanatory variables. It is then called *multiple linear regression* and can be written:

$$y_i = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i} + ... + \beta_k x_{ki} + \varepsilon_i$$

or in matrix form:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

where the design matrix $\boldsymbol{X}$ has k+1 columns. The errors ε_i will be assumed in this section to be Normally distributed, as in Section 3.1. You can fit a multiple linear regression with the `MODEL` and `FIT` directives as before; the only change is that you now give a list of explanatory variates in `FIT`.

Likewise, in Genstat *for Windows*, multiple linear regression is straightforwardly obtained by selecting Multiple Linear Regression in the Regression list box of the Linear Regression menu. There is then an Explanatory Variates box into which you enter the required variates, instead of the (single-variate) Explanatory Variate box given when you select Simple Linear Regression. Alternatively, you can select General Linear Regression to explore different subsets of the explanatory variates. The Maximal Model field in this menu corresponds to the `TERMS` directive (3.2.3) and the subsidiary Change Model menu corresponds to the directives `ADD`, `DROP`, `SWITCH`, `TRY` and `STEP` (3.2.4, 3.2.5 and 3.2.7).

In Example 3.2, data are read from a file attached to the second input channel and a multiple linear regression is fitted for the response variable `Heat` on the four explanatory variables `X[1...4]`; the `RESTRICT` directive is used to confine the analysis to those samples that have 3.2% gypsum. Notice that a message is printed to warn that `X[2]` and `X[4]` are nearly aliased (see 3.1.2 for more details).

Example 3.2

```
  2  "  Multiple linear regression of the heat given out by setting cement
 -3     on four chemical constituents. Data from Woods, Steinour & Starke
 -4     (1932); analysed by Draper & Smith (1981) p.629."
  5  OPEN 'Cement.Dat'; CHANNEL=2
  6  READ [PRINT=data; CHANNEL=2] X[3,1,4,2],%gypsum,Heat

   1   6  7 60 26 3.2  78.5   15  1 52 29 3.2  74.3
   2   8 11 20 56 3.2 104.3    8 11 47 31 3.2  87.6
   3   6  7 33 52 3.2  95.9    9 11 22 55 3.2 109.2
   4   9 11 22 55 4.3 108.0    9 11 22 55 *   110.2
   5  17  3  6 71 3.2 102.7   22  1 44 31 3.2  72.5
   6  18  2 22 54 3.2  93.1    4 21 26 47 3.2 115.9
   7   4 21 26 47 6.5 114.0   23  1 34 40 3.2  83.8
   8   9 11 12 66 3.2 113.3    8 10 12 68 3.2 109.4
   9  18  1 61 17 3.2   *
  7  " Analyse only those samples with 3.2% gypsum."
  8  RESTRICT Heat; %gypsum==3.2
  9  MODEL Heat
 10  " Constituents are: X[1]  tricalcium aluminate
-11                      X[2]  tricalcium silicate
-12                      X[3]  tetracalcium aluminoferrite
-13                      X[4]  beta-dicalcium silicate"
 14  FIT [FPROBABILITY=yes; TPROBABILITY=yes] X[]

Regression analysis
===================

 Response variate: Heat
     Fitted terms: Constant, X[1], X[2], X[3], X[4]


Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       4      2667.90      666.975    111.48  <.001
Residual         8        47.86        5.983
Total           12      2715.76      226.314
```

```
Percentage variance accounted for 97.4
Standard error of observations is estimated to be 2.45.


Estimates of parameters
-----------------------

Parameter      estimate         s.e.      t(8)  t pr.
Constant           62.4         70.1      0.89  0.399
X[1]              1.551        0.745      2.08  0.071
X[2]              0.510        0.724      0.70  0.501
X[3]              0.102        0.755      0.14  0.896
X[4]             -0.144        0.709     -0.20  0.844

* MESSAGE: the variance of some parameter estimates is seriously inflated,
            due to near collinearity or aliasing between the following
            parameters, listed with their variance inflation factors.
X[2]              254.42
X[4]              282.51
```

One common task in multiple regression is find the subset of explanatory variables that gives the most satisfactory fit. You can search for this subset by a process of sequential modelling using the ADD, DROP, SWITCH, TRY and STEP directives. Each of these directives makes and reports changes to the current regression model, and STEP can be used to perform stepwise regression (3.2.7). It is advisable to use the TERMS directive before you start the process of sequential modelling, to define a common set of units for the regression.

An alternative is to use the RSEARCH procedure (3.2.8), which automates the various stepwise procedures, and can also evaluate all subsets of the available explanatory terms. Another way of deciding which model to fit is to perform screening tests on the available model terms. This can be done using procedure RSCREEN (3.2.9).

3.2.1 Extensions to the FIT and RDISPLAY directives in multiple linear regression

You would usually want to divide the explained variation between explanatory variables. The summary analysis of variance from the PRINT options of FIT and RDISPLAY does not do this, but there is a further setting accumulated. This divides the variation according to the order in which you listed the variables in the parameter of the FIT directive: therefore, the sum of squares for each variable ignores the effects of variables fitted later and eliminates the effect for variables already fitted. This contrasts with the t-statistics from PRINT=estimates which can be used to test the effect of each variable after eliminating the effects of all the other variables. You will find the accumulated setting useful also for summarizing changes in the regression model that you might make by the directives described later in this section. Here is the accumulated summary produced after Example 3.2.

Example 3.2.1

```
  15  RDISPLAY [PRINT=accumulated; FPROBABILITY=yes]

Regression analysis
===================

Accumulated analysis of variance
--------------------------------

Change             d.f.         s.s.         m.s.      v.r.  F pr.
+ X[1]                1     1450.076     1450.076    242.37  <.001
+ X[2]                1     1207.782     1207.782    201.87  <.001
+ X[3]                1        9.794        9.794      1.64  0.237
+ X[4]                1        0.247        0.247      0.04  0.844
Residual              8       47.864        5.983
```

```
Total                12      2715.763       226.314
```

The table shows the sum of squares and degrees of freedom attributable to each individual change in the model. As for the summary analysis of variance, if you set the FPROBABILITY option at the same time as PRINT=accumulated you get an extra column in the table with F-probabilities. By default the variance ratios are obtained by dividing the mean squares by the mean square corresponding to the smallest residual sum of squares in the table; that is from the model with fewest residual degrees of freedom.

If you do not want the sum of squares and the degrees of freedom to be subdivided between changes to the explanatory variables that you make within a statement, you should set option POOL=yes. There would then be just one entry in the table for each statement. The main use of POOL is with the ADD, DROP and SWITCH directives (3.2.4). With FIT, the POOL option merely gives the same table as you would get using the summary setting of the PRINT option.

The lines of the accumulated table are usually labelled by the names of the model terms that have been added or dropped, as shown in Example 3.2.1. When POOL=yes, however, this may become rather too long or complicated, so you can then use the AOVDESCRIPTION option (in FIT, ADD, DROP and SWITCH) to supply your own description. If you supply a missing text, the line is omitted from the table.

The DENOMINATOR option of the FIT and RDISPLAY directives can be set to produce variance ratios in the summary based on the smallest residual mean square, rather than on the mean square corresponding to the smallest residual sum of squares. You might, for example, know in advance of doing the regression that certain variables are unlikely to have a relationship with the response variable. So you would want to be able to include the sum of squares for these variables in the residual sum of squares for the other explanatory variables. You can do that by listing the interesting variables first, and these potentially uninteresting variables last, and setting DENOMINATOR=ms.

Sometimes you will find that the effect of an explanatory variable turns out to be exactly zero. This is no problem if it happens because the correlation of the explanatory variable with the response variable is itself zero. But it is a problem if it happens because the explanatory variable is a linear combination of other explanatory variables. We call this *collinearity* or *aliasing* of the explanatory variables. There is then no unique set of parameter estimates, and the method of computing information about the regression would break down, since it involves inverting a singular matrix $\boldsymbol{X}'\boldsymbol{X}$. The method also becomes unstable if the explanatory variables are nearly linearly related. Therefore Genstat tests for such a linear relationship, and will not include an explanatory variable that fails the test (3.2.3). A warning message is displayed, telling you which variable is not being included and the form of the linear relationship that has been found (see Examples 3.3.4f and 3.5.1). You can prevent the message appearing by using the aliasing setting of the NOMESSAGE option of the FIT directive.

If you then change the model, Genstat will continue to try to include this problem variable unless it is explicitly dropped. This is because the changes in the model may cause the original collinearity to disappear. If the variable is successfully included, a message is printed; again you can prevent the message appearing by the aliasing setting of the NOMESSAGE option.

3.2.2 Saving information about individual regression terms: the RKESTIMATES directive

RKESTIMATES directive

Saves estimates and other information about individual terms in a regression analysis.

Options

FACTORIAL = *scalar* Limit on number of factors and variates in a model term;

	default 3
`Y` = *variate*	Response variate for which results are to be saved; default is the last response variate in the save structure
`SAVE` = *identifier*	Provides the regression save structure for the analysis from which the estimates are to be saved; default * takes the save structure from the most recent regression

Parameters

`TERMS` = *formula*	Model terms for which information is required
`ESTIMATES` = *tables* or *scalars*	Table or scalar to store the estimated regression coefficients for each term
`SE` = *tables* or *scalars*	Table or scalar to store the standard errors of the estimated regression coefficients
`VCOVARIANCE` = *symmetric matrices*	Symmetric matrix or scalar to store the variances and covariances between the estimates of each term
`DF` = *scalars*	Number of degrees of freedom for each term
`POSITIONS` = *tables* or *scalars*	Positions of the estimates in the variate of estimates as saved from `RKEEP` when option `EXPAND=yes`

`RKESTIMATES` allows you to save estimates and other information about the individual terms in a regression analysis. Example 3.2.2 shows how you can use `RKESTIMATES` to save the estimates and standard errors from the regression in Example 3.2.

Example 3.2.2

```
16  RKESTIMATES X[]; ESTIMATES=Est[1...4]; SE=se[1...4]
17  PRINT Est[1],se[1],Est[2],se[2],Est[3],se[3],Est[4],se[4]; FIELD=10,8

  Est[1]   se[1]    Est[2]   se[2]    Est[3]   se[3]    Est[4]   se[4]
   1.551  0.7448    0.5102  0.7238    0.1019  0.7547   -0.1441  0.7091
```

The first parameter, `TERMS`, of `RKESTIMATES` specifies the terms in the regression model about which you wish to save information. Here we have a list of x-variates, `X[]`, meaning X[1], X[2], X[3] and X[4]. In more complicated situations, like those described in Section 3.3, it can be a model formula (1:1.6.3). The `FACTORIAL` option then sets a limit on the number of factors and variates in each term. Any term containing more than that limit is deleted. You can include the single-line text `'constant'` (in any case) to refer to the constant term.

The subsequent parameters allow you to specify identifiers of data structures to store the various types of information for each of the terms that you have specified. The `ESTIMATES` parameter saves estimates for each term. These are stored in a scalar if the term involves only variates (as here), or in a table if the term involves factors (see 3.3). Similarly the `SE` parameter saves standard errors for the estimates. The `VCOVARIANCE` parameter saves the variance (in a scalar) of a term that involves only variates, or the variances and covariances between the estimates (in a symmetric matrix) of a term that involves factors. The `DF` parameter saves the number of degrees of freedom for the terms, in scalars. Finally, the `POSITIONS` parameter saves the positions where the estimates can be found in the variate of estimates that would be saved by the `ESTIMATES` parameter of `RKEEP` when its option `EXPAND=yes`. (This allows you, for example, to obtain correlations between the estimates of different terms out of the variance-covariance matrix that can be saved by the `VCOVARIANCE` parameter of `RKEEP`.)

By default the results are saved from the most recent regression analysis, that is for the last y-variate in the most recent `MODEL` statement. Alternatively, you can use the `SAVE` option to

specify the save structure from another analysis (see the SAVE option of MODEL: 3.1.1). Again, the default is to save the information for the last y-variate, but you can use the Y option to specify another one.

3.2.3 Defining the maximal model: the TERMS directive

TERMS directive

Specifies a maximal model, containing all terms to be used in subsequent linear, generalized linear, generalized additive and nonlinear models.

Options

PRINT = *string tokens*	What to print (correlations, wmeans, SSPM, monitoring); default *
FACTORIAL = *scalar*	Limit for expansion of model terms; default 3
FULL = *string token*	Whether to assign all possible parameters to factors and interactions (yes, no); default no
SSPM = *SSPM*	Gives sums of squares and products on which to base calculations; default *
TOLERANCE = *scalar*	Criterion for testing for linear dependence; default is $10^7\varepsilon$, where ε is the smallest real value such that $1+\varepsilon$ is greater than 1 on the computer
DESIGNMATRIX = *matrix*	Saves the design matrix for the maximal model
MVINCLUDE = *string token*	Whether to include units with missing values in the explanatory factors and variates (explanatory); default * i.e. omit these
RIDGE = *scalar* or *variate*	Supplies values to add to the diagonal of the sums-of-squares-and-products matrix, to enable ridge methods to be used; default 0
CLDESIGNMATRIX = *text*	Saves the column labels of the design matrix for the maximal model i.e. the names of the parameters estimated in the maximal model
CLSSP = *text*	Saves the labels of the sum-of-squares-and-products matrix

Parameter

formula	List of explanatory variates and factors, or model formula

It is sensible to use the TERMS directive before starting to explore different subsets of explanatory variables, so that Genstat can define a common set of units for the regression. The directives that allow you to search through the different subsets, ADD, DROP, SWITCH, TRY and STEP, are described later in this section. TERMS initializes Genstat ready for the exploration. It overrules any model that has already been fitted with FIT, and resets the current model to be the null model.

TERMS is not essential, but problems can arise if you omit TERMS when the explanatory variates have missing values or restrictions. All the regression commands exclude any unit from the analysis if any of the response or explanatory variates has a missing value. So the set of available units will change if you include a new explanatory variate that has a missing value in a unit that was not missing for the response variate or for any of the explanatory variates already in the model. The new model will use fewer units, and so have a smaller total number of degrees of freedom. This can also happen if the new explanatory variate is restricted but the response

variate and the existing explanatory variates are not. If there is a change in the set of units like this, then the directive that makes the change to the model will display a message to draw attention to the fact. The previous model is automatically refitted with the new set of units before the new model is fitted, and the accumulated summary will show only these two fits. The message about the change in units can be suppressed by using the option setting NOMESSAGE=df in any of the fitting directives. So, although it may be convenient to omit TERMS, you should check first that there are no uneven patterns of missing values amongst the explanatory variates.

The formula specified by the parameter of TERMS should contain all the explanatory variables that you may wish to use in the subsets; if you later need to include others, you should give another TERMS statement. For multiple regression, the formula is a simple list of variates; it may include the response variates, but this is not necessary. Here is an example.

Example 3.2.3

```
  18  TERMS [PRINT=correlation] X[]

Degrees of freedom
------------------

Correlations:     11


Correlation matrix
------------------

         Heat   1  1.000
         X[1]   2  0.731  1.000
         X[2]   3  0.816  0.229  1.000
         X[3]   4 -0.535 -0.824 -0.139  1.000
         X[4]   5 -0.821 -0.245 -0.973  0.030  1.000
                       1      2      3      4      5
```

The TERMS directive actually fits a model: the null model containing only the constant term (in this case a mean). It also calculates the sums of squares and products and the means (SSPM) of the variates, including any response variates: the matrix of SSPMs is $\boldsymbol{X}'\boldsymbol{X}$, augmented by rows and columns for response variables, and is the basis of the regression calculations. The matrix is weighted if you have specified
weights in the MODEL statement, and the calculations are made within groups if you have specified a grouping factor. All units of the variates are used unless there are restrictions or missing values. You are not allowed to have different restrictions on the different vectors. Thus you can define the set of units that Genstat uses in the calculations by putting a restriction on any one of: a response variate, an explanatory variate, the weight variate, the offset variate or the groups factor. A missing value in any of these structures except a response variate will also exclude the corresponding unit. You should not alter the restriction applied to the vectors between the TERMS statement and subsequent fitting statements.

The model containing all the terms specified by the parameter of TERMS, excluding the response variates, is called the *maximal model*.

The PRINT option controls printed output, with settings:

SSPM	sums of squares and products between the variates in the model (including the response variates and dummy variates set up to represent any factors and their interactions), the means of the variates and the degrees of freedom;
correlation	the matrix of correlations between variables;
wmeans	group means for a within-group regression;
monitoring	monitoring information from the fit of the null model.

The FACTORIAL and FULL options are relevant only if there are factors in the model (3.3.1 and 3.3.2).

The SSPM option lets you use values that you have already calculated for an SSPM structure (1:2.7.2). You might find this especially useful when you are analysing very large sets of data: you can accumulate the SSPM sequentially to avoid storing all the data at once (1:4.10.3). Later regression calculations will be based on the supplied values of the SSPM, though no fitted values, residuals or leverages will be available. The values of a supplied SSPM are accepted without checking by the TERMS directive: Genstat simply assumes you are giving it something sensible.

The TOLERANCE option controls the detection of aliasing in subsequent model fitting. By default, a parameter in a linear or generalized linear model will be deemed to be aliased if the ratio between the original diagonal value of the SSPM corresponding to this parameter and the current diagonal value of the partially inverted SSPM is less than 10ε. The quantity ε depends on the computer and is defined to be the smallest real number such that the computer recognizes $1.0 + \varepsilon$ as greater than 1.0. Any positive value can be supplied by the TOLERANCE option to replace this default criterion in subsequent linear regression and generalized linear regression.

The DESIGNMATRIX option allows you to save the design matrix corresponding to the maximal model. With the RKEEP directive, you can only extract a design matrix corresponding to the currently fitted model (excluding columns corresponding to intrinsically or extrinsically aliased parameters). The CLDESIGNMATRIX option can save the column labels of the design matrix without saving the design matrix itself. (These are the names of the parameters estimated in the maximal model.)

The MVINCLUDE option allows units with missing values with missing values in factors or variates in the model to be included (by default these are excluded). Where this occurs, the factor or variate is taken to make no contribution to the fitted value for the unit concerned. This is an option that should be set only under very special circumstances. For example it is required internally by some of the procedures that fit hierarchical generalized linear models (see HGANALYSE; 3.5.11), and it may be relevant in some specialized meta analyses. It should *not* be used during ordinary analysis.

The RIDGE option enables ridge methods to be implemented. It can be set to a scalar, to define a constant to add to all the diagonal elements of the sums-of-squares-and-products matrix that correspond to the parameters in the model. Alternatively you can set RIDGE to a variate, to add a different value to each diagonal element. You may then want to use the CLSSP option to save the row labels of the sum-of-squares-and-products matrix, so that you see which rows correspond to model parameters, and which ones correspond to the y-variates. By default nothing is added (i.e. RIDGE = 0).

3.2.4 Modifying the model: the ADD, DROP and SWITCH directives

The directives ADD, DROP and SWITCH all have identical options and parameters.

ADD directive

Adds extra terms to a linear, generalized linear, generalized additive or nonlinear model.

DROP directive

Drops terms from a linear, generalized linear, generalized additive or nonlinear model.

SWITCH directive

Adds terms to, or drops them from a linear, generalized linear, generalized additive or nonlinear model.

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, confidence); default mode,summ,esti
NONLINEAR = *string token*	How to treat nonlinear parameters between groups (common, separate, unchanged); default unch
CONSTANT = *string token*	How to treat the constant (estimate, omit, unchanged, ignore); default unch
FACTORIAL = *scalar*	Limit for expansion of model terms; default * i.e. that in previous TERMS statement
POOL = *string token*	Whether to pool ss in accumulated summary between all terms fitted in a linear model (yes, no); default no
DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality, vertical, df, inflation); default *
FPROBABILITY = *string token*	Printing of probabilities for variance and deviance ratios (yes, no); default no
TPROBABILITY = *string token*	Printing of probabilities for t-statistics (yes, no); default no
SELECTION = *string tokens*	Statistics to be displayed in the summary of analysis produced by PRINT=summary, seobservations is relevant only for a Normally distributed response, and %cv only for a gamma-distributed response (%variance, %ss, adjustedr2, r2, seobservations, dispersion, %cv, %meandeviance, %deviance, aic, bic, sic); default %var, seob if DIST=normal, %cv if DIST=gamma, and disp for other distributions
PROBABILITY = *scalar*	Probability level for confidence intervals for parameter estimates; default 0.95
AOVDESCRIPTION = *text*	Description for line in accumulated analysis of variance (or deviance) table when POOL=yes

Parameter

formula	List of explanatory variates and factors, or model formula

---

You use the directives ADD, DROP and SWITCH to change the current model. Broadly, ADD lets you add extra explanatory variables, DROP lets you remove variables, and SWITCH lets you simultaneously add and remove variables.

The directives have a common syntax, which is also much the same as the syntax of the FIT directive. They modify the current regression model, which may be linear, generalized linear, generalized additive, standard curve or nonlinear. It is best to give a TERMS statement before using any of the three directives, in order to define a common set of units for the regression. If no model is fitted after the TERMS statement before an ADD, DROP or SWITCH statement (or after a MODEL statement if you decide to omit TERMS), the current model is taken to be the null model.

Here is some output that continues the example from the beginning of this section:

Example 3.2.4

```
  19  ADD [PRINT=deviance,estimates; TPROBABILITY=yes] X[1,2,4]

Regression analysis
===================

Residual d.f. 9, s.s. 47.97; Change d.f. -3, s.s. -2667.79

Estimates of parameters
-----------------------

Parameter      estimate         s.e.      t(9)  t pr.
Constant           71.6         14.1      5.07  <.001
X[1]              1.452        0.117     12.41  <.001
X[2]              0.416        0.186      2.24  0.052
X[4]             -0.237        0.173     -1.37  0.205

  20  DROP [PRINT=deviance,estimates; TPROBABILITY=yes] X[4]

Regression analysis
===================

Residual d.f. 10, s.s. 57.90; Change d.f. 1, s.s. 9.93

Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(10)  t pr.
Constant          52.58         2.29     23.00  <.001
X[1]              1.468        0.121     12.10  <.001
X[2]             0.6623       0.0459     14.44  <.001

  21  SWITCH [PRINT=estimates,accumulated; FPROBABILITY=yes;\
  22         TPROBABILITY=yes] X[2,4]

Regression analysis
===================

Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(10)  t pr.
Constant         103.10         2.12     48.54  <.001
X[1]              1.440        0.138     10.40  <.001
X[4]            -0.6140       0.0486    -12.62  <.001


Accumulated analysis of variance
--------------------------------

Change             d.f.         s.s.         m.s.      v.r.  F pr.
+ X[1]                1     1450.076     1450.076    272.04  <.001
+ X[2]                1     1207.782     1207.782    226.59  <.001
+ X[4]                1        9.932        9.932      1.86  0.205
Residual              9       47.973        5.330
- X[4]               -1       -9.932        9.932      1.86  0.205
- X[2]               -1    -1207.782     1207.782    226.59  <.001
+ X[4]                1     1190.925     1190.925    223.43  <.001

Total                12     2715.763      226.314
```

The formula specified by the parameter of each of these directives indicates the terms that are to be added or dropped, as appropriate, from the model. You must have included all of these in the formula of the previous `TERMS` statement (if you have chosen to specify `TERMS`). The terms in the formula (a list of variates in the case of multiple linear regression) are compared with those in the current regression model to form the new model.

For the ADD directive, the new model consists of all terms in the current model together with any terms in the formula; terms may appear in both the current model and the formula, in which case they will remain in the new model.

In Example 3.2.4, remember that the TERMS statement has reset the current model to be the null model. The ADD statement in line 19 thus has the same effect as the statement

```
FIT [PRINT=deviance,estimates] X[1,2,4]
```

If the ADD statement were followed by another, for example

```
ADD X[3,4]
```

then the variate X[3] would be added to the model, which would then be the same as in Example 3.2. (X[4] is already in the model.)

For the DROP directive, the new model consists of all terms in the current model excluding any that are in the formula: terms in the formula that are not in the current model are ignored. You can see this at line 20 of Example 3.2.4. If the DROP statement had instead been

```
DROP [PRINT=deviance,estimates] X[3,4]
```

it would still have had the same effect, since X[3] does not appear in the current model as defined by the previous statements.

Terms in the formula for the SWITCH directive are dropped from the current model if they are already there, and added to it if they are not. For example, if the current model consists of R and S, the effect of

```
SWITCH S,T
```

is to make a new model consisting of R and T (assuming that T was included in the previous TERMS statement).

The options of the ADD, DROP and SWITCH directives are the same as those of the FIT directive, but with the extra NONLINEAR option (see 3.7.3).

The summary analysis of variance produced by the summary setting of PRINT differs slightly from that produced by FIT in that there is an extra line called "Change". This shows the change in the Residual line since the last model. If no previous model has been fitted, the change refers to the null model.

The accumulated summary produced by the accumulated setting of the PRINT option shows all changes made to the model since the last TERMS or FIT statement, including those made by the FIT statement. You can see this after the SWITCH statement in Example 3.2.4: three terms are added, then X[4] is removed, and then X[2] is removed and X[4] reinstated. Notice the two very different sums of squares for X[4]: the smaller is the sum of squares after eliminating X[1] and X[2] while the larger is the sum of squares after eliminating X[1] but ignoring X[2]. The large difference implies that X[2] and X[4] are highly correlated after elimination of X[1]; in fact, the correlation matrix from the TERMS statement shows that they are also highly correlated ignoring X[1].

The variance ratios in the accumulated summary are calculated either from the smallest residual mean square, or from the residual mean square corresponding to the smallest residual sum of squares, depending on how the DENOMINATOR option has been set in the statement that prints the summary. In Example 3.2.4, DENOMINATOR has its default value and so the variance ratios are calculated from the residual mean square corresponding to the smallest residual sum of squares.

The model fitted by ADD, DROP or SWITCH will include a constant term if the previous model included one, and will not include one if the previous model did not. You can, however, change this using the CONSTANT option.

3.2.5 Evaluating changes to the model: the TRY directive

TRY directive

Displays results of single-term changes to a linear, generalized linear or generalized additive model.

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, changes, confidence); default chan
FACTORIAL = *scalar*	Limit for expansion of model terms; default * i.e. that in previous TERMS statement
POOL = *string token*	Whether to pool ss in accumulated summary between all terms fitted in a linear model (yes, no); default no
DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality, vertical, df, inflation); default *
FPROBABILITY = *string token*	Printing of probabilities for variance and deviance ratios (yes, no); default no
TPROBABILITY = *string token*	Printing of probabilities for t-statistics (yes, no); default no
SELECTION = *string tokens*	Statistics to be displayed in the summary of analysis produced by PRINT=summary, seobservations is relevant only for a Normally distributed response, and %cv only for a gamma-distributed response (%variance, %ss, adjustedr2, r2, seobservations, dispersion, %cv, %meandeviance, %deviance, aic, bic, sic); default %var, seob if DIST=normal, %cv if DIST=gamma, and disp for other distributions
PROBABILITY = *scalar*	Probability level for confidence intervals for parameter estimates; default 0.95

Parameter

formula	List of explanatory variates and factors, or model formula

TRY can be used to evaluate potential changes to the model. The essential difference between TRY and SWITCH is that TRY makes no permanent change to the current model. Explanatory variables are added or removed only temporarily.

The current regression model is modified by each term in the formula specified by the parameter of TRY, one term at a time, dropping terms that are in the current model and adding terms that are not. The default setting, changes, of the PRINT option summarises the effects of the changes after they have all been tried. Other settings request further details of the changed models. These are printed after each change. Genstat then restores the original model before trying the next change.

In Example 3.2.5, TRY is used to study the effect of adding either X[2] or X[3].

Example 3.2.5

```
  22  TRY X[2,3]

Changes investigated by TRY
===========================

Change                              d.f.         s.s.         m.s.

+ X[2]                                 1        26.79        26.79
+ X[3]                                 1        23.93        23.93

Residual of initial model             10        74.76         7.48
```

The only circumstances in which TRY does make a permanent change is when the current model includes a term that had been found to be aliased before this TRY statement was reached. If the aliased term can be fitted after dropping one of the terms in the TRY formula, then that is indeed done. The term that was dropped will be aliased thereafter.

The options are as in the FIT directive, except that there is no CONSTANT option. The accumulated setting of the PRINT option will show only one change at a time (see Example 3.5.1). Accumulated summaries produced by later statements will not have any entries for a TRY statement.

3.2.6 Wald tests to assess whether terms can be dropped: the RWALD procedure

RWALD procedure

Calculates Wald and F tests for dropping terms from a regression (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (waldtests); default wald
FACTORIAL = *scalar*	Limit on number of factors in the model terms generated from the TERMS parameter; default 3
Y = *variate*	Y-variate from whose analysis to calculate the statistics; default is the last y-variate in SAVE
RDF = *scalar*	Saves the residual d.f. used to calculate F probabilities when the dispersion is not fixed
SAVE = *regression save structure*	Specifies the save structure (from MODEL) containing the analysis for which to calculate the tests; default is the save structure from the most recent regression

Parameters

TERMS = *formula*	Model terms for which tests are required
WALDSTATISTIC = *scalar* or *pointer* to *scalars*	Saves Wald statistics
DF = *scalar* or *pointer* to *scalars*	Saves d.f. of Wald statistics
PROBABILITY = *scalar* or *pointer* to *scalars*	Saves the probabilities for the Wald statistics if the dispersion is fixed, or the corresponding F statistics if it is estimated

RWALD provides Wald tests to help you decide whether any terms can be dropped from a regression model. It calculates the tests from the output of the existing model, so it is quicker

than TRY (3.2.5) as that assesses the terms by changing and refitting the model. RWALD can thus be used to make a final check before you stop refining a model, or it can be used as part of a backwards stepwise process in which you fit the full model and then drop terms until all the remaining terms are essential.

By default, RWALD produces tests for all the terms that can be dropped from the most recent regression analysis, but you can set the SAVE and Y options to request tests from an earlier analysis. You can use the TERMS parameter to request Wald tests for a specific set of terms. A missing value is then given for any term that cannot be dropped.

If option PRINT=waldtests (the default), RWALD prints a table with columns containing the Wald statistic, its number of degrees of freedom and a probability value. With an ordinary linear regression, RWALD will also print an F statistic, and use this to obtain the probability. Provided there is no aliasing between the parameters of the terms, these F statistics and probabilities will be identical to those that would be printed in the Change lines of the Summary of Analysis if the terms were dropped from the model explicitly by using the DROP or TRY directives. However, as already mentioned, the advantage of RWALD is that the model does not have to be refitted (excluding each term) to calculate the information.

The use of RWALD is illustrated in Example 3.2.6, which uses the data from Example 3.2. All the available x-variates are fitted in line 24, and then RWALD is used to see which ones can be dropped from the model.

Example 3.2.6

```
  24  FIT [FPROBABILITY=yes; TPROBABILITY=yes] X[]

Regression analysis
===================

 Response variate: Heat
     Fitted terms: Constant, X[1], X[2], X[3], X[4]


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       4      2667.90      666.975    111.48  <.001
Residual         8        47.86        5.983
Total           12      2715.76      226.314

Percentage variance accounted for 97.4
Standard error of observations is estimated to be 2.45.


Estimates of parameters
-----------------------

Parameter      estimate         s.e.      t(8)  t pr.
Constant           62.4         70.1      0.89  0.399
X[1]              1.551        0.745      2.08  0.071
X[2]              0.510        0.724      0.70  0.501
X[3]              0.102        0.755      0.14  0.896
X[4]             -0.144        0.709     -0.20  0.844

* MESSAGE: the variance of some parameter estimates is seriously inflated,
           due to near collinearity or aliasing between the following
           parameters, listed with their variance inflation factors.
X[2]           254.42
X[4]           282.51

  25  RWALD
```

```
Wald tests for dropping terms
-----------------------------

Term  Wald statistic  d.f.  F statistic   F pr.
X[1]           4.337     1         4.34   0.071
X[2]           0.497     1         0.50   0.501
X[3]           0.018     1         0.02   0.896
X[4]           0.041     1         0.04   0.844

Residual d.f. 8
```

RWALD can also be used with generalized linear models; see 3.5. When the dispersion is not fixed (as for example with Normal or gamma distributions), it again gives F probabilities. However, when the dispersion is fixed (as with binomial or Poisson distributions), the probabilities are obtained by treating the Wald statistics as chi-square statistics. The deviances and deviance ratios used by TRY and DROP are calculated from the likelihoods of the generalized linear models, whereas the Wald and F statistics are essentially based on weighted sums of squares. So probabilities calculated by RWALD will no longer be identical to those given by TRY and DROP. However, both sets of probabilities are based on the asymptotic properties of their statistics, and so they should give similar conclusions.

The WALDSTATISTIC parameter can save the statistics, and the DF parameter can save their numbers of degrees of freedom. If you are making a Wald test for a single term, you can supply a scalar for each of these parameters. However, if you have several terms, you must supply a pointer which will then be set up to contain as many scalars as there are terms. Similarly the PROBABILITY parameter saves the probabilities for the Wald statistics if the dispersion is fixed, or the corresponding F statistics if it is estimated. The number residual degrees of freedom for the F statistics can be saved, in a scalar, by the RDF option. This contains a missing value if the dispersion is fixed.

3.2.7 Stepwise regression: the STEP directive

STEP directive

Selects terms to include in or exclude from a linear, generalized linear or generalized additive model according to the ratio of residual mean squares.

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, changes, confidence); default mode,summ,esti,chan
FACTORIAL = *scalar*	Limit for expansion of model terms; default * i.e. that in previous TERMS statement
POOL = *string token*	Whether to pool ss in accumulated summary between all terms fitted in a linear model (yes, no); default no
DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality, vertical, df, inflation); default *
FPROBABILITY = *string token*	Printing of probabilities for variance and deviance ratios (yes, no); default no
TPROBABILITY = *string token*	Printing of probabilities for t-statistics (yes, no); default

	no
SELECTION = *string tokens*	Statistics to be displayed in the summary of analysis produced by PRINT=summary, seobservations is relevant only for a Normally distributed response, and %cv only for a gamma-distributed response (%variance, %ss, adjustedr2, r2, seobservations, dispersion, %cv, %meandeviance, %deviance, aic, bic, sic); default %var, seob if DIST=normal, %cv if DIST=gamma, and disp for other distributions
INRATIO = *scalar*	Criterion for inclusion of terms; default 1.0
OUTRATIO = *scalar*	Criterion for exclusion of terms; default 1.0
MAXCYCLE = *scalar*	Limit on number of times to repeat stepwise selection, unless no change is made; default 1
PROBABILITY = *scalar*	Probability level for confidence intervals for parameter estimates; default 0.95

Parameter

formula	List of explanatory variates and factors, or model formula

Example 3.2.7a shows how you can use STEP to pick the "best" change to make to the set of explanatory variables at any stage.

Example 3.2.7a

```
  26  FIT [PRINT=*] X[1]
  27  STEP [INRATIO=4; OUTRATIO=4] X[1...4]

Step 1: Residual mean squares
-----------------------------

       5.790    Adding   X[2]
       7.476    Adding   X[4]
     115.062    No change
     122.707    Adding   X[3]
     226.314    Dropping X[1]

Chosen action: adding X[2].


Regression analysis
===================

 Response variate: Heat
     Fitted terms: Constant, X[1], X[2]


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       2      2657.86     1328.929    229.50  <.001
Residual        10        57.90        5.790
Total           12      2715.76      226.314

Change          -1     -1207.78     1207.782    208.58  <.001

Percentage variance accounted for 97.4
Standard error of observations is estimated to be 2.41.

* MESSAGE: the following units have high leverage.
```

```
         Unit     Response    Leverage
           12       115.90        0.55


Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(10)  t pr.
Constant          52.58         2.29     23.00  <.001
X[1]              1.468        0.121     12.10  <.001
X[2]             0.6623       0.0459     14.44  <.001
```

Example 3.2.7a starts by fitting a model containing just the term `X[1]`. Then the `STEP` statement tries, one at a time, to drop `X[1]` and to add `X[2]`, `X[3]` and `X[4]`. After each of these it reverts to the original model. Thus far, therefore, it is like a `TRY` statement. But then `STEP`, unlike `TRY`, permanently modifies the current model according to the change that was most successful. This means (putting it loosely at the moment) that if, for example, dropping `X[1]` "improves" the model, then `X[1]` is permanently removed; or, when no removals are worthwhile, if adding `X[2]` gives the biggest "improvement", then `X[2]` is permanently included. We see in fact that the latter happened, and so the current model is now as displayed at the end of Example 3.2.7a.

We now define what constitutes an "improvement" in the model. The current model is modified by each term in the formula specified by the parameter of `STEP`, one term at a time, as with `TRY` (3.2.5). For each term, the residual sum of squares and the residual degrees of freedom are recorded; then Genstat reverts to the original model before trying the next term.

The current model is finally modified by the best term, according to a criterion based on the variance ratios. Suppose that the residual sum of squares and residual degrees of freedom of the current model are s_0 and d_0, and of the model after making a one-term change are s_1 and d_1. If the variance ratio for any term that is dropped is less than the value of the setting of the `OUTRATIO` option, then the term that most reduces or least increases the residual mean square is dropped. That is, when the dispersion is being estimated, a term will be dropped only if at least one term has

$$\{(s_1-s_0) / (d_1-d_0)\} / \{s_0/d_0\} < \texttt{OUTRATIO}$$

When the dispersion is fixed, the equation becomes

$$\{(s_1-s_0) / (d_1-d_0)\} < \texttt{OUTRATIO}$$

If you have set `OUTRATIO=*`, then no term is dropped. Note that, though the criteria are ratios of variances, you should not interpret them as F-statistics with the usual interpretation of significance. The probability levels would need be adjusted to take account of correlations between the explanatory variables concerned, and the number of changes being considered.

If no term satisfies the criterion for dropping, then the term that most reduces the residual mean square will be added to the model if its variance ratio is greater than the setting of the `INRATIO` option. That is, when the dispersion is being estimated, if

$$\{(s_0-s_1) / (d_0-d_1)\} / \{s_1/d_1\} > \texttt{INRATIO}$$

When the dispersion is fixed, the equation becomes

$$\{(s_0-s_1) / (d_0-d_1)\} > \texttt{INRATIO}$$

Likewise, if you have set `INRATIO=*`, no term will be added.

If neither criterion is met, the current model is left unchanged.

The `changes` setting of the `PRINT` option produces a list of terms with the corresponding residual mean squares and residual degrees of freedom, ordered according to the sizes of the residual mean squares; you can see this in Example 3.2.7a. Note that this list is not available for display later by the `RDISPLAY` directive. The `INRATIO` and `OUTRATIO` options are explained above. The rest of the options are as in the `FIT` directive, except that there is no `CONSTANT` option.

In Example 3.2.7a, `STEP` is making a single step of a stepwise regression. The `MAXCYCLE` option allows you to request stepping to continue for a given number of cycles, or until the set

of explanatory variables stops changing. So, you can make STEP do forward selection by setting MAXCYCLE to a sufficient number of steps and setting option OUTRATIO=*: for example,

```
TERMS X[]
STEP [OUTRATIO=*; MAXCYCLE=4] X[]
```

(Four steps in enough here as we have only four potential explanatory variates.)

Similarly, you can make STEP do backward elimination, by setting MAXCYCLE with option INRATIO=*. For example:

```
TERMS X[]
FIT X[]
STEP [INRATIO=*; OUTRATIO=4; MAXCYCLE=4] X[]
```

Alternatively, you can use MAXCYCLE while supplying values for both INRATIO and OUTRATIO to do full automatic stepwise regression, as shown in Example 3.2.7b.

Example 3.2.7b

```
 27  TERMS X[]
 28  STEP [PRINT=changes; INRATIO=4; OUTRATIO=4; MAXCYCLE=10] X[]

Step 1: Residual mean squares
-----------------------------

       80.35    Adding   X[4]
       82.39    Adding   X[2]
      115.06    Adding   X[1]
      176.31    Adding   X[3]
      226.31    No change

Chosen action: adding X[4].


Step 2: Residual mean squares
-----------------------------

       7.476    Adding   X[1]
      17.574    Adding   X[3]
      80.352    No change
      86.888    Adding   X[2]
     226.314    Dropping X[4]

Chosen action: adding X[1].


Step 3: Residual mean squares
-----------------------------

       5.330    Adding   X[2]
       5.648    Adding   X[3]
       7.476    No change
      80.352    Dropping X[1]
     115.062    Dropping X[4]

Chosen action: adding X[2].


Step 4: Residual mean squares
-----------------------------

       5.330    No change
       5.790    Dropping X[4]
       5.983    Adding   X[3]
       7.476    Dropping X[2]
      86.888    Dropping X[1]

Chosen action: dropping X[4].
```

```
Step 5: Residual mean squares
-----------------------------

       5.330    Adding   X[4]
       5.346    Adding   X[3]
       5.790    No change
      82.394    Dropping X[1]
     115.062    Dropping X[2]

Chosen action: no change.

  29  RDISPLAY [FPROBABILITY=yes; TPROBABILITY=yes]

Regression analysis
===================

 Response variate: Heat
     Fitted terms: Constant, X[1], X[2]


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       2      2657.86     1328.929    229.50  <.001
Residual        10        57.90        5.790
Total           12      2715.76      226.314

Change           0     -2657.86            *

Percentage variance accounted for 97.4
Standard error of observations is estimated to be 2.41.

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
           12       115.90        0.55


Estimates of parameters
-----------------------
Parameter      estimate         s.e.     t(10)  t pr.
Constant          52.58         2.29     23.00  <.001
X[1]              1.468        0.121     12.10  <.001
X[2]             0.6623       0.0459     14.44  <.001
```

The `STEP` statement produces output for each step, so it is advisable to set the `PRINT` option, for example to `changes`, if you do not need the full details of each model in the search path. The example takes five steps to converge. First it adds `X[4]` because this term by itself gives the greatest reduction in the residual mean square. Then it adds `X[1]` similarly, followed by `X[2]`, because these terms too give greater reductions in the residual mean square than the supplied ratio of 4 (from the `INRATIO` option). But the next step is to drop `X[4]` back out of the model. This occurs because of the correlations between the explanatory variables: much of the effect of `X[4]` ignoring the other variables can be ascribed alternatively to the pair of explanatory variables `X[1]` and `X[2]`. The effect of dropping `X[4]` actually increases the residual mean square, but by less than the supplied ratio of 4 (from the `OUTRATIO` option). Finally, the last step establishes that no further single-term change can be made with the supplied criteria.

Usually, the `INRATIO` and `OUTRATIO` options will either be set to the same value or, as already explained, one will be set to `*` to enforce the method of backward elimination or of forward selection respectively. However, if the options are set to different non-missing values, it is possible for the search to alternate between two models, or get into a more complicated loop. Genstat will detect alternation, and stop; but it is not able to detect a more complicated loop and will continue cycling until the limit on the number of cycles is reached.

The `STEP` directive can be used with of difference for predictions generalized linear models as well as with linear models, but it cannot be used with nonlinear models of any kind.

3.2.8 Searching for the best regression model: the RSEARCH procedure

RSEARCH procedure

Helps search through models for a regression or generalized linear model (P.W. Goedhart).

Options

PRINT = *string token*	Printed output required (model, results); default mode, resu
METHOD = *string tokens*	Model selection method to employ (allpossible, forward, backward, fstepwise, bstepwise, accumulated, pooled); default allp
FORCED = *formula*	Model formula to include in every model; default *
CONSTANT = *string token*	How to treat the constant (estimate, omit); default esti
FACTORIAL = *scalar*	Limit for expansion of all model terms; default 3
DENOMINATOR = *string token*	Whether to base ratios in accumulated summaries on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
INRATIO = *scalar*	Criterion for inclusion of terms for forward selection, backward elimination and stepwise regression; default 1.0
OUTRATIO = *scalar*	Criterion for exclusion of terms for forward selection, backward elimination and stepwise regression; default 1.0
MAXCYCLE = *scalar*	Limit on number of times to repeat stepwise selection methods, unless no change is made; default 50
CRITERION = *string token*	Criterion for selecting best models among all possible models (r2, adjusted, cp, ep, aic, bic, sic, meandeviance, deviance); default adju
EXTRA = *string token*	Criterion which is also printed for the selected best models (r2, adjusted, cp, ep, aic, bic, sic, meandeviance, deviance); default cp when DISPERSION=*, and mean otherwise
AFACTORIAL = *scalar*	Limit for expansion of FREE model terms for the fitting of all possible models; default 3
PENALTY = *scalar*	Penalty for Mallows Cp and Akaike's information criterion AIC; default 2
NTERMS = *scalar*	Limit on the number of terms to be fitted when fitting all possible models; default 16
NBESTMODELS = *scalar*	Number of best models printed for each subset size; default 8
PPROBABILITY = *scalar*	When METHOD=allpossible, only models with all probabilities less than PPROBABILITY are printed; default 1 i.e. all models are printed
FINALMODELS = *pointer*	Pointer to save the final models for forward, backward, fstepwise and bstepwise regression methods
ALLMODELS = *pointer*	Pointer to save formulae for all possible regression models containing the fitted terms of all the models; every formula includes the FORCED formula if set
ESTIMATES = *pointer*	Pointer to save variates for all possible regression models containing the parameter estimates

SE = *pointer*	Pointer to save variates for all possible regression models containing standard errors of the parameter estimates
RESULTS = *pointer*	Pointer to save variates for all possible regression models containing the criteria (r2, adjusted, cp, ep, aic, sic, deviance, meandeviance), degrees of freedom for residual and the total number of fitted parameters p
STATISTICS = *pointer*	Pointer to save variates for all possible regression models containing the test statistics. These are F-to-delete statistics (i.e. deviance ratios) when the DISPERSION option of the MODEL directive is set to *, and Chi-square-to-delete statistics (i.e. deviance differences scaled by the dispersion parameter) for a fixed dispersion parameter
DF = *pointer*	Pointer to save variates for all possible regression models containing the degrees of freedom for the numerator of the test statistics
PROBABILITIES = *pointer*	Pointer to save variates for all possible regression models containing the probabilities of the test statistics
MARGINALTERMS = *string token*	How to treat terms that are marginal to other terms in the FREE formula (forced, free); default forc

Parameter

FREE = *formula*	Model formula specifying the candidate model terms

The forward selection, backward elimination and stepwise regression methods provided by the STEP directive (3.3.5) result in only one model, and alternative models with an equivalent or even better fit are easily overlooked. In observational studies with many correlated (or non-orthogonal) variables, there can be many alternative models, and selection of just one well-fitting model may be unsatisfactory and perhaps misleading. A preferable method may be to fit all possible regression models, and to evaluate these according to some criterion. In this way several best regression models can be selected. However the fitting of all possible regression models is very computer-intensive. It should also be used with caution, because models can be selected that appear to have a lot of explanatory power, but contain only noise variables (see for example Flack & Chang 1987). This may occur particularly when the number of parameters is large in comparison to the number of units. Terms should therefore not be selected on the basis of a statistical analysis alone.

RSEARCH can be used to perform these model selection methods. It must be preceded by a MODEL statement (3.1.1) to define the response variate and, if required, any other aspects of the model (e.g. link and distribution of a generalized linear model; see 3.5.1). Only one response variate is allowed unless the DISTRIBUTION option of MODEL is set to multinomial (3.5.5). The FREE parameter specifies the candidate model terms. These may include variates, factors and interactions (see 3.3), and regression functions like POL and SSPLINE (3.4). The METHOD option controls which model selection methods are employed:

accumulated	prints an accumulated analysis of deviance in which all model terms are added one by one to the model in the given order;
pooled	prints an accumulated analysis of deviance in which terms with the same number of identifiers, e.g. main effects or two-factor interactions, are pooled;
forward	prints an accumulated analysis of deviance resulting from

	forward selection;
backward	prints an accumulated analysis of deviance resulting from backward elimination;
fstepwise	prints an accumulated analysis of deviance resulting from stepwise regression starting with no candidate terms in the model;
bstepwise	prints an accumulated analysis of deviance resulting from stepwise regression starting with all candidate terms in the model;
allpossible	prints summary statistics for a number of best models among all possible models.

For each model with METHOD=allpossible, the selection criterion and the degrees of freedom of the included terms are printed. The probability for the hypothesis that an included term can be deleted as the last term is also printed. These probabilities are based on F-to-delete statistics (or deviance ratios for generalized linear models) when the DISPERSION option of the MODEL directive is set to *, and Chi-square-to-delete statistics (i.e. deviance differences scaled by the dispersion parameter) for a fixed dispersion parameter.

The PPROBABILITY option allows you to reduce the amount of output when METHOD=allpossible. If this is set, only models where all the probabilities are less than PPROBABILITY are printed. (By default PPROBABILITY=1, and so they are all printed.)

It is sometimes desirable to include specific terms in every model. Such terms may be specified by means of the FORCED option. The FORCED model terms are always fitted first. The CONSTANT option controls whether the constant parameter is included in the model. The limit for expanding the FREE and FORCED model formulae can be set with the FACTORIAL option, which has default value 3. The PRINT option controls the output from RSEARCH.

The criteria for inclusion and exclusion of terms for forward selection, backward elimination and stepwise regression can be specified by the INRATIO and OUTRATIO options respectively. The MAXCYCLE option specifies the number of steps. These operate exactly as in the STEP directive (3.2.7). The DENOMINATOR option controls the way in which variance ratios are calculated in accumulated analysis of deviance summaries.

All possible regression models are fitted only when the number of candidate FREE model terms does not exceed 16. If the FREE formula specifies a main effects model, i.e. a model without interactions, the main effects are the candidate terms. When the FREE formula contains interactions, the default is to remove any terms marginal to an interaction from the FREE formula, and include them instead in the FORCED formula. However, you can set option MARGINALTERMS to free to retain them in the FREE formula. Note that RSEARCH considers only models that obey the principle of marginality. This states that a model that includes an interaction term must also include all its marginal terms. For example, a model that includes the interaction A.B must also include the main effects A and B. See 3.3.1.

The AFACTORIAL option can be used to limit the expansion of the FREE model terms for the fitting of all possible regression models. The expansion is limited in addition to the limitation imposed by the FACTORIAL option. As an example, the following calls to RSEARCH result in identical candidate model terms, namely a.b, a.c, b.c and d, for all possible regression models:

```
RSEARCH [METHOD=forward,backward,allpossible;\
        FACTORIAL=3; AFACTORIAL=2] a*b*c + d
RSEARCH [METHOD=forward,backward,allpossible;\
        FACTORIAL=2; AFACTORIAL=2; FORCED=a+b+c] a*b*c + d
```

However, forward selection starts with no terms in the first call and with the model a+b+c in the second call. Backward elimination starts with the full model including the three factor interaction a.b.c in the first call, while this term is not fitted in the second call.

The CRITERION option controls the selection of the best models among all possible regression models. The criteria employed in RSEARCH are defined as follows:

r2	$100 \times [1 - \text{Dev} / \text{Dev0}]$
adjusted	$100 \times [1 - (\text{Dev} / (n-p)) / (\text{Dev0} / (n-p_0))]$
cp	$\text{Dev} / f + 2 \times p - n$
ep	$\text{Dev} \times (n+1) \times (n-2) / [n \times (n-p) \times (n-p-1)]$
aic	$\text{Dev} / f + 2 \times p$
sic or bic (synonyms)	$\text{Dev} / f + \text{Ln}(n) \times p$
deviance	Dev
meandeviance	$\text{Dev} / (n-p)$

where

Dev	is the deviance of the current model;
Dev0	is the deviance of the null model;
p	is the number of fitted parameters of the current model;
p_0	is the number of fitted parameters of the null model;
n	is the number of units;
f	is the dispersion parameter.

The null model is the model with only a constant term, which may include the fitting of a grouping factor for a within groups regression and/or the fitting of cut-points for an ordinal response model.

The dispersion parameter f is specified by the DISPERSION option of the MODEL directive or, when DISPERSION is set to *, is estimated by the mean deviance of the model with all the candidate terms. In ordinary linear regression R^2, adjusted R^2 and Mallows Cp are widely used. When R^2 is used, there is no penalty for adding a term, i.e. R^2 always improves with the addition of a term. When adjusted R^2 or Cp is employed, there is a penalty for adding a term. Adjusted R^2 improves when the F-ratio due to the addition of the term is larger than 1, while Cp improves when the F-ratio is larger than 2. Clearly, Cp is the more conservative criterion and will tend to select models with fewer terms as compared to R^2 and adjusted R^2. Minimizing Cp minimizes the mean squared error of prediction in ordinary linear regression in the case where predictions will be made at the same values as are present in the current data set. Models with negligible bias have Cp » p. For predictions at new random values, as is common in observational studies, Ep estimates the mean squared error of prediction; then Ep should be minimized. Thompson (1978) and Miller (1990) discuss Cp and Ep in detail.

Criteria suggested for generalized linear models are the Akaike information criterion (AIC) and the Schwarz (Bayesian) information criterion (SIC, or its synonym BIC). The definition of both criteria used here is different from that in the literature. The deviance is used instead of the maximum value of the log-likelihood, which implies a constant shift for distributions without dispersion parameter. Moreover, in the spirit of generalized linear models, the deviance is scaled by the dispersion parameter. This makes AIC equivalent to Cp. Clearly, SIC is the more conservative criterion, especially when the number of units is large.

Note that the best models have a small Cp, Ep, AIC, SIC, deviance and mean deviance, but a large R^2 and adjusted R^2. The default penalty of 2 in the definition of Cp and AIC can be altered by setting the PENALTY option, in which case Cp and AIC improves when the F-ratio is larger than PENALTY. The EXTRA option specifies an extra criterion which is printed alongside the selection criterion. The default for CRITERION is adjusted. The default for EXTRA is cp when DISPERSION is set to *, and meandeviance otherwise.

The NTERMS option specifies the maximum number of candidate terms in a model. This can be used when only models with few candidate terms are relevant or to reduce the computational burden. For example with 12 candidate terms there are 4096 different models, while there are only 299 models with maximally three terms. Specifying NTERMS=3 then saves a considerable amount of computing time. The NBESTMODELS option specifies the number of best models

within each subset size for which summary statistics are printed.

The FINALMODEL option can be used to save the last models for forward selection, backward elimination and fstepwise and bstepwise regression. Results of the fitting of all possible regression models can be saved by means of the parameters ALLMODELS, ESTIMATES, SE, RESULTS, STATISTICS, DF and PROBABILITIES. This saves results from all the fitted models not only from those that are printed. This includes the constant model.

All regression warnings are suppressed. This is to prevent the printing of long lists of similar warnings like "Iterative weights have become 0, or have been held at a limit". Note that the printed output of all possible regression models is adjusted to the width of the output file.

Example 3.2.8 examines all possible subsets of the explanatory variates in Example 3.2. The results confirm that there are several candidate models amongst those with two explanatory variables.

Example 3.2.8

```
  30  RSEARCH [METHOD=allpossible] X[1...4]

Model selection
===============

 Response variate: Heat
  Number of units: 13
     Forced terms: Constant
        Forced df: 1
      Free terms: X[1] + X[2] + X[3] + X[4]


All possible subset selection
=============================

* MESSAGE: probabilities are based on F-statistics, i.e. on variance ratios.

 Best subsets with 1 term

  Adjusted        Cp  Df       X[1]      X[2]      X[3]      X[4]
     64.50    138.73   2          -         -         -      .001
     63.59    142.49   2          -      .001         -         -
     49.16    202.55   2       .005         -         -         -
     22.10    315.15   2          -         -      .060         -

 Best subsets with 2 terms

  Adjusted        Cp  Df       X[1]      X[2]      X[3]      X[4]
     97.44      2.68   3       .000      .000         -         -
     96.70      5.50   3       .000         -         -      .000
     92.23     22.37   3          -         -      .000      .000
     81.64     62.44   3          -      .000      .006         -
     61.61    138.23   3          -      .687         -      .526
     45.78    198.09   3       .037         -      .587         -

 Best subsets with 3 terms

  Adjusted        Cp  Df       X[1]      X[2]      X[3]      X[4]
     97.64      3.02   4       .000      .052         -      .205
     97.64      3.04   4       .000      .000      .209         -
     97.50      3.50   4       .001         -      .070      .000
     96.38      7.34   4          -      .006      .000      .000

 Best subsets with 4 terms

  Adjusted        Cp  Df       X[1]      X[2]      X[3]      X[4]
     97.36      5.00   5       .071      .501      .896      .844
```

3.2.9 Screening tests for terms in a regression model: the RSCREEN procedure

RSCREEN procedure

Performs screening tests for generalized or multivariate linear models (H. van der Voet).

Options

PRINT = *string tokens*	Printed output required (model, pool, starscheme, tests, pvalues); default mode, pool, star
CONSTANT = *string token*	How to treat the constant (estimate, omit); default esti
FACTORIAL = *scalar*	Limit for expansion of model terms; default 3
NOMESSAGE = *string tokens*	Which warning messages to suppress when fitting the complete model (aliasing, marginality): warning messages are always suppressed when fitting models for individual tests; default *
EXCLUDEHIGHER = *string token*	Whether to exclude higher-order interactions in the conditional regression model for each tested term (yes, no); default no
FORCED = *formula*	Terms always included in the model (no tests on these terms); default *
TESTED = *text*	To save the names of individual terms which are tested
NELEMENTS = *variate*	To save the number of identifiers composing each individual term
MARGINAL = *pointer*	To save results from marginal tests for each tested term in a pointer containing the test statistic, corresponding degrees of freedom and the calculated probability
CONDITIONAL = *pointer*	To save results from conditional tests for each tested term in a pointer containing the test statistic, corresponding degrees of freedom and the calculated probability
MVINCLUDE = *string token*	Whether to include units with missing values in non-relevant explanatory variates or factors when calculating conditional and marginal tests (yes, no); default no

Parameter

FREE = *formula*	List of explanatory variates and factors, or model formula; each term from the expanded FREE formula is tested in a marginal and in a conditional test, unless the term is also part of the FORCED formula

RSCREEN provides sets of marginal and conditional tests for assessing individual terms of a linear regression model, a generalized linear model (3.5.1) or a multivariate linear model (6.6.2). RSCREEN also performs pooled testing of all main effects, of all 2-factor interactions, etc. These tests are particularly useful if you are using the regression facilities to fit a factorial model to unbalanced data, when the ordinary sequential analysis-of-variance (see Sections 4.8.1 and 4.7.4) may not give sufficient information.

A call to RSCREEN must be preceded by a MODEL statement (3.1.1) which defines the response variate(s) and, if required, a vector of weights, an offset and other aspects of a generalized linear model (3.5.1). If you define more than one response variable multivariate linear regression models are fitted (see procedure RMULTIVARIATE, Section 6.6.2), and tests are based on Rao's F approximation of Wilks' Lambda; this is possible only for ordinary linear models. If you supply

a single response variable, the tests are based on (scaled) deviances or deviance ratios, according to the setting of the DISPERSION option in the MODEL directive. Deviance ratios are always based on the mean deviance of the full model.

The FREE parameter specifies the model terms to be tested. The limit for expanding the FREE model formula can be set using the FACTORIAL option with default value 3. Two tests are performed for each term in the expanded model formula:

1. a marginal test: the term is added to the simplest possible model. For example, the main effect of A is added to the null model and the interaction term A.B is added to a model containing only main effects A and B.
2. a conditional test: the term is added to the most complex possible model containing no terms involving the term which is tested. For example, interaction A.B is added to the model with all terms except those involving A.B, like for example the interaction A.B.C. Note that e.g. the interaction C.D.E will be included in the model when testing A.B. The inclusion of any higher-order term can be prevented by setting option EXCLUDEHIGHER=yes.

It is sometimes desirable to include specific terms in every model. Such terms may be specified by means of the FORCED option. The FORCED model formula is fitted first and no test results are given for the FORCED terms. The CONSTANT option controls whether the constant parameter is included in the model.

By default any units with missing values in any of the explanatory variates or factors will be excluded from all of the tests. However, if you have many missing values that spread unevenly over the explanatory variables, there may be few units with non-missing values for every variable. If you have only a single y-variate, you may then want to set option MVINCLUDE=explanatory. RSCREEN will then use all the available units when constructing each marginal or conditional test. So it ignores missing values in any explanatory variable that is not involved in the test. This provides more information for each test, but the tables of tests should be interpreted with care as different tests may be based on different sets of units.

The PRINT option controls output. The model setting gives a description of the model. The pool setting prints an accumulated analysis of variance or deviance in which terms with the same number of identifiers, e.g. main effects or two-factor interactions, are pooled. PRINT=tests prints both marginal and conditional test statistics, while setting pvalues prints (approximate) probability values from chi-square or F-tests. Finally, PRINT=starscheme prints significance levels by a conventional star notation. The default setting of PRINT is model, pool, starscheme.

Output can be saved by means of options TESTED, NELEMENTS, MARGINAL and CONDITIONAL. TESTED saves the individual model terms in a text structure, while NELEMENTS saves the number of identifiers composing each individual term. MARGINAL and CONDITIONAL save test results in a pointer which contains four variates. These variates save the test statistic, the corresponding degrees of freedom for numerator and denominator, and the calculated (approximate) probability. For chi-square tests the degrees of freedom for the denominator are set to missing. For multivariate linear regression models, Rao's F-statistic and the corresponding degrees of freedom are saved. Note that, when MVINCLUDE=no, units with one or more missing values in any term are excluded from the analysis. This implies that FIT used for a subset of the terms may give different results than RSCREEN.

All regression warnings are suppressed, except when fitting the full model. This is to prevent the printing of long lists of similar warnings like "Iterative weights have become 0, or have been held at a limit".

If RSCREEN is used for log-linear models, with the option EXCLUDEHIGHER set to yes, the marginal and conditional tests are equal to the marginal and partial tests of Brown (1976). RSCREEN can also be used to implement the model selection strategy used in GLIMPSE, as described in McCullagh & Nelder (1989), pages 91-93. However, RSCREEN does not use

approximations for models that require an iterative fitting process.

`Rscreen` is most relevant when the regression model has terms involving factors, and especially when the terms are non-orthogonal. This is particularly likely in generalized linear models, and so an example of `RSCREEN` is given in Section 3.5.1 (Example 3.5.1).

3.3 Linear regression with grouped or qualitative data

You can incorporate the effects of grouped variables (i.e. factors) into a regression model. These are sometimes called qualitative variables to distinguish them from the quantitative ones that we have discussed so far in this chapter. For example, you could fit a separate constant term for each level of some classification: you would then get a series of parallel regression lines of the response variable on the quantitative variable. You might also want to fit separate slopes for the quantitative variable at each level of the classification.

In Example 3.3, the data from a cloud-seeding experiment include two qualitative variables, referred to as `A` and `E`; their effects are included in a linear model along with the effects of four quantitative variables referred to as `D`, `S`, `C` and `Lp`.

Example 3.3

```
  2  "  Comparison of multiple linear regressions of rainfall on associated
 -3     variables in the presence and absence of cloud seeding.
 -4     Data from Woodley et al. (1975); analysed by Weisberg (1985) p169."
  5  OPEN 'CLOUD.DAT'; CHANNEL=2
  6  FACTOR A,E
  7  READ [PRINT=data; CHANNEL=2] A,D,S,C,P,E,Y; \
  8    FREPRESENTATION=labels,4(*),levels,*

   1  NS  0 1.75 13.4 0.274 2 12.85     S  1 2.70 37.9 1.267 1  5.52
   2   S  3 4.10  3.9 0.198 2  6.29    NS  4 2.35  5.3 0.526 1  6.11
   3   S  6 4.25  7.1 0.250 1  2.45    NS  9 1.60  6.9 0.018 2  3.61
   4  NS 18 1.30  4.6 0.307 1  0.47    NS 25 3.35  4.9 0.194 1  4.56
   5  NS 27 2.85 12.1 0.751 1  6.35     S 28 2.20  5.2 0.084 1  5.06
   6   S 29 4.40  4.1 0.236 1  2.76     S 32 3.10  2.8 0.214 1  4.05
   7  NS 33 3.95  6.8 0.796 1  5.74     S 35 2.90  3.0 0.124 1  4.84
   8   S 38 2.05  7.0 0.144 1 11.86    NS 39 4.00 11.3 0.398 1  4.45
   9  NS 53 3.35  4.2 0.237 2  3.66     S 55 3.70  3.3 0.960 1  4.22
  10  NS 56 3.80  2.2 0.230 1  1.16     S 59 3.40  6.5 0.142 2  5.45
  11   S 65 3.15  3.1 0.073 1  2.02    NS 68 3.15  2.6 0.136 1  0.82
  12   S 82 4.01  8.3 0.123 1  1.09    NS 83 4.65  7.4 0.168 1  0.28
   9  " Variables are: A  Action (NS not seeded, S seeded)
 -10                   D  Days after first day of experiment
 -11                   S  Suitability for seeding (from model)
 -12                   C  Percent cloud cover
 -13                   P  Previous rainfall (in 10**7 cubic m)
 -14                   E  Type of cloud (1 or 2)
 -15                   Y  Subsequent rainfall (in 10**7 cubic m)"
  16  CALCULATE Lp,Ly = LOG10(P,Y)
  17  MODEL Ly
  18  TERMS A*(D+S+C+Lp+E)
  19  FIT [PRINT=model,estimates; FPROBABILITY=yes; TPROBABILITY=yes]\
  20      A+S+D+C+Lp+E

Regression analysis
===================

 Response variate: Ly
     Fitted terms: Constant + A + S + D + C + Lp + E
```

```
Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(17)  t pr.
Constant          1.030        0.381      2.70  0.015
A S               0.274        0.149      1.84  0.083
S                -0.0817       0.0966    -0.85  0.410
D               -0.00604      0.00359    -1.68  0.111
C                -0.0049       0.0119    -0.41  0.689
Lp                0.348        0.240      1.45  0.165
E 2               0.340        0.195      1.74  0.099

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                   A  NS
                   E  1

  21  ADD [PRINT=model,estimates,accumulated; FPROBABILITY=yes;\
  22      TPROBABILITY=yes] S.A

Regression analysis
===================

 Response variate: Ly
     Fitted terms: Constant + A + S + D + C + Lp + E + S.A


Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(16)  t pr.
Constant          0.614        0.368      1.67  0.115
A S               1.670        0.559      2.99  0.009
S                 0.107        0.111      0.96  0.350
D               -0.00925      0.00336    -2.76  0.014
C                -0.0128       0.0108    -1.18  0.253
Lp                0.379        0.208      1.82  0.087
E 2               0.470        0.177      2.66  0.017
S.A S            -0.430        0.167     -2.57  0.021

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                   A  NS
                   E  1


Accumulated analysis of variance
--------------------------------

Change                          d.f.         s.s.         m.s.      v.r.  F pr.
+ A                                1      0.19498      0.19498      2.20  0.158
+ S                                1      0.38967      0.38967      4.39  0.052
+ D                                1      0.86460      0.86460      9.75  0.007
+ C                                1      0.00214      0.00214      0.02  0.878
+ Lp                               1      0.08127      0.08127      0.92  0.353
+ E                                1      0.35882      0.35882      4.05  0.061
+ S.A                              1      0.58560      0.58560      6.60  0.021
Residual                          16      1.41926      0.08870

Total                             23      3.89635      0.16941
```

Before we go into details, look at the `FIT` statement in lines 19 and 20. `A` is a factor with two levels labelled `NS` and `S`, and `E` also has two levels, 1 and 2. This statement fits a multiple linear regression of the variate `Ly` on the variates `S`, `D`, `C` and `Lp`; the model also includes the *main effects* of the factors `A` and `E`. This means that for each factor an additive constant is estimated, representing the mean difference between the responses at the two levels of the factor. In other words, a set of parallel linear regressions is fitted, one for each combination of levels of the two factors.

Now look at the ADD statement. Here the interaction between the factor A and the variate S is included too. This means that different effects of the variate S are estimated for each level of A. In other words, separate linear regressions are fitted as before, except that the fitted relationships between Ly and S for each level of A are not constrained to be parallel.

We now make some more formal definitions, after which we shall return to this example.

You store data from qualitative variables in factors (1:2.3.3). After factors have been declared and assigned values, their effects can be included in regression models. You do this by putting their identifiers in directives such as FIT and TERMS, along with the identifiers of variates storing the values of quantitative explanatory variables.

You represent the *main effect* of a factor by its identifier as a single term: a model including such a main effect has a separate constant or intercept for each level of the factor.

Interactions between factors allow more detailed modelling of the constant term for combinations of levels of more than one factor. They are represented by terms consisting of the dot operator between factor identifiers in formulae.

Interactions between factors and variates allow modelling of the changes in the regression coefficient of the variate between combinations of levels of factors. They too are represented by terms including dot operators.

"Interactions" between quantitative variables can also be expressed in this way. They simply represent the product of two or more variates.

3.3.1 Formulae in parameters of regression directives

Formulae are described in 1:1.6.3, and further details are given in 4.1.1. In regression directives you cannot use the // operator, nor the functions POLND and REGND. The functions POL, COMPARISON, REG, SSPLINE and LOESS can be used to represent polynomial effects, general sets of contrasts and nonparametric smoothed effects; these are described in 3.4. The basic operators are those of summation (+) and dot product (.), and if you want you can write all formulae using just these two. The other operators provide a shorthand for representing complicated formulae. Of particular use in regression are the cross-product operator (*)

```
A*B = A + B + A.B
```

and the nesting operator (/)

```
A/B = A + A.B
```

For more complicated formulae, remember that the nesting operator is not distributive (see 1:1.6.3 and 4.1.1): for example,

```
(A + B)/C = A + B + A.B.C
```

Terms are ignored if they are put in an invalid order. For example the formula A.B + A becomes just A.B, since A is *marginal* to A.B. Genstat takes care to avoid fitting uninterpretable models that violate the principles of marginality, and will not accept any model where a term is specified before any of its margins (3.3.3).

If a formula contains commas, they are treated in the same way as + operators together with pairs of brackets. For example, X,Y*A is the same as (X+Y)*A, which is X+Y+A+X.A+Y.A.

The expansion of formulae into constituent terms is controlled in all regression directives by the FACTORIAL option. The default setting is 3, which excludes all interactions involving more than three identifiers. For example,

```
FIT [FACTORIAL=2] A*B*C
```

will fit a model that includes the terms A, B, C, A.B, A.C and B.C, but excludes A.B.C. However, following a TERMS statement, the default of FACTORIAL in other regression statements is whatever was set or implied by default in TERMS.

3.3.2 Parameterization of factors

A regression model that includes the main effect of a single factor and omits the constant (3.1.2), contains one parameter for each level of the factor: this parameter represents the constant term for that level. If an explicit constant term is also included in the model, then some constraint must be applied to the parameters for the factors. In Genstat, the parameter corresponding to the *reference level* of the factor is set to zero. The reference level is specified using the REFERENCELEVEL option of the FACTOR directive (1:2.3.3). If it is not set, as in line 6 of Example 3.3, Genstat takes the first level of the factor as the reference level. For example, in the first model fitted by lines 19 and 20 of Example 3.3, the parameter estimates are:

Example 3.3.2a

```
Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(17)  t pr.
Constant          1.030        0.381      2.70  0.015
A S               0.274        0.149      1.84  0.083
S               -0.0817       0.0966     -0.85  0.410
D              -0.00604      0.00359     -1.68  0.111
C               -0.0049       0.0119     -0.41  0.689
Lp                0.348        0.240      1.45  0.165
E 2               0.340        0.195      1.74  0.099
```

No parameter estimate is shown for "A NS" or for "E 1". You can interpret the constant term here as the constant when both these factors are at their reference levels, level 1: that is, on days when there was no seeding and the cloud was of Type 1. Thus the parameter labelled "A S" is the difference between the constant for days with and without seeding. The same is true for factors with more than two levels: the parameters all represent differences from the first level. So it makes sense to use the level representing the standard conditions (for example the placebo in a drug trial, or the control variety in a variety trial) as the reference level. If, however, there are no observations at the reference level of a factor, any fitting statement will display a warning, and will change the reference level to the first level of that factor for which there are observations.

This form of parameterization makes it easy to compare each level of a factor with the reference level. In the example, the t-statistic of the estimate for "A S" shows that the difference between the constants for the levels of A is not quite significant at the 5% level.

You may not necessarily find these parameters very convenient for summarizing the effect of a factor, especially when there are several levels, or several factors in a model. Instead you may wish to use the PREDICT directive to produce summaries (3.3.4), unless the methods of Chapter 4 or 5 are relevant.

You can obtain other parameterizations by modifying the definition of the model. For example, you can fit a constant for each level of factor A by setting option CONSTANT=omit in FIT:

Example 3.3.2b

```
  23  MODEL Ly
  24  FIT [PRINT=estimates; CONSTANT=omit; TPROBABILITY=yes] A+S+D+C+Lp+E

Regression analysis
===================

Estimates of parameters
-----------------------
```

```
Parameter      estimate         s.e.     t(17)  t pr.
A NS              1.371        0.432      3.17  0.006
A S               1.645        0.480      3.43  0.003
S               -0.0817       0.0966     -0.85  0.410
D              -0.00604      0.00359     -1.68  0.111
C               -0.0049       0.0119     -0.41  0.689
Lp                0.348        0.240      1.45  0.165
E 1              -0.340        0.195     -1.74  0.099
E 2                   0            *         *      *
```

Since there is no constant term in this model, no constraint needs to be imposed on the parameters representing factor A. However, the parameterization of factor E must still be constrained as before. Genstat always chooses to parameterize the first factor in the model fully when the constant is omitted; so to get E fully parameterized you should put E before A in the FIT statement.

If you want to fit a sequence of models and use any form of parameterization other than the standard one (including the constant), you must set option FULL=yes in the TERMS statement. This is because TERMS allocates the number of parameters for each term in the model, and automatically imposes constraints when there is over-parameterization. The setting FULL=yes specifies that a parameter is to be associated with every level of each factor, regardless of the presence of a constant term. If you include a constant term in a model as well as some factors, you will again find that one of the parameters of each factor will be aliased. Similarly, if you omit the constant and fit more than one factor, each factor other than the first will also have an aliased parameter. If you set CONSTANT=omit and try to fit a model containing factors without setting FULL=yes, Genstat gives a failure diagnostic. The diagnostic can be suppressed by setting CONSTANT=ignore in FIT, ADD, DROP or SWITCH, but this should be done only in special circumstances (for example this setting is used inside the procedure HGANALYSE which fits hierarchical generalized linear models; 3.5.11).

Example 3.3.2c

```
  25  TERMS [FULL=yes] A*(D+S+C+Lp+E)
  26  FIT [PRINT=estimates; CONSTANT=omit; TPROBABILITY=yes] A+S+D+C+Lp+E

Regression analysis
===================

Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(17)  t pr.
A NS              1.371        0.432      3.17  0.006
A S               1.645        0.480      3.43  0.003
S               -0.0817       0.0966     -0.85  0.410
D              -0.00604      0.00359     -1.68  0.111
C               -0.0049       0.0119     -0.41  0.689
Lp                0.348        0.240      1.45  0.165
E 1              -0.340        0.195     -1.74  0.099
E 2                   0            *         *      *
```

The last level of the factor E is aliased in both Example 3.3.2b and Example 3.3.2c since this is the last parameter to be fitted, and its estimate is left as 0. Notice that no reports are given on partial aliasing of terms involving factors when the constant is omitted or when FULL=yes, regardless of the setting of the NOMESSAGE option of the FIT directive.

Factor effects are also fully parameterized if an SSPM structure, supplied through the SSP option of the TERMS directive, was declared by an SSPM statement (1:2.7.2) with option FULL set to yes.

3.3.3 Parameterization of interactions, and marginality

The parameters representing interactions in a model are also constrained to remove over-parameterization.

For example, suppose A and B are factors with two and three levels respectively with their first levels as reference level. If the model A*B is fitted (including a constant), the parameters will be: Constant, A2, B2, B3, A2.B2 and A2.B3. No parameter is assigned to A1 because there is a constant, and none to B1 or A1.B1. Similarly, no parameter is assigned to A2.B1 because the main effect of A is included, and none to A1.B2 nor A1.B3 because the main effect of B is included. The terms A and B are described as being *marginal* to the term A.B. The constant term is also marginal to A and B, and to the term A.B.

In general, one term is marginal to a second if the second can be written as an interaction between the first term and a third term involving factors only; for example, A is marginal to A.B and to A.B.C.D. Whenever one term is marginal to a second, some parameters of the full set of the second term are aliased with the first term. Genstat will automatically constrain selected parameters to be zero to avoid aliasing. The automatic constraint can be removed by setting the FULL option of the TERMS directive.

In the analysis fitted in lines 21 and 22 of Example 3.3, the fitted model is

```
A + S + D + C + Lp + E + S.A
```

The term S.A is an interaction between a factor and a variate, and so represents variations in the effect of the variate between levels of the factor: that is, the regression lines of Ly on S are allowed to have separate slopes for the days with and without seeding, as well as separate intercepts. The linear model is

$$y_{ijk} = \alpha + \gamma_{1i} + \beta_1 x_{1ijk} + \beta_2 x_{2ijk} + \beta_3 x_{3ijk} + \beta_4 x_{4ijk} + \gamma_{2j} + \delta_i x_{1ijk} + \varepsilon_{ijk}$$
$$\text{for } i = 1, 2; j = 1, 2; k = 1 \ldots N_{ij}$$

where α represents the constant term, and is the intercept for "A NS" and "E 1". The parameters γ_{1i} and γ_{2j} represent the main effects of A and E: γ_{1i} is the difference between the intercept for the ith level of A and that for the first level (labelled "A NS"), so that γ_{11} is zero. The parameter β_1 represents the variate S, and is the slope for "A NS" and "E 1". Lastly, the parameters δ_i represent the interaction term S.A; δ_i is the difference between the slope for the ith level of A and that for the first level, so that δ_1 is zero. In this model, the constant is marginal to the terms A and E, and S is marginal to S.A.

Again, you can present the results differently, either using the PREDICT directive (3.3.4) or by modifying the model. The parameters can be made to be the actual slopes by omitting S from the model, as long as you have set option FULL=yes in TERMS:

Example 3.3.3a

```
  27  FIT [PRINT=estimates; CONSTANT=omit; TPROBABILITY=yes] A+D+C+Lp+E+S.A

Regression analysis
===================

Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(16)  t pr.
A NS              1.084        0.391      2.77  0.014
A S               2.754        0.600      4.59  <.001
D              -0.00925      0.00336     -2.76  0.014
C               -0.0128       0.0108     -1.18  0.253
Lp                0.379        0.208      1.82  0.087
E 1              -0.470        0.177     -2.66  0.017
E 2                   0            *         *      *
S.A NS            0.107        0.111      0.96  0.350
S.A S            -0.323        0.126     -2.57  0.021
```

If option FULL had been left at its default setting no, the FIT statement would fail:

Example 3.3.3b

```
  28  TERMS A*(D+S+C+Lp+E)
  29  FIT [PRINT=*; CONSTANT=omit] A+D+C+Lp+E+S.A

* MESSAGE: term A cannot be added because term Constant is marginal to it
and is not in the model.

* MESSAGE: term E cannot be added because term Constant is marginal to it
and is not in the model.

* MESSAGE: term S.A cannot be added because term S is marginal to it
and is not in the model.
```

The messages about marginality can be suppressed by using the marginality setting of the NOMESSAGE option of the FIT directive.

As an alternative to setting FULL=yes, you could omit the marginal terms from the TERMS statement as well; above you would need to omit the effect of S. However, the constant cannot be omitted in TERMS.

3.3.4 Forming predictions: the PREDICT directive

PREDICT directive

Forms predictions from a linear or generalized linear model.

Options

PRINT = *string token*	What to print (description, lsd, predictions, se, sed, vcovariance); default desc, pred, se
CHANNEL = *scalar*	Channel number for output; default * i.e. current output channel
COMBINATIONS = *string token*	Which combinations of factors in the current model to include (full, present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment (marginal, equal); default marg
WEIGHTS = *table*	Weights classified by some or all of the factors in the model; default *
OFFSET = *scalar*	Value of offset on which to base predictions; default mean of offset variate
METHOD = *string token*	Method of forming margin (mean, total); default mean
ALIASING = *string token*	How to deal with aliased parameters (fault, ignore); default faul
BACKTRANSFORM = *string token*	What back-transformation to apply to the values on the linear scale, before calculating the predicted means (link, none); default link
SCOPE = *string token*	Controls whether the variance of predictions is calculated on the basis of forecasting new observations rather than summarizing the data to which the model has been fitted (data, new); default data
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, nonlinear); default *
DISPERSION = *scalar*	Value of dispersion parameter in calculation of s.e.s;

	default is as set in the MODEL statement
DMETHOD = *string token*	Basis of estimate of dispersion, if not fixed by DISPERSION option (deviance, Pearson); default is as set in the MODEL statement
NBINOMIAL = *scalar*	Supplies the total number of trials to be used for prediction with a binomial distribution (providing a value *n* greater than one allows predictions to be made of the number of "successes" out of *n*, whereas the value one predicts the proportion of successes); default 1
PREDICTIONS = *tables* or *scalars*	Saves predictions for each y variate; default *
SE = *tables* or *scalars*	Saves standard errors of predictions for each y variate; default *
SED = *symmetric matrices*	Saves standard errors of differences between predictions for each y variate; default *
LSD = *symmetric matrices*	Saves least significant differences between predictions for each y variate (models with Normal errors only); default *
LSDLEVEL = *scalar*	Significance level (%) to use in the calculation of least significant differences; default 5
VCOVARIANCE = *symmetric matrices*	Saves variance-covariance matrices of predictions for each y variate; default *
SAVE = *identifier*	Specifies save structure of model from which to predict; default * i.e. that from latest model fitted

Parameters

CLASSIFY = *vectors*	Variates and/or factors to classify table of predictions
LEVELS = *variates*, *scalars* or *texts*	To specify values of variates, levels of factors
PARALLEL = *identifiers*	For each vector in the CLASSIFY list, allows you to specify another vector in the CLASSIFY list with which the values of this vector should change in parallel (you then obtain just one dimension in the table of predictions for these vectors)
NEWFACTOR = *identifiers*	Identifiers for new factors that are defined when LEVELS are specified

The PREDICT directive provides a convenient way of summarizing the results of a regression, by using the fitted relationship to predict the values of the response variate at particular values of the explanatory variables. In simple or multiple linear regression, the parameters of the model may be sufficient summaries in themselves, but these may not provide a very clear description when the model contains factors and their interactions. PREDICT can also be used to answer "what-if" questions, effectively predicting what fitted values would have been obtained if the data had been balanced in some way.

The simplest use of PREDICT is to make estimates from a simple linear regression for specific values of the explanatory variable. For example, if we had regressed Ly on just S in the example above, we could get the predicted value of Ly at S = 3.5 (say) by putting

```
PREDICT S; LEVELS=3.5
```

If we wanted the predicted values at 3.5 and 4, we would have to put these into a variate. The easiest way to do that is to use an unnamed variate (1:1.4.3):

```
PREDICT S; LEVELS=!(3.5,4)
```

Suppose now that we had regressed Ly on both S and C, and wanted to predict the value of Ly at S = 3.5 and 4 and C = 4, 8 and 12. We would then put 3.5 and 4 into one variate, and 4, 8 and 12 into another:

```
PREDICT S,C; LEVELS=!(3.5,4),!(4,8,12)
```

This would give six predicted values, one for each combination of 3.5 and 4 with 4, 8 and 12.

If we had also included the factor E in the regression, we might want to predict Ly for S equal to 3.5 at both levels 1 and 2 of E:

```
PREDICT E,S; LEVELS=!(1,2),3.5
```

This would produce two predicted values, classified by the levels of E. Since C is not mentioned in the PREDICT statement, the predictions will be based on the mean value of C by default. It is not actually necessary to list the levels of E if predictions are wanted for all of them; we could thus have put:

```
PREDICT E,S; LEVELS=*,3.5
```

If the factor A was also in the model, we could still use either of the previous two statements to get a summary of the effects of E. Since there is no mention of A, the predictions would automatically be averaged over the levels of A, as described later in this section.

For more complicated structures the rules are more intricate, as we shall see. But the basic ideas remain the same as in the simpler cases. In Example 3.3.4a, we summarize the model fitted at line 21 and 22 of Example 3.3, for every combination of levels of the two factors.

Example 3.3.4a

```
  20  FIT [PRINT=*] A+S+D+C+Lp+E+S.A
  31  PREDICT A,E

Predictions from regression model
---------------------------------

These predictions are estimated mean values.

The predictions have been formed only for those combinations of factor levels
for which means can be estimated without involving aliased parameters.

The predictions are based on fixed values of some variates:
        Variate   Fixed value   Source of value
              D         35.33   Mean of variate
              S         3.169   Mean of variate
              C         7.246   Mean of variate
             Lp       -0.6489   Mean of variate

The standard errors are appropriate for interpretation of the predictions as
summaries of the data rather than as forecasts of new observations.

Response variate: Ly

            E           1                       2
               Prediction        s.e.  Prediction        s.e.
            A
           NS       0.2883      0.0995      0.7582      0.1583
            S       0.5958      0.0918      1.0656      0.1799
```

The four values are estimates, based on the fitted model, of the mean logged rainfall at the mean values of the four explanatory variates.

By using the LEVELS parameter, we can ask for the summary to be calculated for cloud-type 2 only, for a range of suitability values (variate S), and as if all observations were made on the first day of the experiment (D=0).

Example 3.3.4b

```
  32  PREDICT S,A,E,D; LEVELS=!(1...4),*,2,0

Predictions from regression model
---------------------------------

These predictions are estimated mean values.

The predictions have been formed only for those combinations of factor levels
for which means can be estimated without involving aliased parameters.

The predictions are based on fixed values of some variates:
        Variate   Fixed value   Source of value
              D            0.   Supplied
              C         7.246   Mean of variate
             Lp       -0.6489   Mean of variate

The predictions are calculated at fixed levels of some factors:
         Factor   Fixed level
              E  2

The standard errors are appropriate for interpretation of the predictions as
summaries of the data rather than as forecasts of new observations.

Response variate: Ly

            A          NS                        S
               Prediction        s.e.  Prediction        s.e.
            S
            1       0.852      0.2123       2.093      0.3871
            2       0.960      0.1633       1.770      0.2854
            3       1.067      0.1820       1.447      0.2115
            4       1.174      0.2538       1.124      0.1991
```

The first parameter, CLASSIFY, specifies those variates or factors in the current regression model whose effects you want to summarize. Any variate or factor in the current model that you do not include will be standardized in some way, as described below.

The LEVELS parameter specifies values at which the summaries are to be calculated, for each of the structures in the CLASSIFY list. For factors, you can select some or all of the levels, while for variates you can specify any set of values. A single level or value is represented by a scalar; several levels or values must be combined into a variate (which may of course be unnamed). Alternatively, if the factor has labels, you can use these to select the levels for the summaries by setting LEVELS to a text. A missing value in the LEVELS parameter is taken by Genstat to stand for all the levels of a factor, or for the mean value of a variate.

The PARALLEL parameter allows you to indicate that a factor or variate should change in parallel to another factor or variate. Both of these should have same number of values specified for it by the LEVELS parameter of PREDICT. The predictions are then formed for each corresponding set of values rather than for every combination of these values. For example, suppose we had fitted a quadratic model with explanatory variates X and Xsquared. We could then put

```
PREDICT Xsquared,X; PARALLEL=X,*;\
        LEVELS=!(0,4,16,36,64,100),!(0,2,4,6,8,10)
```

The PARALLEL parameter specifies that Xsquared should change in parallel to X, so that we obtain predictions only for matching values.

When you specify LEVELS, PREDICT needs to define a new factor to classify that dimension of the table. By default this will be an unnamed factor, but you can use the NEWFACTOR parameter to give it an identifier. The EXTRA attribute of the factor is set to the name of the corresponding factor or variate in the CLASSIFY list; this will then be used to label that

dimension of the table of predictions.

You can best understand how Genstat forms predictions by regarding its calculations as consisting of two steps. The first step, referred to below as Step A, is to calculate the full table of predictions, classified by every factor in the current model. For any variate in the model, the predictions are formed at its mean, unless you have specified some other values using the LEVELS parameter; if so, these are then taken as a further classification of the table of predictions. The second step, referred to as Step B, is to average the full table of predictions over the classifications that do not appear in the CLASSIFY parameter: you can control the type of averaging using the COMBINATIONS, ADJUSTMENT and WEIGHTS options. By default, the predictions are made at the mean of any offset variate (see 3.5.1), but option OFFSET can be used to specify another value at which the predictions should be made instead.

Printed output is controlled by settings of the PRINT option:

description	describes the standardization policies used when forming the predictions,
predictions	prints the predictions
se	produces predictions and standard errors,
sed	prints standard errors for differences between the predictions,
lsd	prints least significant differences between the predictions (ordinary linear regression models or generalized linear models with the Normal distibution only), and
vcovariance	prints the variance and covariances of the predictions.

By default descriptions, predictions and standard errors are printed. The standard errors (and sed's) are relevant for the predictions when considered as means of those data that have been analysed (with the means formed according to the averaging policy defined by the options of PREDICT). The word *prediction* is used because these are predictions of what the means would have been if the factor levels been replicated differently in the data; see Lane & Nelder (1982) for more details. The LSDLEVEL option specifies the significance level (%) to use in the calculation of least significant differences (default 5%).

Example 3.3.4c prints standard errors of differences and least significant differences for the predictions formed in Example 3.3.4b.

Example 3.3.4c

```
  33  PREDICT [PRINT=sed,lsd] S,A,E,D; LEVELS=!(1...4),*,2,0

Predictions from regression model
---------------------------------

The standard errors are appropriate for interpretation of the predictions as
summaries of the data rather than as forecasts of new observations.

Response variate: Ly

Standard errors of differences of predictions
---------------------------------------------

 S 1  A NS   1          *
  S 1  A S   2     0.3976          *
 S 2  A NS   3     0.1115     0.3477          *
  S 2  A S   4     0.3124     0.1258     0.2454          *
 S 3  A NS   5     0.2229     0.3295     0.1115     0.2183          *
  S 3  A S   6     0.2624     0.2516     0.1766     0.1258     0.1356
 S 4  A NS   7     0.3344     0.3480     0.2229     0.2448     0.1115
  S 4  A S   8     0.2678     0.3774     0.1839     0.2516     0.1441
                       1          2          3          4          5
```

```
 S 3  A S   6          *
S 4  A NS   7     0.1744          *
 S 4  A S   8     0.1258     0.1805          *
                       6          7          8


Least significant differences of predictions (5% level)
-------------------------------------------------------

S 1  A NS   1          *
 S 1  A S   2     0.8428          *
S 2  A NS   3     0.2363     0.7371          *
 S 2  A S   4     0.6624     0.2667     0.5202          *
S 3  A NS   5     0.4725     0.6985     0.2363     0.4627          *
 S 3  A S   6     0.5562     0.5334     0.3744     0.2667     0.2874
S 4  A NS   7     0.7088     0.7376     0.4725     0.5189     0.2363
 S 4  A S   8     0.5676     0.8000     0.3898     0.5334     0.3055
                       1          2          3          4          5


 S 3  A S   6          *
S 4  A NS   7     0.3698          *
 S 4  A S   8     0.2667     0.3826          *
                       6          7          8
```

By default, the standard errors (and sed's) are not augmented by any component corresponding to the estimated variability of a new observation. (Hence the comment in the output of Examples 3.3.4a and 3.3.4b: "The standard errors are appropriate for interpretation of the predictions as summaries of the data rather than as forecasts of new observations.") However, you can set option `SCOPE=new` to request that the variance of predictions should be calculated on the basis of forecasting new observations rather than of summarizing the data to which the model has been fitted. This setting cannot be used if the predictions are to be standardized for the effects of any factors in the model; in other words, all factors in the current model must be listed in the `CLASSIFY` parameter of the `PREDICT` statement. In addition, it cannot be used when making predictions from generalized linear models with option `BACKTRANSFORMATION=none` (3.5.3), nor with weighted regression (see 3.1.1). The effect of `SCOPE=new` is to form variances for each predicted value by combining the variance of the estimated mean value of the prediction (as produced for `SCOPE=data`) together with the estimated variance of a new observation with the same values of explanatory variates and factors:

"new" variance = "data" variance + (dispersion × variance function)

The `DISPERSION` and `DMETHOD` options allow you to change the method by which the variance of the distribution of the response values is obtained for calculating the standard errors. These options operate like the corresponding options of `MODEL` (except that they apply only to the current statement). The default is to use the method as originally defined by the `MODEL` statement.

You can send the output to another channel, or to a text structure, by setting the `CHANNEL` option.

The `COMBINATIONS` option specifies which cells of the full table in Step A are to be filled for averaging in Step B. The default, `COMBINATIONS=estimable`, uses all the cells other than those that involve parameters that cannot be estimated, for example because of aliasing. Alternatively, you can set `COMBINATIONS=present` to exclude cells for factor combinations that do not occur in the data, as shown in Example 3.3.4i below, or `COMBINATIONS=full` to use all the cells. In the examples above, however, this would make no difference because all four cells in the `A` by `E` table contain some values.

When `COMBINATIONS` is set to `estimable` or `present` the `LEVELS` parameter is overruled. Any subsets of factor levels in the `LEVELS` parameter are ignored, and predictions are formed for all the factor levels that occur in the data or are estimable. Likewise, the full table cannot then be classified by any sets of values of variates; the `LEVELS` parameter must then supply only

single values for variates.

The ADJUSTMENT and WEIGHTS options define how the averaging is done in Step B. Values in the full table produced in Step A are averaged with respect to all those factors that you have not included in the settings of the CLASSIFY parameter. By default, the levels of any such factor are combined with what we call *marginal weights*: that is, by the number of occurrences of each of its levels in the whole dataset. Line 34 of Example 3.3.4d uses the TABULATE directive (1:4.11.1) to display the occurrences of combinations of levels of the factors A and E, and then line 35 produces a summary of the effects of A alone, averaging over E.

Example 3.3.4d

```
  34  TABULATE [PRINT=counts; CLASSIFICATION=A,E; MARGINS=yes]

                    Count
           E            1           2       Count
           A
          NS            9           3          12
           S           10           2          12
       Count           19           5          24

  35  PREDICT A

Predictions from regression model
---------------------------------

These predictions are estimated mean values, adjusted with respect to some
factors as specified below.

The predictions have been formed only for those combinations of factor levels
for which means can be estimated without involving aliased parameters.

The predictions are based on fixed values of some variates:
        Variate   Fixed value   Source of value
              D         35.33   Mean of variate
              S         3.169   Mean of variate
              C         7.246   Mean of variate
             Lp       -0.6489   Mean of variate

The predictions have been standardized by averaging over the levels of some
factors:
         Factor  Weighting policy  Status of weights
              E  Marginal weights  Constant over levels of other factors

The standard errors are appropriate for interpretation of the predictions as
summaries of the data rather than as forecasts of new observations.

Response variate: Ly

              Prediction        s.e.
           A
          NS      0.3862     0.08897
           S      0.6937     0.09091
```

In forming the averages for A, the data from the two levels of E have been combined with weights 19 and 5, since these are the frequencies with which they occur in all the data. Because we are using the default settings of ADJUSTMENT and WEIGHTS, these weights are constant over the levels of the other factors: that is, the same weights are used when forming the prediction for each level of A, even though the levels of E occurred with different frequencies at the different levels of A. The effect, therefore, is to *standardize* the prediction for the estimated effects of E.

The ADJUSTMENT and WEIGHTS options allow you to change the weights. The setting ADJUSTMENT=equal specifies that the levels are to be weighted equally, when the predictions are averaged over the standardizing factors. (This corresponds to the default weighting used by VPREDICT; see 5.5.1.) The weights would then be 1 and 1 instead of 19 and 5, as shown in

Example 3.3.4e.

Example 3.3.4e

```
  36  PREDICT [ADJUSTMENT=equal] A

Predictions from regression model
---------------------------------

These predictions are estimated mean values, adjusted with respect to some
factors as specified below.

The predictions have been formed only for those combinations of factor levels
for which means can be estimated without involving aliased parameters.

The predictions are based on fixed values of some variates:
        Variate   Fixed value   Source of value
              D         35.33   Mean of variate
              S         3.169   Mean of variate
              C         7.246   Mean of variate
             Lp       -0.6489   Mean of variate

The predictions have been standardized by averaging over the levels of some
factors:
         Factor  Weighting policy  Status of weights
              E     Equal weights  Constant over levels of other factors

The standard errors are appropriate for interpretation of the predictions as
summaries of the data rather than as forecasts of new observations.

Response variate: Ly

                Prediction        s.e.
            A
           NS      0.5232      0.0984
            S      0.8307      0.1122
```

The WEIGHTS option is more powerful than the ADJUSTMENT option, allowing you to specify an explicit table of weights. This table can be classified by any, or all, of the factors over whose levels the predictions are to be averaged; the levels of remaining factors will be weighted according to the ADJUSTMENT option. Moreover, you can classify the weights by the factors in the CLASSIFY parameter as well, to provide different weightings for different combinations of levels of these factors. If you supply explicit weights in the WEIGHTS option, any setting of the COMBINATIONS option is ignored.

You will find explicit weights useful in particular when you have population estimates of the proportions of each level of a factor – proportions which may not be matched well in the available data. For example, you might know that these proportions for Type of cloud are in the ratio 2:1 rather than the 19:5 observed in the data. You might then specify these weights with the WEIGHTS option, as shown in Example 3.3.4f.

Example 3.3.4f

```
  37  TABLE [CLASSIFICATION=E; VALUES=2,1] Wte
  38  PREDICT [WEIGHTS=Wte] A

Predictions from regression model
---------------------------------

These predictions are estimated mean values, adjusted with respect to some
factors as specified below.

The predictions have been formed only for those combinations of factor levels
for which means can be estimated without involving aliased parameters.
```

```
The predictions are based on fixed values of some variates:
        Variate   Fixed value   Source of value
              D         35.33   Mean of variate
              S         3.169   Mean of variate
              C         7.246   Mean of variate
             Lp       -0.6489   Mean of variate

The predictions have been standardized by averaging over the levels of some
factors:
         Factor  Weighting policy  Status of weights
              E  Supplied weights  Constant over levels of other factors

The standard errors are appropriate for interpretation of the predictions as
summaries of the data rather than as forecasts of new observations.

Response variate: Ly

              Prediction        s.e.
           A
          NS      0.4449     0.08957
           S      0.7524     0.09731
```

If a model contains any aliased parameters, predicted values cannot be formed for some cells of the full table without assuming a value for the aliased parameters. With the default setting, `COMBINATIONS=estimable`, no predictions are formed for these cells. When `COMBINATIONS=full`, if the aliased parameters simply represent effects of variates that are correlated with other explanatory variables in the model, it may be sufficient just to ignore them. This can be done by setting the `ALIASING` option to `ignore`. The aliased parameters are then taken to be zero, and fitted values are calculated for all cells of the table from the remaining parameters in the model.

Aliasing can also occur if there are some combinations of factors that do not occur in the data, and here it may be more sensible to set option `COMBINATIONS=present` so that these cells are all excluded from the calculation of predictions.

To illustrate the action of the `ALIASING` and `COMBINATIONS` options, in Example 3.3.4g we fit a new model to the cloud-seeding data. The factor `Sf` is formed by grouping the values of the variate `S`; it happens that there were no days in the experiment when the suitability `S` was 2 or less and seeding was done, so one parameter of the interaction between `A` and `Sf` cannot be fitted.

Example 3.3.4g

```
 39  GROUPS S; FACTOR=Sf; LIMITS=!(2,3,4)
 40  TERMS A*Sf
 41  FIT [PRINT=estimates] A*Sf

* MESSAGE: term A.Sf cannot be fully included in the model because 1 parameter
is aliased with terms already in the model.

(A S .Sf 4.175) = (A S) - (A S .Sf 2.525) - (A S .Sf 3.350)

Regression analysis
===================

Estimates of parameters
-----------------------

Parameter           estimate         s.e.     t(17)  t pr.
Constant               0.446        0.236      1.89  0.076
A S                    0.369        0.354      1.04  0.312
Sf 2.525               0.348        0.373      0.93  0.364
Sf 3.350              -0.054        0.298     -0.18  0.858
Sf 4.175              -0.398        0.373     -1.07  0.300
A S .Sf 2.525         -0.362        0.500     -0.72  0.479
A S .Sf 3.350         -0.192        0.448     -0.43  0.673
```

```
A S .Sf 4.175              0           *         *      *

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                   A  NS
                  Sf  1.600
```

When the model is fitted, the last parameter of the interaction term `A.Sf` is aliased, and the form of the aliasing relationship is shown in the message. This relationship appears complicated because it is the first level of `Sf` that has no observations when `A` takes level `'S'`, and parameters are usually differences from the first level. When this happens, the parameters become differences with the last level instead, and the parameter for the last level becomes aliased.

The default setting, `estimable`, of `COMBINATIONS` suppresses any prediction which includes a contribution from a factor combination that is not represented in the data. This makes it clear that there is not enough information to form the value in question without making further assumptions. So when we form predictions for `A` in Example 3.3.4h, none is formed for level `S`.

Example 3.3.4h

```
  42  PREDICT [PRINT=prediction; ADJUST=equal; COMBINATIONS=estimable] A

Predictions from regression model
---------------------------------

Response variate: Ly

              Prediction
           A
          NS      0.4201
           S           *
```

Alternatively, we could set `COMBINATIONS=present`, to specify that predictions are to be formed only for the cells of the full table in Step A that have observations. So, in the `A` by `Sf` table, in Example 3.3.4i, no prediction is formed for `A` level `S and Sf` 1.60.

Example 3.3.4i

```
  43  PREDICT [PRINT=prediction; COMBINATIONS=present] A,Sf

Predictions from regression model
---------------------------------

Response variate: Ly

              Prediction
          Sf       1.600       2.525       3.350       4.175
           A
          NS      0.4462      0.7944      0.3919      0.0478
           S           *      0.8013      0.5686      0.4165
```

The use of `COMBINATIONS=present` has consequences on any averaging that is done. Here there is none, but if we were to give the statement

```
PREDICT [COMBINATIONS=present] A
```

the averages for the two levels of `A` would not be formed with the same weights for `Sf`: that for level `NS` would include a contribution from level 1 of `Sf`, whereas that for level `S` would not; see Example 3.3.4j, This must be borne in mind when interpreting the results.

Example 3.3.4j

```
  44  PREDICT [PRINT=prediction; ADJUST=equal; COMBINATIONS=present] A

Predictions from regression model
---------------------------------

Response variate: Ly

              Prediction
           A
          NS      0.4201
           S      0.5955
```

If you do want to form predictions for all the combinations, you need to make some assumptions about the aliasing. If we simply set `COMBINATIONS=full`, a warning message appears.

Example 3.3.4k

```
  45  PREDICT [PRINT=prediction; COMBINATIONS=full] A,Sf

******** Warning, code RE 36, statement 1 on line 45

Command: PREDICT [PRINT=prediction; COMBINATIONS=full] A,Sf
Predictions cannot be formed.
Option ALIAS is set to 'fault' and 1 parameter is aliased.
```

If you want to assume that the missing parameter (the difference between the first and last levels of `Sf` when `A` is `'S'`) is actually zero, then you can just set option `ALIASING=ignore`.

Example 3.3.4l

```
  46  PREDICT [PRINT=prediction; COMBINATIONS=full; ALIASING=ignore] A,Sf

Predictions from regression model
---------------------------------

Response variate: Ly

              Prediction
          Sf       1.600       2.525       3.350       4.175
           A
          NS      0.4462      0.7944      0.3919      0.0478
           S      0.8149      0.8013      0.5686      0.4165
```

An alternative way to overcome aliasing is to supply explicit weights using the `WEIGHTS` option.

We have assumed in this section that averaging is the appropriate way of combining predicted values over levels of a factor. But sometimes summation is needed, for example in the analysis of counts by log-linear models (3.5.1). You can achieve this by setting the `METHOD` option to `total`. The rules about weights and so on still apply. The `BACKTRANSFORM` and `NBINOMIAL` options are also relevant only to generalized linear models (3.5.3).

The `PREDICTIONS`, `SE`, `SED`, `LSD` and `VCOVARIANCE` options let you save the results of `PREDICT` as well as, or instead of, printing them. We use this in Example 3.3.4m to produce 95% confidence limits for predictions of the amount of rainfall at each level of A, transformed back to the natural scale of the original data. (The `EDT` function is described in 1:4.2.9.)

Example 3.3.4m

```
47  TERMS A*(D+S+C+Lp+E)
48  FIT [PRINT=*] A+S+D+C+Lp+E+S.A
49  PREDICT [PRINT=*; PREDICTION=Pa; SE=Sa] A
50  RKEEP DF=df
51  CALCULATE High,Low = Pa + 1,-1*Sa*EDT(0.95; df)
52  & Low,Pa,High = 10**Low,Pa,High
53  PRINT Low,Pa,High

                Low          Pa        High
      A
     NS       1.702       2.433       3.479
      S       3.427       4.939       7.118
```

The SAVE option allows you to specify the regression save structure of the analysis on which the predictions are based. If SAVE is not set, the most recent regression model is used.

3.3.5 Comparisons between predictions: the RCOMPARISONS and RTCOMPARISONS procedures

Two procedures are available to calculate comparisons within a table of predicted means. RCOMPARISONS calculates comparisons amongst the levels of one of the factors in the table, and can assess how they vary over the other factors in the table. Alternatively, RTCOMPARISONS can calculate comparisons between any cells of a multi-way table. So it differs from RCOMPARISONS in that the comparison can be across the levels of more than one of the factors of the table. It can also take tables of means from an analysis of variance (see Chapter 4) as well as those from regression.

RCOMPARISONS procedure

Calculates comparison contrasts amongst regression means (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (aov, contrasts); default aov, cont
COMBINATIONS = *string token*	Factor combinations for which to form the predicted means (full, present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when forming the predicted means (marginal, equal, observed); default marg
PSE = *string tokens*	Types of standard errors to be printed with the contrasts (contrasts, differences, lsd); default cont
WEIGHTS = *table*	Weights classified by some or all of the factors in the model; default *
OFFSET = *scalar*	Value of offset on which to base predictions; default mean of offset variate
METHOD = *string token*	Method of forming margin (mean, total); default mean
ALIASING = *string token*	How to deal with aliased parameters (fault, ignore); default faul
BACKTRANSFORM = *string token*	What back-transformation to apply to the values on the linear scale, before calculating the predicted means (link, none); default link
SCOPE = *string token*	Controls whether the variance of predictions is

	calculated on the basis of forecasting new observations rather than summarizing the data to which the model has been fitted (`data`, `new`); default `data`
`NOMESSAGE` = *string tokens*	Which warning messages to suppress (`dispersion`, `nonlinear`); default `*`
`DISPERSION` = *scalar*	Value of dispersion parameter in calculation of s.e.s; default is as set in the `MODEL` statement
`DMETHOD` = *string token*	Basis of estimate of dispersion, if not fixed by `DISPERSION` option (`deviance, Pearson`); default is as set in the `MODEL` statement
`NBINOMIAL` = *scalar*	Supplies the total number of trials to be used for prediction with a binomial distribution (providing a value *n* greater than one allows predictions to be made of the number of "successes" out of *n*, whereas the value one predicts the proportion of successes); default 1
`LSDLEVEL` = *scalar*	Significance level (%) for least significant differences; default 5
`SAVE` = *identifier*	Regression save structure for the analysis from which the comparison contrasts are to be calculated

Parameters

`FACTOR` = *factors*	Factor whose levels are compared
`CONTRASTS` = *matrices*	Defines the comparisons to be estimated
`ORDER` = *scalars*	Number of comparisons to estimate; default is the number of rows of the `CONTRASTS` matrix
`GROUPS` = *factors* or *pointers*	Set if comparisons are to be made at different combinations of another factor or factors
`ESTIMATES` = *variates* or *pointers*	Saves the estimated contrasts in a variate if `GROUPS` is unset, or in a pointer to a set of tables
`SE` = *variates* or *pointers*	Saves standard errors of the contrasts in a variate if `GROUPS` is unset, or in a pointer to a set of tables
`SED` = *pointers*	Pointer to a set of symmetric matrices to save standard errors for differences between the contrasts estimated for different levels of the `GROUPS` factor(s)
`LSD` = *pointers*	Pointer to a set of symmetric matrices to save least significant differences for the contrasts estimated for different levels of the `GROUPS` factor(s)
`DEVIANCES` = *variates*	Saves sums of squares or deviances of the contrasts
`DF` = *variates*	Saves degrees of freedom for the contrasts

`RCOMPARISONS` makes comparisons amongst the levels of a factor classifying a table of predicted means from a linear or generalized linear regression. The `SAVE` option can be used to specify the regression save structure from the analysis for which the comparisons are to be calculated (see the `SAVE` option of the `MODEL` directive). If `SAVE` is not specified, the comparisons are calculated from the most recent regression analysis.

The factor amongst whose levels the comparisons are to be calculated is specified by the `FACTOR` parameter. The `CONTRASTS` parameter supplies a matrix to specify the comparisons to be calculated. This has a column for each level of the `FACTOR`, and a row for each comparison. You can set the `ORDER` parameter to a scalar, *n* say, to indicate that only the comparisons in the first *n* rows of the `CONTRASTS` matrix are to be calculated (otherwise they are all calculated).

By default the comparisons are calculated between the means in the one-way table classified by FACTOR. However, you can set the GROUPS parameter to some other factor to indicate that the comparisons are to be made for each level of that factor, or you can set it to a pointer of factors to make the comparisons for every combination of the levels of those factors.

RCOMPARISONS calculates the means using the PREDICT directive. As explained in Section 3.3.4, the first step (A) of the calculation forms the full table of predictions, classified by every factor in the model. Then the second step (B) averages the full table over the factors that do not occur in the table of means. The COMBINATIONS option specifies which cells of the full table are to be formed in Step A. The default setting, estimable, fills in all the cells other than those that involve parameters that cannot be estimated, for example because of aliasing. Alternatively, setting COMBINATIONS=present excludes the cells for factor combinations that do not occur in the data, or COMBINATIONS=full uses all the cells. The ADJUSTMENT option then defines how the averaging is done in Step B. The default setting, marginal, forms a table of marginal weights for each factor, containing the proportion of observations with each of its levels; the full table of weights is then formed from the product of the marginal tables. The equal setting weights all the combinations equally. Finally, the observed setting uses the WEIGHTS option of PREDICT to weight each factor combination according to its own individual replication in the data (calculated using the TABULATE directive; see 1:4.11.1). Alternatively, you can supply your own table of weights, using the WEIGHTS option. There are also options OFFSET, METHOD, ALIASING, BACKTRANSFORM, SCOPE, NOMESSAGE, DISPERSION, DMETHOD and NBINOMIAL to control further aspects of the calculations; these operate exactly as in the PREDICT directive.

The PRINT option controls printed output, with settings:

aov	to print an analysis of variance (for an ordinary linear regression) or an analysis of deviance (for a generalized linear model), giving the sums of squares (or deviances) and so on for the comparisons;
contrasts	to print the contrasts.

By default these are both printed. The PSE option controls the types of standard errors that are produced to accompany the contrasts, with settings:

contrasts	for standard errors of the contrasts;
differences	for standard errors for differences between pairs of contrasts calculated for the different GROUPS;
lsd	for least significant differences for contrasts calculated for the GROUPS.

The default is contrasts. The LSDLEVEL option sets the significance level (as a percentage) for the least significant differences.

The ESTIMATES parameter of RCOMPARISONS allows you to save the estimated contrasts. These are in a variate if GROUPS is unset, or in a pointer containing a table classified by GROUPS for each comparison otherwise. The SE parameter saves the standard errors of the contrasts, in a variate or pointer similarly to ESTIMATES. If GROUPS is set, you can also save standard errors for differences between the contrasts estimated for different levels of the GROUPS factor(s). This is again a pointer, with a symmetric matrix for each comparison. Finally, the DF parameter can save a variate containing the degrees of freedom of the contrasts, and the DEVIANCES parameter can save a variate with their deviances (for a generalized linear model) or sums of squares (for an ordinary linear regression).

Example 3.3.5a studies the effect of diet on the weight gains of rats. There were six treatments arising from two treatment factors: the source of protein (beef, pork or cereal), and its amount (high or low). The 60 rats that provided the experimental units were allocated at random into six groups of ten rats, one group for each treatment combination. The model

`Source*Amount` in the `FIT` statement in line 18 of the analysis fits three terms in addition to the constant: `Source` (main effect of source of protein), `Amount` (main effect of the amount of protein) and `Source.Amount` (the interaction between source and amount of protein). In line 21, the `RCOMPARISONS` statement in line 21 makes comparisons between animal and cereal sources of protein, and between beef and pork. Then, in line 22, it sees how the contrasts differ according to the amount of protein. This data set is also analysed by the `ANOVA` directive in Sections 4.1 and 4.5.

Example 3.3.5a

```
  2  " 3x2 factorial experiment (Snedecor & Cochran 1980, p.305)."
  3  UNITS [NVALUES=60]
  4  FACTOR [LABELS=!T(beef,cereal,pork); VALUES=(1...3)20] Source
  5  & [LABELS=!T(high,low); VALUES=3(1,2)10] Amount
  6  READ Gain

   Identifier   Minimum      Mean   Maximum    Values   Missing
         Gain     49.00     87.87     120.0        60         0

 17  MODEL        Gain
 18  FIT          [FPROBABILITY=yes; TPROBABILITY=yes] Source*Amount

Regression analysis
===================

 Response variate: Gain
     Fitted terms: Constant + Source + Amount + Source.Amount


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       5        4613.        922.6      4.30  0.002
Residual        54       11586.        214.6
Total           59       16199.        274.6

Percentage variance accounted for 21.9
Standard error of observations is estimated to be 14.6.


Estimates of parameters
-----------------------

Parameter                     estimate         s.e.     t(54)  t pr.
Constant                        100.00         4.63     21.59  <.001
Source cereal                   -14.10         6.55     -2.15  0.036
Source pork                      -0.50         6.55     -0.08  0.939
Amount low                      -20.80         6.55     -3.18  0.002
Source cereal .Amount low        18.80         9.26      2.03  0.047
Source pork .Amount low           0.00         9.26      0.00  1.000

Parameters for factors are differences compared with the reference level:
             Factor  Reference level
             Source  beef
             Amount  high

 19  MATRIX       [ROWS=!T('animal vs cereal','beef vs pork'); COLUMNS=3; \
 20               VALUES=0.5,-1,0.5,1,0,-1] Compare
 21  RCOMPARISONS Source; CONTRASTS=Compare

Comparisons between means
-------------------------

Variate: Gain

Contrasts
---------
```

```
                    Estimate        s.e.       t(54)       t pr.
 animal vs cereal       4.450       4.011        1.11       0.272
     beef vs pork       0.500       4.632        0.11       0.914

Analysis of variance
--------------------

Source                  s.s.        d.f.        m.s.        v.r.       F pr.
animal vs cereal         264           1       264.0        1.23       0.272
beef vs pork               3           1         2.5        0.01       0.914
residual               11586          54       214.6

  22  RCOMPARISONS [PSE=contrasts,differences,lsd]\
  23               Source; CONTRASTS=Compare; GROUPS=Amount

Comparisons between means
-------------------------

Variate: Gain

Contrasts: animal vs cereal
---------------------------

                Estimate        s.e.       t(54)       t pr.
       Amount
         high      13.850       5.673        2.44       0.018
          low      -4.950       5.673       -0.87       0.387

Standard errors of differences between estimated contrasts
----------------------------------------------------------

         high           *
          low       8.023           *
                     high         low

Least significant differences (at 5.0%) for estimated contrasts
---------------------------------------------------------------

            1           *
            2       16.08           *
                        1           2

Contrasts: beef vs pork
-----------------------

                Estimate        s.e.       t(54)       t pr.
       Amount
         high      0.5000       6.551        0.08       0.939
          low      0.5000       6.551        0.08       0.939

Standard errors of differences between estimated contrasts
----------------------------------------------------------

         high           *
          low       9.264           *
                     high         low

Least significant differences (at 5.0%) for estimated contrasts
---------------------------------------------------------------

            1           *
            2       18.57           *
                        1           2

Analysis of variance
--------------------

Source                  s.s.        d.f.        m.s.        v.r.       F pr.
animal vs cereal        1442           2       721.1        3.36       0.042
beef vs pork               3           2         1.3        0.01       0.994
residual               11586          54       214.6
```

RTCOMPARISONS procedure

Calculates comparison contrasts within a multi-way table of means (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (contrasts); default cont
COMBINATIONS = *string token*	Factor combinations for which to form the predicted means (full, present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when forming the predicted means (marginal, equal, observed); default marg
WEIGHTS = *table*	Weights classified by some or all of the factors in the model; default *
OFFSET = *scalar*	Value of offset on which to base predictions; default mean of offset variate
METHOD = *string token*	Method of forming margin (mean, total); default mean
ALIASING = *string token*	How to deal with aliased parameters (fault, ignore); default faul
BACKTRANSFORM = *string token*	What back-transformation to apply to the values on the linear scale, before calculating the predicted means (link, none); default link
SCOPE = *string token*	Controls whether the variance of predictions is calculated on the basis of forecasting new observations rather than summarizing the data to which the model has been fitted (data, new); default data
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, nonlinear); default *
DISPERSION = *scalar*	Value of dispersion parameter in calculation of s.e.s; default is as set in the MODEL statement
DMETHOD = *string token*	Basis of estimate of dispersion, if not fixed by DISPERSION option (deviance, Pearson); default is as set in the MODEL statement
NBINOMIAL = *scalar*	Supplies the total number of trials to be used for prediction with a binomial distribution (providing a value *n* greater than one allows predictions to be made of the number of "successes" out of *n*, whereas the value one predicts the proportion of successes); default 1
SAVE = *identifier*	Regression or ANOVA save structure for the analysis from which the comparisons are to be calculated

Parameters

CONTRAST = *tables*	Defines the comparisons to be estimated
ESTIMATES = *scalars*	Saves the estimated contrasts
SE = *scalars*	Saves standard errors of the contrasts

RTCOMPARISONS makes comparisons within a multi-way tables of predicted means from a linear or generalized linear regression or an analysis of variance (Chapter 4). The model should previously have been fitted by the directives FIT (3.1.2) or ANOVA (4.1.2) in the usual way. The SAVE option can be used to specify the save structure from the analysis for which the comparisons are to be calculated; see the SAVE option of the MODEL directive (3.1.1) or ANOVA directive (4.1.2). If SAVE is not specified, the comparisons are calculated from the most recent

regression analysis.

Each comparison is specified in a table supplied by the CONTRAST parameter. For a regression or generalized linear models analysis, RTCOMPARISONS calculates the means using the PREDICT directive (3.3.4), and has COMBINATIONS, ADJUSTMENT and WEIGHTS options to control the process, just like those of RCOMPARISONS. However, these options are irrelevant if the SAVE structure is from an ANOVA analysis (4.1.2); the means are then obtained using AKEEP (4.6.1), and are the same as those that would be printed by ANOVA. The options OFFSET, METHOD, ALIASING, BACKTRANSFORM, SCOPE, NOMESSAGE, DISPERSION, DMETHOD and NBINOMIAL are also relevant only to regression, and operate exactly as in the PREDICT directive.

The PRINT option controls printed output, with setting:

contrasts	to print the contrasts (default).

The ESTIMATE parameter allows you to save the estimated contrast, and the SE parameter can save its standard errors.

Example 3.3.5b makes some comparisons with the Amount-by-Source means from Example 3.3.5a: Comp1 compares high protein from beef with low protein from cereal, while Comp2 compares the average of high protein from beef and high protein from pork with low protein from cereal.

Example 3.3.5b

```
 24  TABLE [CLASSIFICATION=Amount,Source] Comp1,Comp2;\
 25        VALUES=!(1,0,0,0,-1,0),!(0.5,0,0.5,0,-1,0)
 26  PRINT Comp1

                  Comp1
      Source       beef      cereal        pork
      Amount
        high     1.0000      0.0000      0.0000
         low     0.0000     -1.0000      0.0000


 27  &            Comp2

                  Comp2
      Source       beef      cereal        pork
      Amount
        high     0.5000      0.0000      0.5000
         low     0.0000     -1.0000      0.0000


 28  RTCOMPARISONS Comp1,Comp2

Comparisons between means
-------------------------

Variate: Gain

Contrast    estimate        s.e.       t(54)         pr.
   Comp1      16.100       6.551        2.46       0.017
   Comp2      15.850       5.673        2.79       0.007
```

3.3.6 Plots of estimates: the RDESTIMATES procedure

RDESTIMATES procedure

Plots one- or two-way tables of regression estimates (R.W. Payne).

Options

GRAPHICS = *string token*	Type of graph (highresolution, lineprinter); default high

METHOD = *string token*	What to plot (estimates, lines); default esti
XFREPRESENTATION = *string token*	How to label the x-axis (levels, labels); default labels uses the XFACTOR labels, if available
PSE = *string token*	What s.e. to plot to represent variation (average, individual); default aver
SAVE = *regression save structure*	Save structure of the analysis to display; default * shows the most recently fitted regression

Parameters

XFACTOR = *factors*	Factor providing the x-values for each plot
GROUPS = *factors*	Factor identifying the different sets of points from a two-way table of estimates
XVARIATES = *variates*	X-variates for regression coefficients or pointer
NEWXLEVELS = *variates*	Values to be used for XFACTOR instead of its existing levels
TITLE = *texts*	Title for the graph; default defines a title automatically
YTITLE = *texts*	Title for the y-axis; default ''
XTITLE = *texts*	Title for the x-axis; default is to use the identifier of the XFACTOR

RDESTIMATES helps you study factors in a regression model by plotting their estimates. By default these are taken from the most recent regression, but you use the SAVE option to specify the save structure from the MODEL statement (3.1.1) of some other analysis.

The XFACTOR parameter indicates the factor against whose levels the estimates are plotted. You can also specify a second factor, using the GROUPS parameter, to plot a two-way table of estimates. A separate set of points is then plotted for every level of GROUPS.

By default, the estimates will be for the model term XFACTOR (if GROUPS is not set) or XFACTOR.GROUPS (if GROUPS is set). You can also specify one, or more, variates for the term, using the XVARIATES parameter. If XVARIATES is set to a single variate, xvar say, the term will be XFACTOR.xvar or XFACTOR.GROUPS.xvar (representing regression coefficients for xvar). Alternatively, it can be set to a pointer containing several variates, for example x1var and x2var. The term will be then be XFACTOR.x1var.x2var or XFACTOR.GROUPS.x1var.x2var (representing regression coefficients for the product of the variates x1var and x2var).

The NEWXLEVELS parameter enables different levels to be supplied for XFACTOR if the existing levels are unsuitable. If XFACTOR has labels, these are used to label the x-axis unless you set option XFREPRESENTATION=levels.

Usually, each estimate is represented by a point (using pens 1, 2, and so on for each level in turn of the GROUPS factor). However, with high-resolution plots, the METHOD option can be set to lines to draw lines between the points. The GRAPHICS option controls whether a high-resolution or a line-printer graph is plotted; by default GRAPHICS=high.

The PSE option specifies how to represent the variability of the estimates, as follows:

average	plots an error bar showing the average standard error of the estimates;
individual	plots a bar around each estimate showing plus and minus its standard error.

The TITLE, YTITLE and XTITLE parameters allow you to supply titles for the graph, the y-axis and the x-axis respectively.

Figure 3.3.6 shows a plot of the Source estimates from Example 3.3.5, obtained by the statement

```
RDESTIMATES Source
```

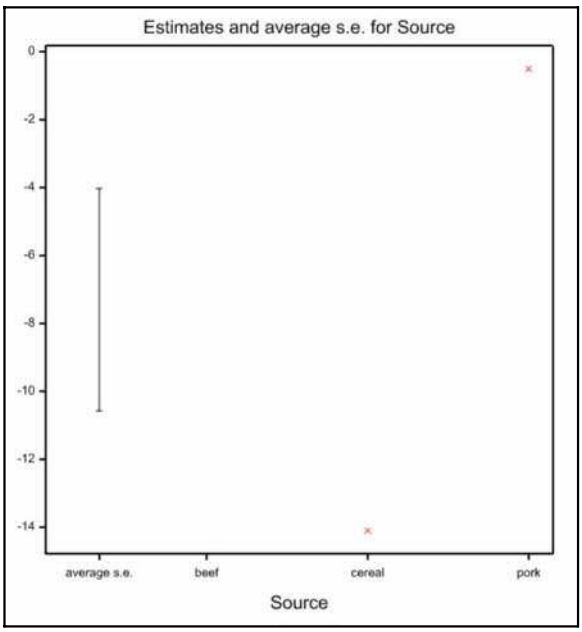


Figure 3.3.6

3.4 Polynomials and additive models

This section describes how to fit regression models containing functions of explanatory variables. The POL function allows you to specify polynomial contrasts representing quadratic, cubic or quartic curves. The COMPARISON and REG functions allow you to specify your own contrasts, provided they are linear in the parameters (nonlinear models are described in later sections of this chapter). With REG the contrasts are orthogonalized (and this also allows you fit orthogonal polynomials), but with COMPARISON they are not. The SSPLINE function, or S for short, provides general smoothing splines. These are actually cubic splines with constraints to ensure smoothness, but they are usually regarded as nonparametric effects of variables. The LOESS function provides an alternative smoothing method, by locally weighted regression. Models containing SSPLINE or LOESS are referred to as *additive models*.

SSPLINE and LOESS be used only with explanatory variates. However, it is easy to use CALCULATE to form a variate from a factor, as in

```
CALCULATE V = F
```

or

```
CALCULATE V = NEWLEVELS(F; W)
```

and then use a function of the variate in the regression model.

You can fit interactions involving the functions POL, REG and COMPARISON. However, interactions involving SSPLINE or LOESS fit different linear trends over the variates in the functions, but have common nonlinear components.

3.4.1 Polynomial regression

You can fit a polynomial model simply by using the CALCULATE statement before FIT. For example, the following statements fit the quadratic regression of Y on X:

```
CALCULATE X2 = X**2
MODEL Y
FIT X,X2
```

However, you can do this more quickly, and using less storage space, with the POL function:

```
MODEL Y
```

```
FIT POL(X; 2)
```

The latter method also has the advantage that the PREDICT directive can produce predictions for specific values for X: with the former method, PREDICT treats X and X2 as if they varied separately rather than having a fixed relationship.

Example 3.4.1a shows the fitting of a cubic relationship between two variables measured on children with diabetes. The fitted polynomial curve is plotted by RGRAPH in Figure 3.4.1.

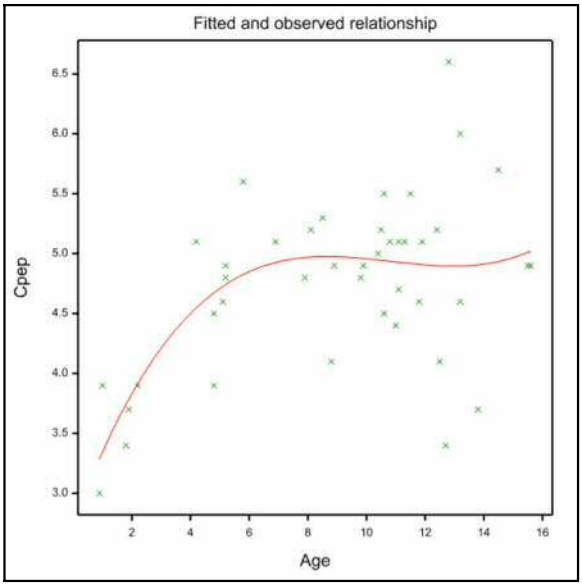


Figure 3.4.1

Example 3.4.1a

```
  2  "  Relationship between serum C-peptide and measured variables
 -3     in children with diabetes. Data from Sochett et al. (1987);
 -4     analysed by Hastie & Tibshirani (1990) p304."
  5  OPEN     '%GENDIR%/Examples/GuidePart2/Diabetes.dat'; CHANNEL=2
  6  READ     [CHANNEL=2; PRINT=data] Age,Base,Cpep

    1   5.2  -8.1 4.8    8.8 -16.1 4.1   10.5  -0.9 5.2   10.6  -7.8 5.5
    2  10.4 -29.0 5.0    1.8 -19.2 3.4   12.7 -18.9 3.4   15.6 -10.6 4.9
    3   5.8  -2.8 5.6    1.9 -25.0 3.7    2.2  -3.1 3.9    4.8  -7.8 4.5
    4   7.9 -13.9 4.8    5.2  -4.5 4.9    0.9 -11.6 3.0   11.8  -2.1 4.6
    5   7.9  -2.0 4.8   11.5  -9.0 5.5   10.6 -11.2 4.5    8.5  -0.2 5.3
    6  11.1  -6.1 4.7   12.8  -1.0 6.6   11.3  -3.6 5.1    1.0  -8.2 3.9
    7  14.5  -0.5 5.7   11.9  -2.0 5.1    8.1  -1.6 5.2   13.8 -11.9 3.7
    8  15.5  -0.7 4.9    9.8  -1.2 4.8   11.0 -14.3 4.4   12.4  -0.8 5.2
    9  11.1 -16.8 5.1    5.1  -5.1 4.6    4.8  -9.5 3.9    4.2 -17.0 5.1
   10   6.9  -3.3 5.1   13.2  -0.7 6.0    9.9  -3.3 4.9   12.5 -13.6 4.1
   11  13.2  -1.9 4.6    8.9 -10.0 4.9   10.8 -13.5 5.1
  7  CLOSE    2
  8  MODEL    Cpep
  9  FIT      [FPROBABILITY=yes; TPROBABILITY=yes] POL(Age; 3)

Regression analysis
===================

 Response variate: Cpep
     Fitted terms: Constant + Age
        Submodels: POL(Age; 3)


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       3         7.95       2.6515      7.46  <.001
Residual        39        13.85       0.3552
Total           42        21.81       0.5192

Percentage variance accounted for 31.6
Standard error of observations is estimated to be 0.596.
```

```
* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
            7        3.400       -2.58
           22        6.600        2.94

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
            8        4.900        0.37
           15        3.000        0.32
           24        3.900        0.29
           29        4.900        0.34


Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(39)  t pr.
Constant          2.740        0.508      5.39  <.001
Age Lin           0.665        0.250      2.66  0.011
Age Quad        -0.0641       0.0339     -1.89  0.066
Age Cub         0.00198      0.00134      1.47  0.149
```

The FIT statement in Example 3.4.1a fits a cubic curve relating the response to a single explanatory variate. You can also use POL functions in multiple regression models with some or all the explanatory variates, and with different orders (quadratic, cubic, and so on). The maximum order for POL is 4. This limit is used because polynomial models with high orders can be very unstable; higher orders are allowed for orthogonal polynomials with the REG function.

When using POL, or the other functions, you must follow the syntax of model formulae (1:1.6.3). This means that you cannot use commas between functions: for example,

```
FIT POL(X; 3), POL(Z; 3), F
```

would be faulted. Instead, you should use the plus operator, as in

```
FIT POL(X; 3) + POL(Z; 3) + F
```

However, you can use commas inside the function, so this model is the same as the previous one:

```
FIT POL(X,Z; 3) + F
```

The models specified by POL are simple polynomials: they are not orthogonalized. Thus, the parameter estimates are simply the linear coefficients of powers of an explanatory variate. This can result in computational problems with some data, when successive polynomial effects can be highly correlated; this would be evidenced in Genstat by a report of linear dependence and the omission of some of the effects. For this reason, it can be better to use the REG function to fit orthogonal polynomials, though the estimated parameters are then not so easy to interpret. Example 3.4.1b shows the correlations between the estimated parameters.

Example 3.4.1b

```
  10  RGRAPH   Age
  11  RDISPLAY [PRINT=correlations]

Regression analysis
===================

Correlations between parameter estimates
----------------------------------------

Parameter                ref    correlations

Constant                   1    1.000
Age Lin                    2   -0.899  1.000
Age Quad                   3    0.799 -0.975  1.000
Age Cub                    4   -0.721  0.928 -0.986  1.000
```

```
                                    1       2       3       4
```

Functions can also be used in the TERMS directive. If a variate appears in a POL, REG or COMPARISON function in the model formula of TERMS, then the fitting statements that follow will fit the function of the variate rather than just its ordinary (linear) effect, whether or not the function name and parentheses are given. If a particular variate has already been fitted in the model, the default order for the POL, REG or COMPARISON function is the order already fitted; otherwise it is the order used in the TERMS directive. The order specified by TERMS cannot be exceeded (unless a new TERMS statement is given). It may be changed to a lower value whenever the variate is added to the model, or in a FIT statement. Attempts to change the order of a function already in the model by any other directive apart from FIT and SWITCH are ignored. For example, you can give the following statements to compare a quadratic with a cubic model:

```
TERMS POL(X; 3)
FIT POL(X; 2)
SWITCH POL(X; 3)
```

If you use POL with a factor, the default is to use the factor levels as the x-values for the polynomials. However, as in ANOVA (4.5), you can use the third argument of POL to specify alternative values for the factor levels if those declared for the factor are unsuitable. The use of factors differs from that of variates in that, if you specify a factor without its POL, REG or COMPARISON function while fitting a sequence of regressions, Genstat interprets this as the factor itself (not any function of the factor). So it is easy to switch between fitting contrasts for a factor and fitting the factor itself This enables you to assess how well the polynomials fit the effects of the factor (similarly to the use of the deviations line produced by ANOVA; see Example 4.5b). In Example 3.4.1c, we form an eight-level factor Agegroup from the variate Age. We fit a cubic polynomial, and then switch this with Agegroup itself. The POOL option of SWITCH is set so that the results of the switch are presented all together in a single line, providing the "deviations" that we need. The results differ slightly from those with the variate, Age, as the x values are now the medians of the eight groups (calculated as the levels of Agegroup by GROUPS; see 1:4.6.1). However it seems clear from the value, 0.22, of the variance ratio for deviations that there is no need for any higher order of polynomial.

Example 3.4.1c

```
  12  GROUPS   [NGROUPS=8] Age; FACTOR=Agegroups
  13  FIT      [FPROBABILITY=yes; TPROBABILITY=yes] POL(Agegroups;3)

Regression analysis
===================

 Response variate: Cpep
     Fitted terms: Constant + Agegroups
        Submodels: POL(Agegroups; 3)


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       3         8.21       2.7350      7.84  <.001
Residual        39        13.60       0.3488
Total           42        21.81       0.5192

Percentage variance accounted for 32.8
Standard error of observations is estimated to be 0.591.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
            7        3.400       -2.63
           22        6.600        2.94
```

```
          28        3.700       -2.31


Estimates of parameters
-----------------------

Parameter            estimate         s.e.     t(39)  t pr.
Constant                2.441        0.658      3.71  <.001
Agegroups Lin           0.754        0.337      2.24  0.031
Agegroups Quad        -0.0731       0.0468     -1.56  0.126
Agegroups Cub         0.00229      0.00190      1.21  0.235

  14  SWITCH   [PRINT=model,estimates,accumulated; POOL=yes;\
  15           FPROBABILITY=yes; TPROBABILITY=yes] Agegroups

Regression analysis
===================

 Response variate: Cpep
     Fitted terms: Constant + Agegroups


Estimates of parameters
-----------------------

Parameter             estimate         s.e.     t(35)  t pr.
Constant                 3.580        0.275     13.00  <.001
Agegroups 4.950          1.053        0.373      2.82  0.008
Agegroups 7.900          1.520        0.390      3.90  <.001
Agegroups 8.900          1.220        0.390      3.13  0.003
Agegroups 10.60          1.480        0.390      3.80  <.001
Agegroups 11.20          1.320        0.373      3.54  0.001
Agegroups 12.50          1.300        0.390      3.34  0.002
Agegroups 14.15          1.387        0.373      3.72  <.001

Parameters for factors are differences compared with the reference level:
             Factor  Reference level
          Agegroups  1.800


Accumulated analysis of variance
--------------------------------

Change                          d.f.         s.s.         m.s.      v.r.  F pr.
+ POL(Agegroups; 3)                3       8.2051       2.7350      7.21  <.001
- POL(Agegroups; 3)
+ Agegroups                        4       0.3273       0.0818      0.22  0.928
Residual                          35      13.2747       0.3793

Total                             42      21.8070       0.5192
```

3.4.2 Orthogonal polynomials and general functions

The REG function can be used in exactly the same way as the POL function to fit polynomial effects. The difference is that REG will fit orthogonalized effects. It is also possible to fit orthogonalized effects by calculating them in advance with the ORTHPOLYNOMIAL procedure, as in the following statements:

```
ORTHPOLYNOMIAL [ORDER=4] X; POLYNOMIAL=P
FIT P[1...4]
```

The same model can be fitted more easily using REG as follows:

```
FIT REG(X; 4)
```

Using the REG function in this way results in the automatic calculation of orthogonal polynomials internally, by the same method as used in procedure ORTHPOLYNOMIAL. Consequently REG uses more storage space than POL. The use of orthogonal polynomials is not as straightforward in regression as in ANOVA (see Chapter 4), as there the designs must be balanced. So if the polynomials are orthogonal overall in an analysis of variance, they will also

be orthogonal within each level of a factor involved in an interaction with the polynomial. This need not be so in regression, and interactions involving the REG function will then be less easy to interpret.

In Example 3.4.2, we fit the same model as in Example 3.4.1a but using orthogonalized polynomials for comparison; note that there is now no correlation between the parameter estimates.

Example 3.4.2

```
  16  FIT      [PRINT=estimates,correlations; TPROBABILITY=yes] REG(Age; 3)

Regression analysis
===================

Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(39)  t pr.
Constant         4.7465       0.0909     52.22  <.001
Age Reg1         0.0831       0.0229      3.63  <.001
Age Reg2       -0.01490      0.00562     -2.65  0.012
Age Reg3        0.00198      0.00134      1.47  0.149


Correlations between parameter estimates
----------------------------------------

Parameter                 ref    correlations

Constant                    1    1.000
Age Reg1                    2    0.000  1.000
Age Reg2                    3    0.000  0.000  1.000
Age Reg3                    4    0.000  0.000  0.000  1.000
                                    1      2      3      4
```

The REG and COMPARISON functions can be used to specify general functions of a variate or factor. For a variate, you must form these functions yourself, for example by using the CALCULATE directive, and put the results into a matrix for use in the third argument of REG. This matrix must have as many columns as there are values of the variate. The number of rows is the maximum order of the function and it must be greater than or equal to the setting of the second parameter the function. For a factor, the matrix has as many columns as the number of levels of the factor, and the rows specify the coefficients to use for the levels of the factor for each contrast. No examples are given of the use of REG and COMPARISON with factors in regression, but the same conventions are used in ANOVA and are illustrated in Examples 4.5a and 4.5c.

With REG the columns are orthogonalized, by adjusting the second column to be orthogonal to the first, and then the third to be orthogonal to the first and second, and so on. For example, the following statements form a matrix Xpol3 containing X, X**2 and X**3 and use it as the third argument REG. This would give the same result as using REG with no third argument.

```
CALCULATE X2 = X**2
& X3 = X**3
MATRIX [ROWS=3; COLUMNS=X; VALUES=#X,#X2,#X3] Xpol3
FIT REG(X; 3; Xpol3)
```

The values of the variate X are not actually used in the analysis, but must nevertheless be present.

With COMPARISON, the columns are fitted exactly as they are specified. So, the regression parameter for each comparison will be adjusted for every other one. The results of the COMPARISON function in regression thus differ from those in ANOVA, where each comparison if fitted ignoring the other comparisons (see 4.5). Likewise, the RCOMPARISONS procedure (3.3.5) makes comparisons between regression means where each comparison is fitted ignoring the other

comparisons. The COMPARISON function can thus be used to specify and fit the whole design matrix of a model. Suppose that the matrix M contains the design matrix, as would be the case if it were formed by RKEEP after an analysis:

```
RKEEP DESIGNMATRIX=M
```

Then the model corresponding to this design matrix can be fitted by the statements

```
CALCULATE Mt = TRANSPOSE(M)
& N = NCOLUMNS(M)
FIT [CONSTANT=omit] COMPARISON(X; N; Mt)
```

where X is any of the explanatory variates fitted in the model.

The use of the REG and COMPARISON functions in regression directives other than FIT is the same as described for POL in 3.4.1, apart from the PREDICT directive: after fitting a model that includes a REG or a COMPARISON function, it is not possible to form predictions.

Note: in releases before Release 6.1, contrasts specified using the third argument of the REG function were not orthogonalized. Now, however, all REG contrasts are orthogonalized, and the COMPARISON is provided to fit unorthogonalized contrasts.

3.4.3 Cubic smoothing splines

The SSPLINE function, or S for short, specifies a cubic smoothing spline for the effect of a variate. Smoothing splines are complicated functions, constructed from segments of cubic polynomials between the distinct values of the variate, and constrained to be "smooth" at the junctions. Models that contain such a function are no longer linear, but are described as *additive models* because the effects of separate explanatory variates are still combined additively. Another way of describing the effects of a variate that has been smoothed in this way is *nonparametric*: in fact, there is a complicated parameterization of the fitted smooth curve, but it is unlikely to be of use for interpretation. See Hastie & Tibshirani (1990) for further details of these models. The main uses of smoothed terms in regression are to investigate the shape of a relationship with a view to later parametric fitting, and to remove the effect of nuisance variables so as to concentrate on the variables of interest.

The degree of smoothness can be controlled, effectively increasing or relaxing the constraints. For example,

```
FIT SSPLINE(X; 4)
```

would fit a spline for X that has four effective degrees of freedom. This curve will be similar to the curve fitted by

```
FIT REG(X; 4)
```

However, the smoothing spline does not exhibit the awkward end-effects of the polynomial, where the curve by its parametric nature tends to bend much more sharply than the observed data would suggest. The smoothing spline with one degree of freedom has the same effect as a linear fit, although the iterative fitting process may not give exactly the same results. At the other extreme, if the variate X has precisely N values, all distinct, then the statement

```
FIT SSPLINE(X; N)
```

would fit a curve that actually passes through each data point (and so would be of little practical use). By default, if the second parameter of SSPLINE is omitted, four effective degrees of freedom are assigned. For an explanation of effective degrees of freedom, see Hastie & Tibshirani (1990).

Example 3.4.3a shows a smoothing spline fitted to the relationship in the previous examples. The resulting fit is displayed in Figure 3.4.3a using the procedure RGRAPH.

Example 3.4.3a

```
  17  FIT      [FPROBABILITY=yes; TPROBABILITY=yes] SSPLINE(Age; 3)

Regression analysis
===================

 Response variate: Cpep
     Fitted terms: Constant + Age
        Submodels: SSPLINE(Age; 3)


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       3         7.88       2.6269      7.36  <.001
Residual        39        13.93       0.3571
Total           42        21.81       0.5192

Percentage variance accounted for 31.2
Standard error of observations is estimated to be 0.598.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
            7        3.400       -2.66
           22        6.600        2.87

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
            8        4.900        0.29
           15        3.000        0.26
           24        3.900        0.24
           29        4.900        0.27


Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(39)  t pr.
Constant          3.996        0.226     17.66  <.001
Age Lin          0.0831       0.0229      3.62  <.001

  18  RGRAPH   Age
```

Note that the linear component of the smoothing spline is reported in the same way as when just the linear effect of a variate is fitted and, in fact, has the same value. No other parameters of the smoothed effect are available from Genstat.

If a TERMS statement is given before fitting a smoothed variate, the same function must be defined for the variate in TERMS. Again the default number of degrees of freedom is four, but if a number is given in the second argument of the SSPLINE function in TERMS it becomes the default for subsequent fitting statements, until another number is specified in a fitting statement, as with variates in POL (3.4.1). The order of SSPLINE can be

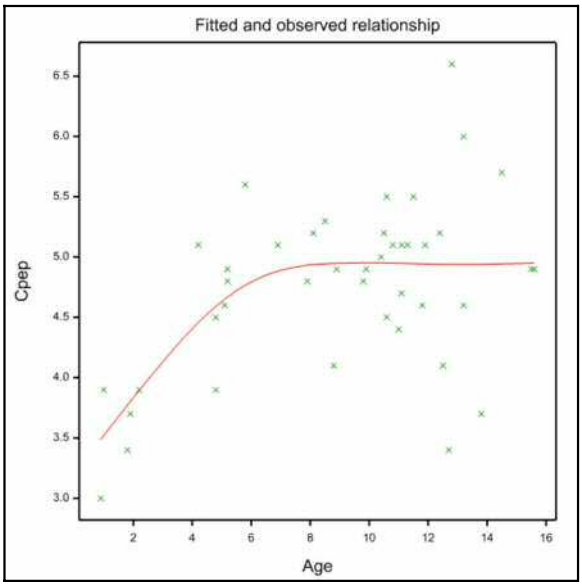


Figure 3.4.3a

either increased or decreased in subsequent SWITCH statements, and whenever the variate is re-introduced into the model after being dropped. Unlike POL, REG and COMPARISON, there is no theoretical maximum number of degrees of freedom; the number available, however, is one less than the number of distinct values in the variate. After you have used the SSPLINE function to fit a smooth function of a variate, you can revert to fitting just the linear effect by specifying the variate, without the function, in either SWITCH or FIT. However, attempts to change the order of an SSPLINE function already in the model, by any directive other than SWITCH or FIT, will be ignored.

Example 3.4.3b shows the effect of a second smoothed variable Base being added to Age, after first giving a TERMS statement.

Example 3.4.3b

```
  19  TERMS    SSPLINE(Age,Base)
  20  FIT      [PRINT=model,deviance] SSPLINE(Age)

Regression analysis
===================

 Response variate: Cpep
     Fitted terms: Constant + Age
        Submodels: SSPLINE(Age; 4)


Residual d.f. 38, s.s. 13.72; Change d.f. -4, s.s. -8.08
  20  ADD      [FPROBABILITY=yes; TPROBABILITY=yes] S(Base)

Regression analysis
===================

 Response variate: Cpep
     Fitted terms: Constant + Age + Base
        Submodels: SSPLINE(Age; 4)
                   SSPLINE(Base; 4)


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       8       12.356       1.5446      5.56  <.001
Residual        34        9.451       0.2780
Total           42       21.807       0.5192

Change          -4       -4.273       1.0683      3.84

Percentage variance accounted for 46.5
Standard error of observations is estimated to be 0.527.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           22        6.600        2.70

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
            5        5.000        0.85
           10        3.700        0.52


Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(34)  t pr.
Constant          4.494        0.243     18.46  <.001
Age Lin          0.0617       0.0208      2.97  0.005
Base Lin         0.0374       0.0117      3.19  0.003
```

Finally, Figure 3.4.3b is produced by the statements

```
SWITCH SSPLINE(Age; 20)
RGRAPH Age
```

This shows how SWITCH can be used to change the order of the smoothing function, in this case increasing it to a point where the curve follows individual fluctuations too closely to be of much practical use.

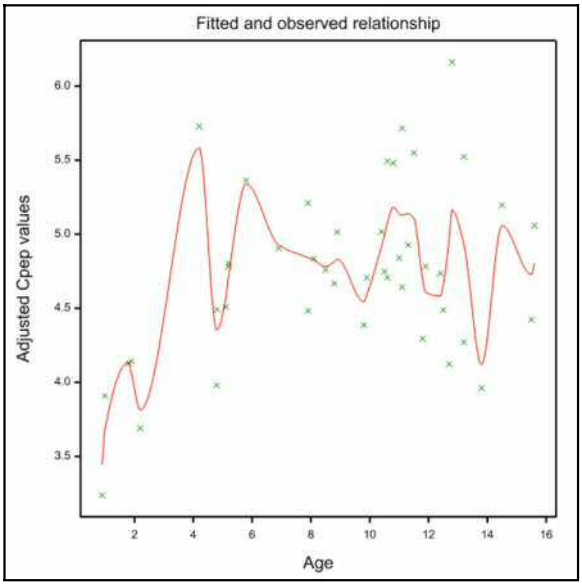


Figure 3.4.3b

After fitting spline functions or loess functions (3.4.4), you can access the fitted effects with the RKEEP directive (3.1.4). The STERMS parameter can be used to store a pointer to those variates whose effects in the model are smoothed. The SCOMPONENTS parameter stores a pointer to variates, one for each smoothed variate in the same order as in STERMS, containing the fitted nonlinear component of each smoothed variate – this does not include the linear component or the constant term.

The PREDICT directive cannot be used to form predictions at specific values of a variate that has been smoothed. If predictions are formed for other explanatory variates or factors in the model, only the linear effect of the smoothed variate will be incorporated in the predictions.

When a spline or loess function is included in the model, it has to be fitted iteratively using a technique known as *back-fitting*. This iterative process can be monitored if required in the same way as the iterative process for generalized linear models (3.5.8). Because an iterative method is needed, Genstat will analyse only the first response variate, even if several have been listed in the MODEL statement. Similarly, it is not possible to fit additive models based on sequentially accumulated SSPM structures (3.2.3), nor can individual changes to the model be summarized separately in an accumulated analysis of variance (3.2.1).

3.4.4 Locally weighted regression

The LOESS function performs locally weighted regression. The algorithm is based on the public-domain software created and kindly made available by Cleveland, Grosse & Shyu of AT&T. (See Cleveland 1979, Cleveland & Grosse 1991, Cleveland & Devlin 1988 and Cleveland, Devlin & Grosse 1988 for details.) The Genstat implementation is the responsibility of the Genstat developers, however, and neither the original authors nor AT&T make any representation or warranty of any kind concerning the merchantability of this software or its fitness for any particular purpose.

Local regression methods fit regression models based on one or more x-variates. The assumption in the modelling is that locally, around any point, the regression surface can be approximated by a function from a particular class: for loess, the class consists of polynomials of order 1 (linear) or 2 (quadratic). In Genstat, local regression is specified by the use of the function LOESS within a regression model:

`LOESS(x;d;s;o)` fits a locally weighted regression of order `o` with approximately `d` degrees of freedom or using smoothing parameter `s`: `x` is a variate for univariate smoothing, or a pointer to up to four variates for multivariate smoothing; when `x` is a variate `o` is a scalar, when `x` is a pointer it is either a scalar or a variate with an element for each variate

in the pointer.

The first and last arguments of the function specify the polynomial model. For example, suppose we have two x-variates `u` and `v`, and specify order 1 by

```
LOESS(!p(u,v); d; s; 1)
```

The local regression will then fit a polynomial consisting of the terms (or *monomials*): *constant*, `u` and `v`. Alternatively, if we specify order 2 by

```
LOESS(!p(u,v); d; s; 2)
```

the polynomial will consist of the monomials: *constant*, `u`, `v`, `uv`, u^2 and v^2. Finally, we can put in a variate for the order, to include a quadratic for one of the variates but not the other. For example:

```
LOESS(!p(u,v); d; s; !(2,1))
```

defines a polynomial consisting of the monomials: constant, `u`, `v`, `uv` and u^2.

The loess method fits the polynomials in local zones within the space of the x-values, thus fitting a smoothed surface to represent the response to the x-variates. The regression is weighted so that data make less of a contribution as you move away from the point of interest. The span, or smoothing parameter, `s` indicates what proportion of the points are used to fit the regression model at x. Let t be the distance of mth closest data point from position x, where m is `s` multiplied by the number of observed points. The weight used for another point at distance d is

$(1 - (d/t)^3)^3$

if $d < t$,

or 0

if $d \geq t$.

When there are several x-variates, it may be sensible to normalize them unless they are on the same natural scale (e.g. geographical distances). This can be done straightforwardly beforehand by using `CALCULATE`.

Example 3.4.4 shows the use of loess to study how the amount of nitric oxide and nitrogen dioxide exhaust produced by an engine is affected by the equivalence ratio (a measure of the richness of the fuel and air mixture). The function

```
LOESS(e; !(*); span; 2)
```

fits a locally quadratic regression with a span of 2/3 (calculated in line 15). The fitted relationship is plotted in Figure 3.4.4.

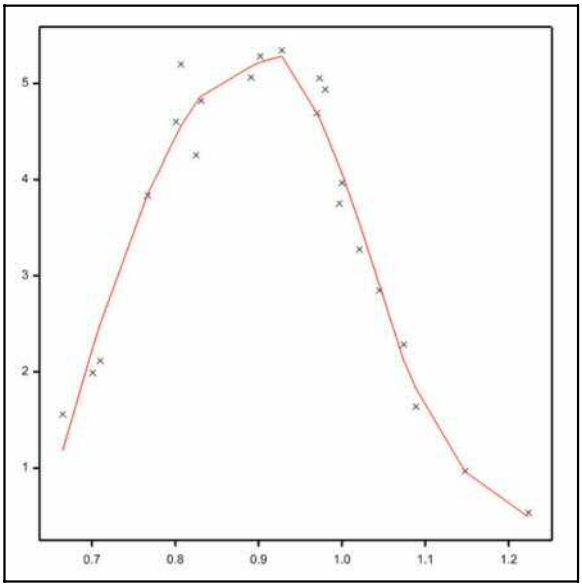


Figure 3.4.4

Notice that the degrees of freedom (the second argument of the function) takes precedence over the span (the third argument), so you need to supply a missing value for the degrees of freedom if you want to set the span. If you set the degrees of freedom, Genstat searches for the appropriate span to generate a smoothing model with the required degrees of freedom. Note, however, that there is not a smooth relationship between span and degrees of freedom, so it may not always be possible to deliver exactly the number of degrees of freedom requested.

Example 3.4.4

```
 2  " LOESS modelling: see Cleveland, Grosse and Shyu (1992, A Package
-3    of C and Fortran Routines for Fitting Local Regression Models).
-4    Gas data: 22 observations from an industrial experiment studying
```

```
 -5    nitric oxide and nitrogen dioxide exhaust from a one-cylinder
 -6    engine versus the equivalence ratio at which the engine was run
 -7    (Brinkman, N.D., 1981, SAE Transactions, 90, No. 810345, 1410-1424)."
  8  VARIATE [VALUES= 4.818, 2.849, 3.275, 4.691, 4.255, 5.064, \
  9           2.118, 4.602, 2.286, 0.97,  3.965, 5.344, 3.834, 1.99, \
 10           5.199, 5.283, 3.752, 0.537, 1.64,  5.055, 4.937, 1.561] nox
 11  &       [VALUES= 0.831, 1.045, 1.021, 0.97,  0.825, 0.891, \
 12             0.71, 0.801, 1.074, 1.148, 1,     0.928, 0.767, 0.701, \
 13           0.807, 0.902, 0.997, 1.224, 1.089, 0.973, 0.98,  0.665] e
 14  " Locally quadratic loess model, span set to 2/3."
 15  CALCULATE span = 2/3 : open '3-4-4.001';4;graph : devi 4
 16  MODEL     nox
 17  TERMS     LOESS(e; !(*); span; 2)
 18  FIT       [PRINT=model,summary,estimates,fitted; FPROBABILITY=yes;\
 19             TPROBABILITY=yes] LOESS(e; !(*); span; 2)

Regression analysis
===================

 Response variate: nox
     Fitted terms: Constant + e
        Submodels: LOESS(e; *; 0.67; *; 2)


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       5       48.450       9.6899     87.75  <.001
Residual        16        1.767       0.1104
Total           21       50.217       2.3913

Percentage variance accounted for 95.4
Standard error of observations is estimated to be 0.332.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           15        5.199        2.17

* MESSAGE: the residuals do not appear to be random;
           for example, fitted values in the range 2.255 to 4.131
           are consistently larger than observed values
           and fitted values in the range 4.459 to 4.688
           are consistently smaller than observed values.

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
           18        0.537        0.92
           22        1.561        0.66


Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(16)  t pr.
Constant          6.139        0.454     13.51  <.001
e Lin            -2.803        0.485     -5.77  <.001


Fitted values and residuals
---------------------------

                                  Standardized
         Unit     Response Fitted value residual Leverage
            1        4.818        4.866    -0.16     0.21
            2        2.849        2.903    -0.18     0.19
            3        3.275        3.549    -0.90     0.15
            4        4.691        4.688     0.01     0.18
            5        4.255        4.798    -1.84     0.21
            6        5.064        5.177    -0.42     0.35
            7        2.118        2.500    -1.30     0.22
            8        4.602        4.459     0.49     0.21
            9        2.286        2.124     0.57     0.25
```

```
            10        0.970        0.961     0.03     0.31
            11        3.965        4.066    -0.33     0.16
            12        5.344        5.283     0.22     0.29
            13        3.834        3.849    -0.05     0.21
            14        1.990        2.255    -0.92     0.25
            15        5.199        4.559     2.17     0.21
            16        5.283        5.220     0.23     0.33
            17        3.752        4.131    -1.24     0.16
            18        0.537        0.492     0.48     0.92
            19        1.640        1.826    -0.66     0.27
            20        5.055        4.637     1.39     0.18
            21        4.937        4.498     1.45     0.17
            22        1.561        1.185     1.94     0.66

Mean                  3.547        3.547     0.04     0.28
```

```
 19  RKEEP     nox; FITTED=f
 20  PEN       2; METHOD=line; SYMBOL=0
 21  DGRAPH    nox,f; e; PEN=1,2
```

3.4.5 Interactions with SSPLINE or LOESS functions

A term of the form *factor*.SSPLINE(*variate*) or *factor*.LOESS(*variate*) represents separate linear effects of the variate for each level of the factor together with a common smoothed effect for each level. A model containing such a term can therefore be represented as a set of parallel smooth curves with additional linear trends for each level of the factor.

The examples in this section illustrate the different types of models that can be fitted with smoothed effects. The data come from a rotational experiment on sugar beet with plots having a range of soil phosphate levels, and include measurements of weight of harvested beet, percentage sugar in the beet and soil phosphate level in each of four successive years. First, a smooth curve is fitted with the SSPLINE function, choosing three degrees of freedom for the smoother, and ignoring the difference in years.

The TERMS statement includes extra terms S(P[]) that will be used in Example 3.4.5d to fit different splines for each year. There is one of these for each year, containing the soil phosphate level for that year, and missing values for the other years. The option MVINCLUDE=yes is set so that the units with missing values in these variates are still included in the fit. Instead the variates will not contribute to the model where they are missing, as you will see later in Example 3.4.5d. For now though, the option can be ignored.

Example 3.4.5a

```
  2  " A rotational experiment with plots having a range of soil phosphate
 -3    levels provides measurements of weight of beet, %sugar in the beet
 -4    and soil phosphate level in each of four successive years. "
  5  FACTOR [LEVELS=4; VALUES=16(1...4)] Year
  6  OPEN   '%GENDIR%/Examples/GuidePart2/Beet.dat'; CHANNEL=2
  7  READ   [PRINT=data; CHANNEL=2] Beetwt,%sugar,SoilP

    1   7.23 18.5  5.4   7.69 18.0  5.4  24.64 20.1  7.8  26.67 19.8  8.0
    2  39.78 19.5 18.0  44.98 19.3 15.6  41.59 19.7 30.4  44.08 19.8 33.8
    3  48.37 19.4 50.4  44.76 19.0 51.0  49.73 18.6 44.0  51.54 18.5 40.2
    4  47.69 19.0 57.2  45.66 19.4 65.0  50.18 18.6 27.0  47.69 18.7 30.0
    5
    6   8.82 13.8  5.6   1.81 13.9  4.8  15.82 14.5 10.2   9.04 14.0  8.6
    7  24.41 15.0 21.6  22.60 14.1 17.2  26.45 15.2 36.4  20.80 15.3 37.2
    8  28.30 14.2 44.4  22.60 14.7 44.4  14.24 13.5 41.0  35.94 15.6 30.2
    9  25.54 15.8 60.8  27.13 15.6 47.0  31.42 15.6 27.0  34.13 15.4 29.0
   10
   11  19.90 16.1  3.0  20.60 16.0  2.0  34.70 16.7  6.2  35.40 16.4  6.2
   12  46.80 17.1 19.8  40.50 16.9 17.2  43.00 16.9 29.6  48.60 17.1 28.0
   13  47.30 17.0 42.8  41.30 17.1 46.2  44.30 17.0 36.6  47.60 16.6 40.0
   14  45.60 17.0 42.2  44.60 17.0 52.0  44.00 17.2 23.4  40.10 16.6 28.0
   15
   16  14.35 16.1  4.0  14.35 15.5  3.8  26.71 16.6  8.0  25.12 16.4  6.4
```

```
    17  33.39 17.2 18.2  33.79 16.2 14.8  36.68 17.0 35.0  33.69 16.8 29.6
    18  34.98 17.0 37.2  35.78 17.0 40.0  42.06 17.2 39.6  38.77 17.3 36.8
    19  40.66 17.3 52.4  37.28 17.2 45.6  34.68 17.3 22.0  32.59 17.2 26.0
    20  :
   8  CLOSE 2
   9  CALCULATE Sugar = Beetwt * %sugar / 100
  10  MODEL  Sugar
  11  CALCULATE P[1...4] = MVINSERT(SoilP; Year/=1...4)
  12  TERMS  [MVINCLUDE=explanatory] S(SoilP)*Year+S(P[])
  13  " 1) A common curve for all years."
  14  FIT    [PRINT=model,estimates; TPROBABILITY=yes] S(SoilP; 3)

Regression analysis
===================

 Response variate: Sugar
     Fitted terms: Constant + SoilP
        Submodels: SSPLINE(SoilP; 3)


Estimates of parameters
-----------------------

Parameter       estimate         s.e.     t(60)  t pr.
Constant           3.523        0.476      7.41  <.001
SoilP Lin         0.0790       0.0146      5.42  <.001

  15  RKEEP  FITTED=f
  16  CALCULATE Fit[1...4] = f
  17  RESTRICT Fit[]; Year==1...4
  18  PEN    1...4; SYMBOL=0; LABEL='1','2','3','4';\
  19         COLOUR='red','limegreen','blue','aqua'
  20  PEN    5...9; METHOD=mono; SYMBOL=0; LINESTYLE=1...5;\
  21         COLOUR='black','red','limegreen','blue','aqua'
  22  XAXIS  3; TITLE='Soil Phosphorus'
  23  YAXIS  3; TITLE='Sugar yield'
  24  DGRAPH [WINDOW=3; KEY=0; TITLE='Common smooth curve']\
  25         Sugar,Fit[]; SoilP; PEN=Year,4(5)
```

The fitted model is shown in Figure 3.4.5a, alongside Figure 3.4.5b which shows the model fitted in Example 3.4.5b including a separate additive effect for each year. There is clearly a large difference between the yields in each year, caused by climatic differences, and much of this difference is accounted for by fitting parallel smoothed curves.

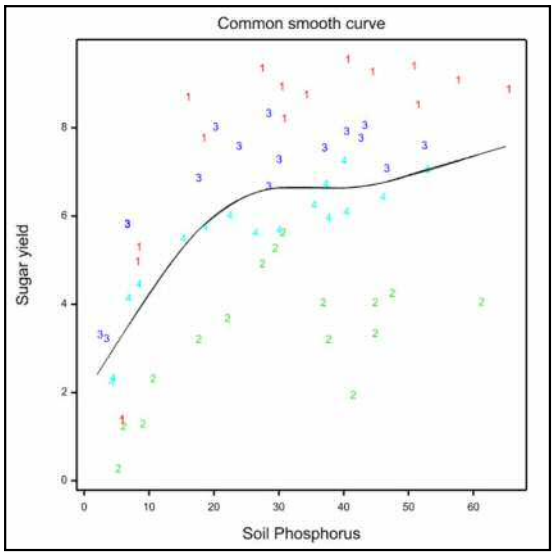


Figure 3.4.5a

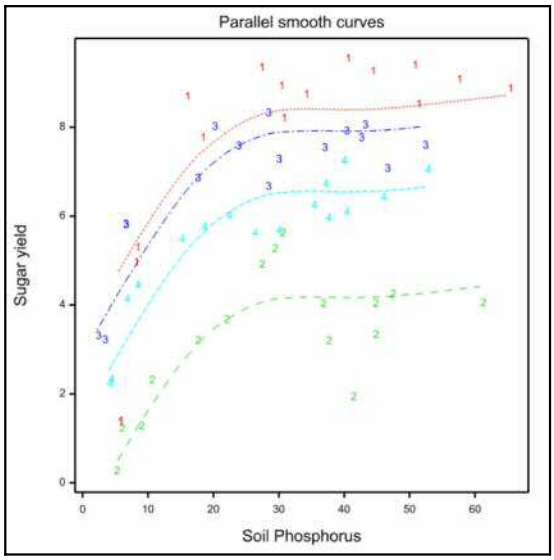


Figure 3.4.5b

Example 3.4.5b

```
  26  " 2) Parallel curves for each year."
  27  ADD    [PRINT=model,estimates; TPROBABILITY=yes] Year

Regression analysis
===================

 Response variate: Sugar
     Fitted terms: Constant + SoilP + Year
        Submodels: SSPLINE(SoilP; 3)


Estimates of parameters
-----------------------

Parameter       estimate         s.e.     t(57)  t pr.
Constant           5.165        0.334     15.48  <.001
SoilP Lin        0.07897      0.00741     10.66  <.001
Year 2            -4.232        0.347    -12.20  <.001
Year 3            -0.485        0.348     -1.39  0.169
Year 4            -1.852        0.348     -5.32  <.001

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                Year  1

  28  RKEEP  FITTED=f
  29  CALC   Fit[1...4] = f
  30  DGRAPH [WINDOW=3; KEY=0; TITLE='Parallel smooth curves']\
  31         Sugar,Fit[]; SoilP; PEN=Year,6...9
```

We now allow separate linear trends for each year, to see whether there is any evidence that the effect of year differences increases or decreases across the range of phosphorus availability.

Example 3.4.5c

```
  32  " 3) Parallel curves with additional trends for each year."
  33  ADD    [PRINT=model,estimates; TPROBABILITY=yes] S(SoilP).Year

Regression analysis
===================

 Response variate: Sugar
     Fitted terms: Constant + SoilP + Year + SoilP.Year
        Submodels: SSPLINE(SoilP; 3)


Estimates of parameters
-----------------------

Parameter               estimate         s.e.     t(54)  t pr.
Constant                   3.981        0.419      9.51  <.001
SoilP Lin                 0.1192       0.0117     10.23  <.001
Year 2                    -2.448        0.616     -3.97  <.001
Year 3                     1.340        0.600      2.23  0.030
Year 4                    -0.369        0.606     -0.61  0.545
SoilP Lin .Year 2        -0.0612       0.0179     -3.42  0.001
SoilP Lin .Year 3        -0.0651       0.0182     -3.58  <.001
SoilP Lin .Year 4        -0.0525       0.0186     -2.83  0.007

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                Year  1

  34  RKEEP  FITTED=f
  35  CALC   Fit[1...4] = f
```

```
36  DGRAPH [TITLE='Parallel smooth curves with separate trends';\
37         WINDOW=3; KEY=0] Sugar,Fit[]; SoilP; PEN=Year,6...9
```

There is certainly some evidence of such a difference, shown by the size of the t-statistics. The model is shown in Figure 3.4.5c.

Finally, we fit separate smooth curves for each year. This can be done by restricting the response variate to each year in turn and fitting the curve. Alternatively, with SSPLINE, you can use copies of the explanatory variate with missing values for all except the units in a particular year. That is not possible with LOESS, and so the RLOESSGROUPS procedure (3.4.6) is provided to automate the fitting of curves separately for each group.

The copies of the explanatory variate were set up by this statement in line 11 of Example 3.4.5a

```
CALC P[1...4] = MVINSERT(SoilP; Year/=1...4)
```

The splines will therefore use values only from the year concerned. Setting the option MVINCLUDE=explanatory in the TERMS statement in line 12 ensures that the units with these missing values are not excluded from the analysis (and the splines will contribute to the model only where they are not missing). Line 39 of Example 3.4.5d removes the previous smoothed effects, and replaces them with separate smooths on each of these four dummy variates. So we have now fitted separate smoothed effects for each year. The fitted model is shown in Figure 3.4.5d.

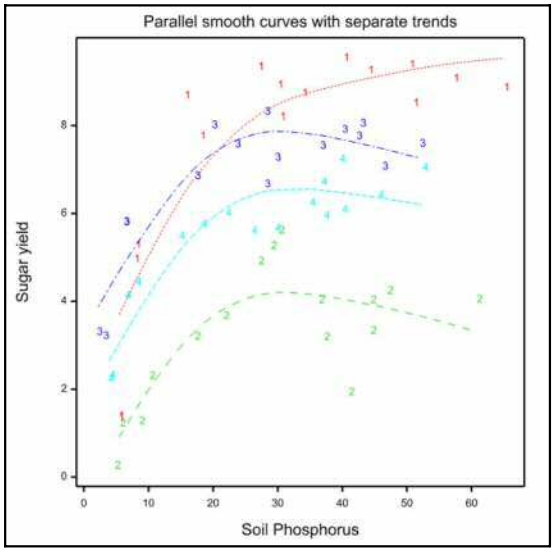


Figure 3.4.5c

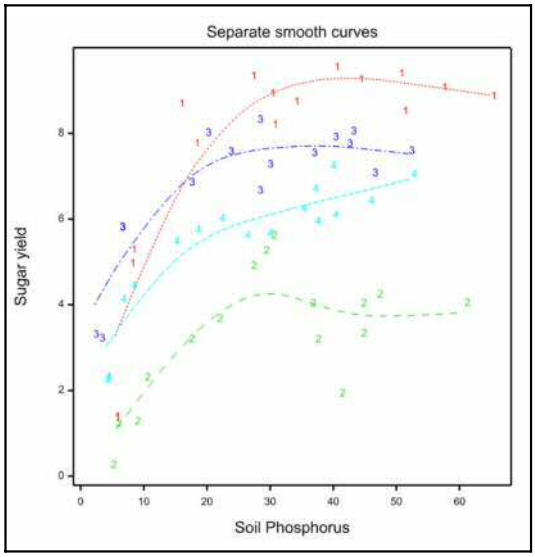


Figure 3.4.5d

Example 3.4.5d

```
38  " 4) Separate curves for each year."
39  SWITCH [PRINT=model,estimates; TPROBABILITY=yes] S(SoilP)/Year+S(P[]; 3)


Regression analysis
===================

 Response variate: Sugar
     Fitted terms: Constant + Year + P[1] + P[2] + P[3] + P[4]
        Submodels: SSPLINE(P[1]; 3)
                   SSPLINE(P[2]; 3)
                   SSPLINE(P[3]; 3)
```

```
                  SSPLINE(P[4]; 3)

Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(48)  t pr.
Constant          4.314        0.366     11.78  <.001
Year 2           -2.576        0.539     -4.78  <.001
Year 3            0.567        0.524      1.08  0.285
Year 4           -1.004        0.530     -1.89  0.064
P[1] Lin          0.1025       0.0102    10.05  <.001
P[2] Lin          0.0528       0.0119     4.44  <.001
P[3] Lin          0.0721       0.0122     5.91  <.001
P[4] Lin          0.0817       0.0127     6.46  <.001

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                Year  1

  40  RKEEP  FITTED=f
  41  CALC   Fit[1...4] = f
  42  DGRAPH [WINDOW=3; KEY=0; TITLE='Separate smooth curves']\
  43         Sugar,Fit[]; SoilP; PEN=Year,6...9
  44  " Show an analysis of parallelism. "
  45  RDISPLAY [PRINT=accumulated; FPROBABILITY=yes]


Regression analysis
===================

Accumulated analysis of variance
--------------------------------

Change                          d.f.         s.s.         m.s.      v.r.  F pr.
+ SSPLINE(SoilP; 3)                3     153.4486      51.1495     86.65  <.001
+ Year                             3     170.9255      56.9752     96.52  <.001
+ SSPLINE(SoilP; 3).Year           3      13.1022       4.3674      7.40  <.001
- SSPLINE(SoilP; 3).Year
- SSPLINE(SoilP; 3)
+ SSPLINE(P[1]; 3)
+ SSPLINE(P[2]; 3)
+ SSPLINE(P[3]; 3)
+ SSPLINE(P[4]; 3)                 6      13.3294       2.2216      3.76  0.004
Residual                         48      28.3346       0.5903

Total                            63     379.1402       6.0181
```

The analysis of parallelism shows that the final step is not statistically significant at the 5% level, so we could conclude that the parallel curves with separate trends are the best representation of the data with this type of model. However, from a biological point of view, it might be better to use an exponential curve, as can be fitted with the `FITCURVE` directive, because it is expected that yield does not increase without limit as soil phosphorus increases.

A limitation of models with smoothed terms and factors is that the `RGRAPH` procedure cannot be used to display them. The figures here were drawn by saving the fitted values, taking separate copies for each year, and using the `DGRAPH` directive to join the fitted points smoothly.

3.4.6 The RLOESSGROUPS procedure

RLOESSGROUPS procedure

Fits locally weighted regression models (loess) to data with groups (D.B. Baird).

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, confidence, groups, submodels); default mode, summ, esti
PLOT = *string tokens*	What to plot (fittedvalues, residuals); default * - no plots
FINALMODEL = *string token*	What to model to fit as the final model (common, parallel, separateslopes, full); default full
CONSTANT = *string token*	How to treat the constant (estimate, omit); default esti
DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality, vertical, df, inflation); default *
FPROBABILITY = *string token*	Printing of probabilities for variance and deviance ratios (yes, no); default no
TPROBABILITY = *string token*	Printing of probabilities for t-statistics (yes, no); default no
PROBABILITY = *scalar*	Probability level for confidence intervals for parameter estimates; default 0.95
MAXCYCLE = *scalar*	Maximum number of iterations for the back-fitting algorithm; default 100
DEVIANCE = *scalar*	Saves the residual deviance
DF = *scalar*	Saves the residual d.f.

Parameters

X = *variate*	Explanatory x-variate to be fitted
GROUPS = *factor*	Groups to be fitted
SMOOTH = *scalar*	Smoothing value to be used in the loess term; default 4
SMTYPE = *string token*	Type of value provided in SMOOTH (df, smoothing); default df
ORDER = *scalar*	Order of regression used in loess term (1 or 2); default 1
RESIDUALS = *variates*	Simple residuals from the fitted loess model
FITTEDVALUES = *variates*	Fitted values from the fitted loess model
ACCUMULATED = *pointer*	Saves the accumulated analysis-of-variance (or deviance) table as a pointer with a variate or text for each column (source, d.f. etc.)
SAVE = *pointer*	Save structure for the fitted model

RLOESSGROUPS is provided to allow the full interaction between a loess smooth on an explanatory variate X and a factor GROUPS to be fitted. It is not possible to include LOESS(X)*GROUPS in the TERMS directive, so the procedure loops around the groups to fit individual models for each group, and then combines the results.

The use of RLOESSGROUPS is similar to FIT (3.1.2). It must be preceded by a MODEL statement (3.1.1), and it can be followed by RDLOESSGROUPS and RKLOESSGROUPS to display and save the results, which operate similarly to RDISPLAY (3.1.3) and RKEEP (3.1.4) respectively. It has options PRINT, CONSTANT, DENOMINATOR, NOMESSAGE, FPROBABILITY, TPROBABILITY and PROBABILITY that operate like those of FIT. However, the PRINT option has two extra settings: submodel to print the three submodels (explained below), and groups to print the individual fits for each group. The output from each submodel or group will use the other settings of PRINT.

The form of the loess curve can be specified by the SMOOTH, SMTYPE and ORDER parameters which specify the arguments to the LOESS function. If SMTYPE=df, SMOOTH gives the number of degrees of freedom used in the function (which should be 2 or greater), while if SMTYPE=smooth, SMOOTH gives the smoothing parameter (which should be between 0 and 1). The ORDER parameter is 1 for a linear loess model and 2 for a quadratic one.

RLOESSGROUPS fits a sequence of models, starting with a common line (ignoring the groups). The next, parallel, model fits a common slope and loess curve, but different intercepts for the groups. The third model (separate slopes) has a common loess curve but different slopes and intercepts. Finally, the fourth (full) model has different loess curves, slopes and intercepts. Groups with less than four observations should be restricted out when fitting the full model, as these cannot be fitted by a loess model. To fit the full model, RLOESSGROUPS uses SUBSET to break up the data into separate groups. It fits these individually using FIT, and then combines the results. The results of these individual fits are printed only if the groups setting is included in the PRINT option.

The FINALMODEL option specifies how far to take the sequence of models, with settings common, parallel, separateslopes and the default, full, corresponding to the models just described. Results from the models earlier than the requested final model are printed only if the submodels setting is included in the PRINT option. Further output displayed by RDLOESSGROUPS and information saved by RKLOESSGROUPS will only be from the final model.

The DEVIANCE option saves the residual deviance, and the DF option saves the residual number of degrees of freedom. The RESIDUALS and FITTEDVALUES parameters save the residuals and fitted values, respectively. The ACCUMULATED parameter saves the accumulated analysis-of-variance (or deviance) table as a pointer. The suffixes of ACCUMULATED for the last 4 columns in the pointer depend on whether it is an analysis of variance ('s.s.', 'm.s.', 'v.r.','F pr.') or an analysis of deviance table ('deviance','mean dev.','dev. r.', 'approx F pr.').

The SAVE parameter can save a pointer, with information about the analysis, for use by the procedures RDLOESSGROUPS and RKLOESSGROUPS. These are very similar to the corresponding RDISPLAY and RKEEP directives. Details are in Part 3 of the *Genstat Reference Manual*.

Example 3.4.6 fits a sequence of loess models to the data from a rotational experiment on sugar beet in Example 3.4.5.

Example 3.4.6

```
" 3.4.6) A sequence of LOESS models "
RLOESSGROUPS [PRINT=model,estimates,accumulated,submodels]\
             SoilP; GROUPS=Year
```

```
Regression analysis
===================

 Response variate: Sugar
     Fitted terms: Constant + SoilP
        Submodels: LOESS(SoilP; 4; *; 1)
```

```
Estimates of parameters
-----------------------

Parameter       estimate         s.e.     t(59)
Constant           3.523        0.473      7.44
SoilP Lin         0.0790       0.0145      5.45


Regression analysis
===================

 Response variate: Sugar
     Fitted terms: Constant + SoilP + Year
        Submodels: LOESS(SoilP; 4; *; 1)


Estimates of parameters
-----------------------

Parameter       estimate         s.e.     t(56)
Constant           5.107        0.322     15.84
SoilP Lin        0.07920      0.00716     11.07
Year 2            -4.194        0.335    -12.52
Year 3            -0.389        0.336     -1.16
Year 4            -1.781        0.336     -5.30

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                Year  1


Regression analysis
===================

 Response variate: Sugar
     Fitted terms: Constant + SoilP + Year + SoilP.Year
        Submodels: LOESS(SoilP; 4; *; 1)


Estimates of parameters
-----------------------

Parameter               estimate         s.e.     t(53)
Constant                   3.873        0.393      9.86
SoilP Lin                 0.1214       0.0109     11.10
Year 2                    -2.464        0.578     -4.26
Year 3                     1.653        0.563      2.94
Year 4                    -0.245        0.569     -0.43
SoilP Lin .Year 2        -0.0599       0.0168     -3.57
SoilP Lin .Year 3        -0.0732       0.0171     -4.29
SoilP Lin .Year 4        -0.0548       0.0174     -3.15

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                Year  1


Regression analysis
===================

 Response variate: Sugar
     Fitted terms: Constant + SoilP + Year + SoilP.Year
        Submodels: LOESS(SoilP; 4; *; 1)


Estimates of parameters
-----------------------

     Parameter  estimate      s.e.     t(45)
    Constant 1     4.314    0.2913     14.81
    Year Lin.1     0.102    0.0081     12.63
    Constant 2     1.737    0.2399      7.24
    Year Lin.2     0.053    0.0072      7.32
```

```
    Constant 3     4.881    0.2685     18.18
    Year Lin.3     0.072    0.0087      8.27
    Constant 4     3.310    0.2252     14.70
    Year Lin.4     0.082    0.0074     10.99

Accumulated analysis of variance
--------------------------------

Change                          d.f.      s.s.       m.s.       v.r.    F pr.
+ LOESS(SoilP; 4; *; 1)            4     159.3      39.84     146.16  < 0.001
+ Year                             3     169.6      56.53     207.40  < 0.001
+ SoilP.Year                       3      14.2       4.73      17.36  < 0.001
+ LOESS(SoilP; 4; *; 1).Year       8      23.7       2.97      10.89  < 0.001
Residual                          45      12.3       0.27          *
Total                             63     379.1       6.02          *
```

As in Example 3.4.5, the analysis shows that different loess curves (and slopes and intercepts) are needed.

3.5 Generalized linear models

Generalized linear models extend the ordinary regression framework to situations where the data do not follow a Normal distribution, or where a transformation (known as the *link function*) needs to be applied before a linear model can be fitted. Section 3.5.1 contains a brief account of the essential concepts, but for more information see Dobson (1990) or McCullagh and Nelder (1989).

Example 3.5a shows a probit analysis (Finney 1971). This is a particular type of generalized linear model which models the relationship between a stimulus, like a drug, and a quantal response (recorded simply as success or failure). In probit analysis it is assumed that for each subject there is a certain level of stimulus below which it will be unaffected, but above which it will respond. This level of stimulus, known as the tolerance, will vary from subject to subject within the population. The assumption in Example 3.5a is that the tolerance of the mice to the logarithm of the dose will have a Normal distribution; so, if we were to plot the proportion of the population with each tolerance against log dose, we would obtain the familiar bell-shaped curve. Likewise, if we plotted the probability that a randomly-selected individual will respond, against the logarithm of dose, we would obtain the sigmoid (S-shaped) cumulative-Normal curve limited below by zero and above by one. To make the relationship linear, then, we could transform the y-axis to Normal equivalent deviates or *probits* (see 3.5.1). Thus, in this example, we need a probit link function in order to fit a linear model.

The data in Example 3.5a consist of observations, in each of which a particular dose of one of the drugs was applied to a group of mice, and the number that responded was counted. The data can thus be assumed to follow a binomial distribution, instead of the Normal distribution assumed for the examples earlier in this chapter.

As Example 3.5a shows, you can fit generalized linear models using exactly the same directives as for linear regression: the only difference is that you need to set extra options in the `MODEL` directive to specify the distribution and the link function, and, for binomial data, an extra parameter to define the total number of subjects at each observation. Likewise the generalized linear models menus in Genstat *for Windows* are very similar to the ordinary linear regression menus. The most general menu (General Model) contains extra fields for you to specify these settings, while the more specialized menus may set them automatically.

One important practical difference with generalized linear models is that the entire model is fitted at once rather than one term at a time as in ordinary regression models. As a result the terms are pooled into a single line in the analysis of deviance table. If you want to see the contributions of the individual terms, you need to fit them one at a time, either explicitly by using `ADD` as in Example 3.5a, or automatically by using the `FITINDIVIDUALLY` procedure

(3.5.3). Alternatively, you could use procedure RSCREEN (3.2.9) to produce screening tests, as in Example 3.5.1.

Example 3.5a

```
  2  " Comparison of effectiveness of 3 analgesic drugs to a standard drug,
 -3    morphine. Data from Grewal (1952), analysed by Finney (1971) p.103.
 -4    Four drugs were compared at several doses for their effect on groups
 -5    of mice; the numbers of mice that responded were recorded."
  6  FACTOR [LABELS=!T(Morphine,Amidone,Phenadoxone,Pethidine)] Drug
  7  READ [PRINT=data] Drug,Dose,Ntest,Nrespond

  8  1 1.50 103 19   1 3.00 120 53   1 6.00 123 83
  9  2 1.50  60 14   2 3.00 110 54   2 6.00 100 81
 10  3 0.75  90 31   3 1.50  80 54   3 3.00  90 80
 11  4 5.00  60 13   4 7.50  85 27   4 10.00 60 32
 12                  4 15.00 90 55   4 20.00 60 44 :
 13  " Fit standard probit models, relating the number of responses to the
-14    logarithm of the dose.  The probit model is a generalized linear
-15    model, assuming a binomial distribution for the number of responses
-16    and a probit link function (cumulative Normal distribution function)
-17    between the number of responses and the logarithm of the dose."
 18  CALCULATE Logdose = LOG10(Dose)
 19  MODEL [DISTRIBUTION=binomial; LINK=probit] Nrespond; NBINOMIAL=Ntest
 20  TERMS Logdose*Drug
 21  " Fit a model ignoring the types of drug used."
 22  FIT [NOMESSAGE=leverage,residual; FPROB=yes; TPROB=yes] Logdose

Regression analysis
===================

 Response variate: Nrespond
  Binomial totals: Ntest
     Distribution: Binomial
    Link function: Probit
     Fitted terms: Constant + Logdose


Summary of analysis
-------------------

                                        mean  deviance approx
Source        d.f.     deviance     deviance     ratio chi pr
Regression       1         39.4        39.41     39.41  <.001
Residual        12        210.6        17.55
Total           13        250.0        19.23

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.


Estimates of parameters
-----------------------

Parameter      estimate         s.e.      t(*)  t pr.
Constant        -0.2976       0.0663     -4.49  <.001
Logdose          0.5972       0.0959      6.23  <.001

* MESSAGE: s.e.s are based on dispersion parameter with value 1.

 23  " Fit parallel responses (on the probit scale) for the drugs; morphine
-24    has been assigned as the first level of the factor so that Genstat
-25    will automatically compare the other drugs to it."
 26  ADD [FPROB=yes; TPROB=yes] Drug

Regression analysis
===================

 Response variate: Nrespond
  Binomial totals: Ntest
```

```
     Distribution: Binomial
    Link function: Probit
     Fitted terms: Constant + Logdose + Drug


Summary of analysis
-------------------

                                        mean  deviance approx
Source        d.f.     deviance     deviance     ratio chi pr
Regression       4      246.090      61.5225     61.52  <.001
Residual         9        3.868       0.4298
Total           13      249.958      19.2275

Change          -3     -206.682      68.8940     68.89  <.001

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.


Estimates of parameters
-----------------------

Parameter              estimate         s.e.      t(*)  t pr.
Constant                 -1.379        0.114    -12.08  <.001
Logdose                   2.468        0.173     14.30  <.001
Drug Amidone              0.238        0.108      2.20  0.028
Drug Phenadoxone          1.360        0.130     10.49  <.001
Drug Pethidine           -1.180        0.133     -8.87  <.001

* MESSAGE: s.e.s are based on dispersion parameter with value 1.

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                Drug  Morphine

  27  " Fit separate models for the different drugs"
  28  ADD [PRINT=accumulated; FPROB=yes] Logdose.Drug

Regression analysis
===================

Accumulated analysis of deviance
--------------------------------

                                                  mean  deviance approx
Change                          d.f.     deviance     deviance     ratio chi pr
+ Logdose                          1      39.4079      39.4079     39.41  <.001
+ Drug                             3     206.6821      68.8940     68.89  <.001
+ Logdose.Drug                     3       1.5336       0.5112      0.51  0.675
Residual                           6       2.3344       0.3891

Total                             13     249.9579      19.2275

* MESSAGE: ratios are based on dispersion parameter with value 1.

  29  " There is no evidence of non-parallelism, so return to the parallel
 -30    model and display it with procedure RGRAPH."
  31  DROP [PRINT=*] Logdose.Drug
  32  RGRAPH
```

The graph of the fitted model drawn by the RGRAPH procedure is shown in Figure 3.5. By default, the model is plotted on the natural scale (here percentages). However, if you want to check the linearity of the response on the transformed scale (here probits), you can set option BACKTRANSFORM to either none or axis in the RGRAPH statement. These settings differ in that axis includes axis markings, back-transformed onto the natural scale, on the right-hand side of the y-axis. However, this is not available for log-ratio, power, reciprocal or calculated links. The transformed marks and labels are plotted using pen 4.

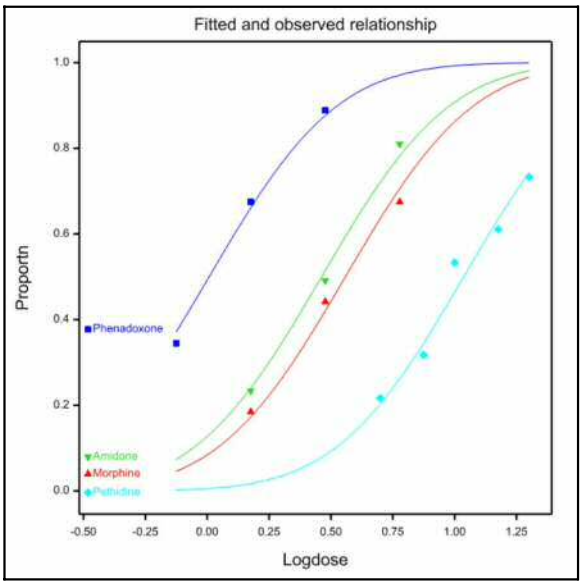


Figure 3.5

This analysis does not include information on LD50s or similar quantities that are usually used to characterize the effectiveness of drugs (see Finney 1971). Functions of the parameters like these can be calculated, with standard errors, using the RFUNCTION directive (3.7.5). There is also a special procedure PROBITANALYSIS, which produces these automatically, as well as being able to estimate natural mortality and immunity; for details see 3.5.9. Alternatively, LD50s and *relative potencies* (which compare one drug with another) can be calculated by procedure FIELLER.

When you have fitted parallel lines, you can use procedure FIELLER to calculate relative potencies of the drugs compared to the standard one (here Morphine), as shown in Example 3.5b. The SLOPE parameter defines the position of the slope in the in the list of parameters that are estimated in the regression model (see Example 3.5a). The TREATMENT parameter defines the positions of the intercepts for the treatments that are to be compared with the standard. The VALUE parameter saves the relative potencies, and the LOWER and UPPER parameters save the lower and upper fiducial limits. For more details about FIELLER see Part 3 of the *Genstat Reference Manual* or the on-line help.

Example 3.5b

```
 33  FIELLER [RELATIVE=yes] SLOPE=2; TREATMENT=3,4,5; \
 34     VALUE=relp[2...4]; LOWER=low[2...4]; UPPER=up[2...4]

  Relative potency   Lower 95%   Upper 95%
           0.09636     0.01032      0.1861

  Relative potency   Lower 95%   Upper 95%
            0.5508      0.4615      0.6465

  Relative potency   Lower 95%   Upper 95%
           -0.4780     -0.5578     -0.3970

 35  " The results are the log-potency and 95% fiducial limits:
-36    transform these to the natural scale."
 37  CALCULATE relp[],low[],up[] = 10**relp[],low[],up[]
 38  PRINT relp[2],low[2],up[2] & relp[3],low[3],up[3] & relp[4],low[4],up[4]

    relp[2]      low[2]       up[2]
      1.248       1.024       1.535


    relp[3]      low[3]       up[3]
      3.554       2.894       4.431
```

```
relp[4]      low[4]       up[4]
 0.3327      0.2768      0.4008
```

3.5.1 Introduction to generalized linear models

Generalized linear models are natural generalizations of ordinary linear regression models. The ordinary regression model can be written as:

$$y_i = \mu_i + \varepsilon_i , \qquad i=1...N$$
$$= \alpha + \Sigma\{\beta_j x_{ji}\} + \varepsilon_i$$

where x_{ji} is the ith observation of the jth explanatory variable and y_i is the ith observation of the response variable; and

$$\mathrm{Var}(y_i) = \sigma^2$$

where σ^2 is constant for all observations. The residuals ε_i are assumed to be uncorrelated, and usually the model is specialized further by assuming the observations y_i to be Normally distributed. So y_i follows a Normal distribution with *expected value* μ_i and variance σ^2.

In a generalized linear model the expected value is still

$$\mathbf{E}(y_i) = \mu_i , \qquad i=1...N$$

but now the linear model describes η_i, the *linear predictor*,

$$\eta_i = \alpha + \Sigma\{\beta_j x_{ji}\}$$

and η_i is related to μ_i by

$$\eta_i = G(\mu_i)$$

where $G()$ is a monotonic and differentiable function called the *link function*. Also,

$$\mathrm{Var}(y_i) = \varphi\, V(\mu_i) , \qquad i=1...N$$

where φ is a *dispersion parameter*, known or unknown. Again the model is usually specialized further, now so that the observations y_i have some distribution such as the Normal, Poisson, binomial, negative binomial, exponential or gamma from the exponential family. $V()$ is a differentiable function, called the *variance function*.

The model could equally well be expressed using the inverse of the link function:

$$\mu_i = G^{-1}(\eta_i)$$

However, the convention is to use G rather than G^{-1} because the model is then similar to fitting a linear model to the link transformation of the response. For example, if G is the log function,

$$\log(\mathbf{E}(y_i)) = \alpha + \Sigma\{\beta_j x_{ji}\}$$

is similar to

$$\mathbf{E}(\log(y_i)) = \alpha + \Sigma\{\beta_j x_{ji}\}$$

but they are not identical – the logarithm of the expectation of a random variable is not the same as the expectation of the logarithm.

Ordinary linear regression is in the class of generalized linear models, with $G()$ being the identity function, φ being σ^2, and $V()$ being constant. Many other familiar statistical models are in this class too.

(a) The model used in the probit analysis of proportions is a generalized linear model with $G(\mu)=\Phi^{-1}(\mu/n)$, where Φ is the cumulative Normal distribution function, $\varphi=1$, and $V(y)=\mu(1-\mu/n)$, n being the number of trials of which y respond. The distribution is usually assumed to be binomial. Example 3.5a shows such an analysis.

(b) The log-linear model for contingency tables is a generalized linear model with $G(\mu)=\log(\mu)$, $\varphi=1$, and $V(y)=\mu$. The distribution for the counts is usually stipulated to be Poisson or multinomial. This is illustrated in Example 3.5.1.

(c) Logistic regression models are very similar to models used in probit analysis, except that they use the logit link function, $G(\mu)=\log(\mu/(n-\mu))$, rather than the probit. Data can often be analysed in two equivalent forms: units may correspond to individuals that are tested, so that the response is always 0 or 1; alternatively, groups of individuals with common values of explanatory variables may be treated as units, so that each data value is the number responding out of the number in the group. Example 3.5.2 shows an analysis using the latter form.

(d) Dilution assays are usually analysed by a model that has $G(\mu)=\log(-\log(1-\mu/n))$, $\varphi=1$, and the binomial distribution. The logarithm of the dilution is included in the model as an offset variable, similarly to the offset in Example 3.5.1, as described later in this subsection.

(e) The proportional-odds and proportional-hazards models for ordinal response variables can be treated as generalized linear models with a multinomial distribution for the response. The link functions for the two models are the logit as in (c) and the complementary-log-log as in (d); the former is shown in Example 3.5.5.

(f) Inverse polynomial models are generalized linear models with $G(\mu)=1/\mu$. They are usually used for response variables with constant coefficient of variation, $V(y)=\mu^2$, rather than constant variance, so the distribution is taken to be gamma.

You can fit these and other models using the options of the MODEL directive. The DISTRIBUTION option specifies the characteristic form of the variance function $V()$, according to these rules:

Distribution	**Variance function, *V***
Normal	1
Poisson	μ
Binomial	$\mu\,(1-\mu/n)$
Bernoulli	$\mu\,(1-\mu)$
Negative Binomial	$\mu+\mu^2/k$
Geometric	$\mu+\mu^2$
Multinomial	$\mu\,(1-\mu/n)$
Exponential	μ^2
Gamma	μ^2
Inverse Normal	μ^3

If you use the binomial distribution, you must put the number of successes (or the number of failures) into the response variate, and supply the total numbers (that is successes plus failures) in another variate using the NBINOMIAL parameter of the MODEL directive. For example:

```
VARIATE [VALUES=3,5,6] Nsuccess
&       [VALUES=5,9,17] Ntrial
MODEL [DISTRIBUTION=binomial] Nsuccess; NBINOMIAL=Ntrial
```

Alternatively, if you have units for each individual, the total numbers will all be 1 and the above statements would be replaced by:

```
VARIATE [VALUES=3(1),2(0), 5(1),4(0), 6(1),11(0)] Nsuccess
MODEL [DISTRIBUTION=binomial] Nsuccess; NBINOMIAL=1
```

This special case of the binomial is known as the Bernoulli distribution. So, instead you could put

```
MODEL [DISTRIBUTION=bernoulli] Nsuccess
```

You must supply the parameter k for the negative binomial distribution using the AGGREGATION option of the MODEL directive. The default value of k is set at 1, which corresponds to the geometric distribution; k must be positive, and as it increases to infinity the distribution approaches the Poisson distribution. To fit a negative binomial generalized linear model, while estimating the aggregation parameter at the same time, you can use procedure RNEGBINOMIAL (see Part 3 of the *Genstat Reference Manual*).

The multinomial distribution can be used only for ordinal response models (3.5.5). A list of response variates is required, one for each category of the response; the number of trials for each unit is determined automatically by adding the values of each response.

When you use the Normal, gamma or inverse Normal distribution, the dispersion parameter φ is usually unknown and is assumed to be constant over all observations. For the Normal distribution this is the constant variance, usually written as σ^2, and for the gamma distribution it is the reciprocal of the index, written either as σ^2 or as ν^{-1}. Sometimes, however, you may

know a value for the dispersion parameter. For example, you may know that the response variable has a Normal distribution with a variance that you can estimate from previous experiments or surveys. In this case, you can fix the value of the dispersion parameter using the DISPERSION option of MODEL (3.1.1). The effect of this is that standard errors and other measures of variability for the fit of the model will be based on the given fixed value rather than on a value estimated from the data. The DFDISPERSION option allows you to specify the number of degrees of freedom for a value specified by the DISPERSION option. If DFDISPERSION is not set, the supplied dispersion is assumed to be known exactly.

The Poisson and binomial distributions do not have any dispersion parameter, so Genstat fixes it at 1.0. This has the effect described above: the variance of an observation is a function only of its mean, and so no estimator of variance is required from the observations as a whole. The exponential distribution also has a dispersion parameter fixed at 1.0 (in fact it is a special case of the gamma distribution with dispersion parameter set to 1.0).

You may sometimes want to include a dispersion parameter even though you are using the binomial, multinomial or Poisson distributions. An example is the *heterogeneity factor* of probit analysis: the distribution of the observations is taken to be "superbinomial", in the sense that the variance is greater than what would be expected for a binomial distribution; specifically, $V(y)=\theta\mu(1-\mu/n)$, where θ is the heterogeneity factor (Finney 1971). This can be achieved by setting the DISPERSION option to *:

```
MODEL [DISTRIBUTION=binomial; DISPERSION=*] Nsuccess;\
  NBINOMIAL=Ntrial
```

By default the dispersion parameter is estimated using the residual deviance but, if you have a Poisson distribution, you can set option DMETHOD=Pearson to request the Pearson chi-square statistic to be used instead.

Data for which a "superbinomial", "supermultinomial" or "superPoisson" distribution incorporating such a heterogeneity factor are needed are called *overdispersed*, or *underdispersed* if θ is less than 1; see McCullagh & Nelder (1989) for more details.

Using a heterogeneity factor means formally that the method of analysis is no longer based on maximum likelihood, because there is no probability distribution in the exponential family to provide a likelihood to be maximized. Instead, the method requires a *quasi-likelihood*, which relies solely on the description of the relationship between variance and mean. However, the model can still be analysed and interpreted in the same way as with a given distribution; see McCullagh & Nelder (1989).

The link function is specified by the LINK option of the MODEL directive. The link functions available in Genstat are as follows:

Link function	$G(\mu)$	$G^{-1}(\eta)$
identity	μ	η
logarithm	$\log(\mu)$	$\exp(\eta)$
logit	$\log(\mu/(n-\mu))$	$n\exp(\eta)/(1+\exp(\eta))$
reciprocal	$1/\mu$	$1/\eta$
power	μ^{power}	$\eta^{(1/\text{power})}$
square root	$\mu^{1/2}$	η^2
probit	$\Phi^{-1}(\mu/n)$	$n\times\Phi(\eta)$
complementary log-log	$\log(-\log(1-\mu/n))$	$n\,(1-\exp(-\exp(\eta)))$
log-ratio	$\log(\mu/(\mu+k))$	$k\times\exp(\eta)/(1-\exp(\eta))$

In the original definition the probit was equal to the Normal equivalent deviate Φ^{-1} plus five but, for simplicity, in Genstat the five is omitted. Similarly, the logit transformation is sometimes defined with a multiplier of ½, but this too is omitted in Genstat.

By default, the power setting uses the exponent −2; you can specify other values using the EXPONENT option, for example:

```
MODEL [DISTRIBUTION=gamma; LINK=power; EXPONENT=1.5] Y
```

The parameter *k* in the log-ratio link can be set using the KLOGRATIO option. The default value is taken from the AGGREGATION option.

For each of the available distributions, one of the links is known as the *canonical link*. This has special properties. In particular, a model with its canonical link always provides a unique set of parameter estimates, whereas with other models this may not be so. There are often practical scientific reasons for using the canonical link, but there may sometimes also be very good reasons for using a non-canonical link. If you do not set the LINK option, the default is the canonical link of the chosen distribution:

Normal	Identity
Poisson	Log
Binomial	Logit
Bernoulli	Logit
Negative Binomial	Log-ratio
Geometric	Log-ratio
Exponential	Reciprocal
Gamma	Reciprocal
Inverse Normal	Power, with exponent −2
Multinomial	Logit

The MODEL directive also allows you to specify your own distributions or link functions or both. There is an example in 3.5.4.

When the binomial distribution is used, it is usually natural to choose the logit, probit or complementary-log-log link function; and vice versa. If another link is chosen with the binomial distribution, it is assumed to relate the expected proportion of responses (rather than the expected number of responses) to the linear predictor. Similarly, if one of the above three links is chosen with a distribution other than the binomial, the number of trials is assumed to be 1.

Only the logit or complementary-log-log links can be used with the multinomial distribution.

An *offset* variable is a variable that appears in the linear predictor without a parameter. It provides for each observation a fixed offset, o_i say, from the estimated constant:

$$G(\mu_i) = o_i + \alpha + \Sigma\{\beta_j x_{ji}\}$$

You set an offset by the OFFSET option of the MODEL directive. Offsets arise naturally in the standard analysis for dilution assay, involving a complementary-log-log link function. The model then takes the form:

$$E(y_i) = n_i \exp(-d_i \exp(\alpha)) = n_i \exp(-\exp(\log(d_i)+\alpha))$$

where y_i is the number of positives out of n_i samples tested at dilution d_i, and α is the unknown concentration. So the logarithm of the dilution is an offset. This model contains no explanatory variables other than the dilution, but the concentration can sometimes be expressed as a linear function of variables such as time. Dilution assays can conveniently be analysed in Genstat using the DILUTION procedure.

Offset variables also occur naturally in log-linear models for rates where each cell has a different exposure time. Example 3.5.1 shows an analysis of data of this kind where the offset adjusts for the different lengths of service of some ships. Notice that the table of estimates has an extra column, giving the antilogarithms of the estimates. These represent multiplicative effects on the natural scale. The column of antilogarithms is produced for generalized linear models with logit link as well as with the log link. With the logit, they represent multiplicative effects on the odds ratio.

Example 3.5.1

```
  2  " Analysis of the damage caused by waves to forward sections of
 -3    cargo-carrying ships. The data, from McCullagh & Nelder (1989) p.204,
 -4    are counts of damage incidents for each combination of three risk
 -5    factors: the type of ship, the year of construction, and the
```

```
  -6    period of operation."
   7  UNITS [NVALUES=40]
   8  FACTOR [LABELS=!T(A,B,C,D,E)] Type
   9  & [LABELS=!T('1960-64','1965-69','1970-74','1975-79')] Construction
  10  & [LABELS=!T('1960-74','1975-79')] Operation
  11  GENERATE Type,Construction,Operation
  12  " Read the number of months service and number of damage incidents."
  13  OPEN '%GENDIR%/Examples/GuidePart2/Ship.dat'; CHANNEL=2
  14  READ [CHANNEL=2] Service,Damage

    Identifier   Minimum      Mean   Maximum    Values   Missing
       Service    0.0000      4674     44882        40         5    Skew
        Damage    0.0000     10.17     58.00        40         5    Skew

  15  CLOSE 2
  16  " Use the log of the number of months of service as an offset in the
 -17    model; CALCULATE turns zeroes into missing values, which will then
 -18    be excluded by TERMS as required for a correct analysis."
  19  CALCULATE Logservice = LOG(Service)

******** Warning 18, code CA 7, statement 1 on line 19

Command: CALCULATE Logservice = LOG(Service)
Invalid value for argument of function.
The first argument of the LOG function in unit  34 has the value 0.0000

  20  MODEL [DISTRIBUTION=poisson; LINK=log; OFFSET=Logservice] Damage
  21  TERMS [FACTORIAL=2] Type * Construction * Operation
  22  " Fit the main effects."
  23  FIT [FPROB=yes; TPROB=yes] Type + Construction + Operation

Regression analysis
===================

 Response variate: Damage
     Distribution: Poisson
    Link function: Log
   Offset variate: Logservice
     Fitted terms: Constant + Type + Construction + Operation


Summary of analysis
-------------------

                                        mean  deviance approx
Source        d.f.     deviance     deviance     ratio chi pr
Regression       8       107.63       13.454     13.45  <.001
Residual        25        38.70        1.548
Total           33       146.33        4.434

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           21         6.00        3.01
           22         2.00       -2.29
           30        11.00        2.30
           36         7.00        2.15

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
            9        39.00        0.70
           11        58.00        0.64
           12        53.00        0.65
           14        44.00        0.59
           16        18.00        0.56
           38        12.00        0.56
```

```
Estimates of parameters
-----------------------

                                                              antilog of
Parameter                 estimate         s.e.      t(*)  t pr.   estimate
Constant                    -6.406        0.217    -29.46  <.001   0.001652
Type B                      -0.543        0.178     -3.06  0.002     0.5808
Type C                      -0.687        0.329     -2.09  0.036     0.5029
Type D                      -0.076        0.291     -0.26  0.794     0.9269
Type E                       0.326        0.236      1.38  0.167      1.385
Construction 1965-69         0.697        0.150      4.66  <.001      2.008
Construction 1970-74         0.818        0.170      4.82  <.001      2.267
Construction 1975-79         0.453        0.233      1.94  0.052      1.574
Operation 1975-79            0.384        0.118      3.25  0.001      1.469

* MESSAGE: s.e.s are based on dispersion parameter with value 1.

Parameters for factors are differences compared with the reference level:
             Factor  Reference level
               Type  A
       Construction  1960-64
          Operation  1960-74

  24  " Try adding the two-factor interactions."
  25  TRY [PRINT=accumulated; FPROB=yes]\
  26      Type.Construction + Type.Operation +  Construction.Operation

Regression analysis
===================

Accumulated analysis of deviance
--------------------------------

                                                    mean  deviance approx
Change                        d.f.     deviance     deviance     ratio chi pr
+ Type
+ Construction
+ Operation                      8      107.633       13.454     13.45  <.001
+ Type.Construction             12       24.108        2.009      2.01  0.020
Residual                        13       14.587        1.122

Total                           33      146.328        4.434

* MESSAGE: ratios are based on dispersion parameter with value 1.


Regression analysis
===================

Accumulated analysis of deviance
--------------------------------

                                                    mean  deviance approx
Change                        d.f.     deviance     deviance     ratio chi pr
+ Type
+ Construction
+ Operation                      8      107.633       13.454     13.45  <.001
+ Type.Operation                 4        4.939        1.235      1.23  0.294
Residual                        21       33.756        1.607

Total                           33      146.328        4.434

* MESSAGE: ratios are based on dispersion parameter with value 1.


* MESSAGE: term Construction.Operation cannot be fully included in the model
because 1 parameter is aliased with terms already in the model.

(Construction 1975-79 .Operation 1975-79) = (Construction 1975-79)

Regression analysis
===================
```

```
Accumulated analysis of deviance
--------------------------------

                                                    mean  deviance approx
Change                          d.f.     deviance     deviance     ratio chi pr
+ Type
+ Construction
+ Operation                        8      107.633       13.454     13.45  <.001
+ Construction.Operation           2        1.787        0.894      0.89  0.409
Residual                          23       36.908        1.605

Total                             33      146.328        4.434

* MESSAGE: ratios are based on dispersion parameter with value 1.

  27  " Perform screening tests for the terms in the model."
  28  RSCREEN [FACTORIAL=2] Type * Construction * Operation

Screening of terms
==================

 Response variate: Damage
     Distribution: Poisson
    Link function: Log
 Free formula: Type*Construction*Operation

* MESSAGE: P-values are from likelihood ratio approximate chi-square tests
           scaled by the dispersion parameter with value 1

* MESSAGE: term Construction.Operation cannot be fully included in the model
because 1 parameter is aliased with terms already in the model.

(Construction 1975-79 .Operation 1975-79) = (Construction 1975-79)


Pooled accumulated analysis of variance or deviance
---------------------------------------------------

pooled terms                      df    deviance     P-value
Terms with 1 element               8      107.63      0.0000
Terms with 2 elements             18       31.84      0.0230
Residual                           7        6.86      0.4439



Significance indications for marginal and conditional tests
-----------------------------------------------------------

Coding: ~ .05<p<=.10; * .01<p<=.05; ** .001<p<=.01; *** p<=.001
Chance frequencies of symbols in 12 tests are:

       ~       *      **     ***
     0.6    0.48   0.108   0.012


                 term     mstar     cstar
                 Type       ***       ***
         Construction       ***       ***
            Operation       ***        **

                 term     mstar     cstar
     Type.Construction         *         *
        Type.Operation
Construction.Operation
```

At the end of the example `RSCREEN` (3.2.9) is used to perform screening tests for the various terms in the model. The marginal tests (the column headed `mstar`) show the effect of adding each term to the simplest possible model: so `Type` is added to a model containing only the constant, while `Type.Construction` is added to a model containing the constant, `Type` and `Construction`. The conditional tests (the column headed `cstar`) show the effect of adding

each term to the most complex possible model: so `Type` is added to a model containing the constant, `Construction`, `Operation` and `Construction.Operation`, while `Type.Construction` is added to a model containing every other term. The degree of non-orthogonality between the model terms is not too serious, so the results of the two tests are similar to each other and to the results from the `TRY` directive.

3.5.2 The deviance

You can assess how well a linear regression fits by doing an analysis of variance. Based on the assumption that the residuals have independent Normal distributions with equal variances, the variance ratio (mean square due to the regression divided by the residual mean square) has an F distribution.

With generalized linear models, there is no similarly simple exact distributional property. However, you can get approximate assessments of the quality of the fit from a statistic called the *scaled deviance*. This is defined as minus twice the log-likelihood ratio between the model you have fitted and a full model that explains all the variation in the data. The scaled deviance has approximately a χ^2_d distribution, d being the number of residual degrees of freedom. The approximation is better for large numbers of observations than for small numbers, and is poor when there are many extreme observations (such as zeroes for the Poisson distribution). In particular, in the special case of a binary response variable (with values 1 and 0), the scaled deviance is absolutely uninformative about the fit of the model.

The scaled deviance is a function of the dispersion parameter, and so its distribution depends also on any estimate of that parameter. Usually you would obtain the estimate from a model that you believe explains all systematic variation – a *maximal model*, as in the analysis of variance for linear regression. You can assess the importance of a term in any generalized linear model by considering the difference between the scaled deviances of that model and the model excluding the term. The difference in scaled deviances also has an approximate χ^2_t distribution, where t is the number of degrees of freedom of the term; in fact this approximation is better than that for the scaled deviance itself.

Alternatively, you can consider ratios of mean scaled deviances between competing models, one of which is nested inside the other. (The mean scaled deviance is the scaled deviance divided by the corresponding number of degrees of freedom.) The resulting ratios do not involve the dispersion parameter. Such a ratio has approximately an F distribution – exact for linear regression models with Normal errors.

Genstat reports the *deviance* of the data for each type of model, which is equivalent to the scaled deviance multiplied by the dispersion parameter. The deviance is otherwise known as the log-likelihood ratio statistic.

You can summarize the fit of a sequence of nested models by an *analysis of deviance*, which you interpret in much the same way as an analysis of variance (but do not forget that the distributions have only approximate χ^2 distributions).

Here are the formulae for the deviance for each distribution; the ith response is represented by y_i, and the corresponding fitted value by f_i:

Normal	$\Sigma(y_i-f_i)^2$
Poisson	$2\,\Sigma\{y_i \log(y_i/f_i)-(y_i-f_i)\}$
Binomial	$2\,\Sigma\{y_i \log(y_i/f_i)+(n_i-y_i) \log((n_i-y_i)/(n_i-f_i))\}$
Bernoulli	$2\,\Sigma\{y_i \log(y_i/f_i)+(1-y_i) \log((1-y_i)/(1-f_i))\}$
Negative Binomial	$2\,\Sigma\{(y_i+k) \log((f_i+k)/(y_i+k)) + y_i \log(y_i/f_i)\}$
Geometric	$2\,\Sigma\{(y_i+1) \log((f_i+1)/(y_i+1)) + y_i \log(y_i/f_i)\}$
Exponential	$2\,\Sigma\{(y_i-f_i)/f_i-\log(y_i/f_i)\}$
Gamma	$2\,\Sigma\{(y_i-f_i)/f_i-\log(y_i/f_i)\}$
Inverse Normal	$\Sigma\{(y_i-f_i)^2/(y_i f_i^2)\}$
Multinomial	$2\,\Sigma\Sigma\{y_{ij} \log(y_{ij}/f_{ij})\}$

Sometimes parameter estimates cannot be obtained. The commonest cause with models using the binomial or Poisson distribution is the presence of observations at the extremes (0 for Poisson, 0 or *n* for binomial). One or more of the parameters may then need to be infinite to maximize the likelihood: in practice, approximate convergence will usually be achieved with the parameters large but finite (the meaning of "large" being dependent on the link function).

This is illustrated in Example 3.5.2: all subjects at level 1 of the factor `Li` responded positively (that is, they were disease-free for three years). Hence, on the logit scale which is the default link function for the binomial distribution, the difference between the two levels is infinite. Genstat achieves convergence here, so the only indications of the problem are the large estimates and standard errors for the constant and "`Li 2`". The `PREDICT` statement shows what is happening: all the predicted proportions at level 1 of `Li` are almost exactly 1.0.

Example 3.5.2

```
  2  " Logistic regression including a factor with a 100% response rate.
 -3    Data from Goorin et al. (1987).
 -4    46 patients were studied, to determine predictors of non-metastatic
 -5    sarcoma: this analysis uses Li (Lymphocytic infiltration), Sex,
 -6    and Aop (any osteoid pathology). The response variable is the number
 -7    disease free for three years."
  8  FACTOR [NVALUES=8; LEVELS=2] Li,Sex,Aop
  9  GENERATE Li,Sex,Aop
 10  VARIATE [VALUES=3,2,4,1,5,3,5,6] Nfree
 11  &       [VALUES=3,2,4,1,5,5,9,17] Nstudy
 12  MODEL [DISTRIBUTION=binomial] Nfree; NBINOMIAL=Nstudy
 13  TERMS Sex,Aop,Li
 14  ADD [PRINT=*] Sex
 15  & Aop
 16  & [PRINT=estimates,accumulated; FPROB=yes; TPROB=yes] Li

Regression analysis
===================

Estimates of parameters
-----------------------

                                                        antilog of
Parameter      estimate         s.e.      t(*)  t pr.   estimate
Constant           13.9         90.4      0.15  0.878   1060002.
Sex 2            -1.636        0.912     -1.79  0.073     0.1947
Aop 2            -1.220        0.771     -1.58  0.114     0.2951
Li 2              -11.8         90.4     -0.13  0.896  7.764E-06

* MESSAGE: s.e.s are based on dispersion parameter with value 1.

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                 Sex  1
                 Aop  1
                  Li  1


Accumulated analysis of deviance
--------------------------------

                                          mean  deviance approx
Change             d.f.     deviance     deviance     ratio chi pr
+ Sex                 1       5.8795       5.8795      5.88  0.015
+ Aop                 1       5.0105       5.0105      5.01  0.025
+ Li                  1       6.9148       6.9148      6.91  0.009
Residual              4       1.6279       0.4070

Total                 7      19.4327       2.7761

* MESSAGE: ratios are based on dispersion parameter with value 1.

  17  PREDICT Sex,Aop,Li
```

```
Predictions from regression model
---------------------------------

These predictions are estimated mean proportions, formed on the scale of the
response variable, corresponding to one binomial trial.

The predictions have been formed only for those combinations of factor levels
for which means can be estimated without involving aliased parameters.

The standard errors are appropriate for interpretation of the predictions as
summaries of the data rather than as forecasts of new observations.

Response variate: Nfree

                     Li           1                      2
                          Prediction        s.e.  Prediction        s.e.
          Sex       Aop
            1         1      1.0000     0.00009      0.8917     0.09345
                      2      1.0000     0.00029      0.7083     0.17530
            2         1      1.0000     0.00044      0.6157     0.15173
                      2      1.0000     0.00148      0.3211     0.10912


* MESSAGE: s.e's, variances and lsd's are approximate, since the model is
not linear.

* MESSAGE: s.e's are based on dispersion parameter with value 1
```

Occasionally, the iterative process may converge only very slowly when a parameter needs to be infinite: you can increase the limit on the number of cycles with the RCYCLE directive (3.5.4), though this may not always help. Most of the results of the analysis are usually reliable, with the exception of the affected parameter estimates and standard errors, and also the leverages. If a parameter representing the reference level of a factor needs to be infinite, the side-effects on other parameters can be reduced to choosing another level to be the reference level (3.3.2). Very rarely you may even get divergence; this can also happen when the initial guesses for the fitted values are very bad, and the deviance appears to increase after the first cycle. But usually in such cases, the model would not fit the data satisfactorily anyway.

Failure to find a solution may occur when estimates from a fit take impossible values. For example, the gamma distribution is defined in the range (0, ∞), but some sets of data may produce an estimated mean that is negative. In such cases, you should consider a different link, or try a new fit omitting those explanatory variables whose parameters were estimated as negative.

3.5.3 Modifications to output and the RKEEP and PREDICT directives

Some aspects of the results of fitting generalized linear models differ from those described for linear regression, because of the iterative process that is involved. We call any generalized linear model other than linear regression an *iterative* model.

Genstat will analyse only one response variate if the model is iterative, except for models for ordinal response where several response variates are involved in each set of data (3.5.5). If the Y parameter of the MODEL statement contains more than one variate, Genstat will analyse only the first. This is because the fitting process involves weights that depend on the fitted values, which would thus differ from response variate to response variate (see ITERATIVEWEIGHTS below).

The SELECTION option of the fitting directives and of RDISPLAY controls which statistics accompany the summary of analysis. The default for the Normal distribution provides the percentage variance accounted for together with the standard error of the observations. With the gamma distribution, the default setting is cv%, which displays the percentage coefficient of variation of the observations (equal to the square root of the dispersion parameter). For other

distributions, the default setting is `dispersion`, which displays the estimate of the dispersion parameter or the assumed value if the dispersion is fixed, as with the Poisson, binomial, Bernoulli and exponential distributions. `SELECTION` also has two settings, `%meandeviance` and `%deviance`, that are specifically for generalized linear models. These provide analogous summaries, in terms of deviance, to the percentage variance and sum of squares accounted for by linear models (and settings `%variance` and `%ss` are interpreted as requesting `%meandeviance` and `%deviance`, respectively, if the distribution is not Normal).

In generalized linear models with the log or logit link function, an extra column is included in the table of parameter estimates, produced by the `estimates` setting of the `PRINT` option. This gives the antilogarithm of the estimates, which can be then interpreted as multiplicative effects on the scale of the response or on the odds ratio scale respectively.

The standard errors of the parameter estimates are only approximate for iterative models; the same applies to the t-statistics, and to the correlations produced by the `correlations` setting. The `TPROBABILITY` option can still be used to request probabilities, but you should bear in mind that the adequacy of the approximation depends on the model and the context, and should use the values only as a guide. You can get a better test of the corresponding parameter by dropping it from the model and then assessing the change in the deviance.

Genstat displays leverages with the `fittedvalues` setting of the `PRINT` option and allows them to be stored by the `LEVERAGE` parameter of the `RKEEP` directive. With iterative models, the formula for the *i*th leverage is:

$$l_i = u_i w_i \{\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{U}\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}'\}_{ii}, \quad i = 1...N$$

where $\boldsymbol{U}$ is a diagonal matrix consisting of the iterative weights u_i (defined below). These values are also used in the standardization of residuals, according to the formula given in 3.1.1. However, no leverages are formed for ordinal response models, because there is no analogous quantity for assessing influence in these effectively multivariate generalized linear models; the standardized residuals, therefore, contain no adjustment for relative influence.

By default, the residuals are deviance residuals, as described in 3.1.1: each residual is the signed square root of the contribution to the deviance. (See 3.5.2 for the definition of deviance for each distribution.) The standardization of the residuals uses the leverages, l_i, described above, and the weights, w_i, if specified; by default, if the `WEIGHTS` option of `MODEL` is not set the weights are 1.0. The *i*th residual is

$$r_i = \text{sign}(y_i - f_i) \sqrt{\{w_i d_i / (s^2(1 - l_i))\}}$$

where d_i is the contribution to the deviance from unit i, and s^2 is the estimated or fixed dispersion. For example, the deviance residuals for a model with the Poisson distribution are given by:

$$r_i = \text{sign}(y_i - f_i) \sqrt{\{2\,(y_i \log(y_i/f_i) - (y_i - f_i)) / (1 - l_i)\}}$$

If you set the `RMETHOD` option of the `MODEL` directive to `Pearson`, Genstat forms the residuals by adjusting the ordinary residuals for their estimated variance:

$$r_i = (y_i - f_i) \sqrt{\{w_i / (V(f_i) s^2 (1 - l_i))\}}$$

With the binomial distribution, the table produced by the `fittedvalues` setting includes a column for the binomial totals specified by the `NBINOMIAL` parameter of the `MODEL` directive. For the multinomial distribution, a separate table is printed for each category.

The `accumulated` setting of the `PRINT` option produces an accumulated analysis of deviance for iterative models, just as for linear models except that all contributions from one statement are pooled. The `POOL` option of directives like `FIT`, has no effect with iterative models. Thus you cannot calculate the change in deviance attributable to each individual term unless you add the terms into the model individually. For example, these statements would provide a full analysis of deviance for two factors A and B and their interaction:

```
TERMS A*B
ADD [PRINT=*] A
& B
& [PRINT=accumulated] A.B
```

The alternative is to use procedure FITINDIVIDUALLY, which will automatically fit a regression (or generalized linear) model one term at a time. It is used exactly like FIT. It must be preceded by a MODEL statement, and can be followed by RCHECK, RDISPLAY, RGRAPH, RKEEP, ADD, DROP, SWITCH and so on. It has a TERMS parameter to specify the terms to be fitted, like the parameter of FIT. It also has options PRINT, CONSTANT, FACTORIAL, POOL, DENOMINATOR, NOMESSAGE, FPROBABILITY, TPROBABILITY, SELECTION just like those of FIT. So we could equivalently obtain a full analysis of deviance for factors A and B and their interaction by

```
FITINDIVIDUALLY [PRINT=accumulated] A.B
```

The monitoring setting of the PRINT option provides a report on the progress of the fit. Example 3.5.3 shows how convergence was achieved in Example 3.5.2 above.

Example 3.5.3

```
  18  FIT [PRINT=monitoring] Sex,Aop,Li

Convergence monitoring
----------------------

Scoring cycle  Deviance    Current parameters
1              12.618057    -0.695419    -0.801977    -0.470358
2              4.2247979     -1.22439     -1.03232     -1.58180
3              2.4242137     -1.54368     -1.18072     -2.66966
4              1.9038448     -1.62716     -1.21515     -3.72636
5              1.7275255     -1.63560     -1.21979     -4.75119
6              1.6642328     -1.63614     -1.22030     -5.76057
7              1.6411534     -1.63620     -1.22037     -6.76404
8              1.6326904     -1.63620     -1.22038     -7.76531
9              1.6295808     -1.63620     -1.22038     -8.76578
10             1.6284373     -1.63620     -1.22038     -9.76595
11             1.6280167     -1.63620     -1.22038     -10.7660
12             1.6278620     -1.63620     -1.22038     -11.7660

Convergence in scoring loop at cycle 12.
```

The criteria of the STEP directive (3.2.7) use the residual deviance from the model rather than the residual sum of squares as used for a linear model.

Three of the parameters of the RKEEP directive are relevant only for saving results of iterative models. The LINEARPREDICTOR parameter lets you save the linear predictor; that is

$$p_i = a + o_i + \Sigma\{b_j x_{ij}\} , \qquad i = 1...N$$

where a and b_j are estimates of α and β_j. The values of the linear predictor are the same as the fitted values if the link function is the identity function. You can save standard errors for the linear predictor using the SELINEARPREDICTOR parameter.

The ITERATIVEWEIGHTS parameter saves a variate containing the iterative weights used in the last cycle of the iteration. The weight for unit i is

$$\{ V(f_i) \}^{-1} \{ p_i' \}^{-2}$$

where $V()$ is the variance function (3.5.1) and p_i' is the derivative of the linear predictor with respect to the mean. The iterative weights do not contain any contribution from the weights that can be specified whether or not the model is iterative by the WEIGHTS option of the MODEL directive. The iterative weights are 1.0 for ordinary linear regression.

The YADJUSTED parameter saves the adjusted response variate Z that was used in the last cycle of the iteration:

$$z_i = p_i + (y_i - f_i)p_i'$$

With the identity link function this is the same as the response variate.

The Pearson chi-square statistic can be saved using the PEARSONCHI parameter of RKEEP. It is defined as

$$\Sigma\{ (y_j - f_i)^2 / V(f_i) \}$$

and can be used as an alternative to the deviance for testing goodness of fit; see Nelder & McCullagh (1989) page 37.

The EXIT parameter of RKEEP provides a code that indicates the success or type of failure when fitting a generalized linear model (codes for nonlinear models are given in 3.7.4).

0 Successful fitting
8 Data incompatible with model
9 Predicted mean or linear predictor out of range
10 Invalid calculation for calculated link or distribution
11 All units have been excluded from the analysis
12 Iterative process has diverged
13 Failure due to lack of space or data access

With a generalized linear model, the EXIT code is usually the only information that you can save if the fit has been unsuccessful. Howevber, if you set option IGNOREFAILURE=yes, RKEEP will save any information that may be available. (You may thus, for example, be able to discover more about the cause of the failure.)

The DISPERSION and DMETHOD options of RKEEP are also relevant only for generalized linear models. They operate in the same way as those options of MODEL (3.1.1), and allow you to change the way in which the deviance is calculated for the quantities saved by RKEEP.

The PREDICT directive forms summaries of the fit of an iterative model as for a linear model. However, note that averaging is done by default on the scale of the original response variable, not on the scale transformed by the link function. In other words, linear predictors are formed for all the combinations of factor levels and variate values specified by PREDICT, and then transformed by the link function back to the natural scale. This back transformation may be useful when you are reporting results, since the tables from PREDICT can then be interpreted as natural averages of means predicted by the fitted model. You can set option BACKTRANSFORM=none if you would prefer the averaging to be done on the scale of the linear predictor; PREDICT will then form averages and report predictions on the transformed scale. You could then use the BACKTRANSFORM procedure to transform these back onto the natural scale.

The NBINOMIAL option of PREDICT is also relevant only to generalized linear models, allowing you to specify a total number of trials to use when forming predictions from a binomial distribution. Genstat then predicts the number of successful trials. The default for NBINOMIAL is 1, which gives the predicted proportion of successful trials.

The OFFSET option of PREDICT directive is most likely to be used when forming predictions from a generalized linear model. Bu default, predictions are made at the mean of the offset variate, but you can set the OFFSET option to any value to produce predictions at that value. Thus, for example, the contribution from the offset can be excluded altogether by setting OFFSET=0.

PREDICT calculates the standard errors of predictions from iterative models by using first-order approximations that allow for the effect of the link function. Thus you should interpret them only as a rough guide to the variability of individual predictions.

3.5.4 The RCYCLE directive

RCYCLE directive

Controls iterative fitting of generalized linear, generalized additive and nonlinear models, and specifies parameters, bounds etc for nonlinear models.

Options

MAXCYCLE = *scalars* — Maximum number of iterations for Fisher-scoring algorithm (used in generalized linear models), back-

	fitting algorithm (used in additive models) and nonlinear algorithms; single setting implies the same limit for all; default 15, 15, 30
TOLERANCE = *scalar* or *variate*	Scalar or first unit of a variate defines the convergence criterion for the relative change in deviance and, if required, the second element of a variate defines the criterion for convergence to a zero deviance; default !(0.0001,1.0E-11)
FITTEDVALUES = *variate*	Initial fitted values for generalized linear model; default *
METHOD = *string token*	Algorithm for fitting nonlinear model (GaussNewton, NewtonRaphson, FletcherPowell); default Gaus, but Newt for scalar minimization
LINEARPARAMETERS = *scalars*	Scalars to hold current values of linear parameters used in nonlinear model, for reference within model calculations

Parameters

PARAMETER = *scalars*	Nonlinear parameters in the model
LOWER = *scalars*	Lower bound for each parameter
UPPER = *scalars*	Upper bound for each parameter
STEPLENGTH = *scalars*	Initial step length for each parameter
INITIAL = *scalars*	Initial value for each parameter

The parameters of the RCYCLE directive are ignored when generalized linear models are fitted; see 3.7.6 and 3.8.1 for their use in nonlinear models.

The MAXCYCLE option allows you to change the limit on the number of cycles in the iterative estimation process. Usually, the algorithm converges in four or five cycles, but when there are many extreme observations more cycles may be needed; however, the resulting fit is then often uninformative.

The TOLERANCE option can be set to a scalar or a variate to control the criterion for convergence in generalized linear and generalized additive models. A scalar or the first unit of a variate defines the convergence criterion for the relative change in deviance (default 0.0001). The iteration stops when the absolute change in deviance in successive cycles is less than the tolerance multiplied by the current value of the deviance. The second element of a variate defines the criterion for convergence to a zero deviance. If TOLERANCE is unset, or if it is set to a scalar, the default criterion for zero deviance is 1.0E−11.

When additive terms are included in the model (3.4.3, 3.4.4), Genstat fits the resulting *generalized additive model* by nested iteration (Lane & Hastie 1992). This means that at each cycle of the iterative fit required by the presence of a non-identity link function or non-Normal distribution or both, the iterative search described in 3.4.3 will take place. This can, of course, be a time-consuming operation, particularly if the number of units is large. Nested iterations also take place when you are fitting generalized nonlinear models (3.5.8); these extend the ordinary generalized linear models by the inclusion of nonlinear parameters into model for the linear predictor. These iterative processes are all controlled by the settings of MAXCYCLE and TOLERANCE.

The algorithm has to start by estimating an initial set of fitted values. Genstat usually obtains these by a simple transformation of the observed responses. It may be that better estimates are available, for example from a previously fitted model; if so, you can supply these by the FITTEDVALUES option.

The METHOD option is relevant only for nonlinear models, as described in 3.8.1.

3.5.5 Models for multinomial and ordinal responses

The models in this section may be relevant when a response variable can take one out of a fixed set of possible values. A response variable of this kind is called *polytomous*, and the possible values are called *response categories*.

When the categories are purely nominal – that is, with no concept of an ordering – it is natural to assume that the data are allocated to the categories according to a multinomial distribution. This can be fitted using the FITMULTINOMIAL procedure. The counts must all be put into a single y-variate, and a "response" factor must be defined (with one level for each category) to record the category shown by each observation. The procedure has a RESPONSEFACTOR option to specify which is the response factor, and a CLASSIFICATION option that can be used to specify the explanatory factors that classify the subjects. The model to be fitted is specified by the TERMS parameter, and the factors in that model provide the default for CLASSIFICATION if that is not set. The other options, PRINT, RESPONSEFACTOR, CLASSIFICATION, FACTORIAL, POOL, DENOMINATOR, NOMESSAGE, FPROBABILITY, TPROBABILITY and SELECTION operate in the same way as in FIT (3.1.2). The model is fitted with the ordinary generalized linear models commands as it if were a log-linear model, by using the fact that a multinomial distribution can be generated by taking the sum of several Poisson variables (one for each outcome of the multinomial), and then constraining their sum to be equal to the multinomial total (see McCullagh & Nelder 1989, or any book on probability distributions).

So FITMULTINOMIAL first fits a model defined as all factorial combinations of the CLASSIFICATION factors. This imposes the constraint that the Poisson variables sum to the totals of the multinomial distribution. The effects of these terms assess how the design has been set up – i.e. how the subjects have been allocated to the treatments – but they have no information on the effects of the treatments on the response.

It then fits RESPONSEFACTOR. This represents the overall distribution of the response categories across the subjects, and is analogous to the grand mean in an ordinary analysis. (This must be fitted, and so FITMULTINOMIAL has no CONSTANT option.) Finally it fits the interactions of the terms in TERMS with RESPONSEFACTOR. These show how the distribution of subjects to response categories is affected by the treatment terms – which is the main interest of the analysis.

Alternatively, if the categories are on an interval scale, so that differences between categories can be compared quantitatively, the response variable can be analysed as for a continuous variable, using linear regression or some generalized linear model with an appropriate distribution.

The main topic of this section, however, is the analysis of ordinal data. With ordinal categories, there is a known ordering of the categories but no concept of distance between them. Genstat provides two possible models for the relationship between explanatory variables and the division into categories. These are both cumulative models, describing the relationship between numbers of observations up to a particular category and the explanatory values. They are described in Chapter 5 of McCullagh & Nelder (1989), where they are called the *proportional-odds model* and the *proportional-hazards model*. They have the following form:

$$G(\gamma_{ij}) = \theta_j - \Sigma\{\beta_i x_{ij}\}$$

where $G()$ is the logit or complementary-log-log link function, respectively, and γ_{ij} is the probability that the response for unit i is in category j or lower. The quantities θ_j are referred to as the *cut-points*, and provide a quantification of the difference between successive categories on the scale of the chosen link function. It is conventional to have the minus sign in this model, rather than the plus sign that would be expected in a multiple linear model: this convention ensures that as the linear predictor increases, the probability of the response lying in the higher categories also increases.

Example 3.5.5 uses the proportional odds model. Note that it is necessary to set the option YRELATION=cumulative in the MODEL statement, as well as DISTRIBUTION=multinomial;

this is to allow for further models using the multinomial distribution in the future.

Example 3.5.5

```
  2  " Analysis of a tasting experiment with ordinal response categories.
 -3    Data from McCullagh & Nelder (1989) p.175.
 -4    Four types of cheese were rated by 52 panellists on a nine-point
 -5    'hedonic scale' for taste, ranging from 'strong dislike' (1) to
 -6    'excellent taste' (9)."
  7  READ [PRINT=data] Taste[1...9]

  8   0  0  1  7  8  8 19  8  1
  9   6  9 12 11  7  6  1  0  0
 10   1  1  6  8 23  7  5  1  0
 11   0  0  0  1  3  7 14 16 11 :
 12  FACTOR [LABELS=!t(A,B,C,D); VALUES=1...4] Cheese
 13  " Specify the proportional-odds model (LINK=logit is the default)
-14    and ask for Pearson residuals rather than deviance residuals,
-15    since these are reported by McCullagh and Nelder."
 16  MODEL [DISTRIBUTION=multinomial; YRELATION=cumulative; \
 17        RMETHOD=Pearson] Taste[]
 18  " Use full parameterization to get differences with Cheese D, as in
-19    McCullagh & Nelder, rather than with Cheese A."
 20  TERMS [FULL=yes] Cheese
 21  FIT   [FPROB=yes; TPROB=yes] Cheese

Regression analysis
===================

Response variates: ordinal model for categories defined by
                   Taste[1], Taste[2], Taste[3], Taste[4], Taste[5],
                   Taste[6], Taste[7], Taste[8], Taste[9]
     Distribution: Multinomial
    Link function: Logit
     Fitted terms: Cheese


Summary of analysis
-------------------

                                        mean  deviance approx
Source        d.f.     deviance     deviance     ratio chi pr
Regression       3       148.45      49.4846     49.48  <.001
Residual        21        20.31       0.9671
Total           24       168.76       7.0318

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.


Response variate: Taste[4]

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
            1         7.00        2.23


Response variate: Taste[6]

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
            2         6.00        2.30


Estimates of parameters
-----------------------

                                                        antilog of
Parameter           estimate         s.e.      t(*)  t pr.   estimate
Cut-point 0/1         -7.080        0.562    -12.59  <.001  0.0008416
Cut-point 1/2         -6.025        0.475    -12.67  <.001   0.002418
```

```
Cut-point 2/3         -4.925        0.427    -11.53  <.001   0.007260
Cut-point 3/4         -3.857        0.390     -9.88  <.001    0.02114
Cut-point 4/5         -2.521        0.343     -7.35  <.001    0.08042
Cut-point 5/6         -1.569        0.309     -5.08  <.001     0.2083
Cut-point 6/7         -0.067        0.266     -0.25  0.801     0.9353
Cut-point 7/8          1.493        0.331      4.51  <.001      4.450
Cheese A              -1.613        0.378     -4.27  <.001     0.1993
Cheese B              -4.965        0.474    -10.47  <.001   0.006981
Cheese C              -3.323        0.425     -7.82  <.001    0.03606
Cheese D                   0            *         *      *      1.000

* MESSAGE: s.e.s are based on dispersion parameter with value 1.
```

3.5.6 Non-standard distributions and link functions

If you want a non-standard distribution for the response variable or a non-standard link function, you can specify your own. It will then be up to you to ensure that the iterative process is suitable and to decide how to interpret the resulting fit (if convergence is achieved). Formally, the methods for generalized linear models are suitable only for distributions in the exponential family, and for a monotonic differentiable link function.

To specify your own distribution, you need to set `DISTRIBUTION=calculated` in the `MODEL` statement. You must then supply expression structures with the `DCALCULATION` option to calculate the deviance and the variance function for each unit of the response variate, using the current values of the fitted-values variate. You must also set the `FITTEDVALUES`, `DEVIANCE` and `VFUNCTION` parameters of the `MODEL` statement to indicate which identifiers are used to represent these in the expressions.

For example, the following statements specify the calculations for the gamma distribution (though it would be more efficient of course just to set `DISTRIBUTION=gamma`). The deviance is calculated by expression `Dc[1]` and placed into the scalar `D`, and the variance function `V` is defined by expression `Dc[2]`.

```
EXPRESSION Dc[1]; VALUE=!e(D=2*((Y-F)/F-log(Y/F)))
& Dc[2]; VALUE=!e(V=F*F)
MODEL [DISTRIBUTION=calculated; LINK=reciprocal; \
  DCALCULATION=Dc[]] Y; FITTED=F; VFUNCTION=V; DEVIANCE=D
FIT X
```

To specify your own link, you need to set `LINK=calculated` and provide expressions for two other calculations to form the fitted values and the derivative of the link function for each unit of the response variate, using the current values of the linear predictor. You must also set the `FITTEDVALUES`, `LINEARPREDICTOR` and `DERIVATIVE` parameters to specify the identifiers used to represent these in the calculations. In addition, you must provide initial values for the linear predictor, so that the iterative process can get started: often this can be done just by applying the link function to the response variate itself, but it may be necessary to modify extreme values such as 0 that may be mapped to infinity by the link function.

Example 3.5.6 defines a link function for a probit model, incorporating a known control mortality. (If the control mortality is not known, the model cannot be treated as a generalized linear model, but the `PROBITANALYSIS` procedure, described in 3.5.9, can be used instead or you could define a generalized nonlinear model – see 3.5.8.) The inverse of the link function here takes the form

$$\mu = n(c + (1-c)\Phi(\eta))$$

where c is the control mortality, and the derivative of the link is

$$d = \sqrt{(2\pi)}\exp(\eta^2/2)/(n(1-c))$$

Example 3.5.6

```
  2  " Analysis of toxicity of derris roots to grain beetle, using
 -3    probit analysis with allowance for control mortality.
```

```
  -4    Data from Martin (1940), analysed by Finney (1971) p131."
   5  READ [PRINT=data] Conc,Nspray,Ndead

   6  1480 142 142  1000 127 126   480 128 115   120 126  58
   7   619 125 125   458 117 115   310 127 114   149  51  40
   8  37.1 132  37  :
   9  FACTOR [LABELS=!t(w213,w214); VALUES=4(1),5(2)] Root
  10  CALCULATE Logconc = LOG10(Conc)
  11  " Estimate of control mortality is 17% "
  12  SCALAR [VALUE=0.17] Cm
  13  " Give calculations for probit link with control mortality."
  14  EXPRESSION [VALUE=Fv1=Nspray*(Cm+(1-Cm)*NORMAL(Lp1))] E[1]
  15  & [VALUE=Ld1=SQRT(2*C('pi'))*EXP(Lp1**2/2)/Nspray/(1-Cm)] E[2]
  16  MODEL [DISTRIBUTION=binomial; LINK=calculated; LCALCULATION=E[1,2]] \
  17    Ndead; NBINOMIAL=Nspray; LINEARPRED=Lp1; FITTED=Fv1; DERIVATIVE=Ld1
  18  " Initialize the linear predictor."
  19  CALCULATE Lp1 = NED((Ndead+0.5)/(Nspray+1))
  20  FIT [FPROB=yes; TPROB=yes] Logconc,Root

Regression analysis
===================

 Response variate: Ndead
  Binomial totals: Nspray
     Distribution: Binomial
    Link function: Calculated from: E[1], E[2]
     Fitted terms: Constant, Logconc, Root


Summary of analysis
-------------------

                                        mean  deviance approx
Source        d.f.     deviance     deviance     ratio chi pr
Regression       2      450.778      225.389    225.39  <.001
Residual         6        7.391        1.232
Total            8      458.168       57.271

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
            4        58.00        0.70


Estimates of parameters
-----------------------

Parameter       estimate         s.e.      t(*)  t pr.
Constant          -6.222        0.462    -13.46  <.001
Logconc            2.795        0.184     15.16  <.001
Root w214          0.668        0.146      4.57  <.001

* MESSAGE: s.e.s are based on dispersion parameter with value 1.

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
                Root  w213
```

The methods described above are suitable for all straightfoward user-specified generalized linear models. The GLM procedure provides an alternative for situations where you cannot specify the link or distribution by straightforward expressions. Here the information is supplied by user-defined subsidiary procedures, called by GLM, so can use any Genstat command to carry out the calculations. Full details are in Part 3 of the *Genstat Reference Manual*.

3.5.7 Generalized additive models

The use of the SSPLINE and LOESS functions to define additive models is described in Sections 3.4.3 and 3.4.4. When they are included within the context of a generalized linear model, the models are called *generalized additive models* (Hastie & Tibshirani 1990). The Genstat specification simply combines the constructs already described in Sections 3.4.3, 3.4.4 and 3.5.1. Example 3.5.7 presents an example from Hastie & Tibshirani (1990). The data here have a Bernoulli distribution (i.e. binomial with NBINOMIAL=1), and we use the default logit link.

Example 3.5.7

```
   2  " Generalized additive model:
  -3    Data on 83 patients undergoing corrective spinal surgery;
  -4    determine risk factors for kyphosis (forward flexion of the spine).
  -5    Data from Hastie & Tibshirani p.301."
   6  FILEREAD [PRINT=summary; NAME='Kyphosis.dat'] \
   7           Unit,Kyphosis,Age,Number,Start; FGROUPS=no

Summary
-------

The file Kyphosis.dat is assumed to contain 5 structure(s), with one value for
each structure on each record.

The file contains 83 values for each of the following structures:

  Identifier      Type   Missing
        Unit   variate         0
    Kyphosis   variate         0
         Age   variate         0
      Number   variate         0
       Start   variate         0

   8  " Fit smooth effects of Age, Number and Start. "
   9  MODEL [DISTRIBUTION=binomial] Kyphosis; NBINOMIAL=1
  10  TERMS SSPLINE(Age,Number,Start; 3)
  11  FIT   [FPROB=yes; TPROB=yes] SSPLINE(Age)+SSPLINE(Number)+SSPLINE(Start)

Regression analysis
===================

 Response variate: Kyphosis
  Binomial totals: 1
     Distribution: Binomial
    Link function: Logit
     Fitted terms: Constant + Age + Number + Start
        Submodels: SSPLINE(Age; 3)
                   SSPLINE(Number; 3)
                   SSPLINE(Start; 3)


Summary of analysis
-------------------

                                        mean  deviance approx
Source        d.f.     deviance     deviance     ratio chi pr
Regression       9        39.76       4.4177      4.42  <.001
Residual        73        47.04       0.6444
Total           82        86.80       1.0586

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.

* MESSAGE: the residuals do not appear to be random;
           for example, fitted values in the range 0.00 to 0.07
           are consistently larger than observed values
           and fitted values in the range 0.58 to 0.77
           are consistently smaller than observed values.
```

```
* MESSAGE: the error variance does not appear to be constant;
           large responses are more variable than small responses.

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
           15         0.00        0.53
           25         0.00        0.34
           28         1.00        0.87
           45         0.00        0.39
           55         1.00        0.35


Estimates of parameters
-----------------------

                                                        antilog of
Parameter        estimate         s.e.      t(*)  t pr.   estimate
Constant            -1.95         1.44     -1.35  0.176     0.1424
Age Lin           0.01156      0.00785      1.47  0.141      1.012
Number Lin          0.379        0.190      1.99  0.046      1.460
Start Lin         -0.1872       0.0760     -2.46  0.014     0.8292

* MESSAGE: s.e.s are based on dispersion parameter with value 1.
```

3.5.8 Generalized nonlinear models

Generalized nonlinear models are models that include some nonlinear parameters, but are otherwise in the form of generalized linear models. Such models are fitted relatively efficiently by fitting a standard generalized linear model at each stage of an iterative search for optimum values of the nonlinear parameters.

These models can be fitted with the FIT directive, and modified with directives like ADD just as for generalized linear models. The nonlinear parts of the model are specified using the CALCULATION option of FIT, which should be set to one or more expression structures storing the nonlinear parts of the calculation of fitted values. An RCYCLE statement must be given before FIT to list the nonlinear parameters, and perhaps to set initial values and bounds for them. To avoid confusion with the use of RCYCLE in FITCURVE and FITNONLINEAR, it is the setting of the CALCULATION option that signals the new type of model rather than the use of the RCYCLE directive.

As in FITNONLINEAR (3.8) you can also carry out the calculations using Fortran rather than Genstat expressions. FIT has options OWN, INOWN and OUTOWN for this purpose. The NGRIDLINES option allows you to evaluate the deviance on a grid of parameter values, to study the behaviour of awkward functions.

Example 3.5.8a shows a simple use of the CALCULATION option in FIT to estimate a transformation for an explanatory variate. It could as easily be carried out with the FITNONLINEAR directive (3.8), but with FIT the same idea can also be used in generalized linear models, which FITNONLINEAR cannot handle. The data are measurements of length and age of dugongs. These data are also analysed below, in Section 3.7. There, an asymptotic regression curve is fitted; but here, we attempt to fit a linear relationship between Length and Age transformed with

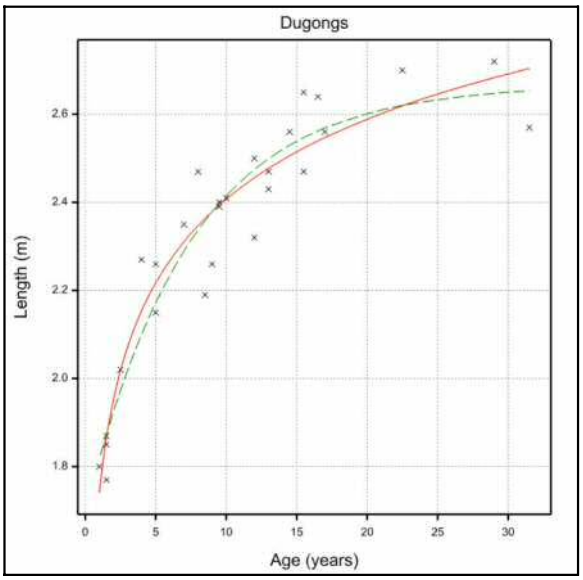


Figure 3.5.8a

the Box-Cox transformation. The expression `Boxcox` transforms `Age` according to the value of the scalar `Bc`, and stores the result in variate `Tage`. The expression is complicated by the definition of the transformation as `LOG(Age)` if the parameter is zero, and to protect the calculation of $\texttt{Age}^{\texttt{Bc}-1}/\texttt{Bc}$ by a logical expression to avoid division by zero. The fit of the model is shown in Figure 3.5.8a as a solid line, with the exponential curve fitted in Section 3.7 shown as a dotted line for comparison.

Example 3.5.8a

```
   2  " Example of estimating transformation of explanatory variable:
  -3    relationship between length and age of dugongs.
  -4    Data from Ratkowsky (1983) p.101."
   5  OPEN   '%GENDIR%/Examples/GuidePart2/Dugong.dat'; CHANNEL=2
   6  READ   [CHANNEL=2] Age,Length

    Identifier   Minimum      Mean   Maximum    Values   Missing
           Age     1.000     10.94     31.50        27         0
        Length     1.770     2.335     2.720        27         0

   7  CLOSE  2
   8  MODEL  Length
   9  RCYCLE Bc; INITIAL=1
  10  EXPRESSION Boxcox; VALUE=\
  11         !e( Tage = LOG(Age)*(Bc==0) + (Age**Bc-1)/(Bc+(Bc==0))*(Bc/=0) )
  12  FIT    [CALCULATION=Boxcox] Tage

Nonlinear regression analysis
=============================

 Response variate: Length
Nonlinear parameters: Bc
  Model calculations: Boxcox
     Fitted terms: Constant, Tage


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression       2       1.7870     0.893491    123.03
Residual        24       0.1743     0.007262
Total           26       1.9613     0.075434

Percentage variance accounted for 90.4
Standard error of observations is estimated to be 0.0852.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           11       2.1900       -2.04


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
Bc                  -0.066        0.135
* Linear
Constant             1.740
Tage                0.3122

  13  FIT    [PRINT=estimates; CALCULATION=Boxcox; SELINEAR=yes] Tage

Nonlinear regression analysis
=============================

Estimates of parameters
-----------------------
```

```
Parameter          estimate          s.e.
Bc                   -0.066         0.135
* Linear
Constant             1.7404        0.0552
Tage                 0.3122        0.0724
```

The maximum-likelihood estimate of the Box-Cox transformation parameter `Bc` is very nearly zero, so the relationship between `Length` and `Age` is approximately linear on a log-scale.

The output from `FIT` when the `CALCULATION` option is set is the same as would be expected from the `FITNONLINEAR` directive. In particular, the residuals are no longer completely standardized: they are scaled only by the residual mean square and not also by the leverage of each point; the leverages are not available. Standard errors of parameters are produced only for the nonlinear parameters by default. However, as in `FITNONLINEAR` you can set the option `SELINEAR=yes` to produce all the standard errors, though this can involve a lot of computation if there are many linear parameters.

Example 3.5.8b shows how to fit a probit model with estimation of control mortality. Parallel and non-parallel probit lines with natural mortality and immunity can be fitted automatically by the `PROBITANALYSIS` procedure, which uses either this method or `FITNONLINEAR` internally (3.5.9.). The example is presented for illustrative purposes, and to assist those who may want to fit more complicated models.

The data consist of counts at four concentrations of one derris root and five of another, plus a control count when no root was present: we have arbitrarily decided to represent this as the first root with log-concentration −100: clearly it is not possible to supply an exact value for zero on the logarithmic scale.

The `MODEL` directive has options and parameters to let you define your own link function and distribution (3.5.6). So we define an expression Lc[1], as in Section 3.5.6, to store the calculation of the fitted values from the linear predictor (the inverse of the link function) in this model. For the probit model with no control mortality, the inverse link is

$$\text{fitted value} = n \times \Phi(\text{linear predictor})$$

where n is the number of binomial trials and $\Phi()$ is the probit function (the cumulative Normal distribution function). With control mortality, the inverse link is

$$\text{fitted value} = n \times (c + (1 - c) \times \Phi(\text{linear predictor}))$$

where c is the control mortality expressed as a proportion. The expressions `Lc[2]` and `Lc[3]` define the deviance from the linear predictor for this model, being careful to avoid taking the exponent of too large a number during the search for the best value of c.

Example 3.5.8b

```
  2  READ [PRINT=data] Root,Logconc,Nspray,Ndead

  3     1     2.17     142    142
  4     1     2.00     127    126
  5     1     1.68     128    115
  6     1     1.08     126     58
  7     2     1.79     125    125
  8     2     1.66     117    115
  9     2     1.49     127    114
 10     2     1.17      51     40
 11     2     0.57     132     37
 12     1  -100.00     129     21 :
 13  " In the control, 129 insects were sprayed with the
-14    medium used in the spray, but with no derris; 21 died
-15    therefore have about 17% control mortality."
 16  SCALAR Cm; VALUE=0.17
 17  EXPRESSION Lc[1...3]; VALUE=\
 18    !e(Fv = Nspray*(Cm+(1-Cm)*NORMAL(Lp))),\
 19    !e(Lp = Lp+(6-Lp)*(Lp>6)-(6+Lp)*(Lp<-6)),\
 20    !e(Dv = SQRT(2*C('pi'))*EXP(Lp**2/2)/Nspray/(1-Cm))
```

```
  21  " Calculate initial linear predictor."
  22  CALCULATE Lp = NED((Ndead+0.5)/(Nspray+1))
  23  MODEL [DISTRIBUTION=binomial; LINK=calculated; LCALC=Lc[]] Ndead;\
  24    NBINOMIAL=Nspray; FITTED=Fv; LINEAR=Lp; DERIVATIVE=Dv
  25  RCYCLE Cm
  26  " Set up dummy expression: no work done, but need to set CALC in FIT."
  27  EXPRESSION Fc; VALUE=!e(Cm=Cm)
  28  TERMS Logconc,Root
  29  FIT [PRINT=#,monitoring; CALC=Fc; SELINEAR=yes] Logconc,Root

Convergence monitoring
----------------------

Cycle Eval Move    Function value    Current parameters
    0    1    0         7.4240110     0.170000
                           Steps   0.00850000
                           Steps   0.00212500
    1    3    1         7.4236352     0.169573

Scoring cycle  Deviance    Current parameters
1             7.4288405      2.80515     0.672494
2             7.4288376      2.80545     0.672484

Convergence in scoring loop at cycle 2.

Convergence in Gauss-Newton loop at cycle 1.
    2    8    6     7.4235721  0.169326
                        Steps  0.00212500  -0.0410132  0.0279739  0.00671897
    1   18    0     7.4235721  0.169326    -4.10132      2.79739  0.671897
                        Steps  0.00319181   0.0432144  0.0219911  0.0147921
    1   28    0     7.4235721  0.169326    -4.10132    2.79739    0.671897

Nonlinear regression analysis
=============================

 Response variate: Ndead
  Binomial totals: Nspray
     Distribution: Binomial
    Link function: Calculated from: Lc[1], Lc[2], Lc[3]
Nonlinear parameters: Cm
  Model calculations: Fc
     Fitted terms: Constant, Logconc, Root


Summary of analysis
-------------------

                                        mean  deviance
Source        d.f.     deviance     deviance     ratio
Regression       3      670.014      223.338    223.34
Residual         6        7.424        1.237
Total            9      677.438       75.271

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
Cm                  0.1693       0.0319
* Linear
Constant            -4.101        0.432
Logconc              2.797        0.220
Root                 0.672        0.148

* MESSAGE: s.e.s are based on dispersion parameter with value 1
```

We can fit the model with fixed control mortality with a simple statement

```
FIT Logconc,Root
```

just as in Section 3.5.6. But we are now able to estimate the control mortality by supplying the name of the control mortality parameter in an RCYCLE statement and setting the CALCULATION option of FIT. As mentioned above, we must set this option to make FIT carry out a nonlinear search, even though all the calculations required have already been specified in the MODEL statement. We therefore supply a dummy calculation that has no effect.

The output includes the monitoring trace to show that a nested iteration is taking place. At each step of the search for the nonlinear parameter Cm, FIT is fitting a generalized linear model with the current value of that parameter, which itself requires an iterative search using the scoring algorithm. The monitoring output shows the progress of the inner loop only once in each iteration of the nonlinear (Gauss-Newton) algorithm rather than at each function evaluation. Convergence is very fast here, because the initial value is very close to the solution.

The results show that the maximum-likelihood estimate of control mortality is very little different from the estimate made from the single control observation.

Example 3.5.8c shows how smoothing can also now be incorporated in these models, providing what could be described as *generalized nonlinear additive models*. The data come from an experiment carried out over several years to determine the effect of growing wheat on plots with different lengths of time previously under grass leys, and with different applications of fertilizer nitrogen. There were four plots with each combination of six lengths of ley and six levels of nitrogen. We fit a model estimating a smooth effect of nitrogen and additive effects of the length of ley. But part of the effect of previous grass is to supply nitrogen to a subsequent crop, so we allow for an additive effect of length of ley in addition to the supplied fertilizer. The model can thus be represented as follows:

$$\text{yield}_{ij} = \text{base-yield}_i + \texttt{SSPLINE}(\text{applied-fertilizer}_j + \text{ley-fertilizer}_i), \quad i = 1...6,\ j = 1...6$$

The base-yields can be estimated as linear effects of the factor Ley, and the applied-fertilizer effects are taken just as the quantitative amounts of fertilizer applied. But the effective amounts of fertilizer supplied by the ley treatments must be estimated as nonlinear parameters.

Example 3.5.8c

```
  2  "Effect of Ley-age and N on Yield of wheat."
  3  OPEN    '%GENDIR%/Examples/GuidePart2/Ley.dat'; CHANNEL=2
  4  FACTOR  Ley
  5  READ    [CHANNEL=2] N,Ley,Yield

   Identifier   Minimum      Mean   Maximum    Values   Missing
            N    0.0000     125.0     250.0       144         0
        Yield     3.878     8.482     9.977       144         0    Skew

   Identifier    Values   Missing    Levels
          Ley       144         0         6

  6  CLOSE   2
  7  MODEL   Yield
  8  RCYCLE  Leyfert[2...6]; INITIAL=0; STEP=1
  9  VARIATE [VALUES=6(0)] Vleyfert
 10  EXPRESSION shift[1,2]; VALUE=\
 11          !e(Vleyfert$[2...6] = Leyfert[2...6]),\
 12          !e(Shiftn = N + NEWLEVELS(Ley; Vleyfert))
 13  FIT     [PRINT=#,monitoring; CALC=shift[]] Ley+S(Shiftn; 4)

Convergence monitoring
----------------------

Cycle Eval Move    Function value    Current parameters
    0    1    0         71.939878           0.          0.          0.
           0.          0.
```

```
                          Steps      1.00000      1.00000      1.00000
     1.00000      1.00000
                          Steps      1.10068      1.07437     0.919776
    0.970563     0.947282
    1    7    0         54.317743      18.0864      29.4214      32.3202
     41.0462      58.0221

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1            0.0011480128      2.9982           3      2.9982      0.0093
  2          0.000011943149      2.9982           3      2.9982      0.0093

Convergence in back-fitting loop at cycle 2.

    2   13    0         46.065667      13.3389      54.5473      74.0608
     79.6635      101.480

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1           0.00088151628      3.0136           3      3.0136      0.0093
  2           7.6143800E-06      3.0136           3      3.0136      0.0093

Convergence in back-fitting loop at cycle 2.

    3   19    0         45.971642      15.2035      56.9218      84.1179
     89.4514      108.242

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1           0.00084229450      3.0124           3      3.0124      0.0093
  2          0.000011341945      3.0124           3      3.0124      0.0093

Convergence in back-fitting loop at cycle 2.

    4   25    0         45.966567      14.8062      55.7173      82.4222
     88.3168      108.323

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1           0.00084788696      3.0132           3      3.0132      0.0093
  2          0.000010045004      3.0132           3      3.0132      0.0093

Convergence in back-fitting loop at cycle 2.

                          Steps     0.657211     0.795189      1.06006
     1.14901      1.57098
    5   34    0         45.966420      14.7557      55.6563      82.3446
     88.2230      108.180

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1           0.00050667002      3.0135           3      3.0135      0.0093
  2           5.6878502E-06      3.0135           3      3.0135      0.0093

Convergence in back-fitting loop at cycle 2.

    6   50    0         45.965910      14.4850      55.3805      82.0774
     87.9301      107.865

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1           0.00050348362      3.0138           3      3.0138      0.0093
  2           5.6479849E-06      3.0138           3      3.0138      0.0093

Convergence in back-fitting loop at cycle 2.

                          Steps     0.170258     0.200961     0.263173
    0.282592     0.383779
    7   66    1         45.965910      14.4850      55.3805      82.0774
     87.9301      107.865

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1           0.00018410798      3.0140           3      3.0140      0.0093

Convergence in back-fitting loop at cycle 1.

    8   87    0         45.965910      14.4850      55.3805      82.0774
     87.9301      107.865
```

```
  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1           0.00030282531      3.0139           3      3.0139      0.0093

Convergence in back-fitting loop at cycle 1.

    9  108    0         45.965910      14.4850      55.3805      82.0774
      87.9301      107.865

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1           0.00045411336      3.0138           3      3.0138      0.0093

Convergence in back-fitting loop at cycle 1.


Convergence in Gauss-Newton loop at cycle 9.
   10  121    6         45.967266      14.6494      55.4477      82.0417
      87.8542      107.481

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1          0.000010396825      3.0142           3      3.0142      0.0093

Convergence in back-fitting loop at cycle 1.

                           Steps      1.32810      1.63895      2.18147
      2.36244      3.23981

  Back-fit cycle  Criterion Smooth d.f. Target d.f. Achv.d d.f.      Param.
  1            0.0010258291      3.0136           3      3.0136      0.0093
  2          0.000012284975      3.0136           3      3.0136      0.0093

Convergence in back-fitting loop at cycle 2.

    1  133    0         45.966217      14.6494      55.4477      82.0417
      87.8542      107.481

Nonlinear regression analysis
=============================

 Response variate: Yield
Nonlinear parameters: Leyfert[2], Leyfert[3], Leyfert[4], Leyfert[5],
                      Leyfert[6]
  Model calculations: shift[1], shift[2]
     Fitted terms: Constant + Ley + Shiftn
        Submodels: SSPLINE(Shiftn; 4)


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression      11       129.94      11.8126     33.92
Residual       132        45.97       0.3482
Total          143       175.91       1.2301

Percentage variance accounted for 71.7
Standard error of observations is estimated to be 0.590.


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
Leyfert[2]            14.6         13.8
Leyfert[3]            55.4         16.6
Leyfert[4]            82.0         22.0
Leyfert[5]            87.9         23.8
Leyfert[6]           107.5         32.5
* Linear
Constant             6.604
Ley 2               0.8328
Ley 3               0.9856
Ley 4               0.9417
Ley 5               0.8920
```

```
Ley 6                0.4301
Shiftn Lin         0.006547
```

Part of the monitoring output is included here to show that the back-fitting algorithm is operating at each step of the nonlinear search, to fit the smoothing spline. The results show that there are very different fertilizer effects of the different ley treatments, which could well be represented as linear effects of the length of time under grass. There is also a substantial non-fertilizer difference between the first treatment and the rest. The fit of this model is shown in Figure 3.5.8c.

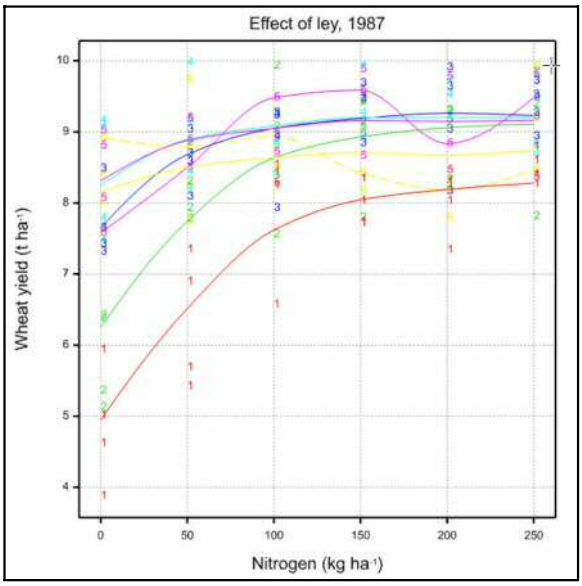


Figure 3.5.8c

A restriction on this analysis is that it is not possible to estimate standard errors of linear parameters when a smoothing spline is included in the model.

There are now three types of iterative estimation in the regression directives: the Fisher-scoring algorithm for fitting generalized linear models, the back-fitting algorithm for additive models, and the alternative algorithms for nonlinear optimization (Gauss-Newton, Newton-Raphson, and Fletcher-Powell). These three types can all be in operation together in a generalized nonlinear additive model. Therefore the `MAXCYCLE` option of the `RCYCLE` directive has been modified to allow a maximum to be set for the number of iterations of each algorithm separately. The setting `MAXCYCLE=50` would, as before, set a limit of 50 iterations for each algorithm. The setting `MAXCYCLE=10,10,50` would set 10 for Fisher-scoring and back-fitting, and 50 for the nonlinear algorithms.

3.5.9 Probit analysis

`PROBITANALYSIS` procedure

Fits probit models allowing for natural mortality and immunity (R.W. Payne).

Options

`PRINT` = *string tokens*	Printed output required (`model`, `summary`, `estimates`, `correlations`, `fittedvalues`, `monitoring`, `effectivedoses`); default `mode`, `summ`, `esti`, `fitt`, `effe`
`TRANSFORMATION` = *string token*	Transformation to be used (`probit`, `logit`, `complementaryloglog`); default `prob`
`MORTALITY` = *string token*	Whether to estimate natural mortality (`omit`, `estimate`); default `omit`
`IMMUNITY` = *string token*	Whether to estimate natural immunity (`omit`, `estimate`); default `omit`
`GROUPS` = *factor*	Defines groups for an analysis of parallelism; default `*` i.e. no groups
`SEPARATE` = *string tokens*	Which parameters (apart from intercept) should be estimated separately for different groups (`slope`,

	`mortality`, `immunity`, `notintercept`); default `*` i.e. none
`LD` = *scalar* or *variate*	Effective, or lethal, doses to be estimated, other than 50
`CIPROBABILITY` = *scalar*	Probability level for the confidence interval of effective doses; default 0.95, i.e. a 95% confidence interval
`LOGBASE` = *string token*	Base of antilog transformation to be applied to LD's (`ten`, `e`); default `*` i.e. none
`DISPERSION` = *scalar*	Controls the use of a heterogeneity factor in the calculation of s.e.s etc; with the default of 1 no factor is used, a missing value `*` estimates the heterogeneity from the residual deviance
`FITMETHOD` = *string token*	Method to use to fit the model (`generalizednonlinear`, `nonlinear`) default `nonl` for Wadley's problem, otherwise `gene`
`MAXCYCLE` = *scalar*	Maximum number of iterations for fitting the model; default 30

Parameters

`Y` = *variates*	Number of subjects responding in each batch
`DOSE` = *variates*	Dose received by each batch of subjects
`NBINOMIAL` = *variates*, *scalars* or *factors*	Variate specifying the number of subjects in each batch, or factor specifying groupings of the observations assumed to have equal expected total numbers of subjects in Wadley's problem; if omitted, assumes Wadleys's problem with all observations having the same expected total number of subjects
`INITIAL` = *variates*	Initial values for parameters
`STEPLENGTHS` = *variates*	Step lengths for parameters
`LDESTIMATES` = *variates*	Saves estimates of the effective, or lethal, doses
`LDLOWER` = *variates*	Saves lower values of the confidence intervals for the estimates of the effective, or lethal, doses (for `FITMETHOD=gene` only)
`LDUPPER` = *variates*	Saves upper values of the confidence interval values for the estimates of the effective, or lethal, doses (for `FITMETHOD=gene` only)

The `PROBITANALYSIS` procedure provides customized facilities for probit analysis. The data consist of observations, in each of which a particular dose of one a drug was applied to a group of subjects, and the number that responded was counted. The `Y` parameter specifies a variate indicating the number of subjects that responded in each batch, the `DOSE` parameter specifies a variate to show the dose given to each batch, and the `NBINOMIAL` parameter defines the total numbers of subjects in each batch.

The `NBINOMIAL` parameter can be omitted if the total numbers cannot be measured, as in some fumigation experiments ("Wadley's problem"; see for example Finney 1971, pages 202-8). The assumption is that the total numbers receiving the doses will come from the same Poisson distribution, and the mean of this distribution is then estimated in the analysis. Alternatively, `NBINOMIAL` can specify a factor to indicate groupings of the doses whose total numbers are expected to come from the same distributions.

The `PRINT` option controls printed output with settings:

`model`	details of the model that has been fitted;

summary	summary analysis-of-variance table;
estimates	parameter estimates and standard errors;
correlations	correlations between parameter estimates;
fittedvalues	fitted values and residuals;
monitoring	information about the fitting process; and
effectivedoses	effective, or lethal, doses (see parameter LD below).

By default, PRINT=mode,summ,esti,fitt,effe.

The TRANSFORMATION option allows other transformations other than the probit to be selected. Putting TRANSFORMATION=logit requests a logit transformation:

$$\text{logit}(P\%) = \log(P\% / (100 - P\%))$$

This is very like the probit but approaches zero (to the left) and one (to the right) rather more slowly. The other possibility is the complementary log-log ($=\log(-\log(100-P\%))$, which is relevant to the "one-hit" model (that is infection processes where just one infected particle is sufficient to cause the response).

Sometimes, subjects may respond even in the absence of any dose. For example, with some short-lived insects, some would have died simply from natural causes during the period of the experiment. By setting option MORTALITY=estimate this natural mortality can be included in the model and estimated. Similarly, there may be subjects that will not respond, no matter how high the dose. Setting option IMMUNITY=estimate will include and estimate a parameter for natural immunity.

It is also often of interest to fit study the way in which the model varies for different groups of subjects. For example, there may be groups of batches of subjects, each of which is given a different drug. The GROUPS option should then specify the group to which each batch of subjects belongs, and option SEPARATE indicates which parameters of the model (slope, mortality, and/or immunity) should have separate estimates. Separate parameters are always fitted for the intercept unless you include the setting notintercept. So, if SEPARATE is left at its default value, parallel lines will be fitted with identical values for any estimates of mortality and immunity.

The LD option can request the estimation of one or more effective (or lethal) doses, specifying a scalar if there is just one, or a variate if there are several. The LOGBASE option is useful if the doses have been transformed to logarithms before calling PROBITANALYSIS. If you use LOGBASE to specify the base of the logarithms (ten or e), the back-transformed lethal doses will be printed as well.

The estimates of the effective (or lethal) doses can be saved, in a variate, by the LDESTIMATES parameter. Also, when model is fitted as a generalized nonlinear model (see the FITMETHOD option, below), the lower and upper values of the confidence intervals for the estimates can be saved by the LDLOWER and LDUPPER parameters, respectively. If LOGBASE is set, these are all back-transformed. The CIPROBABILITY option specifies the probability level for the confidence intervals; the default is 0.95, i.e. 95% confidence intervals.

The DISPERSION option can be used to request use of a heterogeneity factor in the calculation of the standard errors of the slopes and lethal doses (see Finney 1971, pages 70-74). The standard assumptions for probit analysis are that the observations have binomial distributions in probit lines and planes, or Poisson distributions in Wadley's problem. Under these circumstances, the residual deviance will follow a Chi-square distribution. The residual deviance should on average be equal to its number of degrees of freedom. A significantly large value may indicate that there are other (possibly unknown) factors affecting the subjects, for example that the conditions were not uniform during the experiment. Alternatively it may occur because the subjects did not react independently, for example because there were sub-populations of genetically related individuals. If the large Chi-square seems to arise because the residuals are larger in general than expected (overdispersion) and not because of systematic deviations from the fitted relationship, it is sensible to increase the standard errors by a heterogeneity factor equal to the residual mean deviance. This can be requested by setting option DISPERSION=*.

Alternatively DISPERSION can be set to a known value if one is available.

When the FITMETHOD option is set to generalizednonlinear, the model is fitted as a generalized nonlinear model, using the FIT directive (3.5.8). The alternative setting, nonlinear, fits it as a nonlinear model using FITNONLINEAR (3.8). Apart from minor numerical differences, the two methods should generate the same results. Generalized nonlinear models allow a confidence region to be generated for lethal doses, and these are used as default for all situations except Wadley's problem. The nonlinear method is more accurate, and is thus used as the default for the more difficult situation presented by Wadley's problem. However, there is the limitation that you cannot use the notintercept setting of the SEPARATE option with the nonlinear method.

The final two parameters, INITIAL and STEPLENGTHS, allow initial values and step lengths to be specified for the optimization. For a generalized nonlinear model, the order of parameters is: total(s) for Wadley's problem (if appropriate), mortality parameters (if any) and immunity parameters (if any); the slopes and intercepts are fitted as regression parameters. For a nonlinear model, the order of parameters is: LD50(s), slope(s), mortality parameters (if any) and immunity parameters (if any); the totals for Wadley's problem, if required, as fitted as linear parameters. The MAXCYCLE option sets a limit on the number of iteractions used during fitting (default 30). Parameter estimates, fitted values, residuals, and so on, can be saved after running the procedure, by using the RKEEP directive in the usual way.

Example 3.5.9 uses PROBITANALYSIS to analyse the data in Example 3.5.8b, this time though fitting different slope and natural mortality parameters for each type of root. Notice that we need to redefine Root as a factor, and set the control dose to missing instead of the value −100 in Example 3.5.8b.

Example 3.5.9

```
 30  GROUPS          [REDEFINE=yes] Root
 31  CALCULATE       Logconc = MVINSERT(Logconc; Logconc==-100)
 32  PROBITANALYSIS [TRANSFORMATION=probit; MORTALITY=estimate; GROUPS=Root;\
 33                  SEPARATE=mortality,slope; LD=!(50,90)]\
 34                  Ndead; DOSE=Logconc; NBINOMIAL=Nspray

Nonlinear regression analysis
=============================

 Response variate: Ndead
  Binomial totals: Nspray
     Distribution: Binomial
    Link function: Calculated from: Lc[1], Lc[2], CalcControlDoseLinpred,
                   CalcProbitFitted, CalcProbitDerivative
Nonlinear parameters: PrMortality['1'], PrMortality['2']
  Model calculations: !E(...)
     Fitted terms: Root + X['Logconc'].Root


Summary of analysis
-------------------

                                        mean  deviance
Source        d.f.     deviance     deviance     ratio
Regression       5      450.998       90.200     90.20
Residual         4        7.170        1.793
Total            9      458.168       50.908

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.
```

```
Estimates of parameters
-----------------------

Parameter           estimate         s.e.
PrMortality['1']      0.1649       0.0327
PrMortality['2']       0.225        0.101
* Linear
Root 1                -3.467        0.444
Root 2                 -3.18         1.13
X['Logconc'].Root 1    2.827        0.295
X['Logconc'].Root 2    3.046        0.726

* MESSAGE: s.e.s are based on dispersion parameter with value 1


Fitted values and residuals
---------------------------

             Binomial                    Standardized
         Unit    total Response Fitted value residual
            1      142      142       141.55     0.95
            2      127      126       125.48     0.45
            3      128      115       117.35    -0.73
            4      126       58        56.53     0.26
            5      125      125       123.88     1.50
            6      117      115       114.26     0.48
            7      127      114       118.43    -1.46
            8       51       40        37.15     0.92
            9      132       37        37.30    -0.06
           10      129       21        21.28    -0.07

Mean                                     0.22


Effective doses
---------------

       Group          LD    estimate        s.e.   lower 95%   upper 95%
           1       50.00       1.226     0.04812       1.123       1.309
           1       90.00       1.679     0.04744       1.597       1.780
           2       50.00       1.044     0.16509       0.582       1.221
           2       90.00       1.464     0.06287       1.328       1.552
```

3.5.10 Generalized linear mixed models

Generalized linear mixed models can be fitted by the GLMM procedure. You can then display further output with the GLDISPLAY procedure, plot residuals with the GLPLOT procedure, form predictions with the GLPREDICT procedure, save results with the GLKEEP procedure, and perform permutation tests with the GLPERMTEST procedure. These are all described in this section. There is also the GLTOBITPOISSON procedure, which uses the Tobit method to fit a Poisson-log generalized linear mixed model with censored Poisson data. Details are in the *Genstat Reference Manual, Part 3 Procedures*.

GLMM procedure

Fits a generalized linear mixed model (S.J. Welham).

Options

PRINT = *string token*	What output to display (model, components, effects, fittedvalues, means, backmeans, monitoring, vcovariance, waldtests, missingvalues, covariancemodels, deviance); default mode, comp, effe, mean, back, moni, vcov, cova
DISTRIBUTION = *string token*	Error distribution (binomial, poisson, normal,

	gamma, negativebinomial); default bino
LINK = *string token*	Link function (identity, logarithm, logit, reciprocal, probit, complementaryloglog, logratio); default * gives the canonical link
DISPERSION = *scalar*	Value at which to fix the residual variance, if missing the variance is estimated; default 1 for binomial, Poisson and negative binomial distributions, a missing value otherwise
RANDOM = *formula*	Random model *excluding* bottom stratum; this must be set
FIXED = *formula*	Fixed model; default *
ABSORB = *factor*	Absorbing factor to be used at the REML step of the iterations
CONSTANT = string token	Whether to estimate or omit constant term in fixed model (omit, estimate); default esti
FACTORIAL = *scalar*	Limit on number of factors/covariates in a model term; default 3
PTERMS = *formula*	Formula specifying fixed terms for which means or back-transformed means are to be printed; default * prints all the fixed model terms
PSE = *string token*	Standard errors to print with tables of means (se, sesummary, sed, sedsummary, vcovariance, differences, estimates, alldifferences, allestimates); default seds
MVINCLUDE = *string tokens*	Whether to include units with missing values in the explanatory factors and variates and/or the y-variates (explanatory, yvariate); default * i.e. omit units with missing values in either explanatory factors or variates or y-variates
MAXCYCLE = *scalar*	Maximum number of iterations of the GLMM algorithm; default 20
TOLERANCE = *scalar*	Convergence criterion for iterative procedure; default 0.0001
FMETHODGLMM = string token	Specifies fitting method (all, fixed): all indicates the method of Schall (1991); fixed indicates the marginal method of Breslow & Clayton (1993) ; default all
OFFSET = *variate*	Variate holding values to be used as an offset on the linear predictor scale; default *
CADJUST = *string token*	What adjustment to make to covariates for the REML analysis (mean, none); default mean
AGGREGATION = *scalar*	Fixed parameter for negative binomial distribution (parameter k as in variance function var = mean + mean^2/k); default 1
KLOGRATIO = *scalar*	Parameter k for logratio link, in form log(mean / (mean + k)); default as set in AGGREGATION option
OWNDIST = *text*	For non-standard distributions only: text specifying the variance function to be used with dummy variable DUM, e.g. OWNDIST='DUM'
OWNLINK = *text*	For non-standard link functions only: text specifying 3 functions using dummy variable DUM - the link function,

	its inverse and its derivative, e.g. `OWNLINK = !T('log(DUM)','exp(DUM)','1/DUM')`
`CDEFINITIONS` = *text*	Statements to execute to define correlation models; default * i.e. none
`CVECTORS` = *pointer*	Data structures involved in the correlation models
`WORKSPACE` = *scalar*	Number of blocks of internal memory to be set up for use by the `REML` algorithm; default 1
`VCONSTRAINTS` = *string token*	Whether to constrain variance components to be positive (`none`, `positive`); default `posi`
`VMETHOD` = *string token*	Indicates whether to use the standard Fisher-scoring algorithm or the new AI algorithm with sparse matrix methods (`Fisher`, `AI`); default `AI`
`VMAXCYCLE` = *scalar*	Limit on the number of iterations; default 30

Parameters

`Y` = *variates*	Dependent variates
`NBINOMIAL` = *scalars* or *variates*	Number of binomial trials for each unit (must be set if `DISTRIBUTION=binomial`)
`FITTEDVALUES` = *variates*	Variates to save fitted values
`COMPONENTS` = *variates*	Variate to save estimated variance components
`VCOVARIANCE` = *symmetric matrices*	Variance-covariance matrix for the variance components
`MEANS` = *pointers*	Pointer to save tables of means for each `Y` variate
`VARMEANS` = *pointers*	Pointer to save covariance matrices of tables of means for each `Y` variate
`BACKMEANS` = *pointers*	Pointer to save tables of back-transformed means for each `Y` variate
`ITERATIVEWEIGHTS` = *variates*	Saves the iterative weights from the generalized linear model fitting
`INITIALFITTEDVALUES` = *variates*	Defines initial values for the fitted values; if unset, these are formed automatically
`EXIT` = *scalar*	Exit status for the fit of the `GLMM` (0 if successful)
`SAVE` = *REML save structures*	Saves details of the `REML` analysis used to fit the model
`GLSAVE` = *pointer*	Saves details of the `GLMM` analysis

Procedure `GLMM` estimates the parameters of a generalized linear mixed model using either the method of Schall (1991) or the marginal method of Breslow & Clayton (1993). It assumes a generalized linear mixed model, that is a generalized linear model with both fixed and Normally-distributed random effects on the scale of the linear predictor. It estimates the fixed effects together with the variance components associated with the random effects.

The `DISTRIBUTION` option sets the error distribution; the default is to assume a binomial distribution but the Poisson, gamma and negative-binomial distributions are also available. Other distributions can be used via the `OWNDIST` option; this should be set to a text containing the formula for calculating the variance function for the required distribution, in terms of dummy variable `DUM`. The link can be set using the `LINK` option; the default takes the canonical link. Identity, logarithm, logit, reciprocal, probit, complementaryloglog or logratio link functions are also provided, and alternative link functions can be used via the `OWNLINK` option. In this case, `OWNLINK` must be set to a text with three values containing formulae (in terms of dummy variable `DUM`) for calculating the link function, its inverse and its first derivative. For example, instead of specifying a Poisson distribution with log link, the `OWNDIST` and `OWNLINK` options

could be set as

```
OWNDIST='DUM'; OWNLINK=!T(LOG(DUM),EXP(DUM),'1/DUM')
```

Where necessary, these expressions should be constructed so that invalid results (eg. divide by zero or log(zero)) are avoided.

The AGGREGATION option supplies the aggregation parameter for the negative-binomial distribution; default 1. The KLOGRATIO option supplies the parameter *k* to be used in the logratio link, and takes its default from AGGREGATION.

The DISPERSION option specifies the dispersion parameter. The default is 1 for binomial, Poisson and negative binomial distributions, a missing value otherwise (indicating that the dispersion parameter is to be estimated).

The fixed and random models are specified by the FIXED and RANDOM options. The number of factors in the terms of the fixed model can be limited using the FACTORIAL option. By default the variance components are constrained to be positive, but you can set option VCONSTRAINTS to none to allow them to become negative.

The VMETHOD option specifies the algorithm to use in the REML steps of the GLMM algorithm: either Fisher or AI(default). The ABSORB option can specify an absorbing factor for use with the Fisher algorithm. However, if the absorbing factor appears in any of the terms of the FIXED model, no estimates of error will be available for these terms (5.3.3, 5.3.7). The VMAXCYCLE option controls the number of iterations used by the REML algorithm.

By default, a constant term is included in the model; this can be suppressed by setting option CONSTANT=omit. An offset can be included in the linear predictor by setting option OFFSET. By default any covariates are centred for the REML fitting by subtracting their means, weighted according to the iterative weights of the generalized linear model. Alternatively you can set option CADJUST=none to request that the uncentred covariates are used instead. You can save the iterative weights using the ITERATIVEWEIGHTS parameter.

It is also possible to define correlation models on the random terms, although the results should be used with caution as their properties are not yet well understood. To do this, you should set the CDEFINITIONS option to a text containing the Genstat statements required to define the models (e.g. using VSTRUCTURE). You also need to set the CVECTORS option to a pointer containing the data structures involved in the statements. Then, in the statements themselves, you should refer to each of these as CVECTORS[n], where n is the position of the relevant data structure in the pointer. For example:

```
TEXT cdef; VALUE=\
'VSTRUCTURE [CVECTORS[1].CVECTORS[2]] ar,ar;
FACTOR=CVECTORS[1,2]; ORDER=1'
GLMM  [DISTRIBUTION=gamma; LINK=log; FIXED=variety;\
      RANDOM=fieldrow*fieldcolumn; CDEFINITION=cdef;\
      CVECTORS=!p(fieldrow,fieldcolumn)] yield
```

The MVINCLUDE option allows the inclusion of units with missing values, as in the REML directive. By default, units where there is a missing value in the y-variate or in any of the factors or variates in the model terms are excluded. The setting explanatory allows units with missing values in factors or variates in the model to be included. For missing covariate values, this is equivalent to substituting the mean value. The setting yvariate includes units with missing values in the y-variate. This can be useful to retain the balanced structure of the data for use with direct product covariance matrices (see VSTRUCTURE, 5.4.1), or to produce predictions of data values for given values of explanatory factors and/or variates.

The FMETHODGLMM option specifies the method used to form the fitted values and therefore determines the fitting method to be used. The default setting all specifies that both fixed and random terms should be used to form fitted values which gives the method of Schall (1991); setting fixed indicates that only fixed terms are used to form fitted values which gives the marginal method of Breslow & Clayton (1993). For details see the description of GLMM in Part

3 of the *Genstat Reference Manual*.

The PRINT option selects the output to be displayed:

model	description of model fitted,
components	estimates of variance components and estimated parameters of covariance models,
effects	estimates of parameters α and β, the fixed and random effects,
fittedvalues	table containing the y-variate, fitted values, residuals on the natural scale and standardized residuals on the scale of the linear predictor,
means	predicted means for factor combinations,
backmeans	back-transformed means,
monitoring	monitoring information at each iteration,
vcovariance	variance-covariance matrix of the estimated components,
waldtests	Wald tests for fixed terms,
missingvalue	estimates of missing values,
covariancemodels	estimated covariance models, and
deviance	deviance from the generalized linear model.

The default is PRINT=mode, comp, effe, mean, back, moni, vcov, cova.

The deviance represents the variation remaining after fitting the fixed terms and all the random terms. It thus assesses how well those terms explain the random variation in the data.

To avoid problems with 0 and 100% observations, the standardized residuals on the linear-predictor scale are calculated as differences between the adjusted dependent variate and the fitted values on that scale (and then standardized by their standard errors). The fitted values include the random as well as the fixed terms. The GLDISPLAY procedure can print residuals and fitted values where the fitted values are calculated only from the fixed terms.

The PTERMS option can specify which tables of means are printed; by default, tables of means are produced for all the terms in the fixed model.

The PSE option controls the standard errors that are printed with tables of means and effects:

se	standard errors,
sesummary	summary of the standard errors (default),
sed	standard errors of differences between pairs of means,
sedsummary	summary of the standard errors of differences,
vcovariance	variance-covariance matrix for the means,
allestimates	synonym of se,
estimates	synonym of sesummary,
alldifferences	synonym of sed,
differences	synonym of sedsummary.

Some control over the iterative GLMM algorithm is provided by option MAXCYCLE which sets the maximum number of iterations (default 20), and by option TOLERANCE which specifies the criterion for determining convergence of the algorithm (default 0.0001). Convergence is judged to have been attained once the maximum change in the ratio (variance component)/(residual variance) and the change in the residual variance are less than the specified TOLERANCE.

The dependent variate is specified using the Y parameter. The NBINOMIAL parameter must be set when DISTRIBUTION=binomial to specify the total number of trials on each unit, as a variate if the number varies from unit to unit or as a scalar if it is constant over all the units.

The other parameters are used to save results. The variance components and residual variance can be saved in a variate using parameter VCOMPONENTS, with their variance-covariance matrix stored in a symmetric matrix specified by parameter VCOVARIANCE. The tables of means to be saved are determined by the setting of PTERMS. The tables are stored in a pointer specified by parameter MEANS, in the order in which they appear in the FIXED model. Their variance matrices

and tables of back-transformed means are stored similarly in pointers specified by parameters VARMEANS and BACKMEANS. The EXIT parameter saves a scalar indicating the exit status for the fit of the GLMM (0 if successful, 1 otherwise).

You can display further output from the analysis using the GLDISPLAY procedure, and use the GLKEEP procedure to save information in Genstat data structures. The GLPREDICT procedure can form predictions. By default these procedures take the most recent GLMM analysis, but you can use the GLSAVE to save the results of the analysis, to use instead in future calls of these procedures.

Alternatively, VDISPLAY and VKEEP can be used to redisplay or store other results from the internal REML estimation, provided REML has not been used in the interim. You can use the SAVE parameter to save the REML save structure, and use that as input to these directives, if REML may be used for another analysis.

GLMM is illustrated in Example 3.5.10a.

Example 3.5.10a

```
  2  " Example of how to use GLMM procedures:',\
 -3    data from McCullagh & Nelder (1989, Table 14.4);',\
 -4    also see Schall (1991)."
  5  FACTOR  [NVALUES=120; LEVELS=20] Female, Male
  6  &       [LEVELS=4; LABELS=!t(RR,RW,WR,WW)] Cross
  7  VARIATE [NVALUES=120] Mate1
  8  READ    Cross,Male,Female; FREPRESENTATION=labels,2(levels)

   Identifier    Values   Missing    Levels
        Cross       120         0         4
         Male       120         0        20
       Female       120         0        20

 24  READ    Mate1

   Identifier   Minimum      Mean   Maximum    Values   Missing
        Mate1    0.0000    0.5833     1.000       120         0

 29  GLMM       [PRINT=model,components,wald; DISTRIBUTION=binomial; LINK=logit;\
 30             FIXED=Cross; RANDOM=Female+Male] Mate1; NBINOMIAL=1

Generalized linear mixed model analysis
=======================================

Method:            c.f. Schall (1991) Biometrika
Response variate:  Mate1
Binomial totals:   1
Distribution:      binomial
Link function:     logit
Random model:      Female + Male
Fixed model:       Constant + Cross
Dispersion parameter fixed at value 1.000


Estimated variance components
-----------------------------

Random term               component        s.e.
Female                        1.410       0.838
Male                          0.089       0.389
```

```
Residual variance model
-----------------------

Term                          Model(order)  Parameter        Estimate      s.e.
Dispersn                      Identity      Sigma2              1.000     fixed


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Cross                              15.71       3          5.06    39.2   0.005

Dropping individual terms from full fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Cross                              15.71       3          5.06    39.2   0.005

* MESSAGE: denominator degrees of freedom for approximate F-tests are calculated
using numerical derivatives ignoring fixed/boundary/singular variance parameters.
```

You can print additional output with the GLDISPLAY procedure.

GLDISPLAY procedure

Displays further output from a GLMM analysis (R.W. Payne).

Options

PRINT = *string token*	What output to display (model, components, effects, fittedvalues, means, backmeans, vcovariance, waldtests, missingvalues, covariancemodels, deviance); default *
PTERMS = *formula*	Formula specifying fixed terms for which means or back-transformed means are to be printed; default * prints all the fixed model terms
PSE = *string token*	Standard errors to print with tables of means (se, sesummary, sed, sedsummary, vcovariance, differences, estimates, alldifferences, allestimates); default seds
OFFSET = *scalar*	Offset value to use when calculating predicted means; default 0
RMETHOD = *string token*	Which random terms to use when calculating RESIDUALS (final, all); default fina
CFORMAT = *string token*	Whether printed output for covariance models gives the variance matrices or the parameters (variancematrices, parameters); default vari
FMETHOD = *string token*	Controls whether and how to calculate F-statistics for fixed terms (automatic, none, algebraic, numerical); default auto
GLSAVE = *pointer*	Save structure from the GLMM analysis

No parameters

GLDISPLAY allows you to display further output from a GLMM analysis. The possibilities are similar to those within GLMM. However, GLDISPLAY provides more choices for deviances, Wald

tests, fitted values and the output of covariance models.

By default the output is from the most recent GLMM analysis. Alternatively, you can set the GLSAVE parameter to a save structure (saved using the GLSAVE parameter of GLMM) to obtain output from an earlier analysis.

The PRINT option operates in the same way as the PRINT option of GLMM, except that there is no monitoring setting. The default is PRINT= mode, comp, effe, mean, back, moni, vcov, cova. The PTERMS and PSE option are the same as in GLMM.

The RMETHOD option controls the way in which residuals and fitted values are formed. With the default setting RMETHOD=final, the fitted values are calculated from all the fixed and random effects. The setting RMETHOD=all can be used to obtain fitted values constructed from the fixed terms alone, omitting all random terms. (The residuals are then calculated as the differences between the values of the y-variate and the fitted values.) To avoid problems with 0 and 100% observations, the standardized residuals on the linear-predictor scale are calculated as differences between the adjusted dependent variate and the fitted values on that scale (and then standardized by their standard errors).

The OFFSET option specifies the offset value to use when calculating predicted means. The default is zero.

The CFORMAT option controls the type of output produced for the estimated covariance models. The default setting, variancematrices, produces the variance-covariance matrices for the components, whereas the setting parameters prints their parameters.

The FMETHOD option controls whether to accompany the Wald tests for fixed effects with approximate F statistics and corresponding numbers of residual degrees of freedom. The computations, using the method devised by Kenward & Roger (1997), can be time consuming with large or complicated models. So, with the default setting FMETHOD=automatic, Genstat assesses the model itself and decides automatically whether to do the computations and which method to use. The other settings allow you to control what to do yourself:

none	no F statistics are produced;
algebraic	F statistics are calculated using algebraic derivatives (which may involve large matrix calculations);
numerical	F statistics are calculated using numerical derivatives (which require an extra evaluation of the mixed model equations for every variance parameter).

GLDUSPLAY is used in Example 3.5.10b to print the predicted and back-transformed means. GLPLOT is then used to produce the residual plot shown in Figure 3.5.10, below.

Example 3.5.10b

```
  31  GLDISPLAY  [PRINT=means,backmeans; PSE=sed]

Generalized linear mixed model analysis
=======================================


Tables of means and back-transformed means
==========================================


Table of means and back-transformed means for Cross
---------------------------------------------------

                Means Backtransform
        Cross
           RR   1.163        0.7619
           RW   0.784        0.6865
           WR  -1.412        0.1959
           WW   1.015        0.7340
```

```
Standard errors of differences

     Cross RR   1           *
     Cross RW   2      0.6268           *
     Cross WR   3      0.8457      0.8398           *
     Cross WW   4      0.8367      0.8092      0.6793           *
                             1           2           3           4

  32  GLPLOT     [RMETHOD=final]
```

GLPLOT procedure

Plots residuals from a GLMM analysis (R.W. Payne).

Options

RMETHOD = *string token*	Which random terms to use when calculating the residuals (final, all); default all
BACKTRANSFORM = *string token*	Whether to plot residuals on the natural scale (calculated using back-transformed fitted values) or standardized residuals on the linear-predictor scale (link, none); default none
INDEX = *variate* or *factor*	X-variable for an index plot; default !(1,2...)
OFFSET = *scalar*	Value of offset to use when calculating the residuals; default 0
GRAPHICS = *string token*	What type of graphics to use (lineprinter, highresolution); default high
TITLE = *text*	Overall title for the plots; the default is to form a title displaying the identifier of the y-variate and the type of residual
GLSAVE = *pointer*	Save structure from the GLMM analysis; default * uses the GLSAVE structure from the most recent GLMM analysis

Parameters

METHOD = *string tokens*	Type of residual plot (fittedvalues, normal, halfnormal, histogram, absresidual, index); default fitt, norm, half, hist
PEN = *scalars*, *variates* or *factors*	Pen(s) to use for each plot

GLPLOT provides up to four types of residual plots from a GLMM analysis. These are selected using the METHOD parameter, with settings: fitted for residuals versus fitted values, normal for a Normal plot, halfnormal for a half-Normal plot, histogram for a histogram of residuals, absresidual for a plot of the absolute values of the residuals versus the fitted values, and index for a plot against an "index" variable (specified by the INDEX option). The PEN parameter can specify the graphics pen or pens to use for each plot.

The residuals and fitted values are accessed automatically from the analysis specified by the GLSAVE option. If the GLSAVE option has not been set, they are taken from the most recent GLMM analysis.

The RMETHOD option controls which random terms are used to calculate the residuals:

all	all the random effects (default), and
final	only the final random term,

Note that residuals based on the final random term will not be calculated when any of the variance components are negative, as the associated negative correlations can generate very

misleading patterns. GLPLOT will then generate a warning that all the residuals are missing. You should then use RMETHOD=all instead.

In Figure 3.5.10 the residuals are based on the final random term. With binary data, you may often find some large standardized residuals, as the response variate can take only one of two possible values. The large residuals here are from units 27 and 29, where the response was one but the fitted values are close to zero. (You can see the unit numbers by clicking on the Data into button in the Graphics viewer.)

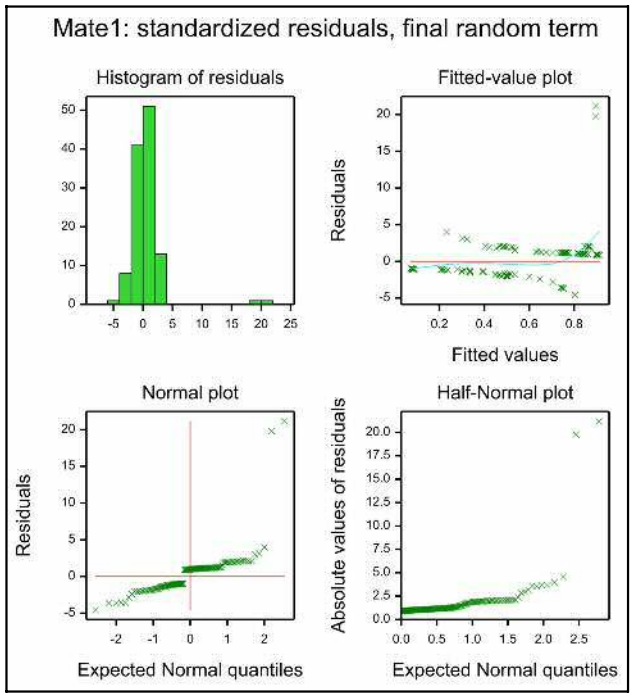


Figure 3.5.10

The BACKTRANSFORM option specifies the scale of the residuals. The default is to plot standardized residuals on the linear-predictor scale. To avoid problems with 0 and 100% observations, these are formed as the difference between the adjusted dependent variate and the fitted values on the linear predictor scale (and then standardized). Alternatively, you can set BACKTRANSFORM=link to plot (unstandardized) residuals on the natural scale.

The OFFSET option specifies the offset value to use when calculating the residuals. The default is zero.

By default, high-resolution graphics are used. Line-printer graphics can be used instead, by setting option GRAPHICS=lineprinter.

The TITLE option can supply an overall title. If this is not set, a default title is formed displaying the identifier of the y-variate and the type of residual.

You can produce predictions with the GLPREDICT procedure.

GLPREDICT procedure

Forms predictions from a GLMM analysis (R.W. Payne).

Options

PRINT = *string tokens*	What to print (description, predictions, backpredictions, se, sesummary, sed, sedsummary, vcovariance); default desc, pred, back, seds
MODEL = *formula*	Indicates which model terms (fixed and/or random) are to be used in forming the predictions; default * includes all the fixed terms and relevant random terms
OMITTERMS = *formula*	Specifies terms to be excluded from the MODEL; default * i.e. none
FACTORIAL = *scalar*	Limit on the number of factors or variates in each term in the models specified by MODEL or OMITTERMS; default 3
PRESENTCOMBINATIONS = *factors*	Lists factors for which averages should be taken across combinations that are present

WEIGHTS = *tables*	One-way tables of weights classified by factors in the model; default *
OFFSET = *scalar*	Value of offset on which to base predictions; default 0
NBINOMIAL = *scalar*	Supplies the total number of trials to be used for prediction with a binomial distribution (providing a value *n* greater than one allows predictions to be made of the number of "successes" out of *n*, whereas the value one predicts the proportion of successes); default 1
PREDICTIONS = *table* or *scalar*	To save the predictions; default *
BACKPREDICTIONS = *table* or *scalar*	To save back-transformed predictions; default *
SE = *table* or *scalar*	To save standard errors of predictions; default *
SED = *symmetric matrix*	To save standard errors of differences between predictions; default *
VCOVARIANCE = *symmetric matrix*	To save variances and covariances of predictions; default *
GLSAVE = *pointer*	Save structure from the GLMM analysis; default * uses the SAVE structure from the most recent GLMM analysis

Parameters

CLASSIFY = *vectors*	Variates and/or factors to classify table of predictions
LEVELS = *variates*, *scalars* or *texts*	To specify values of variates and/or levels of factors for which predictions are calculated
PARALLEL = *identifiers*	For each vector in the CLASSIFY list, allows you to specify another vector in the CLASSIFY list with which the values of this vector should change in parallel (you then obtain just one dimension in the table of predictions for these vectors)
NEWFACTOR = *identifiers*	Identifiers for new factors that are defined when LEVELS are specified

GLPREDICT can be used after the GLMM directive to produce predictions of the values of the response variate at particular values of the variables in the fixed or random models. By default the predictions are from the most recent GLMM analysis, but you can use another analysis by supplying its save structure using the GLSAVE option.

The parameters are the same as those of VPREDICT (5.5.1), which GLPREDICT uses to form the predictions. The CLASSIFY parameter specifies variates or factors that are to be included in the table of predictions, and the LEVELS parameter supplies the values at which the predictions are to be made. For a factor, you can select some or all of the levels, while for a variate you can specify any set of values. A single level or value is represented by a scalar; several levels or values must be combined into a variate (which may of course be unnamed). Alternatively, if the factor has labels, you can use these to select the levels for prediction by setting LEVELS to a text. A missing value in the LEVELS parameter is taken to stand for all the levels of a factor, or the mean value of a variate.

The PARALLEL parameter allows you to indicate that a factor or variate should change in parallel with another factor or variate. Both of these should have the same number of values specified for it by the LEVELS parameter of GLPREDICT. The predictions are then formed for each set of corresponding values rather than for every combination of these values.

When you specify LEVELS, a new factor must be defined to classify that dimension of the table. By default this will be an unnamed factor, but you can use the NEWFACTOR parameter to give it an identifier. The EXTRA attribute of the factor is set to the name of the corresponding

factor or variate in the CLASSIFY list; this will then be used to label that dimension of the table of predictions.

The prediction calculations consist of two steps. The first step is to calculate a table of fitted values. The MODEL, OMITTERMS and FACTORIAL options specify the model to use for this. The formula specified by MODEL is expanded into a list of model terms, deleting any that contain more variates of factors than the limit specified by the FACTORIAL option. Then, any terms in the formula specified by OMITTERMS are removed.

The second step averages the fitted values over the classifications that are not in the list that was supplied by the CLASSIFY parameter. The WEIGHTS option can supply one-way tables classified by any of the factors in the model. These are used to calculate the weight to be used for each fitted value when calculating the averages. Equal weights are assumed for any factor for which no table of weights has been supplied. (Note, this differs from the default in PREDICT, which uses *marginal weights*; see the PREDICT option ADJUSTMENT for details.) In the averaging all the fitted values are generally used. However, if you define a list of factors using the PRESENTCOMBINATIONS option, any combination of levels of these factors that does not occur in the data will be omitted from the averaging. Where a prediction is found to be inestimable, i.e. not invariant to the model parameterization, a missing value is given.

The OFFSET option specifies the offset value to use when calculating predicted means. The default is zero.

The NBINOMIAL parameter can be used to supply the total number of trials to be used for back-transformed predictions with a binomial distribution. If you provide a value *n* greater than one, GLPREDICT predicts the number of "successes" out of *n*. The default, NBINOMIAL=1, predicts the proportion of successes.

Printed output is controlled by settings of the PRINT option with settings:

description	describes the terms and standardization policies used when forming the predictions,
predictions	prints the predictions,
backpredictions	prints back-transformed predictions,
se	prints standard errors of the predictions,
sesummary	prints the minimum, average and maximum standard error,
sed	prints standard errors of differences between the predictions,
sedsummary	prints the minimum, average and maximum standard error of difference,
vcovariance	prints the variance and covariances of the predictions.

The default is to print descriptions, predictions, back-transformed predictions, and a summary of the standard error of differences. Standard errors amd standard errors of differences are prnted only if the predictions themselves are printed.

You can also save the results, using the PREDICTIONS, BACKPREDICTIONS, SE, SED and VCOVARIANCE options.

In Example 3.5.10c, GLPREDICT is used to print and save predicted and back-transformed means with standard errors. Notice that, with the default options, the predictions are the same as those printed by GLDISPLAY in Example 3.5.10b. At the end of the example, GLKEEP is used to save the deviance from the full likelihood together with the degrees of freedom in the fixed model.

Example 3.5.10c

```
 33  GLPREDICT  [PRINT=predictions,backpredictions,sedsummary;\
 34             PREDICTIONS=Predictions; SED=sed] Cross

             Predictions Backtransform
       Cross
          RR       1.163        0.7619
          RW       0.784        0.6865
          WR      -1.412        0.1959
          WW       1.015        0.7340


Minimum standard error of difference       0.6268
Average standard error of difference       0.7729
Maximum standard error of difference       0.8457

 35  PRINT      Predictions & sed

             Predictions
       Cross
          RR       1.163
          RW       0.784
          WR      -1.412
          WW       1.015



                     sed

    Cross RR           *
    Cross RW      0.6268           *
    Cross WR      0.8457      0.8398           *
    Cross WW      0.8367      0.8092      0.6793           *
                 Cross RR    Cross RW    Cross WR    Cross WW
```

GLKEEP procedure

Saves results from a GLMM analysis (R.W. Payne).

Options

FACTORIAL = *scalar*	Limit on number of factors in the model terms generated from the TERMS parameter; default 3
RESIDUALS = *variate*	Residuals from the analysis
FITTEDVALUES = *variate*	Fitted values from the analysis
DISPERSION = *scalar*	Dispersion component
VCOVARIANCE = *symmetric matrix*	Variance-covariance matrix for the estimates of the variance components
VESTIMATES = *variate*	Saves a vector of all parameters in the variance model
VARESTIMATES = *symmetric matrix*	Variance-covariance matrix for the parameters in the variance model (as saved by VESTIMATES)
VLABELS = *text*	Vector of text labels for the VESTIMATES and VARESTIMATES structures
MVESTIMATES = *variate*	Estimates of missing values
MVSE = *variate*	Standard errors of missing-value estimates
MVUNITS = *variate*	Unit numbers of missing values
DEVIANCE = *scalar*	Saves the deviance
MODEL = *pointer*	Information defining the model
RMETHOD = *string token*	Which random terms to use when calculating

	RESIDUALS (final, all); default fina
DFFIXED = *scalar*	Number of degrees of freedom in the fixed model
DFRANDOM = *scalar*	Number of degrees of freedom in the random model
FMETHOD = *string token*	Controls how to calculate F-statistics for fixed terms (automatic, none, algebraic, numerical); default auto
WMETHOD = *string token*	Controls which Wald statistics are saved (add, drop); default drop
OFFSET = *scalar*	Offset value to use when calculating predicted means; default 0
ITERATIVEWEIGHTS = *variate*	Saves the iterative weights from the generalized linear model fitting
LINEARPREDICTOR = *variate*	Linear predictor from a generalized linear model
YADJUSTED = *variate*	Adjusted response variate
ZADJUSTED = *variate*	Adjusted dependent variate on the linear predictor scale
LPRESIDUALS = *variate*	Residuals from the fit on the linear predictor scale
SELPRESIDUALS = *variate*	Standard errors for the residuals from the fit on the linear predictor scale
EXIT = *scalar*	Exit status of the fit (0 if successful)
GLSAVE = *pointer*	Save structure from the GLMM analysis

Parameters

TERMS = *formula*	Model terms for which information is required
COMPONENTS = *scalar* or *pointer to scalars*	Estimated variance components
MEANS = *table* or *pointer* to *tables*	Predicted means for each term
BACKMEANS = *table* or *pointer* to *tables*	Back-transformed means
SEDMEANS = *symmetric matrix* or *pointer* to *symmetric matrices*	Standard errors of differences between means
VARMEANS = *symmetric matrix* or *pointer* to *symmetric matrices*	Variance-covariance matrix for the means
EFFECTS = *table* or *pointer* to *tables*	Effects for each term
SEDEFFECTS = *symmetric matrix* or *pointer* to *symmetric matrices*	Standard errors of differences between effects
VAREFFECTS = *symmetric matrix* or *pointer* to *symmetric matrices*	Variance-covariance matrix for the effects
CADJUSTMENT = *scalar* or *pointer to scalars*	For a term involving covariates, saves the adjustment made to its values during the analysis
WALD = *scalar* or *pointer to scalars*	Wald statistic (fixed terms only)
FSTATISTIC = *scalar* or *pointer to scalars*	F statistics (fixed terms only)
NDF = *scalar* or *pointer to scalars*	Numerator d.f. (fixed terms only)
DDF = *scalar* or *pointer to scalars*	Denominator d.f. (fixed terms only)

GLKEEP saves results from a GLMM analysis. By default the results are from the most recent GLMM analysis. Alternatively, you can set the GLSAVE parameter to a save structure (saved using the GLSAVE parameter of GLMM) to save results from an earlier analysis.

The RESIDUALS and FITTEDVALUES options can specify variates to save the residuals and

fitted values, respectively. The RMETHOD option controls the way in which residuals and fitted values are formed. With the default setting RMETHOD=final, the fitted values are calculated from all the fixed and random effects. The setting RMETHOD=all can be used to obtain fitted values constructed from the fixed terms alone, omitting all random terms. (The residuals are then calculated as the differences between the values of the y-variate and the fitted values.)

The DISPERSION option saves the dispersion coefficient, in a scalar.

The variance-covariance matrix for the estimates of the variance component can be saved using the VCOVARIANCE option. (The estimates themselves are saved using the COMPONENTS parameter, as described below.)

The VESTIMATES option saves a variate containing all the variance parameters estimated in the model. The VARESTIMATES option can supply a symmetric matrix to save the variance-covariance matrix for the estimates of the variance parameters, matching the ordering and contents of VESTIMATES. The vector of labels for these parameters can be saved by the VLABELS option..

The MVESTIMATES option saves a variate containing estimates of the missing values, the MVSE option saves their standard errors, and the MVUNITS option saves a list of the units that are missing.

The DEVIANCE option saves the deviance from the generalized model. This represents the variation remaining after fitting the fixed terms and all the random terms. It thus assesses how well those terms explain the random variation in the data.

The degrees of freedom fitted by the fixed model can be saved by the DFFIXED option, and the degrees of freedom in the random model can be saved by the DFRANDOM option.

The MODEL option can be used to save a pointer, with labels 'distribution', 'link', 'aggregation', 'klogratio', 'owndist', 'ownlink', 'random', 'fixed', 'constant', 'factorial', 'offset', 'cdefinitions', 'cvectors', 'y', and 'nbinomial', storing the settings of the corresponding options and parameters of GLMM. The labels can be specified in either lower or upper case, or any mixture.

The ITERATIVEWEIGHTS parameter saves the iterative weights used in the last cycle of the iteration, and the LINEARPREDICTOR parameter saves the linear predictor. The YADJUSTED parameter saves the adjusted response variate used in the last cycle of the iteration, and the ZADJUSTED parameter similarly saves the adjusted response variate on the scale of the linear predictor. The LPRESIDUALS option saves the residuals from the fit on the linear predictor scale. To avoid problems with 0 and 100% observations, they are calculated as differences between the adjusted dependent variate and the fitted values on that scale. The SELPRESIDUALS option saves their standard errors. The EXIT option saves a scalar indicating the exit status for the fit of the GLMM (0 if successful, 1 otherwise).

The parameters of GLKEEP save information about particular model terms in the analysis. With the TERMS parameter you specify a model formula, which Genstat expands to form the series of model terms about which you wish to save information. The FACTORIAL option sets a limit on the number of factors in each term. Any term containing more than that limit is deleted. The subsequent parameters allow you to specify identifiers of data structures to store various components of information for each of the terms that you have specified.

The MEANS parameter saves tables of predicted means, and the BACKMEANS parameter saves back-transformed means. The OFFSET option specifies the offset value to use when calculating predicted means; the default is zero. The SEDMEANS parameter saves symmetric matrices of standard errors of differences for the means, and the VARMEANS parameter saves symmetric matrices of their vriances and covariances. The EFFECTS parameter saves tables of effects, and the SEDEFFECTS and VAREFFECTS parameter saves symmetric matrices with standard errors for their differences and their variances and covariances, respectively.

If a term involves a covariate, the CADJUSTMENT parameter can save the adjustment that will have been made to its values during the analysis. This will be zero if option CADJUST was set

to none in GLMM. Alternatively, if CADJUST had its default setting of mean, each covariate will have been centred by subtracting its (weighted) mean.

The Wald statistic for fixed terms can be saved in scalars using the WALD parameter. The WMETHOD option controls whether these are from the table where terms are added sequentially to the model, or that where terms are dropped from the full fixed model. The associated F statistic, and its numerator and denominator numbers of degrees of freedom, can be saved in scalars by the FSTATISTIC, NDF and DDF parameters, respectively. The FMETHOD option specifies which algorithm to use to calculate the denominator numbers of degrees of freedom. The default, automatic, will use any stored values that have been calculated for this analysis by earlier GLMM, GLDISPLAY or GLKEEP statements; otherwise it will choose automatically between the two available methods. (See REML for more details.)

If you have a single term, you can supply a table, symmetric matrix or scalar for each of these parameters, as appropriate. However, if you have several terms, you must supply a pointer which will then be set up to contain as many tables, symmetric matrices or scalars as there are terms.

Example 3.5.10d continues from Example 3.5.10c, using GLKEEP to save the Wald statistic and degrees of freedom for Cross.

Example 3.5.10d

```
36  GLKEEP     Cross; WALD=chisq; NDF=df
37  CALCULATE  pr = CUCHISQUARE(chisq; df)
38  PRINT      chisq,df,pr; DECIMALS=*,0,*

     chisq          df          pr
     15.71           3    0.001298
```

The probability for the Wald chi-square is smaller than the probability for the F statistic printed in the Example 3.5.10a, illustrating the biases that can occur when this has to be used instead of the F probability. Nevertheless the evidence is strong enough to be confident of the conclusion. For less clear-cut situations, you can use the GLPERMTEST procedure to do a permutation test.

GLPERMTEST procedure

Does random permutation tests for generalized linear mixed models (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (prwald, criticalwald, ownstatistics, monitoring); default prwa, crit
NTIMES = *scalar*	Number of permutations to make; default 99
NRETRIES = *scalar*	Maximum number of extra samples to take when some analyses fail to converge; default NTIMES
BLOCKSTRUCTURE = *formula*	Model formula defining any blocking to consider during the randomization; default none
EXCLUDE = *factors*	Factors in the block formula whose levels are not to be randomized
SEED = *scalar*	Seed for the random number generator used to make the permutations; default 0 continues from the previous generation or (if none) initializes the seed automatically
BINMETHOD = *string token*	How to permute binomial data (individuals, units; default indi
WMETHOD = *string token*	Controls which Wald statistics are used (add, drop); default add

OWNMETHOD = *string token*	Type of test required for own statistics (twosided, greaterthan, lessthan); default twos
CIPROBABILITY = *scalar*	Probability level for the confidence interval for own statistics; default 0.95

Parameters

GLSAVE = *pointers*	Save structure of the original analysis from GLMM; default * uses the save structure from the most recent GLMM analysis
WALD = *pointers*	Saves a pointer with a variate for each of the fixed terms containing the Wald statistics from the permuted data sets
PRWALD = *pointers*	Saves a pointer with a scalar for each of the fixed terms, containing the test probability obtained from the position of its Wald statistic within those from the permuted data sets
CRITICALWALD = *pointers*	Saves a pointer with variates for the 5%, 1% and 0.1% significance levels containing the corresponding critical values for the fixed terms, obtained from the quantiles of the Wald statistics from the permuted data sets
NNOTCONVERGED = *scalars*	Saves the number of permuted data sets whose analyses failed to converge
OWNDATA = *pointers*	Data required to calculate own statistics
OWNOBSERVEDVALUES = *variates*	Saves observed values of the own statistics
OWNPROBABILITIES = *variates*	Saves bootstrap probabilities for the own statistics
OWNESTIMATES = *variates*	Saves boostrap estimates for the own statistics
OWNSES = *variates*	Saves boostrap standard errors for the own statistics
OWNLOWERCIS = *variates*	Saves boostrap lower values of the confidence intervals for the own statistics
OWNUPPERCIS = *variates*	Saves boostrap upper values of the confidence intervals for the own statistics
OWNSTATISTICS = *pointers*	Saves the own statistics obtained from the permuted data sets, in a pointer with a variate for each statistic

GLPERMTEST performs random permutation tests for fixed terms in a generalized linear mixed model, analysed by GLMM. A problem with these analyses is that their estimates of the variance components are generally biased i.e. the estimates are smaller than the true values. The Wald tests also suffer from bias, in that their test probabilities may be too small. You therefore need to be cautious when the probabilities from the tests are close to their critical values, especially when analysing small data sets or data from a binary distribution.

GLPERMTEST uses random permutation tests to provide an alternative way of assessing the fixed terms. It forms random permutations of the response, analyses those data sets, and records their Wald statistics. The distributions of the Wald statistics, under the null hypothesis of no fixed effects, can be estimated by the sets of statistics obtained from the analyses of the permuted data sets. Test probabilities for the original Wald statistics can therefore be estimated by their locations within those sets.

Before using GLPERMTEST, you need to analyse the original data set by GLMM. The GLSAVE parameter supplies the save structure from that analysis. If this is not specified, GLPERMTEST uses the save structure from the most recent GLMM analysis. The save structure provides the settings of all the options and parameters that GLMM used in that analysis. The analyses of the permuted data sets can therefore be done in exactly the same way as the original analysis.

The NTIMES option defines how many random permutations to perform; by default there are 99. The NRETRIES option specifies the maximum number of extra samples to take when some analyses fail to converge; the default is to use the same number as specified by NTIMES. The NNOTCONVERGED parameter can save a scalar containing the number of permuted data sets whose analyses failed to converge. The results may be unreliable if more than a few analyses fail.

The SEED option allows you to specify the seed to use for the random-number generator that is used for the randomizations to form the permutations. The default, SEED=0, continues the sequence of random numbers from a previous generation or, if this is the first use of the generator in this run of Genstat, it initializes the seed automatically. If NTIMES exceeds the maximum possible number of permutations for the data, an "exact" test is performed in which every permutation is used once. This is feasible only for small data sets. There are *n*! (n factorial) permutations of *n* units: 3!=6, 4!=24, 5!=120, 6!=720, 7!=5040, 8!=40320, and so on.

If the data are from a designed experiment, you may need to use the BLOCKSTRUCTURE option to specify a block model to define how to do the randomization. The EXCLUDE option can then restrict the randomization so that one or more of the factors in the block model is not randomized. See the RANDOMIZE directive for further details.

The BINMETHOD option controls how the permutations are done for binomial data. The original data set will have contained a set of units, each recording a number of "successes" obtained from an observed number of individuals. The default, and recommended, method is to expand the data set to contain individuals themselves, and permute these. Alternatively, you can set BINMETHOD=units if you prefer to permute the units as a whole instead.

The WALD parameter can save a pointer with a variate for each of the fixed terms containing the Wald statistics from the analyses of the permuted data sets. Similarly the PRWALD parameter can save a pointer with a scalar for each of the fixed terms, containing the test probability obtained from the position of its Wald statistic within those from the permuted data sets.

You can define your own statistics to be assessed by the test. They are calculated by a procedure _GLPERMownstatistics, which is called by GLPERMTEST following the GLMM analysis of each permuted data set. Its use is shown in the GLPERMTEST example, which can be modified to calculate your own statistics instead. The information required by _GLPERMownstatistics to do the calculations is supplied, in a pointer, by the OWNDATA parameter. The OWNMETHOD option specifies the type of test to be made. The default, twosided tests whether the statistics differ from zero. The greaterthan setting tests whether they are greater than zero, and the lessthan setting tests whether they are less than zero. Permutation estimates, standard errors and confidence intervals are also calculated, The CIPROBABILITY option specifies the probability for the confidence intervals (default 0.95). The OWNOBSERVEDVALUES parameter can save a variate containing the values of the own statistics from the original data set. The OWNPROBABILITIES can save a variate containing the probabilities from the tests. The OWNESTIMATES can save a variate containing the bootstrap estimates of the statistics (calculated as the mean of the values obtained from the bootstrap samples) The OWNSES can save a variate containing standard errors of bootstrap estimates. The OWNLOWERCIS and OWNUPPERCIS parameters can save variates containing the lower and upper values, respectively, of the confidence intervals. Finally, the OWNSTATISTICS can save the values of the own statistics obtained from the permuted data sets, in a pointer with a variate for each statistic.

Output is controlled by the PRINT option, with settings:

prwald	to print probabilities for the fixed terms, estimated from the locations of their Wald statistics within the sets obtained from the permuted data sets;
criticalwald	to print critical values for the Wald statistics, estimated by quantiles within the sets from the permuted data sets;

`ownstatistics`	to print estimates, standard errors and confidence intervals for the own statistics, and
`monitoring`	to monitor the progress of the anayses.

The default is to print probabilities and critical values.

Example 3.5.10e does a permutation test, printing the probability and critical values. The probability of 0.01 is asa small as we can get with the default number 99 of permutation. So this confirms that there genuinely are differences between the crosses.

Example 3.5.10e

```
  39  GLPERMTEST [SEED=77157]

Probabilities for Wald statistics
---------------------------------

       Source
        Cross       0.010

(determined from 99 random permutations)


Critical values for Wald statistics
-----------------------------------

                        5%          1%        0.1%
       Source
        Cross       6.705      11.110      11.453
```

GLRTEST procedure

Calculates likelihood tests to assess the random terms in a generalized linear mixed model (R.W. Payne).

Options

`PRINT` = *string tokens*	Controls printed output (`tests`); default `test`
`SELECTION` = *string tokens*	Specifies information to print with the tests (`aic`, `sic`, `bic`, `critical`); default `crit`
`CRITICAL` = *variate*	Saves the critical values
`GLSAVE` = *pointer*	Save structure of the original analysis from `GLMM`; default `*` uses the save structure from the most recent `GLMM` analysis

Parameters

`TERMS` = *formula*	Random terms to be tested; default is to test them all
`TESTSTATISTIC` = *scalar* or *pointer* to *scalars*	Test statistics for each term
`DF` = *scalar* or *pointer* to *scalars*	Degrees of freedom of the test statistics
`AIC` = *scalar* or *pointer* to *scalars*	Akaike information coefficients for each term
`SIC` = *scalar* or *pointer* to *scalars*	Schwarz (Bayesian) information coefficients for each term

`GLRTEST` can be used after a `GLMM` analysis to assess the effect of dropping random terms from the model. It uses the `REML` deviances to do this. In the `GLMM` algorithm, `REML` is used to analyse the adjusted dependent variate *z*, with the variate of iterative weights, defined by the generalized linear model. These depend on the current fitted values, and change at each iteration until convergence. The `REML` deviance is taken from the analysis of the final adjusted z-variate with

the final iterative weights.

GLRTEST saves the deviance from the original analysis using VKEEP, and the final adjusted z-variate and variate of iterative weights using GLKEEP. It then does REML analyses with these variates, omitting each random term, saving their deviances, and calculating their differences from the original deviance. Akaike and Schwarz (Bayesian) information coefficients are obtained using the VAIC procedure.

Note that, for compatibility, it is important to use the same adjusted z-variate and the same iterative weights as in the original analysis. With the alternative, of doing GLMM analyses removing each random term, we would be taking deviances from REML analyses with their own adjusted z-variates and weights, which could be very different from those in the original analysis. So we would be comparing REML analyses with different models, different response variates and different weights, which would not provide a valid comparison. Of course this does mean that the results pertain to the REML analysis rather than to the GLMM analysis itself. So they should be used as guidance rather than as a definitive test. Often, however, the random terms will have been defined by the design of the investigation. The tests will then be used more as an indication of the effectiveness of the design than to decide whether to omit terms from the analysis.

By default, GLRTEST produces tests for every random term. However, you can use the TERMS parameter to request tests for a specific set of terms.

The is to print the tests are printed, but you can set option PRINT=* to suppress this. The additional information to be printed with the tests is controlled by the SELECTION option, with settings:

aic	Akaike information coefficients;
sic	Schwarz (Bayesian) information coefficients;
bic	synonym for sic, and
critical	critical values (default).

If the variance components are unconstrained, the critical values are from a chi-square distribution with one degree of freedom. Alternatively, if they are constrained to be positive, the asymptotic distribution of test is a 50:50 mixture of chi-square distributions with zero and one degree of freedom. Essentially this means that the critical values are from a chi-square distribution with one degree of freedom but at double the probability level. See, for example, Lee, Nelder & Pawitan 2006, Section 6.5. The CRITICAL option can save three critical values, in a variate with units for probabilities of 0.05, 0.001 and 0.001.

The TESTSTATISTIC parameter can save the statistics. the DF parameter can save their numbers of degrees of freedom. (These will always be equal to one, but the parameter is included for compatibility with the HGFTEST and HGRTEST procedures.) The AIC and SIC parameters can save the Akaike and Schwarz (Bayesian) information coefficients, respectively. If you are making a test for a single term, you can supply a scalar for each of these parameters. However, if you have several terms, you must supply a pointer which will then be set up to contain as many scalars as there are terms.

Example 3.5.10f assesses the random terms in the analysis of the salamander data. It seems clear that there are differences between the random effects of the female parents, but very little difference in the male random effects.

Example 3.5.10f

```
  40  GLRTEST
Initial random model: Female + Male
Deviance 546.13

Dropping terms from the initial model
-------------------------------------

  Term    Deviance  d.f.
            change
Female       6.112     1
  Male       0.054     1


Critical values (variance components constrained to be positive)
         0.05       2.706
         0.01       5.412
        0.001       9.550
```

3.5.11 Hierarchical generalized linear models

This section describes 11 procedures with the prefix HG, which provide tools for fitting the hierarchical and double hierarchical generalized linear models (HGLMs and DHGLMs) defined by Lee & Nelder (1996, 2001a, 2006) and explained in the book by Lee, Nelder & Pawitan (2006). Procedures HGFIXEDMODEL and HGRANDOMMODEL define the fixed and random models for an HGLM, and HGDRANDOMMODEL can extend it to become a DHGLM. You can also include nonlinear terms in the fixed model, using the HGNONLINEAR procedure. HGANALYSE does the analysis, HGDISPLAY displays the results, HGWALD produces Wald tests for the fixed terms, HGFTEST and HGRTEST calculate likelihood tests for fixed and random terms, HGPREDICT forms tables of predictions, HGPLOT produces model-checking plots, HGGRAPH displays the fitted model, and HGKEEP can save the results. These are all described in this section. There is also the HGTOBITPOISSON procedure, which uses the Tobit method to fit a Poisson-log hierarchical generalized linear model with censored Poisson data. Details are in the *Genstat Reference Manual, Part 3 Procedures*.

HGLMs extend the ordinary generalized linear models (GLMs) to include additional random terms in the linear predictor. They contain generalized linear mixed models (GLMMs) as a special case, but do not constrain the additional terms to follow a Normal distribution and to have an identity link (as in the GLMM). For example, if the basic generalized linear model is a log-linear model (Poisson distribution and log link), a more appropriate assumption for the additional random terms might be a gamma distribution and a log link.

The analysis involves fitting an augmented generalized linear model, known as the *augmented mean model*, to describe the mean of the distribution. This has units corresponding to the original data units, together with additional units for the effects of the random terms; see Lee & Nelder (1996). Then there are further GLMs, with gamma distributions and usually with logarithmic links, to model the dispersion for each random term (including the residual dispersion parameter φ); see Lee & Nelder (2001a). In a DHGLM, some of these dispersion GLMs are themselves extended to become HGLMs by the inclusion of random terms; see Lee & Nelder (2006).

Procedure HGFIXEDMODEL specifies the fixed model terms in the HGLM, and defines the link function and the distribution of the basic GLM.

HGFIXEDMODEL procedure

Defines the fixed model for a hierarchical or double hierarchical generalized linear model (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

DISTRIBUTION = *string token*	Distribution of the data (binomial, poisson, normal, gamma); default norm
LINK = *string token*	Link for the fixed model (identity, logarithm, logit, reciprocal, probit, complementaryloglog); default iden
DISPERSION = *scalar*	Value of dispersion parameter in calculation of s.e.s etc; default * for DIST=norm or gamm, and 1 for DIST=pois or bino
DLINK = *string token*	Link for the dispersion model (logarithm, reciprocal); default loga
DTERMS = *formula*	Dispersion model; default * i.e. none
CONSTANT = *string token*	How to treat the constant (estimate, omit) default esti
FACTORIAL = *scalar*	Limit on number of variates and/or factors in a fixed model term; default 3
WEIGHTS = *variate*	Prior weights; default * i.e. 1
OFFSET = *variate*	Offset variate; default * i.e. none
DOFFSET = *variate*	Offset variate for dispersion model; default * i.e. none
DDISPERSION = *scalar*	Dispersion parameter to use in a dispersion model for the residual dispersion parameter phi; default 1
IDISPERSION = *scalar*	Initial value for the residual dispersion parameter phi; default * i.e. formed automatically

Parameter

TERMS = *formula*	Fixed model

The LINK and DISTRIBUTION options of HGFIXEDMODEL define the link function and distribution of the basic GLM. The TERMS parameter specifies the fixed model, and the FACTORIAL option sets a limit on the number of variates and/or factors in a fixed term (default 3). The CONSTANT option indicates whether or not to include a constant term or intercept in the fixed model (by default this is included), and the OFFSET option allows an offset variate to be specified. The WEIGHTS option can supply a variate of prior weights, and the DISPERSION option allows you to fix the dispersion parameter φ. The DTERMS option allows you to define a *structured dispersion model* by specifying a fixed model to be fitted in the GLM that estimates the residual dispersion parameter φ (the DISPERSION option is then ignored). The DLINK parameter specifies the link to use with the dispersion model, the DOFFSET option allows you to specify an offset variate, and the DDISPERSION option defines the dispersion parameter for the dispersion GLM (default 1).

The random model is defined by HGRAMDOMMODEL.

HGRANDOMMODEL procedure

Defines the random model for a hierarchical or double hierarchical generalized linear model (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

DISTRIBUTION = *string token*	Distribution for the random model (beta, normal, gamma, inversegamma); default norm
LINK = *string token*	Link for the random model (identity, logarithm, logit, reciprocal); default iden

Parameters

TERMS = *formula*	Random model
DLINK = *string tokens*	Link for the dispersion model for each random term (logarithm, reciprocal); default loga
DFORMULA = *formula structures*	Dispersion model for each random term; default * i.e. none
DOFFSET = *variates*	Offset variate for dispersion model for each random term; default * i.e. none
LMATRIX = *matrices*	Linear transformation to apply to design matrix **Z** of each random term, in order to define correlations between its effects; default * i.e. none
DDISPERSION = *scalar*	Dispersion parameter to use in the dispersion model for each random term; default 1
FDISPERSION = *scalar*	Fixed value for the dispersion parameter of each random term; default !s(*) i.e. dispersion is estimated
IDISPERSION = *scalar*	Initial value for the dispersion parameter for each random term; default * i.e. formed automatically

The TERMS parameter defines the additional random terms in the HGLM. These should not include the final (residual) term, unless you want to define a saturated random model as, for example, in the use of a negative binomial distribution in the Fabric example, discussed in Lee, Nelder & Pawitan 2006, Section 6.6.3. The LINK and DISTRIBUTION options specify their distribution and link function respectively.

The DFORMULA option allows you to define a *structured dispersion model* for any of the random terms, by specifying a fixed model to be fitted in the GLM that estimates its dispersion parameter. The DLINK parameter specifies the link to use with each dispersion model, the DOFFSET parameter allows you to specify an offset variate, and the DDISPERSION parameter defines the dispersion parameter for the dispersion GLM (default 1). Alternatively, if you do not define a dispersion model for a random term, you can use the FDISPERSION parameter to fix its dispersion at a specific value.

The LMATRIX parameter allows correlation structures to be defined for random terms, using the method described by Lee & Nelder (2001b). This is done by setting LMATRIX to a matrix **L** that is used as a premultiplier for the **Z** matrix of the random term concerned. Lee & Nelder (2001b) give examples illustrating the types of model that can be defined.

The IDISPERSION parameter allows you to define initial values for the dispersion parameters of the random terms. An initial value for the residual dispersion parameter phi can be defined using the IDISPERSION option of the HGFIXEDMODEL procedure. If you set both of these, the HGANALYSE procedure will then use them to initialize the weights that are involved in the fitting of the augmented mean model; for details see Chapter 6 of Lee, Nelder & Pawitan (2006). The default weights that are formed automatically if either of these is unset are satisfactory in most

circumstances, but you may want to try your own initial values if you encounter convergemce problems.

HGDRANDOMMODEL allows you to extend a hierarchical generalized linear model (HGLM) to become a double hierarchical generalized linear model.

HGDRANDOMMODEL procedure

Defines the random model in a hierarchical generalized linear model for the dispersion in a double hierarchical generalized linear model (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

DISTRIBUTION = *string token*	Distribution for the random model (beta, normal, gamma, inversegamma); default norm
LINK = *string token*	Link for the random model (identity, logarithm, logit, reciprocal); default iden
RANDOMTERM = *formula*	Random term whose dispersion is being modelled; if unset, the model is assumed to be for the residual dispersion parameter (phi)
PHIMETHOD = *string token*	Whether to fix or estimate the residual dispersion parameter in the dispersion HGLM (fix, estimate); default fix

Parameters

TERMS = *formula*	Random model
DLINK = *string tokens*	Link for the dispersion model for each random term (logarithm, reciprocal); default loga
DFORMULA = *formula structures*	Dispersion model for each random term; default * i.e. none
DOFFSET = *variates*	Offset variate for dispersion model for each random term; default * i.e. none
LMATRIX = *matrices*	Linear transformation to apply to design matrix **Z** of each random term, in order to define correlations between its effects; default * i.e. none
DDISPERSION = *scalar*	Dispersion parameter to use in the dispersion model for each random term; default 1
FDISPERSION = *scalar*	Fixed value for the dispersion parameter of each random term; default !s(*) i.e. dispersion is estimated

---

HGDRANDOMMODEL adds some random terms to one of the generalized linear models that is to model one of the dispersion parameters, so that this becomes an HGLM. By default the residual dispersion of this HGLM is fixed, but you can set option PHIMETHOD=estimate to estimate it. The random term whose dispersion is to be modelled by the HGLM is indicated by the RANDOMTERM option. If RANDOMTERM is omitted, the dispersion model is assumed to be for the residual dispersion parameter (φ) of the original HGLM.

The TERMS parameter defines the additional random terms, and the LINK and DISTRIBUTION options specify their distribution and link function respectively. You can specify a dispersion model for any of these additional random terms using the DFORMULA, DLINK, DOFFSET and DDISPERSION parameters, as in the HGRANDOMMODEL procedure. Also, as in the HGRANDOMMODEL procedure, the LMATRIX parameter allows correlation structures to be defined for the additional random terms, and the FDISPERSION parameter allows you to fix their dispersion parameters.

You can include nonlinear terms in the fixed model with the HGNONLINEAR procedure.

HGNONLINEAR procedure

Defines nonlinear parameters for the fixed model of a hierarchical generalized linear model (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

CALCULATION = *expression structures*	Calculation of explanatory variates involving nonlinear parameters
METHOD = *string token*	Algorithm for fitting the nonlinear model (GaussNewton, NewtonRaphson, FletcherPowell); default Gaus
VECTORS = *variates*	Vectors involved in the calculations (data vectors or factors or derived vectors that appear in the fixed model)

Parameters

PARAMETER = *scalars*	Nonlinear parameters in the model
LOWER = *scalars*	Lower bound for each parameter
UPPER = *scalars*	Upper bound for each parameter
STEPLENGTH = *scalars*	Initial step length for each parameter
INITIAL = *scalars*	Initial value for each parameter
DELTA = *scalars*	Parameter increment to use when calculating numerical derivatives

HGNONLINEAR allows you to extend a conjugate HGLM to become a hierarchical generalized nonlinear model by including nonlinear parameters in the fixed model (Payne 2014). *Conjugate HGLMs* have the following combinations of link and distribution for the mean model:

Fixed terms		Random terms	
Distribution	Link	Distribution	Link
Poisson	logarithm	gamma	logarithm
binomial	logit	beta	logit
gamma	reciprocal	inverse-gamma	reciprocal
Normal	identity	Normal	identity

The nonlinear terms are added exactly as in a generalized nonlinear model (see 3.5.8), by defining some calculations to form variates to include as linear terms in the model. So the nonlinear terms have the form

$$B \times \mathrm{f}(p)$$

where B is a (linear) regression coefficient and f() is a function of some nonlinear parameters e.g.

$$B \times R^X$$

defines an exponential term with nonlinear parameter R. (This can be written as $\exp(k \times X)$ where the parameter $R = \exp(k)$.)

The calculations are specified, as a list of Genstat expression structures, by the CALCULATION option. (This corresponds to the CALCULATION option of the FIT directive.) You must also use the VECTORS option to list the vectors that appear in the calculations (either as data vectors or as derived vectors that then appear as linear terms in the fixed model). The METHOD option indicates which algorithm to use to fit the nonlinear model. (This corresponds to the METHOD

option of the RCYCLE directive.)

The parameters of HGNONLINEAR supply information about the nonlinear parameters. Most of these correspond to parameters in the RCYCLE directive. PARAMETER lists the identifiers of the parameters as they appear in the calculations. LOWER and UPPER can define lower and upper bounds. STEPLENGTH can define the step lengths to use for each parameter at the start of the optimization, and INITIAL can define initial values. Genstat will take default initial values if you do not specify these yourself. However, these may not lead to convergence, so you are strongly advised to specify your own. It is often feasible to fit the models in an ordinary generalized nonlinear model, with the random terms included as fixed terms, and then use those estimates as the initial values for the hierarchical generalized nonlinear model.

The final parameter, DELTA, specifies a small increment to each parameter to be used inside the algorithm when calculating derivatives of the fixed model with respect to each nonlinear parameter (needed to calculate leverages).

You can print the model definitions using the HGSTATUS procedure.

HGSTATUS procedure

Displays the current HGLM model definitions (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Option

SAVE = *pointer*	Save structure (from HGANALYSE) to provide details of the HGLM; if omitted, information is printed for the most recently defined or fitted HGLM

No parameters

By default the model definitions are from the most recently defined or fitted HGLM, but you can use the SAVE option to supply the save structure for some other HGLM.

When you are ready, the model can be fitted by HGANALYSE.

HGANALYSE procedure

Analyses data using a hierarchical or double hierarchical generalized linear model (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

PRINT = *string tokens*	Controls printed output (model, fixedestimates, randomestimates, dispersionestimates, likelihoodstatistics, deviance, waldtests, fittedvalues, monitoring, dhgmonitoring); default mode, fixe, disp, devi, like, moni
LMETHOD = *string token*	Whether to use exact likelihood or extended quasi likelihood to obtain the y-variate and weights for the dispersion model (exact, eql); default exac
SEMETHOD = *string token*	Method to use to calculate the se's for the dispersion estimates (approximate, profilelikelihood); default appr
DMETHOD = *string token*	Method to use for the adjusted profile likelihood when calculating the likelihood statistics (automatic, choleski, lrv); default auto
EMETHOD = *string token*	Extrapolation method to use (aitken, adjustedaitken); default aitk
MLAPLACEORDER = *scalar*	Order of Laplace approximation to use in the estimation

	of the mean model (0 or 1); default 0
DLAPLACEORDER = *scalar*	Order of Laplace approximation to use in the estimation of the dispersion components (0, 1 or 2); default 0
MAXCYCLE = *scalars*	Maximum number of iterations of the hierarchical generalized linear model fits, and maximum number of iterations in the fitting of the mean and dispersion models; default 99,50
EXIT = *scalar*	Exit status (0 for success, 1 for failure to converge)
TOLERANCE = *scalar*	Criterion for convergence; default 0.0005
ETOLERANCE = *scalar*	Maximum size of ratio of the original to the new estimates allowed in Aitken extrapolation; default 7.5
GROUPTERM = *formula*	Random term to use as groups when fitting the augmented mean model; default * i.e. none

Parameters

Y = *variate*	Response variate (must be one only)
NBINOMIAL = *variate*	Total numbers for binomial data
RESIDUALS = *variate*	Saves the residuals
FITTEDVALUES = *variate*	Saves the fitted values
SAVE = *pointer*	Saves details of the analysis for use in subsequent HGDISPLAY, HGKEEP, HGPLOT or HGPREDICT statements

The variate to be analysed is supplied by the Y parameter and, if the y-values are binomial responses, the NBINOMIAL parameter should specify the corresponding variate of totals. Residuals and fitted values can be saved using the RESIDUALS and FITTEDVALUES parameters, respectively. Note that only one y-variate can be analysed at once, so any additional variates are ignored (as occurs with the MODEL directive when generalized linear models are defined).

The SAVE parameter allows you to save a pointer containing full details of the analysis. This can then be used to generate further output from HGDISPLAY, HGKEEP, HGPLOT or HGPREDICT. The most recent save structure is kept automatically inside Genstat to use as a default for the SAVE options of HGDISPLAY, HGKEEP, HGPLOT and HGPREDICT. So, you need save the pointer explicitly only if you want to display output from more than one analysis at a time.

The PRINT option specifies what output is required, with settings:

model	details of the model that has been fitted;
fixedestimates	estimates of the fixed effects in the HGLM;
randomestimates	estimates of the random effects in the HGLM;
dispersionestimates	estimates of the parameters in the dispersion models;
likelihoodstatistics	likelihood statistics for assessing the models;
deviance	scaled deviances for assessing goodness of fit;
waldtests	Wald tests of the terms that can be dropped from the fixed model (obtained using the HGWALD procedure);
fittedvalues	table with unit number, response variable, fitted values, residuals and leverages;
monitoring	monitoring of the fitting of the HGLM; and
dhgmonitoring	monitoring of the fitting of the HGLM for the dispersion model in a DHGLM.

The SEMETHOD option specifies which method to use to calculate standard errors for the estimated parameters of the dispersion models. The default, approximate, method is efficient to compute, but it may show downwards bias. However, the alternative profilelikelihood method can be very time-consuming.

The DMETHOD option controls the method used to calculate the adjusted profile likelihood during the calculation of the likelihood statistics. The choleski method is fastest, while the lrv method provides a more robust alternative to use if choleski fails. The default setting, automatic, tries choleski first and then, if that fails, uses lrv instead.

The other options control various aspects of the fitting process. The fitting process involves alternative fits of the augmented GLM for the mean given the current estimates of the dispersion parameters, and of the GLMs that estimate the dispersion parameters. The convergence of the process is assessed by comparing the dispersion estimates from successive fits. The MAXCYCLE option can specify two scalars. The first sets a limit on the number of alternating fits (default 99), and the second controls the number of iterations in the estimation of the mean model and of the dispersion model (default 50). The TOLERANCE option defines the criterion for convergence in the alternating fits (default 0.005). The EMETHOD option determines whether Aitken (default) or adjusted Aitken extrapolation is used in the estimation of the dispersion estimates, or you can set EMETHOD=* to use neither. The ETOLERANCE option sets an upper limit on the ratio of the changed value to the original values in the extrapolations; the default value is 7.5. The GROUPTERM option allows you to specify a random term whose factor combinations should be used as a groups factor during the fitting of the augmented mean model (see the GROUPS option of the MODEL directive). This allows models with large numbers of random effects to be fitted much more efficiently. However, algorithmic complications mean that predictions can then be made by HGPREDICT only using a BLUP for a specific random effect of that term – you cannot form predictions at the expected value of the term. The EXIT option can be set to a scalar which will be set to zero or one according to whether or not the fitting has been successful.

By default HGANALYSE uses exact likelihood to obtain the y-variate and weights for the dispersion model. This produces estimates with less bias than the earlier method, in Releases 6-8, of extended quasi likelihood (EQL). However, option LMETHOD is provided to enable EQL estimates to be obtained if required. For some of the models the DLAPLACEORDER option allows the order of Laplace approximation involved in the estimation of the dispersion components to be increased from the standard value (and default) of 0, to either 1 or 2. This is appropriate for generalized linear mixed models with the binomial or Poisson distributions, where use of Laplace order 0 can lead to serious downwards bias. The MLAPLACEORDER option similarly allows you to set the order of Laplace approximation to use in the estimation of the mean model to 1 instead of 0.

Example 3.5.11a illustrates the fitting of an HGLM by analysing data from Cochran & Cox (1957, page 300) on the breaking angles of cake. Forty five batches of cake mixture were prepared, as fifteen replicates each with a batch of mixture from three different recipes. Each batch was subdivided into ten sub-batches, randomly allocated to be baked at ten different temperatures. (The design is thus a split-plot, as described in 4.2.1, with random terms for the replicates and batches of material, in addition to the usual residual term.) The data values are assumed to follow a generalized linear model with a gamma distribution and reciprocal link. The linear predictor contains additional random variables, with inverse gamma distributions and reciprocal for replicates and batches of cake mixture.

Example 3.5.11a

```
 2  FACTOR   [NVALUES=270; LEVELS=3] Recipe
 3  &        [LEVELS=15] Replicate
 4  &        [LEVELS=!(175,185...225)] Temperature
 5  GENERATE Recipe,Replicate,Temperature
 6  VARIATE  [NVALUES=270] Angle
 7  READ     Angle

  Identifier   Minimum      Mean   Maximum    Values   Missing
       Angle     18.00     32.12     63.00       270         0

23  FACPRODUCT    !p(Replicate,Recipe); Batch
```

```
  24  HGFIXEDMODEL  [DISTRIBUTION=gamma; LINK=reciprocal] Recipe*Temperature
  25  HGRANDOMMODEL [DISTRIBUTION=inversegamma; LINK=reciprocal]\
  26                Replicate+Batch
  27  HGANALYSE     Angle

Monitoring
----------

cycle no., disp. components & max. absolute change
           2      -3.937      -10.72      -11.98      0.4187
           3      -3.933      -10.65      -12.14      0.1573
           4      -3.929      -10.64      -12.22     0.07506
           5      -3.927      -10.63      -12.25     0.03629
           6      -3.926      -10.63      -12.27     0.01796
Aitken extrapolation OK
           7      -3.925      -10.63      -12.29     0.01794
           8      -3.925      -10.63      -12.29   0.0001296

Hierarchical generalized linear model
=====================================

Response variate: Angle

Mean model
----------

Fixed terms: Recipe*Temperature
Distribution: gamma
Link: reciprocal
Random terms: Replicate + Batch
Distribution: inversegamma
Link: reciprocal
Dispersion: free


Dispersion model
----------------

Distribution: gamma
Link: logarithm


Estimates from the mean model
=============================

                           estimate        s.e.        t(*)
constant                  -1.965122    0.001863    -1054.97
Recipe 2                   0.003148    0.001987        1.58
Recipe 3                   0.001846    0.001952        0.95
Temperature 185           -0.002559    0.001681       -1.52
Temperature 195           -0.001821    0.001700       -1.07
Temperature 205           -0.004406    0.001635       -2.69
Temperature 215           -0.008246    0.001543       -5.34
Temperature 225           -0.005675    0.001604       -3.54
Recipe 2 .Temperature 185 -0.000558    0.002468       -0.23
Recipe 2 .Temperature 195 -0.003713    0.002437       -1.52
Recipe 2 .Temperature 205 -0.001506    0.002385       -0.63
Recipe 2 .Temperature 215  0.000315    0.002287        0.14
Recipe 2 .Temperature 225 -0.002884    0.002317       -1.24
Recipe 3 .Temperature 185  0.001361    0.002448        0.56
Recipe 3 .Temperature 195 -0.002316    0.002406       -0.96
Recipe 3 .Temperature 205  0.001119    0.002377        0.47
Recipe 3 .Temperature 215  0.001775    0.002255        0.79
Recipe 3 .Temperature 225 -0.001825    0.002279       -0.80
```

```
Estimates from the dispersion model
===================================

Estimates of parameters
-----------------------

                                                    antilog of
Parameter              estimate         s.e.      t(*)   estimate
phi                     -3.9246       0.0947    -41.46    0.01975
lambda Replicate        -10.626        0.398    -26.68 0.00002428
lambda Batch            -12.287        0.342    -35.89 0.000004609


Likelihood statistics
=====================

-2 * h(y|v)           1517.907
-2 * h                 945.528
-2 * P_v(h)           1622.548
-2 * P_beta,v(h)      1829.042
-2 * EQD(y|v)         1517.019
-2 * EQD               944.639
-2 * P_v(EQD)         1621.659
-2 * P_beta,v(EQD)    1828.153

Fixed parameters in mean model    18
Random parameters in mean model   60
Fixed dispersion parameters        3
Random dispersion parameters       0

Scaled deviances
================

                deviance          df
  Random term
      *units*       223.2       222.3
    Replicate        12.6        12.6
        Batch        17.1        17.1
        Total       252.9       252.0
```

Lee & Nelder (1996) suggest that changes in the fixed model are assessed using changes in the deviance from the adjusted profile likelihood $-2 \times P_v(h)$, while changes in the dispersion models are assessed using $-2 \times P_{\beta,v}(h)$. The deviance of the conditional likelihood $-2 \times h(y|v)$ can be used to calculate the deviance information coefficient (DIC), and $-2 \times h$ is the h-deviance of the mean model. The EQD statistics are approximations to the h-likelihood statistics, calculated using quasi-likelihood instead of exact likelihood. The scaled deviances assess goodness of fit over the variation represented by each random term, and are analogous to the deviance in an ordinary generalized linear model.

Tests based on these likelihoods can be made automatically using `HGRTEST` and `HGFTEST`.

`HGRTEST` procedure

Calculates likelihood tests for random terms in a hierarchical generalized linear model (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

`PRINT` = *string token*	Controls printed output (`tests`); default `test`
`LMETHOD` = *string token*	Whether to use exact likelihood or extended quasi likelihood to obtain the y-variate and weights for the dispersion model (`exact`, `eql`); default `exac`
`DMETHOD` = *string token*	Method to use for the adjusted profile likelihood when calculating the likelihood statistics (`automatic`, `choleski`, `lrv`); default `auto`

EMETHOD = *string token*	Extrapolation method to use (aitken, adjustedaitken); default aitk
MLAPLACEORDER = *scalar*	Order of Laplace approximation to use in the estimation of the mean model (0 or 1); default 0
DLAPLACEORDER = *scalar*	Order of Laplace approximation to use in the estimation of the dispersion components (0, 1 or 2); default 0
MAXCYCLE = *scalars*	Maximum number of iterations of the hierarchical generalized linear model fits, and maximum number of iterations in the fitting of the mean and dispersion models; default 99,50
EXIT = *scalar*	Exit status (0 for success, 1 for failure to converge)
TOLERANCE = *scalar*	Criterion for convergence; default 0.0005
ETOLERANCE = *scalar*	Maximum size of ratio of the original to the new estimates allowed in Aitken extrapolation; default 7.5
GROUPTERM = *formula*	Random term to use as groups when fitting the augmented mean model; default * i.e. none
SAVE = *pointer*	Save structure from the original analysis

Parameters

TERMS = *formula*	Terms to test
TESTSTATISTIC = *pointer* or *scalar*	Saves the test statistics
DF = *pointer* or *scalar*	Saves the degrees of freedom

By default, HGRTEST produces tests for every random term. However, you can use the TERMS parameter to request tests for a specific set of terms. The TESTSTATISTIC parameter can save the statistics, and the DF parameter can save their numbers of degrees of freedom. If you are making a test for a single term, you can supply a scalar for each of these parameters. However, if you have several terms, you must supply a pointer which will then be set up to contain as many scalars as there are terms.

The tests are made by calculating the change in the profile likelihood $P_{\beta,v}(h)$ as the term concerned is dropped from the random model. So, HGRTEST needs to refit the model with the revised random model. The LMETHOD, DMETHOD, EMETHOD, MLAPLACEORDER, DLAPLACEORDER, MAXCYCLE, EXIT, TOLERANCE, ETOLERANCE and GROUPTERM options control how the fitting is done, and the likelihood is calculated. These all operate exactly as in the HGANALYSE procedure, and should generally be set to the same values as in the original analysis (by HGANALYSE). By default, the random terms are dropped from the most recent HGLM analysis, but you can use the SAVE option to supply the save structure from some earlier analysis.

Example 3.5.11b prints likelihood tests for the random terms in Example 3.5.11a. One point to note is that we are testing the random terms against a null hypothesis (that they have zero variance components) which is on the boundary of the parameter space. To allow for this, Lee, Nelder & Pawitan (2006, p. 219) suggest using the critical value for twice the required significance probability or, equivalently, dividing the chi-square probabilities by two. This is not done in the procedure, but is something to bear in mind when assessing the results. Here it is not necessary as the probabilities are <0.001.

Example 3.5.11b

```
28  HGRTEST
```

```
Likelihood tests for dropping HGLM random terms
===============================================

     Term  Test statistic  d.f.          pr.
Replicate           26.45     1       <0.001
    Batch           11.20     1       <0.001
```

HGFTEST procedure

Calculates likelihood tests for fixed terms in a hierarchical generalized linear model (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

PRINT = *string token*	Controls printed output (tests); default test
FACTORIAL = *scalar*	Limit on number of factors in the model terms generated from the TERMS parameter
LMETHOD = *string token*	Whether to use exact likelihood or extended quasi likelihood to obtain the y-variate and weights for the dispersion model (exact, eql); default is to use the same setting as in the original analysis
DMETHOD = *string token*	Method to use for the adjusted profile likelihood when calculating the likelihood statistics (automatic, choleski, lrv); default auto
EMETHOD = *string token*	Extrapolation method to use (aitken, adjustedaitken); default is to use the same setting as in the original analysis
MLAPLACEORDER = *scalar*	Order of Laplace approximation to use in the estimation of the mean model (0 or 1); default is to use the same setting as in the original analysis
DLAPLACEORDER = *scalar*	Order of Laplace approximation to use in the estimation of the dispersion components (0, 1 or 2); default is to use the same setting as in the original analysis
MAXCYCLE = *scalars*	Maximum number of iterations of the hierarchical generalized linear model fits, and maximum number of iterations in the fitting of the mean and dispersion models; default 99,50
EXIT = *scalar*	Exit status (0 for success, 1 for failure to converge with any of the fixed terms)
TOLERANCE = *scalar*	Criterion for convergence; default is to use the same setting as in the original analysis
ETOLERANCE = *scalar*	Maximum size of ratio of the original to the new estimates allowed in Aitken extrapolation; default is to use the same setting as in the original analysis
SAVE = *pointer*	Save structure from the original analysis

Parameters

TERMS = *formula*	Terms to test
TESTSTATISTIC = *pointer* or *scalar*	Saves the test statistics
DF = *pointer* or *scalar*	Saves the degrees of freedom

By default, HGFTEST produces tests for all the fixed terms that can be dropped: that is, for every term that is not marginal to another term in the fixed model. For example, in the formula

```
A + B + C + D + A.B + A.D + B.D
```

the terms C, A.B, A.D and B.D can be dropped as there are no other terms in the model that contain all their factors (i.e. none to which thay are marginal). However, A cannot be dropped until A.B and A.D have been dropped. You can use the TERMS parameter to request tests for a specific set of terms, but a missing value is given for any term that cannot be dropped. The FACTORIAL option sets a limit on the number of factors in each term that is formed from the TERMS formula (default 3).

The TESTSTATISTIC parameter can save the statistics, and the DF parameter can save their numbers of degrees of freedom. If you are making a test for a single term, you can supply a scalar for each of these parameters. However, if you have several terms, you must supply a pointer which will then be set up to contain as many scalars as there are terms.

The tests are made by calculating the change in the profile likelihood $P_v(h)$ as the term concerned is dropped from the fixed model. The LMETHOD, DMETHOD, EMETHOD, MLAPLACEORDER, DLAPLACEORDER, MAXCYCLE, TOLERANCE and ETOLERANCE, options control how the fitting is done, and the likelihood is calculated. These all operate exactly as in the HGANALYSE procedure. The default for DMETHOD is automatic, and the default for MAXCYCLE= is 99,50. For the other options the defaults are to use the same settings as in the HGANALYSE command that performed the original analysis.

By default, the terms are dropped from the most recent HGLM analysis, but you can use the SAVE option to supply the save structure from some earlier analysis.

Example 3.5.11c shows the likelihood test for the interaction Recipe.Temperature, which is the only fixed term that can dropped from the fixed model in Example 3.5.11a.

Example 3.5.11c

```
  29  HGFTEST

Likelihood tests for dropping HGLM fixed terms
==============================================

              Term  Test statistic  d.f.          pr.
Recipe.Temperature           8.918    10        0.540
```

A faster, but more approximate way of assessing the fixed terms, is to use Wald tests. These can be calculated using HGWALD.

HGWALD procedure

Prints or saves Wald tests for fixed terms in an HGLM (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

PRINT = *string token*	Controls printed output (waldtests); default wald
FACTORIAL = *scalar*	Limit on number of factors in the model terms generated from the TERMS parameter; default 3
SAVE = *pointer*	Specifies the save structure (from HGANALYSE) of the analysis from which to calculate the tests; default uses the most recent analysis

Parameters

TERMS = *formula*	Model terms for which tests are required
WALDSTATISTIC = *scalar* or *pointer* to *scalars*	Saves Wald statistics

DF = *scalar* or *pointer* to *scalars*	Saves d.f. of Wald statistics

HGWALD has a similar syntax to HGFTEST. By default it produces tests for all the fixed terms that can be dropped, but you can use the TERMS parameter and FACTORIAL option to request Wald tests for a specific set of terms.

Example 3.5.11d shows the Wald test for the interaction Recipe.Temperature in Example 3.5.11a.

Example 3.5.11d

```
  30  HGWALD

Wald tests for dropping HGLM fixed terms
========================================

              Term  Wald statistic  d.f.  approx. pr.
Recipe.Temperature           8.877    10        0.544
```

HGDISPLAY procedure

Displays results from a hierarchical or displaying double hierarchical generalized linear model analysis (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

PRINT = *string tokens*	Controls printed output (model, fixedestimates, randomestimates, dispersionestimates, likelihoodstatistics, deviance, waldtests, fittedvalues); default *
SEMETHOD = *string token*	Method to use to calculate the se's for the dispersion estimates (approximate, profilelikelihood); default appr
DMETHOD = *string token*	Method to use for the adjusted profile likelihood when calculating the likelihood statistics (automatic, choleski, lrv); default auto
DISPERSIONTERM = *formula*	Model term for output from a dispersion analysis
SAVE = *pointer*	Save structure (from HGANALYSE) to provide details of the analysis; if omitted, output is from the most recent analysis

No parameters

HGDISPLAY allows you to display further output from the analysis. Its options operate almost exactly as in HGANALYSE. However, the PRINT does not provide the settings monitoring and dghmonitoring, which print information during the fitting process. HGDISPLAY also has a SAVE option, to specify the save structure (saved using the SAVE parameter of HGANALYSE) containing details of the analysis. However, you do not need to save or specify this unless you want to display output from more than one analysis at a time.

By default the output is from the analysis of the mean model, but you can set the DISPERSIONTERM option to a formula defining one of the random terms to obtain information from the analysis to model its dispersion parameter.

You can form tables of predictions using HGPREDICT.

HGPREDICT procedure

Forms predictions from a hierarchical or double hierarchical generalized linear model analysis (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

PRINT = *string token*	What to print (description, predictions, se, sed, vcovariance); default desc, pred, se
COMBINATIONS = *string token*	Which combinations of factors in the current model to include (full, present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment (marginal, equal); default marg
WEIGHTS = *table*	Weights classified by some or all of the factors in the model; default *
OFFSET = *scalar*	Value of offset on which to base predictions; default mean of offset variate
METHOD = *string token*	Method of forming margin (mean, total); default mean
ALIASING = *string token*	How to deal with aliased parameters (fault, ignore); default faul
BACKTRANSFORM = *string token*	What back-transformation to apply to the values on the linear scale, before calculating the predicted means (link, none); default none
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, nonlinear); default *
NBINOMIAL = *scalar*	Supplies the total number of trials to be used for prediction with a binomial distribution (providing a value *n* greater than one allows predictions to be made of the number of "successes" out of *n*, whereas the value 1 predicts the proportion of successes); default 1
PREDICTIONS = *table* or *scalar*	To save the predictions; default *
SE = *table* or *scalar*	To save standard errors of predictions; default *
SED = *symmetric matrix*	To save matrices of standard errors of differences between predictions; default *
VCOVARIANCE = *symmetric matrix*	To save variance-covariance matrices of predictions; default *
SAVE = *pointer*	Specifies the save structure (from HGANALYSE) of the analysis from which to predict; default uses the most recent analysis

Parameters

CLASSIFY = *vectors*	Variates and/or factors to classify table of predictions
LEVELS = *variates* or *scalars*	To specify values of variates, levels of factors
NEWFACTOR = *identifiers*	Identifiers for new factors that are defined when LEVELS are specified

HGPREDICT allows you to form predictions for various values of the parameters in the fixed model. It uses the PREDICT directive internally, and its options and parameters are a subset of those of PREDICT (3.3.4). They are used in the same way as in PREDICT, except that back-transformations are possible only with conjugate models. Consequently, the default for option BACKTRANSFORM is none.

The CLASSIFY list can contain factors from either the fixed or random models but you may specify only one level for each random factor. If all the factors in a particular random term are

in the CLASSIFY list, the prediction will use the BLUP (best linear unbiased predictor) for the random effect of the term corresponding to the levels that are specified for its factors. Otherwise, provided that random term was not used as a group term in the analysis (see the GROUPTERM option of HGANALYSE), the predictions will be at the mean value of the random distribution of the term. Alternatively, if that random term was used as a group term, HGPREDICT will make the predictions using the smallest BLUP of the term.

Example 3.5.11e forms predictions for the cake data in Example 3.5.11a.

Example 3.5.11e

```
  31  HGPREDICT     [PRINT=description,prediction,se] Recipe,Temperature

Predictions from regression model
---------------------------------

These predictions are estimated mean values, formed on the scale of the linear
predictor.

The predictions have been formed only for those combinations of factor levels
for which means can be estimated without involving aliased parameters.

The predictions are at the mean value of the distribution of any random term
whose factor levels have not all been fixed.

The standard errors are appropriate for interpretation of the predictions as
summaries of the data rather than as forecasts of new observations.


  Temperature         175                     185
              predictions          se predictions          se
       Recipe
            1    0.034878    0.001863    0.032319    0.001801
            2    0.038026    0.001931    0.034909    0.001852
            3    0.036724    0.001895    0.035526    0.001865


  Temperature         195                     205
              predictions          se predictions          se
       Recipe
            1    0.033058    0.001818    0.030473    0.001758
            2    0.032492    0.001794    0.032115    0.001785
            3    0.032588    0.001793    0.033438    0.001813


  Temperature         215                     225
              predictions          se predictions          se
       Recipe
            1    0.026632    0.001675    0.029204    0.001730
            2    0.030095    0.001738    0.029467    0.001724
            3    0.030254    0.001738    0.029224    0.001715

* MESSAGE: s.e's, variances & lsd's are approximate, since the model is not
linear.
```

You can use HGPLOT to obtain model-checking plots.

HGPLOT procedure

Produces model-checking plots for a hierarchical or double hierarchical generalized linear model analysis (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

MODELTYPE = *string token* — Type of model for which plots are required (mean, dispersion); default mean

RANDOMTERM = *formula*	Random term whose residuals are to be plotted; default * i.e. the residuals from the full model
DHGRANDOMTERM = *formula*	Random model term in a DHGLM whose residuals are to be plotted; default *
RMETHOD = *string token*	Type of residual to use (deviance, Pearson, simple); default devi
INDEX = *variate* or *factor*	X-values to use for an index plot; default !(1,2...)
GRAPHICS = *string token*	What type of graphics to use (lineprinter, highresolution); default high
TITLE = *text*	Overall title for the plots; if unset, the identifier of the y-variate is used
SAVE = *pointer*	Specifies the analysis (by HGANALYSE) from which the residuals and fitted values are to be taken; by default they are taken from the most recent analysis

Parameters

METHOD = *string tokens*	Types of graph (up to four out of the six possible) to be plotted (histogram, fittedvalues, absresidual, normal, halfnormal, index); default hist, fitt, norm, absr
PEN = *scalars*, *variates* or *factors*	Pen(s) to use for each plot

Six types of plot are available, which can be are selected using the METHOD parameter with settings:

histogram	histogram of residuals;
fittedvalues	residuals versus fitted values;
absresidual	absolute values of residuals versus fitted values;
normal	Normal plot;
halfnormal	half-Normal plot; and
index	plot against an "index" variable (specified by the INDEX option).

Up to four can be examined in any call of the procedure. The PEN parameter can be used to specify the graphics pen or pens to use for each plot. The TITLE option can supply an overall title; if this is not set, the identifier of the y-variate is used.

The MODELTYPE option indicates the type of model for which the plots are required. The default setting mean requests plots from the mean GLM, while the alternative setting dispersion obtains plots from the dispersion GLM. The RANDOMTERM option specifies the random term whose residuals (in a mean or dispersion model) are to be plotted; if this is omitted the plot is for the residual term (i.e. for dispersion parameter φ). If a DHGLM has been fitted, you can plot residuals from the HGLM that is being used as a dispersion model by setting the DHGRANDOMTERM parameter to the random term concerned. The type of residual to plot is specified by the RMETHOD option; by default these are deviance residuals.

The fitted model can be displayed using HGGRAPH.

HGGRAPH procedure

Draws a graph to display the fit of an HGLM or DHGLM analysis (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

GRAPHICS = *string token*	Type of graphics to use (lineprinter, highresolution); default high

TITLE = *text*	Title for the graph; default * sets an appropriate title automatically
WINDOW = *number*	Which high-resolution graphics window to use; default 4 (redefined if necessary to fill the frame)
SCREEN = *string token*	Whether to clear the graphics screen before plotting (clear, keep); default clea
BACKTRANSFORM = *string token*	Whether to back-transformation the response scale (link, none, axis); default none
OMITRESPONSE = *string token*	Whether to omit the adjusted response values (no, yes); default no
SAVE = *pointer*	Specifies the save structure (from HGANALYSE) of the analysis from which to predict; default uses the most recent analysis

Parameters

INDEX = *variates* or *factors*	Which variate or factor to display along the x-axis; default * if GROUPS is set, otherwise INDEX is set to the first variate in the fixed model
GROUPS = *factors*	Factor to define groups of points to display; default * if INDEX is set, otherwise GROUPS is set to the first factor in the fixed model

HGGRAPH has a similar role to the RGRAPH procedure in ordinary regression and generalized linear models (3.1.5). It displays the fitted model in one or two dimensions. It usually also displays the observed response values, adjusted for any other explanatory terms in the model, but these can be omitted by setting option OMITRESPONSE=yes.

The dimensions to display are specified by the INDEX and GROUPS parameters. The INDEX vector, which can be either a variate or a factor from the fixed model of the HGLM, defines the x-axis of the plot. (The y-axis corresponds to the response scale.) The GROUPS parameter can be set to another factor from the fixed model. A set of points is then plotted for each level of GROUPS, so that you can study the interaction between GROUPS and INDEX. If INDEX and GROUPS are not set, HGGRAPH takes the first variate (if any) and the first factor in the fixed model.

The TITLE option can be used to supply a title for the graph. By default the graph is plotted on the current high-resolution device, but the GRAPHICS option can be set to line for a line printer plot. The WINDOW option can be used to select a pre-defined window for high-resolution plots; otherwise window 4 is used, and is redefined if necessary to fill the frame. The SCREEN option allows the graph to be added to an existing high-resolution plot. The colours and symbols used in the displays can be controlled by setting the attributes of the following pens with the PEN directive before calling the procedure:

pen 1	labels for lines when drawn for each level of a factor,
pen 2	fitted lines and means,
pen 3	points, and
pen 4	back-transformed axis marks and labels.

The relationship is usually plotted on the scale of the linear predictor but, with a conjugate HGLM, you can set option BACKTRANSFORM=link to use the original scale of the response. Alternatively, you can set BACKTRANSFORM=axis to include axis markings, back-transformed onto the natural scale, on the right-hand side of the y-axis. However, this is not available for the reciprocal link.

Information from the analysis can be saved using HGKEEP.

HGKEEP procedure

Saves information from a hierarchical or double hierarchical generalized linear model analysis (R.W. Payne, Y. Lee, J.A. Nelder & M. Noh).

Options

MODELTYPE = *string token*	Type of model from which to save information (mean, dispersion); default mean
RMETHOD = *string token*	Type of residuals to save using the RESIDUALS parameter (deviance, Pearson, simple); default devi
DMETHOD = *string token*	Method to use for the adjusted profile likelihood when calculating the likelihood statistics (automatic, choleski, lrv); default auto
IGNOREFAILURE = *string token*	Whether to save information even if the fitting of the HGLM failed to converge (yes, no); default no
SAVE = *pointer*	Save structure (from HGANALYSE) to provide details of the analysis; if omitted, information is saved from the most recent analysis

Parameters

RANDOMTERM = *formula*	Random model terms from whose analysis the information is to be saved
DHGRANDOMTERM = *formula*	Random model terms in a DHGLM from whose (HGLM) analysis the information is to be saved
RESIDUALS = *variates*	Residuals
FITTEDVALUES = *variates*	Fitted values
LEVERAGES = *variates*	Leverages
ESTIMATES = *variates*	Estimates of parameters
SE = *variates*	Standard errors of the estimates
VCOVARIANCE = *symmetric matrices*	Variance-covariance matrix of each set of estimates
DEVIANCE = *scalars* or *tables*	Scaled deviances (in a table) for a mean model, or residual deviance (in a scalar) for a dispersion model
DF = *scalars* or *tables*	Residual degrees of freedom
ITERATIVEWEIGHTS = *variates*	Iterative weights
LINEARPREDICTOR = *variates*	Linear predictors
YADJUSTED = *variates*	Adjusted responses
LIKELIHOODSTATISTICS = *variates*	Likelihood statistics
LDF = *variates*	Numbers of fixed and random parameters in the mean and dispersion models

The MODELTYPE option indicates the model (mean or dispersion) from which the information is to be saved; by default this is the model for the mean. The RANDOMTERM parameter specifies the random term from whose analysis the information is to be saved; if this is omitted the information is for the residual term (i.e. dispersion φ). If a DHGLM has been fitted, you can save information from the HGLM that is being used as a dispersion model by setting the DHGRANDOMTERM parameter to the random term concerned.

The LIKELIHOODSTATISTICS parameter saves the likelihood statistics (as given by the likelihoodstatistics setting of the PRINT option of HGANALYSE and HGDISPLAY). The

DMETHOD option controls the method used to calculate the adjusted profile likelihood during the calculation of the likelihood statistics. The choleski method is fastest, while the lrv method provides a more robust alternative to use if choleski fails. The default setting, automatic, tries choleski first and then, if that fails, uses lrv instead.

The LDF parameter saves the numbers of fixed and random parameters in the mean and dispersion models. (These accompany the likelihood statistics in the output, and indicate the numbers of parameters represented by the various statistics.)

The other parameters operate as in the RKEEP directive (3.1.4) except that, for a mean model, DEVIANCE saves tables of scaled deviances and DF saves a table with the corresponding degrees of freedom. Similarly, as in the RKEEP directive, the RMETHOD option indicates the type of residual to form.

By default, HGKEEP will give a warning (and nothing will be saved) if the fitting of the HGLM failed to converge. Alternatively, you can set option IGNOREFAILURE=yes to save information from the final iteration.

3.5.12 Generalized estimating equations

GEE procedure

Fits models to longitudinal data by generalized estimating equations (D.M. Smith & M.G.Kenward).

Options

PRINT = *string token*	What to display (estimates, correlations, scalefactor, wald, monitoring); default esti, corr, scal
DISTRIBUTION = *string token*	Distribution of response (normal, Poisson, binomial, gamma, inversenormal, negativebinomial); default *
LINK = *string token*	Link function (identity, logarithm, logit, reciprocal, power, squareroot, probit, complementaryloglog, logratio); default *
EXPONENT = *scalar*	Exponent for power link; default −2
TERMS = *formula*	Explanatory variates, factors etc
CONSTANT = *string token*	How to treat constant (estimate, omit); default esti
FACTORIAL = *scalar*	Limit for expansion of model terms; default 3
AGGREGATION = *scalar*	Fixed parameter for negative binomial distribution (parameter k as in variance function var = mean + mean^2/k); default 1
KLOGRATIO = *scalar*	Parameter for logratio link, in form log(mean / (mean + k)); default as set in AGGREGATION option
QUADESTIMATION = *string token*	Whether to use quadratic estimation (used, notused); default used
SCALEFACTOR = *string token*	How to calculate the scale factor (fixed, constant, varytime); default varies with distribution, fixed for Poisson and binomial, constant for rest
SFVALUE = *scalar*	Value for scale factor when SCALEFACTOR=fixed; default 1.0 for Poisson and binomial, missing for rest
CRTYPE = *string token*	Form of correlation matrix (independence, unstructured, exchangeable, autoregressive, dependence, antedependence); default *
ORDER = *scalar*	Order in dependence and ante-dependence form of

	correlation matrix; default 1
TIMEDEPENDENT = *string token*	Whether correlation in dependence model changes with time (no, yes); default no

Parameters

Y = *variates*	Response variate for each analysis
NBINOMIAL = *variates*	Denominator in binomial
FITTEDVALUES = *variates*	To store fitted values
RESIDUALS = *variates*	To store residuals
SUBJECT = *factors*	Identifier of subjects
OUTCOME = *factors*	Identifier of outcomes
COUNT = *variates*	Variate of counts of no. outcomes
TIME = *factors*	Times of repeated measures variate
WEIGHT = *variates*	Weight variate
OFFSET = *variates*	Offset variate
SAVE = *pointers*	Structure to save output variables

GEE implements the General Estimating Equation (GEE) methodology of Liang & Zeger (1986) with quadratic estimation for the covariance structure. In the terminology of Liang *et al*. (1992) the methodology implemented is a form of GEE1. Full details of the implementation are given in Kenward & Smith (1995a). GEE, as implemented here, is a comparatively simple non-likelihood method for fitting marginal models to repeated measurements that can be used when the response has a distribution in the exponential family. This includes the Gaussian distribution, for which the procedure implemented here reduces to a form of the EM algorithm, and then produces exact ML or REML estimates, or a close approximation to these depending on the particular correlation structure chosen. For other distributions the resulting estimates are not maximum likelihood but can be shown to have asymptotic properties familiar from quasi-likelihood, such as consistency and asympototic normality.

The standard range of generalized linear models (as in procedure GLM) can be fitted involving a variety of covariance/correlation structures over the times of the repeated measurements. The standard links and distributions can be chosen by setting the options DISTRIBUTION, LINK, EXPONENT, AGGREGATION and KLOGRATIO, as in the MODEL directive (3.1.1). Non-standard ones require the definition of auxiliary procedures to carry out the necessary calculations (see below). The terms in the fitted model are specified by the TERMS option, which may be set to a formula or left unset to fit a null model. The FACTORIAL option (default 3) sets a limit on the number of factors and variates in the terms that are fitted, as in the FIT directive (3.1.2). The CONSTANT option can be used to omit a constant term. Setting the QUADESTIMATION option to used requests the use of quadratic estimation for the data-based covariance/correlation matrix (see Kenward & Smith 1995a). The SCALEFACTOR option specifies the form of scalefactor to be used (fixed to a value specified by the SFVALUE option, constant over times of repeated measurements, or varying over times of repeated measurements). The CRTYPE option specifies the structure of the covariance/correlation matrix over the times of the repeated measurements. The ORDER option specifies the order of the covariance/correlation structures for the dependence and ante-dependence cases, with option TIMEDEPENDENT specifying whether the correlation in a dependence structure changes with the time of the repeated measurement.

The Y parameter must be set to specify the response variate. For a binomial distribution the NBINOMIAL parameter must also be set. The SUBJECT parameter specifies a factor to identify the subjects. Alternatively, where the data consist of outcomes and numbers with those outcomes, the parameter OUTCOME must be set to the identifier of the outcome and the parameter COUNT to the number with the outcome. The parameter TIME must be set to the times of the repeated measurements. The parameters WEIGHT and OFFSET specify weight and offset variates

that may be involved. Neither `Y` nor any of the other input structures must be restricted, and any existing restrictions will be cancelled.

The output from the procedure is controlled by the `PRINT` option; by default estimates, their standard errors, covariances/correlations and scalefactors are given. Two sets of standard errors are provided for the estimates. One is the naive estimate which assumes the specified covariance/correlation structure holds. The other is the sandwich estimate which makes no such assumption. When `PRINT=wald`, Wald tests are produced using both sets of standard errors and correlations.

The fitted values and residuals can be obtained by setting the parameters `FITTEDVALUES` and `RESIDUALS`. The residuals are the Pearson residuals as defined in the Genstat manual.

The `SAVE` parameter can save various details of the analysis, in a pointer with the following suffixes and labels:

`1` or `'scalefactors'`	scalefactor(s),
`2` or `'correlation'` or `'covariance'`	correlations or covariances, according to the type of model (and labelled appropriately),
`3` or `'estimates'`	the estimates of the linear predictor parameters,
`4` or `'naive covariances'`	naive variance-covariance matrix for the estimates,
`5` or `'sandwich covariances'`	sandwich variance-covariance matrix for the estimates,
`6` or `'naive Wald'`	Wald tests calculated using the naive variance-covariance matrix, and
`7` or `'sandwich Wald'`	Wald tests calculated using the sandwich variance-covariance matrix.

The algorithms in the procedure have been set up assuming that the data contain a complete set of observations for each subject. Where there are missing values these must be included explicitly (using the missing value symbol `*`) to create a complete set of observations. Missing values are allowed in both the `Y` variate and the explanatory variates in `TERMS`.

In the case of the Gaussian distribution, a working covariance matrix, rather than correlation matrix, is used. This provides considerable simplification within the algorithm.

Example 3.5.12 uses `GEE` to fit a model with a log link and a gamma distribution with autoregressive errors.

Example 3.5.12

```
" Example of how to use GEE: data from Archer
  (1987, Fertility & Sterility, 47, 559-564)
  about prolactin response to thyroptrin releasing
  hormone in women grouped according fertility status;
  also see Paik (1992, Biometrics, 48, 19-30)."
VARIATE [VALUES=44,45,41,40,72,49,41,31,37,23,15,10,103,79,47,\
        39,51,40,32,15,97,98,76,51,59,55,49,36,97,75,\
        49,38,88,78,61,43,53,40,29,23,66,35,18,16,60,\
        48,32,29,53,47,29,38,111,77,59,58,27,22,18,12,\
        51,62,40,37,28,33,20,15,49,39,32,23,59,49,43,\
        38,155,126,72,48,82,67,54,44,127,99,58,53,75,59,\
        46,29,71,62,49,44,114,110,95,52,172,95,51,43,210,\
        156,117,91,100,90,60,50,86,65,57,42,101,93,68,47] Response
&       [VALUES=16,16,16,16,10,10,10,10,7,7,7,7,8,8,8,8,5,5,5,5,\
        8,8,8,8,7,7,7,7,9,9,9,9,13,13,13,13,3,3,3,3,\
        7,7,7,7,8,8,8,8,17,17,17,17,38,38,38,38,11,11,11,11,\
        12,12,12,12,7,7,7,7,7,7,7,7,26,26,26,26,9,9,9,9,\
        19,19,19,19,12,12,12,12,20,20,20,20,41,41,41,41,4,4,4,4,\
        30,30,30,30,15,15,15,15,36,36,36,36,15,15,15,15,11,11,11,11]\
        Baseline
&       [VALUES=(1...4)30] CTime
FACTOR  [LEVELS=30; VALUES=4(1...30)] Woman
&       [LEVELS=3; VALUES=24(1),48(2),48(3)] Group
&       [LEVELS=4; VALUES=(1...4)30] Time
GEE     [PRINT=estimates,correlations,scalefactor,wald; LINK=log;\
```

```
  27          DISTRIBUTION=gamma; TERMS=Group+CTime+Baseline; \
  28          CRTYPE=autoregressive] SUBJECT=Woman; TIME=Time; Y=Response

Generalized estimating equations
================================

Quadratic estimation operating.


Independence (GLM) case
=======================

 Response variate: workvar
   Weight variate: weight
     Fitted terms: Constant + Group + CTime + Baseline


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression       4        18.78       4.6962     31.02
Residual       115        17.41       0.1514
Total          119        36.20       0.3042

Percentage variance accounted for 50.2
Standard error of observations is estimated to be 0.389.


Estimates of parameters
-----------------------

Parameter      estimate         s.e.    t(115)
Constant          4.483        0.118     38.02
Group 2         -0.0961       0.0978     -0.98
Group 3           0.434        0.106      4.08
CTime           -0.2649       0.0318     -8.34
Baseline        0.00331      0.00398      0.83

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
               Group  1


Correlations between parameter estimates
----------------------------------------

Parameter                ref    correlations

Constant                   1    1.000
Group 2                    2   -0.515  1.000
Group 3                    3   -0.379  0.649  1.000
CTime                      4   -0.674  0.000  0.000  1.000
Baseline                   5   -0.304 -0.105 -0.406  0.000  1.000
                                    1      2      3      4      5


Autoregressive correlation structure
====================================

Scale factor constant over time.

Scale factor      0.1708

Matrix of correlations

            1      1.0000
            2      0.9068      1.0000
            3      0.8223      0.9068      1.0000
            4      0.7457      0.8223      0.9068      1.0000
                        1           2           3           4
```

```
Model estimates of s.e.
-----------------------

                Estimate        s.e.
     Constant       4.418      0.1786
      Group 2      -0.080      0.1940
      Group 3       0.421      0.2111
        CTime      -0.259      0.0179
     Baseline       0.006      0.0079


Correlations
------------

     Constant   1      1.0000
      Group 2   2     -0.6744      1.0000
      Group 3   3     -0.4969      0.6487      1.0000
        CTime   4     -0.2511      0.0000      0.0000      1.0000
     Baseline   5     -0.3983     -0.1052     -0.4056      0.0000      1.0000
                           1           2           3           4           5


Wald tests using model estimates of covariances
-----------------------------------------------

  Source  Wald statistic    d.f.  Chi pr.
   Group            9.00       2    0.011
   CTime          208.84       1   <0.001
Baseline            0.61       1    0.433


Sandwich estimates of s.e.
--------------------------

                Estimate        s.e.
     Constant       4.418      0.1817
      Group 2      -0.080      0.1852
      Group 3       0.421      0.2044
        CTime      -0.259      0.0170
     Baseline       0.006      0.0077


Correlations
------------

     Constant   1      1.0000
      Group 2   2     -0.7951      1.0000
      Group 3   3     -0.4644      0.6309      1.0000
        CTime   4     -0.1349     -0.0607     -0.2471      1.0000
     Baseline   5     -0.4707      0.1237     -0.3772      0.0438      1.0000
                           1           2           3           4           5


Wald tests using sandwich estimates of covariances
--------------------------------------------------

  Source  Wald statistic    d.f.  Chi pr.
   Group            9.22       2    0.010
   CTime          232.90       1   <0.001
Baseline            0.65       1    0.422
```

For full details of the method implemented in this procedure see Kenward & Smith (1995a). A generalized linear model is formulated for the marginal distribution of the observations at each time point using an appropriate link function and error distribution. If the repeated measurements could be assumed to be independent, the well-known iterative weighted least squares fitting procedure could be used to obtain ML estimates of the marginal model parameters. However this ignores the dependence among the repeated measurements. Full likelihood is in general very awkward in this setting so, to avoid a formal introduction of dependence into the model, a

working correlation matrix is introduced into the iterative procedure, changing the least squares from a weighted to a generalized form. The correlation matrix can be introduced in various ways. It can be held constant throughout the iterative procedure. An example of this is the use of the identity matrix, leading to the so-called independence estimating equations for which the process reduces back to that of fitting a univariate generalized linear model. Alternatively an estimated correlation matrix can be introduced into the algorithm which is updated at each cycle using quadratic estimation: essentially the correlation structure is estimated from the residuals using the equations that would be appropriate were the residuals normally distributed. On convergence consistent estimates of the marginal linear model parameters are obtained and, if the correlation structure chosen is appropriate, then this will be consistently estimated as well. It is not necessary for the correlation structure to be correct for the consistency of the marginal parameter estimates, at least when the correlation structure is fixed; indeed the common choice of independence is almost certain not to be appropriate. However the estimates of precision of the marginal parameter estimates do need to be adjusted to allow for the true correlation structure. This correction is done in the so-called "sandwich" estimator provided by the procedure.

The procedures have been written so that it is possible to fit models other than the standard ones. An important example of such a model is the application of the GEE methodology to ordinal categorical data. This application requires the data to be arranged in a particular form (as cummulative logits) and a particular correlation matrix (specified in `_GEECORRELATION`). The type of analyses are explained in Kenward *et al*. (1994) and the methodology described in that paper has been duplicated. Further details are given in Kenward & Smith (1995b).

An option (`SCALEFACTOR`) has been included that allows the user to decide whether or not the scale factor is fixed at its independence distributional default, or is estimated from the scaled residuals as in Liang & Zeger (1986), or is treated as a vector varying over time.

`GEE` has four subsidiary procedures, which can be re-written or replaced, to cater for further user-defined distributions, links and correlation structures:

`_GEEINIT`	calculates initial estimates of the linear predictor in the generalized linear model;
`_GEELINK`	calculates fitted values and derivatives;
`_GEEDISTRIBUTION`	calculates the variance function and deviance;
`_GEECORRELATION`	calculates the correlation matrix and the sandwich matrix involving the residuals. (For the normal distribution the variance/covariance matrices are used not the correlation matrices.)

If the `LINK` option is unset, the procedure will call `_GEEINIT` and `_GEELINK` instead of using those for the various standard link functions. For a logit link function `_GEEINIT` and `_GEELINK` should be defined as follows.

```
PROCEDURE '_GEEINIT'
          "Calculation of initial estimate of linear predictor,
           link unset"
PARAMETER NAME = \
          'Y',              "I: variate; response variate"\
          'LINEARPREDICTOR',"O: variate; linear predictor"\
          'OFFSET',         "I: variate; offset"\
          'NBINOMIAL';      "I: variate; denominator of binomial"\
          SET=3(yes),no;TYPE=4('variate'); \
          COMPATIBLE=*,3(!T(type,nvalues,restriction));\
          PRESENT=yes,no,2(yes)

CALC LINEARPREDICTOR = LOG((Y+0.5)/(NBINOMIAL-Y+0.5)) - OFFSET
ENDPROCEDURE

PROCEDURE '_GEELINK'
          "Calculation of fitted values and derivatives"
PARAMETER NAME = \
          'LINEARPREDICTOR', "I: variate; linear predictor"\
```

```
          'FITTEDVALUES',    "O: variate; estimate of fitted values"\
          'DERIVATIVES',     "O: variate; estimate of derivatives"\
          'OFFSET',          "I: variate; offset"\
          'NBINOMIAL';       "I: variate; denominator of binomial"\
          SET=4(yes),no;TYPE=5('variate'); \
          COMPATIBLE=*,4(!T(type,nvalues,restriction));\
          PRESENT=yes,2(no),2(yes)

GETATTRIBUTE [ATTRIBUTE=NVALUES] LINEARPREDICTOR; SAVE=!P(nobs)

CALC  FITTEDVALUES = NBINOMIAL/(1+EXP(-LINEARPREDICTOR - OFFSET))
&     DERIVATIVES = 1/FITTEDVALUES+1/(NBINOMIAL-FITTEDVALUES)
ENDPROCEDURE
```

If the `DISTRIBUTION` option is unset, the procedure will call `_GEEDISTRIBUTION` instead of using one of the various standard distributions. For a binomial error distribution `_GEEDISTRIBUTION` should be defined as follows.

```
PROCEDURE '_GEEDISTRIBUTION'
          " Calculation of variance function and deviance"
PARAMETER NAME = \
          'Y',           "I: variate; response variate"\
          'FITTEDVALUES',"I: variate; fitted values"\
          'VARIANCE',    "O: variate; variance"\
          'DEVIANCE',    "O: scalar; total deviance"\
          'NBINOMIAL';   "I: variate; denominator of binomial"\
          SET=4(yes),no;TYPE=3('variate'),'scalar','variate'; \
          COMPATIBLE=*,2(!T(type,nvalues,restriction)),*,\
                     !T(type,nvalues,restriction); \
          PRESENT=2(yes),2(no),yes

CALC VARIANCE = FITTEDVALUES*(NBINOMIAL-FITTEDVALUES)/NBINOMIAL
&    DEVIANCE = -2*LLB(Y;NBINOMIAL;(FITTEDVALUES/NBINOMIAL))
ENDPROCEDURE
```

If the `CRTYPE` option is unset, the procedure will call `_GEECORRELATION` instead of using one of the various standard correlation models. For the independence model `_GEECORRELATION` should be defined as follows. Kenward & Smith (1995b) describe how `_GEECORRELATION` should be set up for analysing repeated ordinal categorical data.

```
PROCEDURE   '_GEECORRELATION'
            " Calculation of correlation matrix

              For SANDWICH = NO
                  input is the R matrix as for UNSPECIFIED
                  output is the desired R matrix.
              For SANDWICH = YES
                  input is the (Y-MU)*T(Y-MU) matrix
                  output is the desired modified (Y-MU)*T(Y-MU) matrix.
              N.B. For the normal distribution both the input and
                   output R's should be variance/covariance matrices
                   not correlation matrices."
OPTION      NAME = \
            'CONSTANT', "I: text; how to treat constant (estimate,
                        omit); default e"\
            'SANDWICH'; "I; text; whether the sandwich central matrix
                        product or not) (no,yes); default no"\
            MODE=2(T); NVALUES=2(1); SET=yes;\
            VALUES=!T(ESTIMATE,OMIT),!T(NO,YES); \
            DEFAULT=!T(ESTIMATE),!T(NO);
PARAMETER NAME = \
          'CORRELATIONS',"I/O: matrix; the correlation matrix"\
          'ESTIMATES',   "I: variate; estimates of parameters in
                         model"\
          'Y',           "I: variate; response variate"\
          'RESIDUALS',   "I: variate; residuals"\
```

```
          'FITTEDVALUES',"I: variate; fitted values"\
          'TIME',        "I: variate; times of repeated measures"\
          'MARKER',      "I: factor; identifier of subject or outcome"\
          'DISTRIBUTION',"I: text; identifier of distribution"\
          'SCALEFACTOR', "I: text; scalefactor option in use"\
          'SFVALUE';     "I: scalar; value of scalefactor if FIXED"\
          SET=10(yes);DECLARED=10(yes); \
          TYPE='symmetric',5('variate'),'factor',2('text'),'scalar'; \
          PRESENT=9(yes),no

GETATTRIBUTE [ATTRIBUTE=NVALUES] ESTIMATES; SAVE=!P(ncol)
 &           [ATTRIBUTE=NROWS] CORRELATIONS; SAVE=!P(ntime)

DIAGONALMATRIX [ROWS=ntime;MODIFY=yes] done,wkdm; \
               VALUES=!(#ntime(1)),*

CALC const = 'ESTIMATE' .IN. CONSTANT
 &   sandw = 'NO' .IN. SANDWICH

IF sandw
"
  SCALEFACTOR is as in GEE i.e. FIXED means fixed to SFVALUE
  CONSTANT means the scalefactor is estimated but constant
  across time, and VARYTIME means the scalefactor is estimated
  and varies across time.

  The variate TIME in this PROCEDURE represents the 1...ntime
  distinct times, it is not a FACTOR of length nobs as in GEE.
  It is the levels of the parameter TIME of GEE.
"
  IF DISTRIBUTION.EQS.'NORMAL'
    IF SCALEFACTOR.NES.'VARYTIME'
      IF SCALEFACTOR.EQS.'FIXED'
        CALC wkdm = SFVALUE
      ELSE
        CALC wkdm = TRACE(CORRELATIONS)/ntime
      ENDIF
    ELSE
      CALC wkdm = CORRELATIONS
    ENDIF
    CALC CORRELATIONS = 0 + wkdm
  ELSE
    CALC CORRELATIONS = done
  ENDIF
ENDIF
ENDPROCEDURE
```

If `LINK`, `DISTRIBUTION` or `CRTYPE` are unset, but no user routines are given for `_GEEINIT`, `_GEELINK`, `_GEEDISTRIBUTION` and `_GEECORRELATION`, then those given here (for logit link, binomial error distribution and independence) will be used.

This is a complicated algorithm and some examples may take a while to run. If necessary, however, you can set option `PRINT=monitoring` to see what is happening.

3.5.13 Zero-inflated regression models

`R0INFLATED` procedure

Fits zero-inflated regression models to count data with excess zeros (D.A. Murray).

Options

`PRINT` = *string token*	Controls printed output (`model`, `summary`, `estimates`, `fittedvalues`, `monitoring`); default `mode`, `summ`, `esti`
`DISTRIBUTION` = *string token*	Distribution of response variable (`poisson`, `binomial`,

	negativebinomial); default pois
METHOD = *string token*	Method used for model fitting (em, conditional); default em
CONSTANT = *string token*	How to treat constant for count state (estimate, omit); default esti
ZCONSTANT = *string token*	How to treat constant for zero-inflation state (estimate, omit); default esti
XTERMS = *formula*	List of explanatory variates and factors, or model formula for count state of model
ZTERMS = *formula*	List of explanatory variates and factors, or model formula for zero-inflation state of model
WEIGHTS = *variate*	Variate of weights for weighted zero-inflated regression (EM model only)
OFFSET = *variate*	Offset variate to be used in the model (EM model only)
XGROUPS = *factor*	Absorbing factor defining the groups for within-groups regression for the count state model (EM model only)
ZGROUPS = *factor*	Absorbing factor defining the groups for within-groups regression for the zero-inflation state model (EM model only)
MAXCYCLE = *scalar*	Maximum number of iterations for EM algorithm; default 100
TOLERANCE = *scalar* or *variate*	Convergence criteria for EM algorithm, k and in the generalized linear models; default !(1.E-4, 1.E-4, 1.E-4)
ZPARAMETERIZATION = *string token*	Parameterization of the probability of the zero-inflation model (zero, nonzero): if unset, zero is used for the EM model and nonzero for the conditional model

Parameters

Y = *variates*	Response variate
NBINOMIAL = *scalars* or *variates*	Total numbers for DISTRIBUTION=binomial
RESIDUALS = *variates*	Saves the simple residuals
FITTEDVALUES = *variates*	Saves the fitted values
ESTIMATES = *variates*	Saves the estimates of the parameters
SE = *variates*	Saves the standard errors of the estimates
RSAVE = *identifiers*	Saves the regression structure for the final generalized model fitted for the count model
ZSAVE = *identifiers*	Saves the regression structure for the final binomial regression fitted for the zero-inflation model

Zero-inflated regression models are useful when you have count data with too many zeros. R0INFLATED allows the data to be modelled using two different approaches, according to the setting of the METHOD option.

The first possibility (METHOD=em) is to fit a *zero-inflated Poisson regression model* (ZIP), a *zero-inflated binomial regression model* (ZIB) or a *zero-inflated negative binomial regression model* (ZINB) using an EM algorithm (Lambert 1992). In this analysis, the response variable of counts is assumed to be distributed as a mixture of a distribution (such as Poisson) and a degenerate distribution at zero. In these models, a generalized linear model with a Poisson or negative binomial distribution and log link, or with a binomial distribution and logit link, is used for the count model. A generalized linear model with a binomial distribution and logit link is

used for the zero-inflation model.

The zero-inflated Poisson (mixture) regression model has the distribution

$$\begin{aligned}\Pr(Y=y) &= \omega + (1-\omega)\times\exp(-\lambda) \quad \text{for } y=0\\ &= (1-\omega)\times\exp(-\lambda)\times\lambda^{y}/y! \quad \text{for } y>0\end{aligned}$$

where λ and ω are given by the following models

$$\log(\lambda) = \mathbf{X}\,\boldsymbol{\beta}$$
$$\log(\omega/(1-\omega)) = \mathbf{Z}\,\boldsymbol{\alpha}$$

where $\mathbf{X}$ and $\mathbf{Z}$ are covariate matrices and $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ are vectors of unknown parameters.

The zero-inflated binomial (mixture) regression model has the distribution

$$\begin{aligned}\Pr(Y=y) &= \omega + (1-\omega)\times(1\text{-p})^{n} \quad \text{for } y=0\\ &= (1-\omega)\times p^{y}\times(1-p)^{n-y}\times n!/(y!\times(n-y!)) \quad \text{for } y>0\end{aligned}$$

where p and ω are given by the following models

$$\log(p/(1-p)) = \mathbf{X}\,\boldsymbol{\beta}$$
$$\log(\omega/(1-\omega)) = \mathbf{Z}\,\boldsymbol{\alpha}$$

The zero-inflated negative binomial (mixture) regression model has the distribution

$$\begin{aligned}\Pr(Y=y) &= \omega + (1-\omega)\times(1+\lambda\times k)^{-(1/k)} \quad \text{for } y=0\\ &= (1-\omega)\times\Gamma(y+1/k)/(y!\times\Gamma(1/k))\\ &\quad\times(1+\lambda\times k)^{-(y+1/k)} \quad \text{for } y>0\end{aligned}$$

where λ and ω are given by the same models as for the Poisson distribution, and k is the extra-variation parameter in the negative binomial distribution.

The maximum likelihood estimates for $\boldsymbol{\beta}$, $\boldsymbol{\alpha}$ and k are obtained using an EM algorithm (Lambert 1992). The standard errors for the parameter estimates are derived using the incomplete data observed information matrix as proposed by Lambert (1992). The default parameterization for the mixture models estimates ω, the probability of excess zeros. You can use the `ZPARAMETERIZATION` option to change the parameterization to estimate ω', the probability that an observation is generated through the distribution instead ($\omega' = 1-\omega$).

The alternative (`METHOD=conditional`) is to fit the *conditional model* of Welsh *et al.* (1996), which assumes that the data are in one of two states: a state where zeros are observed, or a state where counts are recorded. A binomial model with a logit link is used for the zero state. A truncated Poisson, truncated binomial or truncated negative binomial model is used for the count state.

In the Poisson case of the conditional model, y has a truncated Poisson distribution (λ). So the probability model is

$$\begin{aligned}\Pr(Y=y) &= \omega \quad \text{for } y=0\\ &= (1-\omega)\times\exp(-\lambda)\times\lambda^{y})/\{\,y!\times(1-\exp(-\lambda)\,\} \quad \text{for y}>0\end{aligned}$$

where λ and ω are given by the following models

$$\log(\lambda) = \mathbf{X}\,\boldsymbol{\beta}$$
$$\log(\omega/(1-\omega)) = \mathbf{Z}\,\boldsymbol{\alpha}$$

In the truncated binomial case, y has a truncated binomial distribution. So the probability model is

$$\begin{aligned}\Pr(Y=y) &= \omega \quad \text{for } y=0\\ &= (1-\omega)\times p^{y}\times(1-p)^{n-y}/(1-(1-p)^{n})\\ &\quad\times n!/(y!\times(n-y!)) \quad \text{for } y>0\end{aligned}$$

where p and ω are given by the following models

$$\log(p/(1-p)) = \mathbf{X}\,\boldsymbol{\beta}$$
$$\log(\omega/(1-\omega)) = \mathbf{Z}\,\boldsymbol{\alpha}$$

In the negative binomial case, y has a truncated negative binomial (λ, k). So the probability model is

$$\begin{aligned}\Pr(Y=y) &= \omega \quad \text{for } y=0\\ &= (1-\omega)\times\Gamma(y+1/k)/(y!\times\Gamma(1/k))\\ &\quad\times(1+k\times\lambda)^{-(y+1/k)}\end{aligned}$$

$$\times (1 - (1 + k \times \lambda)^{-1/k})^{-1}, \quad \text{for } y>0$$

where λ and ω are given by the same models as for the Poisson distribution, and k is the extra-variation parameter in the negative binomial distribution.

The truncated Poisson model is fitted using an iteratively re-weighted least squares algorithm (see Welsh *et al.* 1996). The truncated binomial and negative binomial models are fitted using FITNONLINEAR.. The default parameterization for the mixture models estimates ω' $(=1-\omega)$, the probability of detecting at least one observation given that there is at least one observation, as in Welsh *et al*. (1996). You can use the ZPARAMETERIZATION option to change the parameterization to estimate ω, the probability of detecting a zero observation, instead.

The response variable is supplied, in a variate, using the Y parameter. The NBINOMIAL parameter must also be set when DISTRIBUTION=binomial, to give the number of binomial trials for each unit. The XTERMS and ZTERMS options each specifies a formula, to describe the count model and the zero-inflation model respectively. The CONSTANT and ZCONSTANT options control whether a constant parameter is included in the count and zero-inflation models.

The DISTRIBUTION option specifies the distribution for the count model. Note that a log link is always used for the count model with the Poisson and negative binomial distributions, and a logit link is used with the binomial distribution.

The XGROUPS and ZGROUPS options can specify factors whose effects you want to eliminate from the count or zero-inflation state respectively, before any regression is fitted. This method of elimination is sometimes called absorption. (See the GROUPS option of the MODEL directive.) It gives less information than you would get if you included the factor explicitly in the model. For example, no standard errors are produced. However, it saves space and time when data from many different groups are to be modelled. These options are only available for the EM model.

The ESTIMATES and SE parameters save the parameter estimates and their standard errors. R0INFLATED puts them into variates, using the same order as in the display produced by the PRINT option. The simple residuals and the fitted values can be saved using the RESIDUALS and FITTEDVALUES parameters.

The RSAVE and ZSAVE parameters allow you to specify identifiers for the regression save structures for the count and zero-inflation states of the model. These structures store the final state of the regression models fitted. Note that the standard errors for the parameter estimates in the regression save structures will not be correct and should instead be obtained using the SE parameter or by the R0KEEP procedure.

For the mixture models, the WEIGHTS option can specify a variate holding weights for each unit, and the OFFSET option allows you to include an offset (i.e. a variable in the regression model with a regression coefficient fixed at one).

The PRINT option controls printed output, with settings:

model	gives a description of the model, including response and explanatory variates for count and zero-inflation models;
summary	displays minus twice log-likelihood, the Akaike information coefficient (AIC) and the Schwarz (Bayesian) information coefficient (BIC or SIC);
estimates	gives the estimates of the parameters in the model with standard errors based on the asymptotic variance-covariance matrix derived from the inverse of the observed Fisher information matrix;
fittedvalues	displays a table of unit labels, values of response variate, fitted values and residuals;
monitoring	displays monitoring information of the iterative algorithm.

The iterative process for the EM algorithm is controlled by the MAXCYCLE option which defines the maximum number of cycles, and the TOLERANCE option which sets convergence criteria. The EM algorithm cycle stops when successive values of the log-likelihood are within

a tolerance set by the first element of the TOLERANCE option. The second and third elements of TOLERANCE control the convergence criterion for the aggregation parameter (k) for the negative binomial model and for the generalized linear model, respectively.

Example 3.5.13 fits a conditional model with a truncated negative binomial distribution for the non-zero counts to the data on Leadbeater's possums in Welsh *et al*. (1996).

Example 3.5.13

```
  2  VARIATE    [NVALUES=151] no_lb,stags
  3  READ       no_lb

   Identifier   Minimum      Mean   Maximum    Values   Missing
        no_lb    0.0000     1.371     10.00       151         0    Skew

 10  READ       stags

   Identifier   Minimum      Mean   Maximum    Values   Missing
        stags    0.0000     7.238     31.00       151         0    Skew

 17  CALCULATE  lstags = log(stags+1)
 18  R0INFLATED [PRINT=mod,sum,est; METHOD=conditional; DIST=negative; \
 19             ZTERMS=lstags; XTERMS=lstags] no_lb

Conditional model
=================

    Response variate: no_lb
        Distribution: Truncated negative binomial
                Link: Log
        Fitted Terms: Constant + lstags
Zero-inflation terms: Constant + lstags


Summary of analysis
-------------------

-2 x log-likelihood: 413.1

Binary model residual deviance: 187.0 on 149 d.f.
Count model residual deviance:  64.03 on 53 d.f.


Estimates of count model parameters
-----------------------------------

Parameter    estimate        s.e.
k              0.1189      0.0935
Constant       0.4988      0.3042
lstags         0.3411      0.1270

Estimates of binary model parameters
------------------------------------

Parameter    estimate        s.e.
Constant       -2.079      0.5115
lstags          0.822      0.2485
```

R0KEEP procedure

Saves information from a zero-inflated regression model for count data with excess zeros fitted by R0INFLATED (D.A. Murray).

Options

RESIDUALS = *variate*	Saves the simple residuals
FITTEDVALUES = *variate*	Saves the fitted values
ESTIMATE = *variate*	Saves the parameter estimates

SE = *variate*	Saves the standard errors of the parameter estimates
VCOVARIANCE = *symmetric matrix*	Saves the variance-covariance matrix of estimates for the ZIP and ZINB models
XFITTEDVALUES = *variate*	Saves the fitted values for the count model
XSEFITTEDVALUES = *variate*	Saves the standard errors of the fitted values for the fitted values of the count model
ZFITTEDVALUES = *variate*	Saves the fitted values for the zero model
ZSEFITTEDVALUES = *variate*	Saves the standard errors of the fitted values for the fitted values of the zero model
_2LOGLIKELIHOOD = *scalar*	Saves −2 times the log-likelihood
AIC = *scalar*	Saves the Akaike information coefficient
SIC = *scalar*	Saves the Schwarz (Bayesian) information coefficient

No parameters

R0KEEP allows you to copy information into Genstat data structures from a model that has been fitted by R0INFLATED.

The RESIDUALS and FITTEDVALUES options save the simple residuals and the fitted values. The ESTIMATES and SE options save the parameter estimates and their standard errors. The VCOVARIANCE option saves the variance-covariance matrix of estimates from either a ZIP or ZINB model. The ZFITTEDVALUES and ZSEFITTEDVALUES options save the fitted values and standard errors of fitted values for the zero state. Similarly, the XFITTEDVALUES and XSEFITTEDVALUES options save the fitted values and standard errors of fitted values for the count state. The _2LOGLIKELIHOOD option saves −2 times the log-likelihood, and the AIC and SIC options save the Akaike and Schwarz (Bayesian) information coefficients respectively.

3.6 Generalized least-squares

You can specify a general weight matrix for use in linear regression, supplied as a symmetric matrix using the WEIGHTS option of the MODEL directive. The regression problem is then described as a *generalized least-squares* problem. Similarly, the WEIGHTS option of the FSSPM directive can also be set to a symmetric matrix.

As an example, we fit a model to data measured on a transect, allowing for correlation between the neighbouring, closely-spaced, observations. The measurements are of Zinc content in a polluted soil, taken across the edge of the polluted region where soil cultivation has spread the metal. First, here is the result of fitting a quartic polynomial to the change in Zinc level.

Example 3.6a

```
   2  FILEREAD [PRINT=summary; NAME='DIFFUSE.DAT'] X,Zinc

Summary
-------

The file DIFFUSE.DAT is assumed to contain 2 structure(s), with one value for
each structure on each record.

The file contains 65 values for each of the following structures:

  Identifier      Type   Missing
           X   variate         0
        Zinc   variate         0

   3  " Fit polynomial model without weighting."
   4  MODEL    [RMETHOD=simple] Zinc
   5  FIT      [PRINT=estimates] POL(X; 4)
```

```
Regression analysis
===================

Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(60)
Constant         246.49         2.86     86.09
X Lin           -25.294        0.878    -28.82
X Quad           -2.003        0.205     -9.79
X Cub             0.4338      0.0407     10.66
X Quart          -0.01608     0.00196     -8.19

   6  RGRAPH
```

This fits and plots the model shown in Figure 3.6. As Figure 3.6 shows, it fits the data well in the range sampled, but would not be a sensible model for extrapolation outside the measured region because of the nature of polynomial models. It may be better to fit a smoothing spline, or to use a Fourier curve derived from the equations for diffusion. However, it serves here to show the effect of taking account of the evident correlation between successive observations. This correlation can be estimated from the simple residuals using the CORRELATE directive, as in the Example 3.6b.

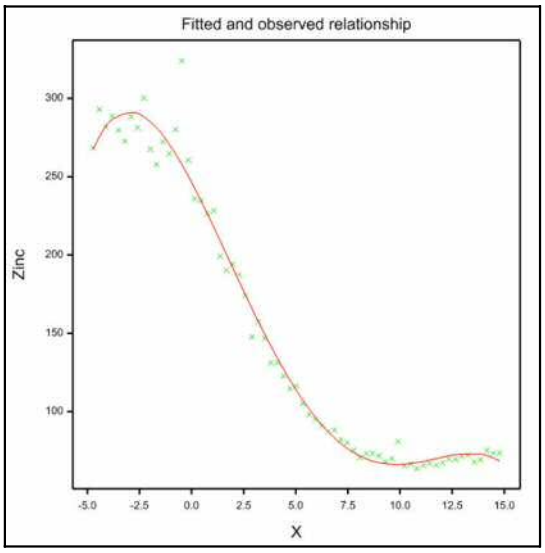


Figure 3.6

Example 3.6b

```
   7  RKEEP     RESIDUALS=Residuals
   8  CORRELATE [PRINT=auto; MAXLAG=5] Residuals

Correlations
------------

         Unit          ACF
            1        1.000
            2        0.286
            3       -0.065
            4       -0.068
            5       -0.175
            6        0.050
```

The estimate of correlation between neighbouring points in the transect is about 0.3, so we use this information to re-fit the model, assuming a simple correlation structure, with all neighbouring points equally correlated. The correlation matrix between all the units has 1.0 on the diagonal, 0.3 just below or above the diagonal, 0.09 (=0.3^2) below or above this, and so on. The weight matrix is the inverse of this correlation matrix, so we could use the INVERSE function to form it. However, the form of the inverse of a matrix with this pattern is well known, and is much more efficiently calculated direct: for correlation r it has the value $1+r^2$ on the diagonal, except for the first and last rows which are 1; $-r$ below and above the diagonal; and

0 elsewhere. So we form the weight matrix directly. The correlation clearly does not affect the parameter estimates much, but the standard errors are larger, by about 30%. This is a well known effect of serial correlation; see, for example, Watson & Hannan (1956).

Example 3.6c

```
  9  " Define weights in terms of correlation R=0.3."
 10  SCALAR    R; VALUE=0.3
 11  CALCULATE N = NVALUES(X)
 12  &         N1 = N-1
 13  SYMMETRIC [ROWS=N] W
 14  CALCULATE W = 0
 15  &         W$[1,N; 1,N] = 1
 16  &         W$[2...N1; 2...N1] = 1 + R**2
 17  &         W$[2...N; 1...N1] = -R
 18  " Fit polynomial model with correlation fixed at R=0.3."
 19  MODEL [WEIGHTS=W] Zinc
 20  FIT   [PRINT=estimates] POL(X; 4)

Regression analysis
===================

Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(60)
Constant         246.43         3.86     63.84
X Lin            -25.24         1.18    -21.40
X Quad           -1.996        0.263     -7.58
X Cub            0.4307       0.0525      8.20
X Quart        -0.01590      0.00253     -6.28
```

A general symmetric weight matrix is also allowed with generalized linear models, and with generalized nonlinear models (3.5). However, the interpretation to be put on the resulting analysis is an open question, since the correlation is being applied on the scale of the linear predictor rather than on the scale of the observations themselves. Matrices of weights cannot be used with the `FITCURVE` or `FITNONLINEAR` directives.

3.7 Standard nonlinear curves

This section describes various standard nonlinear curves that can be fitted using the `FITCURVE` directive (or, in Genstat *for Windows*, using the Standard Curves menu). These standard curves have been found useful in many applications of statistics. They are fitted by a modified Newton method of maximizing the likelihood, using stable forms of parameterization (Ross 1990). Facilities for fitting other user-defined curves are described in 3.8.

The method Genstat uses to fit curves is iterative, using a search procedure to find parameter values that maximize the likelihood. The search is much quicker when Genstat knows the shape of the curve; thus, fitting a curve by the methods in this section is more efficient than using those in 3.8. With standard curves you will not usually need to supply starting values for the search, nor to control the course of the search; in contrast, you will nearly always have to do these things when you are fitting non-standard curves. For more information about nonlinear curve fitting, see Ratkowsky (1983, 1990), Ross (1990), or Seber & Wild (1989).

Example 3.7 fits the exponential curve

$$y_i = \alpha + \beta \rho^{x_i} + \varepsilon_i$$

to the relationship between length and age of dugongs. At line 8 the RGRAPH procedure is used to produce the graph of the fitted curve shown in Figure 3.7.

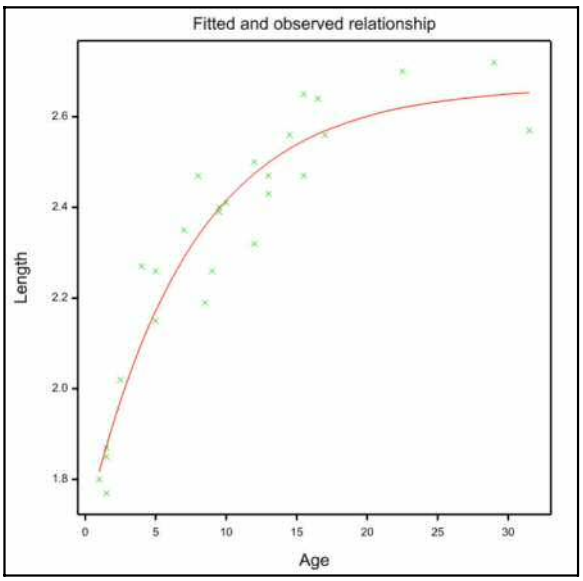


Figure 3.7

Example 3.7

```
   2  " Asymptotic regression (exponential curve) of length
  -3    on age of dugongs. Data from Ratkowsky (1983) p.101."
   4  OPEN     '%GENDIR%/Examples/GuidePart2/Dugong.dat'; CHANNEL=2
   5  READ     [PRINT=data; CHANNEL=2] Age,Length

     1   1.0 1.80   1.5 1.85   1.5 1.87   1.5 1.77   2.5 2.02
     2   4.0 2.27   5.0 2.15   5.0 2.26   7.0 2.35   8.0 2.47
     3   8.5 2.19   9.0 2.26   9.5 2.40   9.5 2.39  10.0 2.41
     4  12.0 2.50  12.0 2.32  13.0 2.43  13.0 2.47  14.5 2.56
     5  15.5 2.65  15.5 2.47  16.5 2.64  17.0 2.56  22.5 2.70
     6  29.0 2.72  31.5 2.57
   6  CLOSE    2
   7  MODEL    Length
   8  FITCURVE [CURVE=exponential; FPROBABILITY=yes] Age

Nonlinear regression analysis
=============================

 Response variate: Length
      Explanatory: Age
     Fitted Curve: A + B*(R**X)
      Constraints: R < 1


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       2       1.7745     0.887257    114.02  <.001
Residual        24       0.1868     0.007782
Total           26       1.9613     0.075434

Percentage variance accounted for 89.7
Standard error of observations is estimated to be 0.0882.

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
            1       1.8000        0.26
           26       2.7200        0.26
           27       2.5700        0.30
```

```
Estimates of parameters
-----------------------

Parameter         estimate         s.e.
R                   0.8735       0.0223
B                  -0.9725       0.0647
A                   2.6666       0.0579

   9  RGRAPH
```

3.7.1 The FITCURVE directive

FITCURVE directive

Fits a standard nonlinear regression model.

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring); default mode, summ, esti
CURVE = *string token*	Type of curve (exponential, dexponential, cexponential, lexponential, logistic, glogistic, gompertz, ldl, qdl, qdq, fourier, dfourier, gaussian, dgaussian, emax, gemax); default expo
SENSE = *string token*	Sense of curve (right, left); default righ
ORIGIN = *scalar*	Constrained origin; default *
NONLINEAR = *string token*	How to treat nonlinear parameters between groups (common, separate); default comm
CONSTANT = *string token*	How to treat the constant (estimate, omit); default esti
FACTORIAL = *scalar*	Limit for expansion of model terms; default as in previous TERMS statement, or 3 if no TERMS given
POOL = *string token*	Whether to pool ss in accumulated summary between all terms fitted in a linear model (yes, no); default no
DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality, vertical); default *
FPROBABILITY = *string token*	Printing of probabilities for variance ratios (yes, no); default no
SELECTION = *string tokens*	Statistics to be displayed in the summary of analysis produced by PRINT=summary (%variance, %ss, adjustedr2, r2, seobservations, dispersion, %cv, %meandeviance, %deviance, aic, bic, sic); default %var, seob

Parameter

formula	Explanatory variate, list of variate and factor, or variate*factor

The parameter of FITCURVE can be set just to the variate that supplies the x-values for the curve, if you simply want to fit a single curve. You can also include a factor if you want to fit separate curves for different groups of the observations: these facilities for *parallel curve analysis* are described in 3.7.3.

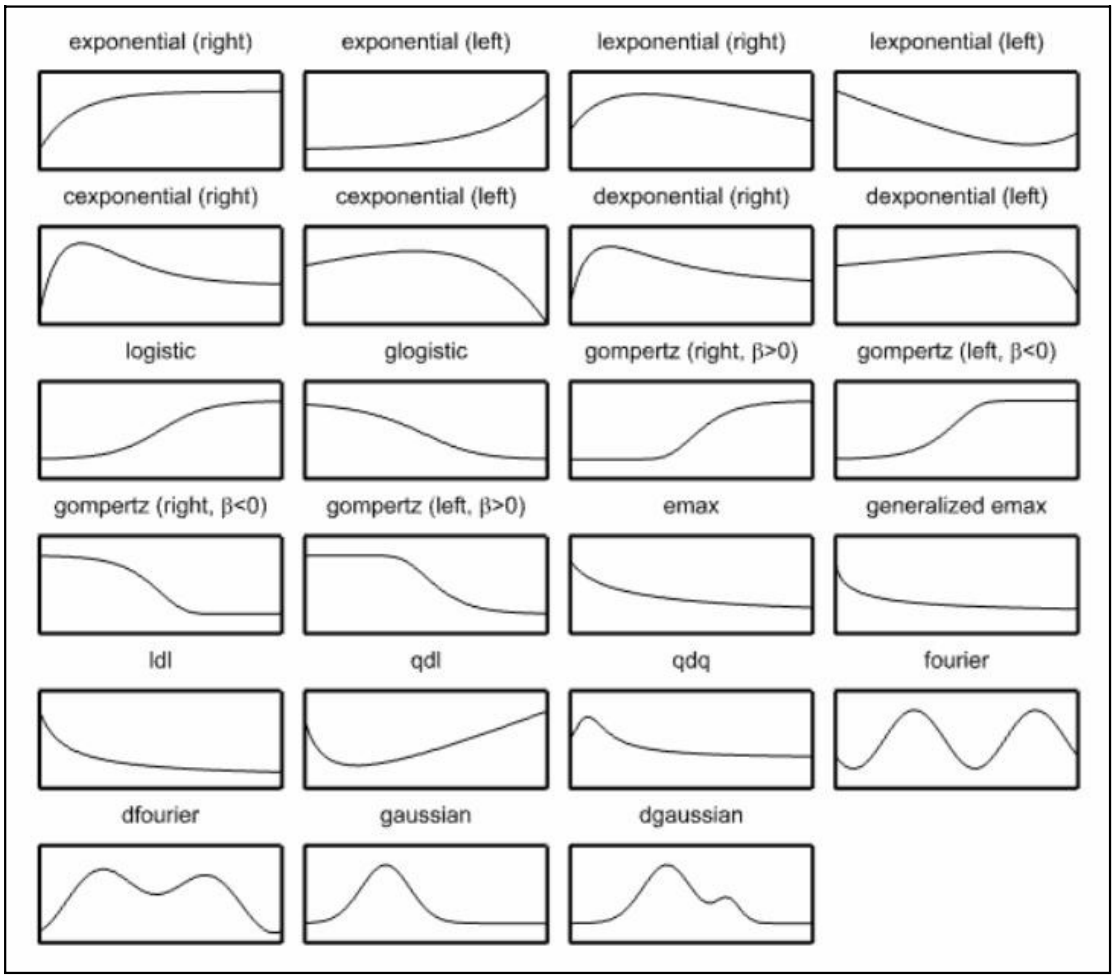


Figure 3.7.1

The CURVE option specifies which of the standard curves is to be fitted. For some of these, the SENSE option lets you choose between alternative forms. Figure 3.7.1 shows the shapes of representative curves of each type, although you should be aware that several of the curves, particularly the rational functions, can exhibit a wide variety of shapes as their parameters vary. Before describing the curves in detail, here is a list for convenient reference:

Exponential

exponential $y_i = \alpha + \beta \rho^{x_i} + \varepsilon_i$

dexponential $y_i = \alpha + \beta \rho^{x_i} + \gamma \sigma^{x_i} + \varepsilon_i$

cexponential $y_i = \alpha + (\beta + \gamma x_i) \rho^{x_i} + \varepsilon_i$

lexponential $y_i = \alpha + \beta \rho^{x_i} + \gamma x_i + \varepsilon_i$

Logistic

logistic $$y_i = \alpha + \frac{\gamma}{1 + \exp(-\beta(x_i-\mu))} + \varepsilon_i$$

glogistic $$y_i = \alpha + \frac{\gamma}{(1+\tau\ \exp(-\beta(x_i-\mu)))^{\tau^{-1}}} + \varepsilon_i$$

gompertz $$y_i = \alpha + \gamma\ \exp(-\exp(-\beta(x_i-\mu))) + \varepsilon_i$$

emax $$y_i = \alpha + \frac{\gamma}{1 + \exp(-\beta(\log(x_i)-\mu))} + \varepsilon_i$$

gemax $$y_i = \alpha + \frac{\gamma}{(1+\tau\ \exp(-\beta(\log(x_i)-\mu)))^{\tau^{-1}}} + \varepsilon_i$$

Rational functions

ldl $$y_i = \alpha + \frac{\beta}{1 + \delta\ x_i} + \varepsilon_i$$

qdl $$y_i = \alpha + \frac{\beta}{1 + \delta\ x_i} + \gamma\ x_i + \varepsilon_i$$

qdq $$y_i = \alpha + \frac{\beta + \gamma\ x_i}{1 + \delta\ x_i + \eta\ x_i^2} + \varepsilon_i$$

Fourier

fourier $$y_i = \alpha + \beta\ \sin\left(\frac{2\pi(x_i-\eta)}{\omega}\right) + \varepsilon_i$$

dfourier $$y_i = \alpha + \beta\ \sin\left(\frac{2\pi(x_i-\eta)}{\omega}\right) + \gamma\ \sin\left(\frac{4\pi(x_i-\varphi)}{\omega}\right) + \varepsilon_i$$

Gaussian

gaussian $$y_i = \alpha + \frac{\beta}{\sqrt{2\pi\sigma^2}}\ \exp\left(\frac{-(x_i-\mu)^2}{2\sigma^2}\right) + \varepsilon_i$$

dgaussian $$y_i = \alpha + \frac{\beta}{\sqrt{2\pi\sigma^2}}\ \exp\left(\frac{-(x_i-\mu)^2}{2\sigma^2}\right) + \frac{\gamma}{\sqrt{2\pi\sigma^2}}\ \exp\left(\frac{-(x_i-\nu)^2}{2\sigma^2}\right) + \varepsilon_i$$

The four exponential curves each arise as solutions of linear ordinary differential equations. These represent processes that increase exponentially with time, for example, or that increase with a law of diminishing returns (that is, for which the rate of increase decreases with time).

The default setting of the `CURVE` option is `exponential`, corresponding to the "asymptotic regression" or Mitscherlich curve. An equivalent form of the equation shown above for this curve is

$$y_i = \alpha + \beta\ \exp(-\kappa\ x_i) + \varepsilon_i$$

where $\rho = \exp(-\kappa)$. The form involving ρ is used in Genstat to avoid problems with large values of κ. The model has only one nonlinear parameter, ρ, which defines the rate of exponential

increase or decrease. FITCURVE estimates the other parameters by linear regression at each stage of an iterative search for the best estimate of ρ. The values of the explanatory variate are automatically scaled to avoid any computational problems near the boundary of the allowed values of ρ. By default, ρ is restricted to the range $0<\rho<1$, giving a curve corresponding to the law of diminishing returns. The alternative is $\rho>1$, which can be requested by setting the SENSE option to left: for all the exponential curves, SENSE=left corresponds to a curve whose asymptote is to the left – that is, as X decreases to $-\infty$. If Genstat finds that a better fit is obtained by the opposite sense to the one specified, the sense is reversed and a warning is printed. The parameter α is the asymptote – to the right if $\rho<1$ and to the left if $\rho>1$; β is the range of the curve between the value at $X=0$ and the asymptote.

The double exponential curve also has two forms: you can choose either $0<\rho<1$ and $0<\sigma<1$ or $\rho>1$ and $\sigma>1$, by using the SENSE option as for the exponential curve. The fitting process is unlikely to find a satisfactory solution for this curve unless there are enough data to estimate both components separately: there should be at least four points for which the fast component is larger than the slow component; the fast component corresponds to the smaller of ρ and σ when SENSE=right, or to the larger of ρ and σ when SENSE=left.

Two limiting cases of the double exponential are provided as special curves. The critical exponential curve can take a variety of shapes like the double exponential, whereas the line-plus-exponential curve is an exponential curve with a non-horizontal asymptote. Again here, the constraint on the parameter ρ depends on the setting of the SENSE option as for the exponential curve.

Another type of standard curve is sigmoid and monotonic, and is often used to model the growth of biological subjects. There are five types of these growth curves in Genstat, each a logistic of some sort. The first type is the generalized logistic without any constraints. In the equation above, α is one asymptote, to the right or to the left according to whether β is positive or negative; μ is the point of inflexion for the explanatory variable; β is a slope parameter; τ is a power-law parameter; and $\alpha+\gamma$ is the other asymptote. To fit this curve you need data for the steep central part and for both flat parts.

There are two special cases of the generalized logistic. The ordinary logistic curve is sometimes known as the autocatalytic or inverse exponential curve. The same curve can be rewritten in several different forms, so you should be alert for concealed equivalences of apparently different curves: otherwise you might be tempted to use FITNONLINEAR, which would be less efficient. The other special case is the Gompertz curve. It is non-symmetrical about the inflexion, $X=\mu$, and has asymptotes at $Y=\alpha$ and $Y=\alpha+\gamma$.

You can also fit these three growth curves to data in which Y decreases as X increases. For the logistic and generalized logistic curves, you are not allowed to constrain the sense of the curve by the SENSE option. This is because the sense depends on both the parameters β and γ. In fact, the logistic curve with parameters α, β, γ and μ is the same as the logistic curve with parameters $(\alpha+\gamma)$, $-\beta$, $-\gamma$ and μ; Genstat will report only one of the two possible versions. For the Gompertz curve, you can set SENSE=left to specify the upside-down Gompertz curve corresponding to $\gamma<0$; otherwise γ is constrained to be positive. When the sign of γ is changed for a response Y that increases with X, the sign of β will also change so that the curve remains an ascending one, and similarly for descending curves. All four possible shapes are shown in Figure 3.7.1. The interpretation of SENSE=left thus depends on the shape of the data; for ascending curves it means that the asymptote is reached more slowly to the left than to the right, but for descending curves it means the opposite.

The final two sigmoid curves, Emax and generalized Emax, are similar to the logistic and generalized logistic except that their equations involve $\log(x)$ instead of x. They are usually used to model decreasing relationships with the parameter β in the equation negative, but Genstat will allow increasing relationships with these curves too.

The three rational functions are ratios of polynomials. The linear-divided-by-linear curve is

a rectangular hyperbola, which occurs for example as the Michaelis-Menten law of chemical kinetics. The quadratic-divided-by-linear curve is a hyperbola with a non-horizontal asymptote. The quadratic-divided-by-quadratic curve is a cubic curve having an asymmetric maximum falling to an asymptote. The SENSE option is ignored for all three rational functions. These curves can have vertical asymptotes at finite values of the explanatory variable. A message is printed to inform you about the asymptotes; such messages can be switched off by setting NOMESSAGE=vertical in the FITCURVE command.

Fourier curves are trigonometric functions, involving the sine function in Genstat's implementation, used to model periodic behaviour. Sometimes the wavelength or period ω is a known constant, such as 2π radians (or 360 degrees), 24 hours, or 12 months; the models are then linear and should be fitted by linear regression using the FIT directive, instead of by FITCURVE. For example, the simple Fourier curve with fixed ω can be expressed in the form:

$$y_i = \alpha + \beta \sin\left(\frac{2\pi x_i}{\omega}\right) + \gamma \cos\left(\frac{2\pi x_i}{\omega}\right) + \varepsilon_i$$

and so can be fitted by statements like the following.

```
CALCULATE X1 = SIN(2*C('pi')*X/W)
& X2 = COS(2*C('pi')*X/W)
FIT X1,X2
```

The parameters β and γ are the amplitudes of the components of the curve. The SENSE option is ignored for Fourier curves.

The Gaussian curve is a bell-shaped curve like the Normal probability density. The double Gaussian is a sum of two overlapping curves of this type, and arises for example in spectography. The parameter α is usually called the *background*, and the parameters μ and ν are the peaks. The parameter σ is the standard deviation: for the double Gaussian, Genstat can deal only with the case of equal standard deviation for the two components. The parameters β and γ represent the strength of a spectrographic signal in each component, excluding the background. The SENSE option is ignored for Gaussian curves.

The PRINT, FACTORIAL, POOL, DENOMINATOR, NOMESSAGE and FPROBABILITY options are as for FIT. The ORIGIN and CONSTANT options are described in 3.7.2, and the NONLINEAR option in 3.7.3.

3.7.2 Distributions and constraints in curve fitting

The curves available with FITCURVE can be fitted in Genstat only with the Normal likelihood. If you set some other distribution in the MODEL statement, you will get a warning message and the distribution will automatically be reset to Normal. However, you can specify a weighted Normal likelihood by providing weights with the WEIGHTS option of the MODEL directive, as for linear regression, and hence mimic other distributions. You can also supply a symmetric matrix of weights, for example to allow for covariances between units. However, if the model contains an explanatory factor, pairs of units with different factor levels must have zero covariances.

You can set the DISPERSION option if you want Genstat to use a known variance for the distribution of the response variate (3.1.1).

FITCURVE ignores the LINK and EXPONENT options of the MODEL directive, and you are not allowed to set the GROUPS option.

You can constrain the exponential and rational curves to pass through a given point. The ORIGIN option of the FITCURVE directive specifies a value for the response variate corresponding to a zero value of the explanatory variate; to specify the response for another value of the explanatory variate you would need to modify the explanatory variate beforehand. For all these standard curves except the double exponential, the supplied origin corresponds to the expression (α+β); in the double exponential it is (α+β+γ). If you constrain the origin in this way, you should probably use some form of weighting, because points near the constraint are

likely to vary less than points further away. You can get approximately log-Normal weighting by using a weight variate with values $1/(Y-\text{origin})^2$. You are not allowed to set the ORIGIN option at the same time as the CONSTANT option.

Another way of constraining the curves is by omitting the constant term – the parameter α in each case. This parameter represents the asymptote: for growth curves with parameter $\beta>0$ it represents the asymptote as $X \to -\infty$, and for those with $\beta<0$ it represents the asymptote as $X \to +\infty$. To constrain the asymptote to be other than 0, you should put the value that you require into every element of the variate in the OFFSET option of the MODEL directive. An example is the exponential curve

$$y_i = o + \beta \rho^{x_i} + \varepsilon_i$$

where o is the constant value to be supplied by the offset variate. Note that the constant cannot be omitted from the Gompertz fitted with SENSE=left.

3.7.3 Parallel curve analysis

When data are grouped, a common requirement in curve fitting is to compare curves fitted to each group. The curves can be constrained to be similar to each other to some degree, governed by restricting some of the parameters to be common to all groups. Genstat provides four levels of similarity to be specified for a single grouping factor.

If you give just a variate in the parameter of the FITCURVE directive, a single curve is fitted to all groups defined by the factor. Thus, for the data in Example 3.7.3 below, the statements

```
FACTOR [LEVELS=4; VALUES=16(1...4)] Solution
MODEL Density
FITCURVE [CURVE=logistic] Log
```

fit the model

$$y_i = \alpha + \frac{\gamma}{1 + \exp(-\beta(x_i-\mu))} + \varepsilon_i , \quad j=1...4$$

in which x_i stands for the explanatory variable (the logarithm of the dilution), y_i stands for the response variable (the optical density of the solution), and j stands for the solution number.

If you specify a variate and a factor, separate curves are fitted for each group, constrained to be parallel: that is, they differ only by a constant (the analogy of what in linear regression would be called the intercept). The statement

```
FITCURVE [CURVE=logistic] Log,Solution
```

fits

$$y_i = \alpha_j + \frac{\gamma}{1 + \exp(-\beta(x_i-\mu))} + \varepsilon_i , \quad j=1...4$$

If you include the interaction between the variate and the factor, the curves are constrained to have common nonlinear parameters, but all linear parameters are estimated separately for each group. So the statement

```
FITCURVE [CURVE=logistic] Log*Solution
```

fits

$$y_i = \alpha_j + \frac{\gamma_j}{1 + \exp(-\beta(x_i-\mu))} + \varepsilon_i , \quad j=1...4$$

You are not allowed to constrain the origin or omit the constant for curves that are constrained in either of the two ways described above.

If you set the NONLINEAR option to separate when the model includes the variate, the factor, and the interaction, Genstat estimates all the parameters independently; only the information about variability is pooled:

```
FITCURVE [CURVE=logistic; NONLINEAR=separate] Log*Solution
```

fits

$$y_i = \alpha_j + \frac{\gamma_j}{1 + \exp(-\beta_j (x_i - \mu_j))} + \varepsilon_i , \quad j=1...4$$

You can modify a model fitted by FITCURVE by using the ADD, DROP or SWITCH directives as for linear models, provided you have given an appropriate TERMS statement before the FITCURVE statement. The alterations must, however, produce a model that would be allowed in the FITCURVE directive: that is, it must contain one variate, or one variate and one factor, or one variate and one factor and their interaction. The NONLINEAR options of the ADD, DROP and SWITCH directives have the same effect as the NONLINEAR option of FITCURVE. Thus you can compare curves between groups of a factor, assessing for example whether they are parallel. The accumulated setting of the PRINT option of these directives allows you to summarize the results. Example 3.7.3 shows such *an analysis of parallelism*.

Example 3.7.3

```
   2  " Model the relationship between dilution and optical density
  -3    for four solutions. Data from Bouvier et al. (1985) p.129."
   4  READ [PRINT=data] Density

   5  1.914 1.878 1.717 1.195 0.587 0.264 0.099 0.114
   6  1.891 1.887 1.703 1.158 0.599 0.277 0.106 0.069
   7  1.876 1.830 1.608 1.099 0.513 0.236 0.096 0.074
   8  1.913 1.847 1.622 1.109 0.536 0.227 0.100 0.086
   9  1.873 1.859 1.707 1.191 0.611 0.262 0.111 0.082
  10  1.877 1.873 1.696 1.185 0.617 0.259 0.122 0.041
  11  1.897 1.800 1.495 0.915 0.417 0.203 0.068 0.047
  12  1.869 1.780 1.500 0.922 0.396 0.165 0.096 0.035 :
  13  FACTOR    [LEVELS=4; VALUES=16(1...4)] Solution
  14  VARIATE   [VALUES=(30,90,270,810,2430,7290,21870,65610)8] Dilution
  15  VARIATE   Log; EXTRA=' dilution'
  16  CALCULATE Log = LOG10(Dilution)
  17  MODEL     Density
  18  TERMS     Log*Solution
  19  FITCURVE  [PRINT=model,estimates; CURVE=logistic] Log

Nonlinear regression analysis
=============================

 Response variate: Density
      Explanatory: Log dilution
     Fitted Curve: A + C/(1 + EXP(-B*(X - M)))


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
B                   -2.816        0.139
M                   2.9973       0.0184
C                   1.8633       0.0329
A                   0.0658       0.0184

  20  ADD [PRINT=model,estimates] Solution

Nonlinear regression analysis
=============================

 Response variate: Density
      Explanatory: Log dilution
  Grouping factor: Solution, constant parameters separate
     Fitted Curve: A + C/(1 + EXP(-B*(X - M)))
```

```
Estimates of parameters
-----------------------

Parameter                      estimate         s.e.
B                               -2.8158       0.0673
M                               2.99728      0.00892
C                                 1.863
A  Solution 1                    0.1043
A  Solution 2                   0.06145
A  Solution 3                   0.09858
A  Solution 4                  -0.01149

  21  ADD [PRINT=model,estimates] Log.Solution

Nonlinear regression analysis
=============================

 Response variate: Density
      Explanatory: Log dilution
  Grouping factor: Solution, all linear parameters separate
     Fitted Curve: A + C/(1 + EXP(-B*(X - M)))


Estimates of parameters
-----------------------

Parameter                      estimate         s.e.
B                               -2.7763       0.0683
M                               3.00329      0.00906
C  Solution 1                    1.891
A  Solution 1                   0.09103
C  Solution 2                    1.866
A  Solution 2                   0.06003
C  Solution 3                    1.877
A  Solution 3                   0.09174
C  Solution 4                    1.846
A  Solution 4                 -0.003673

  22  ADD [PRINT=model,summary,estimates,accumulated; FPROBABILITY=yes;\
  23      NONLINEAR=separate]

Nonlinear regression analysis
=============================

 Response variate: Density
      Explanatory: Log dilution
  Grouping factor: Solution, all parameters separate
     Fitted Curve: A + C/(1 + EXP(-B*(X - M)))


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression      15     34.95313    2.3302090   4742.61  <.001
Residual        48      0.02358    0.0004913
Total           63     34.97672    0.5551860

Change          -6     -0.08419    0.0140311     28.56

Percentage variance accounted for 99.9
Standard error of observations is estimated to be 0.0222.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           12       1.1580       -2.39

* MESSAGE: the residuals do not appear to be random;
           for example, fitted values in the range 0.1620 to 0.4095
           are consistently smaller than observed values
           and fitted values in the range 0.0931 to 0.1140
           are consistently larger than observed values.
```

```
Estimates of parameters
-----------------------

Parameter                    estimate        s.e.
B  Solution 1                 -2.9175      0.0979
M  Solution 1                  3.0491      0.0122
C  Solution 1                  1.8622      0.0217
A  Solution 1                  0.0825      0.0127
B  Solution 2                 -2.8285      0.0967
M  Solution 2                  2.9924      0.0127
C  Solution 2                  1.8572      0.0227
A  Solution 2                  0.0693      0.0126
B  Solution 3                 -2.8693      0.0968
M  Solution 3                  3.0783      0.0125
C  Solution 3                  1.8560      0.0220
A  Solution 3                  0.0655      0.0131
B  Solution 4                 -2.7433      0.0951
M  Solution 4                  2.8610      0.0134
C  Solution 4                  1.8822      0.0245
A  Solution 4                  0.0487      0.0121


Accumulated analysis of variance
--------------------------------

Change                       d.f.          s.s.          m.s.       v.r.  F pr.
+ Log                           3    34.7306063    11.5768688   23562.09  <.001
+ Solution                      3     0.1356465     0.0452155      92.03  <.001
+ Log.Solution                  3     0.0026953     0.0008984       1.83  0.155
+ Separate nonlinear            6     0.0841868     0.0140311      28.56  <.001
Residual                       48     0.0235841     0.0004913

Total                          63    34.9767190     0.5551860
```

The use of the constraint to fit the common nonlinear parameter means that `FITCURVE` is then unable to provide standard errors for the linear parameters. If you need these, you can use the procedure `RCURVECOMMNONNONLINEAR`. This refits the model using the `FIT` directive, with the nonlinear part of the model specified by the `CALCULATION` option, as described in Section 3.5.8. Details are in Section 3.7.7.

3.7.4 Modifications to regression output and the `RKEEP` directive

The output produced by the `PRINT` options of the `FITCURVE` and `RDISPLAY` directives for fitted curves is much like that for iterative generalized linear models with a Normal distribution (3.5.3). In particular, only one response variable is analysed, standard errors are approximate, and the accumulated summary contains pooled contributions for all the terms fitted in one statement.

You cannot get standard errors and correlations for linear parameters in models where you have constrained some parameters of the curve to be equal for all the groups defined by a fitted factor. When you fit separate curves for the groups of a factor, correlations between parameters in different groups are zero and are not shown.

Neither can you get leverages for models in which parameters are constrained to be equal across groups. Genstat therefore does not standardize residuals with respect to the leverages in these models. For other models, the leverages are defined as:

$$l_i = \{\boldsymbol{D}'\boldsymbol{C}\boldsymbol{D}\}_{ii}$$

where $\boldsymbol{D}$ is the matrix of derivatives of the fitted values with respect to the parameters, and $\boldsymbol{C}$ is the variance-covariance matrix of the parameters divided by the estimate of the residual variance.

You can display intermediate results of the iteration by the `monitoring` setting of the `PRINT` option of the `FITCURVE` directive. At each cycle, the current parameter values are displayed together with the total number of times the likelihood function has been evaluated (*`Nfun`*) and an indication of the state of the search (*`Move`*). The possible states are:

Move

0 The current step is acceptable
1 Preconvergence; small adjustments are being made
2 The function is concave in at least one direction
3 Convergence is being approached, but there is distinct curvature
4 A bound has been violated
5 The current step is too large relative to the step lengths
6 Convergence
7 A step has been taken within a boundary plane

The step lengths used in the search are also reported whenever they are changed, and information is given about any temporary scaling used to simplify the search. Example 3.7.4 shows the progress of the search for the curve fitted in Example 3.7.

Example 3.7.4

```
  10  FITCURVE [PRINT=monitoring] Age

Temporary scaling of X by 0.1295

Convergence monitoring
----------------------

Cycle Eval Move    Function value    Current parameters
    0    6    0        0.19066757     0.300000
                           Steps     0.0100000
    1    9    0        0.18676336     0.350168
                           Steps    0.00250000
    2   12    1        0.18675920     0.351721

Convergence in Newton-Raphson  loop at cycle 2.
    3   16    6        0.18675916     0.351887
```

The search may not converge, particularly if the model to be fitted is unsuitable for the data. Genstat will give a warning message to indicate why convergence has not been achieved; often it will also suggest a limiting form of the curve that might be a more suitable description of the data than the one you have specified. You can find out about the final status of the search by the `EXIT` parameter of the `RKEEP` directive. It takes a value according to the following key:

Exit

0 Successful convergence
1 Limit on number of cycles has been reached without convergence
2 Parameter out of bounds
3 Likelihood appears constant
4 Failure to progress towards solution
5 Some standard errors are not available because the information matrix is nearly singular
6 Calculated likelihood may be incorrect because of missing fitted values
7 Curve is close to a limiting form
14 Function returned a missing value

With code 7, the limiting form of the curve is described by the warning diagnostic.

Further messages warn you about vertical asymptotes of rational curves. You can use the `summary` setting of the `PRINT` option to display the value or values of the explanatory variate for which the fitted curve is infinite. A warning is also printed if an asymptote occurs within the range of the data.

The derivatives of the fitted values with respect to each parameter can be stored in variates

using the GRADIENTS parameter of the RKEEP directive. You can use these quantities to assess the relative influence of each observation on a parameter; you can also construct a measure of leverage by summing the gradients for all the parameters.

The RGRAPH procedure can be used to display a fitted curve, as shown in Figure 3.7; it can also display a set of curves fitted for each level of a factor (3.7.3). The RCHECK procedure cannot be used to produce diagnostic information or pictures after curve fitting.

3.7.5 Functions of parameters: the RFUNCTION directive

RFUNCTION directive

Estimates functions of parameters of a linear, generalized linear, generalized additive or nonlinear model.

Options

PRINT = *string tokens*	What to print (estimates, se, correlations); default esti,se
CHANNEL = *identifier*	Channel number of file, or identifier of a text to store output; default current output file
CALCULATION = *expression structures*	Calculation of functions involving nonlinear and/or linear parameters; no default
SE = *variate*	To save approximate standard errors; default *
VCOVARIANCE = *symmetric matrix*	To save approximate variance-covariance matrix; default *
SAVE = *identifier*	Specifies save structure of regression model; default * i.e. that from last model fitted

Parameter

scalars	Identifiers of scalars assigned values of the functions by the calculations

The RFUNCTION directive provides estimates of functions of parameters in regression models, together with approximate standard errors and correlations. It can be used after any linear, generalized linear, generalized additive or nonlinear model, but it probably most useful following the FITCURVE and FITNONLINEAR directives; information about the latter is in 3.8.2. However, if there are any linear parameters in a general nonlinear model for which standard errors have not been estimated, standard errors and correlations cannot be estimated for functions that depend on those parameters (see 3.8.2). In addition, it is not possible to use the RFUNCTION directive after fitting standard curves with separate nonlinear parameters for each level of a factor (option NONLINEAR=separate in FITCURVE, ADD, DROP and SWITCH).

The functions are defined by the expressions supplied by the CALCULATION option of RFUNCTION; these define how to calculate the function from the values of the parameters. Unless initial values have been specified (3.7.6), the parameters in standard curves usually have no identifiers associated with them. If this is the case, you should refer to each parameter by using a text structure containing the name of the parameter as displayed, for example, by the option PRINT=estimates of the FITCURVE directive. The text structure can, of course, just be a string, for example 'R'.

In Example 3.7.5, we use RFUNCTION to provide us with an alternative parameterization of the exponential model fitted in Example 3.7, using the parameter K (3.7.1) instead of R, and reporting -B (i.e. Bneg) instead of B.

Example 3.7.5

```
  11  " Get estimates of parameters in the form
 -12    Y = A - Bneg*EXP(-K*X) "
  13  EXPRESSION E[1,2]; VALUE=!e(Bneg = -'B'),!e(K = -LOG('R'))
  14  RFUNCTION  [CALCULATION=E[]] Bneg,K

Estimates of functions of parameters
====================================

Estimates and standard errors
-----------------------------

Parameter         estimate         s.e.
Bneg                0.9725       0.0647
K                   0.1352       0.0256
```

The parameter of RFUNCTION provides a list of scalars that are to hold the estimated values of the functions. These need not be declared in advance, but will be defined automatically if necessary. The CALCULATION option specifies a list of one or more expressions to define the calculations necessary to evaluate the functions from the parameters of the nonlinear model, and place the results into the scalars. Note that when parameters are referred to by their names, these must match exactly, including case, the names as displayed by FITCURVE.

The PRINT option controls output as usual. By default, the estimates of the function values are formed – as could be done simply by a CALCULATE statement using the expressions if the parameters were available in scalars. In addition, approximate standard errors are calculated, using a first-order approximation based on difference estimates of the derivatives of each function with respect to each parameter. Approximate correlations can also be requested.

The SE and VCOVARIANCE options allow standard errors and the approximate variance-covariance matrix of the functions to be stored; the estimates of the functions themselves are automatically available in the scalars listed by the parameter of RFUNCTION. The SAVE option specifies which fitted model is to be used, as in the RDISPLAY and RKEEP directives.

3.7.6 Controlling the start of the search with the RCYCLE directive

You can use the RCYCLE directive to supply initial values and step lengths for the nonlinear parameters: you might do this, for example, to improve efficiency if you are fitting a standard curve and already have good prior knowledge of the likely values o the parameters. Usually, FITCURVE determines a reasonable starting value for each parameter by a short grid search, or by some manipulation of the data values: this will not be done if you supply initial values. For example

```
RCYCLE PARAMETER=Rate; INITIAL=0.62
FITCURVE [CURVE=exponential] X
```

You must usually give an identifier (here Rate) and an initial value for each nonlinear parameter in the model to be fitted. For logistic curves, however, you must include all the parameters – both nonlinear and linear. The parameters must be listed in the same order as Genstat uses to print them. The RCYCLE directive defines the identifiers as scalars holding the initial values that you have supplied; after the model has been fitted they contain the estimated values of the parameters.

The other parameters of RCYCLE are ignored by FITCURVE: bounds are set up automatically according to the curve to be fitted and the way in which it is parameterized by Genstat (over which you have no control).

You can use the MAXCYCLE option to reset the limit on the number of iterations, but Genstat

ignores the METHOD and TOLERANCE options. For all standard curve fitting Genstat uses a modified Newton method (3.8.1).

3.7.7 Standard errors for linear parameters: the RCURVECOMMONNONLINEAR procedure

RCURVECOMMONNONLINEAR procedure

Refits a standard curve with common nonlinear parameters across groups to provide s.e.'s for linear parameters (R.W. Payne).

Options

PRINT = *string tokens*	Printed output from the analysis (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring); default mode, summ, esti
MAXCYCLE = *variate*	Maximum number of iterations; default 30
METHOD = *string token*	Algorithm for fitting nonlinear model (gaussnewton, newtonraphson, fletcherpowell); default newt
STEPLENGTHS = *scalar* or *variate*	Initial step lengths for the parameters
SAVE = *regression save structure*	Save structure from this analysis
INSAVE = *regression save structure*	Save structure for the curve fitted by FITCURVE, default takes the most recent regression analysis

No parameters

RCURVECOMMONNONLINEAR can be used after a FITCURVE analysis to refit a standard curve that has common nonlinear parameters across groups. It uses the CALCULATION option of FIT, which provides standard errors for the linear parameters. These are unavailable with FITCURVE.

The INSAVE option can provide the regression save structure from the FITCURVE analysis. If this is not set, the save structure from the most recent regression analysis is used. A fault is given if the save structure is not from a FITCURVE analysis with groups and common nonlinear parameters. The SAVE option saves the regression save structure from this analysis.

The PRINT option controls printed output, with the same settings as FIT. The other options control aspects of the optimization. MAXCYCLE specifies the maximum number of iterations to be used to estimate the nonlinear parameters; default 30. METHOD specifies the algorithm to be used. The default is Newton Raphson, which is the same method as FITCURVE. STEPLENGTHS defines step lengths for the estimation of the nonlinear parameters.

FITCURVE uses a different strategy from FIT. It includes nonlinear parameters for all the groups in the model, but constrains them to be equal when they are common across groups. Consequently RCURVECOMMONNONLINEAR may obtain slightly different parameter estimates from the original FITCURVE analysis. Modifying the options may enable you to obtain closer results.

In Example 3.7.7, below, the default settings produce estimates that are reasonably close to those from FITCURVE.

Example 3.7.7

```
  2  " Model the relationship between yield of sugar from sugar beet',\
 -3    and phosphorus in different years."
  4  VARIATE   [NVALUES=64] Beetwt,%sugar,SoilP,Sugar
```

```
   5  READ      Beetwt,%sugar,SoilP

    Identifier   Minimum      Mean   Maximum    Values   Missing
        Beetwt     1.810     33.29     51.54        64         0
        %sugar    13.50      16.88     20.10        64         0
         SoilP     2.000     28.08     65.00        64         0

  25  CALCULATE Sugar = Beetwt * %sugar / 100
  26  FACTOR    [LEVELS=4; VALUES=16(1...4)] Year
  27  MODEL     Sugar
  28  TERMS     SoilP*Year
  29  FITCURVE  [PRINT=*; CURVE=lexp] SoilP
  30  ADD       [PRINT=*] Year
  31  &         SoilP.Year
  32  &         [PRINT=accumulated; NONLINEAR=separate; FPROBABILITY=yes]


Nonlinear regression analysis
=============================

Accumulated analysis of variance
--------------------------------

Change                          d.f.         s.s.         m.s.      v.r.  F pr.
+ SoilP                            3     149.0226      49.6742    132.27  <.001
+ Year                             3     177.0275      59.0092    157.13  <.001
+ SoilP.Year                       6      32.7683       5.4614     14.54  <.001
+ Separate nonlinear               3       2.2959       0.7653      2.04  0.121
Residual                          48      18.0259       0.3755

Total                             63     379.1402       6.0181

  33  " No evidence that the parameter R varies between years,
 -34    therefore fit a common estimate for R."
  35  DROP      [PRINT=model,summary,estimates; NONLINEAR=common]


Nonlinear regression analysis
=============================

 Response variate: Sugar
      Explanatory: SoilP
  Grouping factor: Year, all linear parameters separate
     Fitted Curve: A + B*(R**X) + C*X
      Constraints: R < 1


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression      12       358.82      29.9015     75.04
Residual        51        20.32       0.3985
Total           63       379.14       6.0181

Change           3         2.30       0.7653      2.04

Percentage variance accounted for 93.4
Standard error of observations is estimated to be 0.631.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           27        1.922       -3.12
           28        5.607        2.67


Estimates of parameters
-----------------------

Parameter                     estimate         s.e.
R                               0.8027       0.0246
B  Year 1                       -24.39
C  Year 1                   -0.0009727
```

```
A  Year 1                      9.033
B  Year 2                    -11.40
C  Year 2                  -0.003844
A  Year 2                      4.052
B  Year 3                    -7.086
C  Year 3                   0.003282
A  Year 3                      7.464
B  Year 4                    -7.007
C  Year 4                   0.02934
A  Year 4                      5.268

  36  " Use RCURVECOMMONNONLINEAR to obtain s.e.'s for A (intercept)
 -37    B (regression coefficient for R**Sugar for each year) and
 -38    C (regression coefficient for Sugar for each year)."
  39  RCURVECOMMONNONLINEAR


Nonlinear regression analysis
=============================

 Response variate: Sugar
Nonlinear parameters: Rate
  Model calculations: calcnonlin
     Fitted terms: Year + nonlin.Year + SoilP.Year


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression      12       358.82      29.9015     75.04  <.001
Residual        51        20.32       0.3985
Total           63       379.14       6.0181

Percentage variance accounted for 93.4
Standard error of observations is estimated to be 0.631.


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
Rate                0.8026       0.0348
* Linear
Year 1               9.032        0.632
Year 2               4.052        0.635
Year 3               7.464        0.607
Year 4               5.268        0.599
nonlin.Year 1      -24.40          5.67
nonlin.Year 2      -11.41          2.74
nonlin.Year 3       -7.09          1.32
nonlin.Year 4       -7.01          1.98
SoilP.Year 1       -0.0010       0.0145
SoilP.Year 2       -0.0038       0.0164
SoilP.Year 3        0.0033       0.0170
SoilP.Year 4        0.0293       0.0170
```

3.8 General nonlinear regression, and minimizing a function

You can use the methods described in this section (which correspond to the Nonlinear Models menu of Genstat *for Windows*) to fit any kind of regression. However, you should check first that the model does not belong to any of the categories described earlier in this chapter, for the appropriate directives are then much more efficient. These categories are linear models, generalized linear models and the standard curves provided by `FITCURVE`.

Because the methods described here are very general, they are neither as robust nor as automatic as, for example, the method that is used for fitting linear models. Nonlinear methods make use of iterative optimization algorithms, designed to search for the minimum value of a

function as the parameters vary; for nonlinear regression models, the function involved is the deviance, or minus twice the log-likelihood ratio, so the algorithm searches for the maximum-likelihood solution. It is often necessary to provide the algorithm with good starting values, to set bounds on the parameter values, and sometimes even to define the initial direction of search.

Optimization is easiest with few parameters, approximately quadratic functions, small correlations between parameters and good initial parameter estimates.

Where possible, you can effectively reduce the number of parameters to be optimized by separating linear and nonlinear ones: that is, you can first fit the linear parameters, and treat the resulting residual sums of squares as functions of the nonlinear parameters alone (3.8.2).

Problems with optimization methods are most likely to arise if you neglect the parameterization of the function. You can often transform the parameters to make the function nearly quadratic; after finding a solution, you can then use the `RFUNCTION` directive (3.7.5) to estimate the original parameters. Another source of difficulty is if you try to fit inappropriately many parameters.

You can usually find descriptive statistics based on the data that will provide initial estimates reasonably close to the final parameter estimates. For example, suitably spaced ordinates provide parameters for curve fitting that give much the same likelihood surface whatever curve is being fitted.

For advice on reformulating functions to speed up optimization, see Ross (1990). The methods used for optimization in Genstat are the same as those in MLP, the Maximum Likelihood Program. The MLP Manual (Ross 1987) contains further useful advice on alternative ways of specifying models.

Example 3.8 shows the fitting of a nonlinear model with four parameters. The model has the form

$$y_i = \frac{\theta_1\ \theta_3 \left(x_{2i} - \dfrac{x_{3i}}{1.632} \right)}{1 + \theta_2\ x_{1i} + \theta_3\ x_{2i} + \theta_4\ x_{3i}} + \varepsilon_i$$

which is linear in the parameter θ_1 but nonlinear in θ_2, θ_3 and θ_4. The parameterization of this model is reasonable, and it fits the data well; the algorithm succeeds in finding the solution without requiring the definition of initial values or bounds.

Example 3.8

```
  2  " Nonlinear model for a chemical process, involving four parameters.
 -3    Data from Carr (1960), analysed in Seber & Wild (1989) p.78.
 -4    The response R is the rate of disappearance of n-pentane by catalytic
 -5    isometrization to i-pentane, and the three associated variables
 -6    X1, X2 and X3 are the partial pressures of hydrogen, n-pentane and
 -7    i-pentane. Fit the unweighted model (Seber & Wild, p.83)."
  8  OPEN '%GENDIR%/Examples/GuidePart2/Reaction.dat'; CHANNEL=2
  9  READ [CHANNEL=2] X1,X2,X3,R

  Identifier   Minimum      Mean   Maximum    Values   Missing
          X1     106.6     290.5     470.9        24         0
          X2     68.30     152.3     294.4        24         0
          X3     10.50     74.88     157.1        24         0
           R    0.2680     4.071     11.65        24         0

 10  CLOSE 2
 11  " Change units from psia to atmospheres."
 12  CALCULATE X1,X2,X3 = X1,X2,X3 / 14.7
 13  " Specify how to form the nonlinear component of the model from
-14    the parameters and associated variables."
 15  EXPRESSION E1; VALUE=!e(Z = T3*(X2-X3/1.632)/(1+T2*X1+T3*X2+T4*X3))
 16  MODEL R
 17  " List the nonlinear parameters: attempt optimization from
-18    default starting values of 1 with no bounds."
 19  RCYCLE T2,T3,T4
```

```
  20  " Fit the model, estimating the linear parameter (called theta1 by
 -21    Seber & Wild) by linear regression with no additional constant."
  22  FITNONLINEAR [CALCULATION=E1; CONSTANT=omit; SELINEAR=yes; FPROB=yes] Z

Nonlinear regression analysis
=============================

 Response variate: R
Nonlinear parameters: T2, T3, T4
  Model calculations: E1


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       4      637.254     159.3135    985.09  <.001
Residual        20        3.234       0.1617
Total           24      640.488      26.6870

Percentage variance accounted for 98.5
Standard error of observations is estimated to be 0.402.


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
T2                    1.05         2.68
T3                    0.56         1.60
T4                    2.47         6.46
* Linear
Z                     35.9         11.4
```

3.8.1 Fitting nonlinear models

This subsection describes the preliminary things that you must do before fitting a general nonlinear model. It also gives information about the algorithms that Genstat uses.

Before using the FITNONLINEAR directive to fit a nonlinear model, you must use the MODEL directive to specify either the response variate, or the scalar that is to store the value of a general function (3.8.4). You must use the RCYCLE directive to specify the nonlinear parameters. You can also use the LINEARPARAMETERS option of RCYCLE to specify identifiers for the linear parameters (if any – see Section 3.8.2), so that you can refer to them in the model calculations. The TERMS directive can be used as in linear regression, to list the explanatory variables to be used in modelling. The model calculations themselves are provided in expression structures which are supplied by the CALCULATION option of FITNONLINEAR; in Example 3.8, a single expression called E1 is used. If you have used TERMS you can modify the model using the ADD, DROP and SWITCH directives, as in the previous sections. You can use the RDISPLAY and RKEEP directives to display or save the results. The RCHECK procedure does not work with nonlinear models, but RGRAPH can be used to display the fit of a nonlinear model with respect to some specified variate.

Genstat fits nonlinear regression models by maximum likelihood. The likelihood is usually from a distribution in the exponential family; this is specified using the DISTRIBUTION option of the MODEL directive. With the Normal and the Poisson distribution you can take advantage of linear parameters that the model contains; see 3.8.2. The fitting of models with the other settings of DISTRIBUTION, or with no linear parameters, is described in 3.8.3. To use other forms of likelihood, you should specify how it is to be calculated and set the FUNCTION option of the MODEL directive to a scalar whose value is assigned by the calculation (3.8.4). You can use this same device to minimize a general function with respect to its parameters.

The settings of the LINK and EXPONENT options of the MODEL directive are ignored, and you are not allowed to set the GROUPS option; other options and parameters are as in linear

regression.

Genstat provides three algorithms for fitting general nonlinear models; they work with numerical differences and so do not require you to specify derivatives. The default algorithm is a modified Gauss-Newton method. This takes advantage of the fact that the likelihood function can be expressed as a sum of squares. However, you cannot use it for minimizing a general function (3.8.4). The second algorithm, a modified Newton method, is requested by setting option METHOD=Newton in the RCYCLE statement (3.5.4). This can be used for any nonlinear model. The third algorithm is a modified Fletcher-Powell method, specified by setting METHOD=Fletcher. In fact, this is similar to the Newton method, with an occasional step in the search being determined by the Fletcher-Powell algorithm rather than by the Newton algorithm.

The modification in all these methods is to use estimated numerical differences instead of evaluating derivatives. In nonlinear regression problems, particularly ones with separable linear parameters, specification of the derivatives would be very complex, and so it is much more convenient to estimate them numerically.

You can change the limit on the number of iterations by the MAXCYCLE option of the RCYCLE directive, as for the FITCURVE directive.

You must set the PARAMETER parameter of the RCYCLE directive to the identifiers of scalars that will be used to represent the nonlinear parameters in the model calculations (3.8.2). There must be at least one nonlinear parameter. There is no formal upper limit on the number of nonlinear parameters, but the greater the number of parameters the longer the time required for the search and the smaller the chance of finding a satisfactory solution.

You can set the LOWER and UPPER parameters of RCYCLE to provide fixed bounds for each parameter. By default, the values $\pm 10^9$ are used. Where possible you should always set bounds, particularly to avoid such problems as attempting to take the log of a negative number. You can incorporate more general constraints as logical functions within the calculations. For example you could compute an extra term

```
(Constr > 0) * K * Constr
```

to impose a penalty on exceeding the constraint, controlled by setting different values of K. Often, the best way to impose a constraint is to reparameterize. For example, if a parameter α must be positive, you could replace α by $\exp(\beta)$, and allow β to take any value.

The STEPLENGTH parameter of RCYCLE can be used to provide initial step lengths for the search. By default the step length is 0.05 times the initial value of the corresponding parameter, or precisely 1.0 if the initial value is zero. If you set a step length to zero, Genstat treats the corresponding parameter as being fixed at its initial value. This allows complex problems in many dimensions to be tackled in stages, optimizing some parameters with others fixed, and then optimizing the others in turn.

By default, the initial value of a parameter is taken to be the current value of the scalar that represents it in the calculation, or 1.0 if the value is missing. Other values can be specified using the INITIAL parameter of RCYCLE.

If you can calculate a range within which you expect a parameter to lie, you should choose a step length of about 1% of the width of the range. If the steps are too small, numerical differencing may not work; if they are too large, gradients may be unreliable and you may get premature convergence. Genstat tests convergence by the relationship of final adjustments to step lengths.

The more parameters there are to estimate, and the more scattered are the data, the more iterations are required to find the optimum. The maximum number of iterations is set to 30 by default, but you can reset this with the MAXCYCLE option of RCYCLE (3.5.4). However, if convergence fails with a given setting of MAXCYCLE, you should check the data and consider reparameterizing the model before you indiscriminately increase the number of iterations.

Genstat prints a warning when convergence fails. The only sections of output that are then available are the residual degrees of freedom, the residual deviance, the fitted values, and the

parameter estimates (without standard errors) for the current cycle. The EXIT parameter of the RKEEP directive (3.7.4) allows you to obtain a numerical code indicating why convergence failed.

For any nonlinear model, you can choose just to evaluate the likelihood for a range of combinations of parameter values, rather than to maximize the likelihood with respect to the parameters. You do this by setting the NGRIDLINES option of FITNONLINEAR (3.8.2). The calculated values of the likelihood can be stored in a variate using the GRID parameter of the RKEEP directive (3.1.4), and used to produce pictures of the surface for example with the DCONTOUR or DSURFACE directives. This is illustrated in Example 3.8.4b.

3.8.2 Nonlinear regression for models with some linear parameters

FITNONLINEAR directive

Fits a nonlinear regression model or optimizes a scalar function.

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, grid); default mode, summ, esti or grid if NGRIDLINES is set
CALCULATION = *expression structures*	Calculation of fitted values or of explanatory variates involving nonlinear parameters; default * (valid only if OWN set)
OWN = *scalar*	Option setting for OWN directive if this is to be used rather than CALCULATE; default * requests CALCULATE to be used
CONSTANT = *string token*	How to treat the constant (estimate, omit); default esti
FACTORIAL = *scalar*	Limit for expansion of model terms; default as in previous TERMS statement, or 3 if no TERMS given
POOL = *string token*	Whether to pool ss in accumulated summary between all terms fitted in a linear model (yes, no); default no
DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality, vertical, df); default *
FPROBABILITY = *string token*	Printing of probabilities for variance and deviance ratios (yes, no); default no
SELECTION = *string tokens*	Statistics to be displayed in the summary of analysis produced by PRINT=summary, seobservations is relevant only for a Normally distributed response, and %cv only for a gamma-distributed response (%variance, %ss, adjustedr2, r2, seobservations, dispersion, %cv, %meandeviance, %deviance, aic, bic, sic); default %var, seob if DIST=normal, %cv if DIST=gamma, and disp for other distributions
NGRIDLINES = *scalar*	Number of values of each parameter for a grid of

	function evaluations; default *
SELINEAR = *string token*	Whether to calculate s.e.s for linear parameters (yes, no); default no
INOWN = *identifiers*	Setting to be used for the IN parameter of OWN if used in place of CALCULATE; default *
OUTOWN = *identifiers*	Setting to be used for the OUT parameter of OWN if used in place of CALCULATE; default *

Parameter

formula	List of explanatory variates and/or one factor to be used in linear regression, within nonlinear optimization

If the model is linear in some of the parameters, it may be fitted more efficiently using the methods described in this subsection. To use these the data must either be Normally distributed, or they must follow a Poisson distribution and the model must contain only one explanatory variable and no constant term.

The linear parameters are fitted by a linear regression of the response variate (specified by the parameter of the MODEL statement) on the variates listed by the parameter of FITNONLINEAR. At least one of these variates must depend on the nonlinear parameters in the model but they need not all do so. You can define how to calculate the variates from the nonlinear parameters either by the CALCULATION option or by the OWN, INOWN and OUTOWN options of FITNONLINEAR. If the parameter of FITNONLINEAR is not set, Genstat uses the methods described in either 3.8.3 or 3.8.4.

In Example 3.8, the linear parameter (θ_1 in the equation) is estimated by a regression of the response variate R on the variate Z; expression E1 defines how to form Z from the values of the parameters T2, T3 and T4 (θ_2, θ_3 and θ_4 in the equation) and from the variates X1, X2 and X3. The setting CONSTANT=omit in the FITNONLINEAR statement ensures that there is no constant term.

As already mentioned, the parameter of FITNONLINEAR may include variates that are not changed by the calculations as well as those that are. One factor may also be included so that a separate constant is fitted for each level. Thus

```
FACTOR [LEVELS=3; VALUES=8(1...3)] F
FITNONLINEAR Z,F,X2
```

would fit the model of Example 3.8 modified to include a constant for each of the three levels of F and an additional linear effect of the variable X2. The effect of including the factor is to fit a set of parallel nonlinear regressions. You cannot include interactions between a variate and a factor, as is allowed with FITCURVE; nor can you include POL, REG, COMPARISON, SSPLINE or LOESS functions, nor interactions between variates as allowed with FIT. However, procedure FITPARALLEL allows you to assess the various ways in which nonlinear models can be non-parallel (see 3.7.3 for an explanation of analysis of parallelism with FITCURVE).

If there is a constant in the linear regression, as specified by the CONSTANT option, the factor will be parameterized in terms of differences from the first level – as in linear regression. If you set CONSTANT=omit, the actual constants are fitted; there is no need to set option FULL of the TERMS directive, which is ignored in nonlinear models.

If you specify an offset variate (3.1.1), its values can also be modified by the calculations, and depend on the parameters.

The PRINT option is as for the FIT directive.

You must set one of the CALCULATION and OWN options to define how the nonlinear parameters are included in the model. The CALCULATION option does this by a list of one or more expressions. The expressions are evaluated in turn at every step of the estimation process, just as if they had been given in a sequence of CALCULATE statements. For example:

```
EXPRESSION Diffuse[1]; \
```

```
  VALUE=!E(Xl,Xr=NORMAL((H+1,-1*X)/SQRT(2*D*T))
& Diffuse[2]; VALUE=!E(Z=Xl+Xr-1)
FITNONLINEAR [CALCULATION=Diffuse[1,2]] Z
```

Here, the CALCULATION option is set to the two expressions Diffuse[1] and Diffuse[2], to define a model for one-dimensional diffusion.

Alternatively, you can set the OWN option to specify that the calculation is to be done by executing your own source code, called by a version of the subroutine G5XZXO, as for the OWN directive. Generally, using OWN is likely to be worthwhile only when calculations are very extensive, or when a particular function is needed often. The setting of the OWN option will be passed to G5XZXO in the same way as the setting of the SELECT option of the OWN directive is passed to G5XZXO.

The CONSTANT, FACTORIAL, POOL, DENOMINATOR, NOMESSAGE and FPROBABILITY options are as for the FIT directive, except that the NOMESSAGE option has an additional setting df which controls messages about loss of degrees of freedom occurring during the iterative fitting of the model, when observations may become excluded because of missing values introduced by the calculations.

If you set the NGRIDLINES option to n, say (with $n \geq 2$), the FITNONLINEAR directive evaluates the likelihood at a grid of values of the nonlinear parameters, and does not search for an optimum. For each parameter, the distance between the upper and lower bounds (set by the RCYCLE directive) will be divided into $(n-1)$ equal parts, defining a rectangular grid with n gridlines in each dimension. By setting some upper and lower bounds equal, you can look at the behaviour of the function with respect to a few parameters at a time. The default setting of the PRINT option is grid in this case, and produces a display of the function values. Other settings of the PRINT option are ignored. The calculated grid of values is available from the GRID parameter of the RKEEP directive. This is illustrated in 3.8.4.

By default, standard errors are calculated only for nonlinear parameters. To obtain standard errors for the linear parameters as well, you can set option SELINEAR=yes. Then, after the optimum has been found, Genstat increases the number of dimensions to include the linear parameters and estimates the rate of change of the likelihood in all the dimensions.

The INOWN and OUTOWN options are relevant only when the OWN option is set.

3.8.3 Nonlinear regression models with no linear parameters

If there are no linear parameters in the model, or if the distribution is not one of those that can be handled by the method described in 3.8.2, you should no longer use the parameter of FITNONLINEAR. Instead you should set the FITTEDVALUES parameter in the MODEL statement to the identifier of a variate that is to contain the fitted values for any set of values of the nonlinear parameters. Then define how to calculate the fitted values from the nonlinear parameters and the explanatory variates, using either the CALCULATION or the OWN options of FITNONLINEAR, as in 3.8.2.

Example 3.8.3a shows how to refit the model of Example 3.8 without taking advantage of the linearity of parameter θ_1. Expression E2 in line 24 calculates the variate of fitted values F as T1 (θ_1) multiplied by the variate Z (calculated by the expression E1 used in Example 3.8). F is identified as the fitted-value variate in line 27, initial values are specified for the parameters in line 31, and then the model can be fitted, to obtain the same answers as before.

Example 3.8.3a

```
 23  " Specify how to form the fitted values from Z and the linear
-24    parameter theta 1."
 25  EXPRESSION E2; VALUE=!e(F=T1*Z)
 26  " Supply the name of the variate that will hold fitted values
-27    calculated by the expressions."
 28  MODEL R; FITTED=F
 29  " Include theta1 with the list of nonlinear parameters;
```

```
 -30    use initial values of 1 as before, except for theta 1
 -31    (if this is not done, FITNONLINEAR will not converge)."
  32  RCYCLE T1,T2,T3,T4; INITIAL=36,1,1,1
  33  " Fit the model, with no linear regression involved."
  34  FITNONLINEAR [CALCULATION=E1,E2; FPROB=yes]

Nonlinear regression analysis
=============================

 Response variate: R
Nonlinear parameters: T1, T2, T3, T4
  Model calculations: E1, E2


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       4      637.254     159.3135    985.09  <.001
Residual        20        3.234       0.1617
Total           24      640.488      26.6870

Percentage variance accounted for 98.5
Standard error of observations is estimated to be 0.402.


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
T1                    35.9         11.4
T2                    1.05         2.69
T3                    0.56         1.61
T4                    2.47         6.50
```

The output from the `monitoring` setting of the `PRINT` option, not displayed here, shows that solution takes 18 iterations involving 164 function evaluations compared to 13 and 123 when θ_1 is treated as linear. Moreover, convergence is not achieved here without supplying an initial value for θ_1. So clearly you should exploit linearity where possible.

With the methods described in this section, the distribution can be any of those available from the `DISTRIBUTION` option of the `MODEL` directive, with the exception of the inverse-Normal distribution. Thus, the deviance will be based on the likelihood function of either the Normal, Poisson, binomial, gamma or multinomial distributions, taking account of the settings of the `DISPERSION` and `WEIGHTS` options of the `MODEL` directive. The first four of these distributions were discussed in 3.5.1 and 3.5.2.

The multinomial distribution is used rather differently from the others: it is for fitting distributions. The `DISTRIBUTION` directive (2.2.10) provides a wide range of standard distributions, and is more convenient and efficient than `FITNONLINEAR` for these; but `FITNONLINEAR` allows you to fit other distributions. (Despite the terminology "multinomial", this setting is thus not for fitting models to response variables that take one of a finite set of values for each unit; these can be fitted using generalized linear models as described in 3.5.5.)

To specify and fit your own distribution, you should supply as response variate a set of counts of observations falling into a series of groups; the fitted values should then be a set of expected counts for the groups, calculated from the distribution being considered. The resulting multinomial likelihood is the same as that of the Poisson distribution, but with the constraint $\Sigma f_i = M$, where M is the sum of the counts.

Example 3.8.3b fits a Normal distribution to a set of observations produced by the Genstat pseudo-random number generator. It would be much easier to use the `DISTRIBUTION` directive (2.2.10) for this, but use of this familiar distribution here should make it clear how `FITNONLINEAR` can be used in more complicated situations.

Example 3.8.3b

```
  2  " Fit a Normal distribution to pseudo-random numbers in the range (0,1)
 -3    generated by the functions URAND and EDNORMAL."
  4  CALCULATE Random = EDNORMAL(URAND(25384; 50))
  5  " Define bounds to subdivide the observations."
  6  SCALAR    Limit[1...8]; VALUE=-100,-1,-0.6,-0.2,0.2,0.6,1,100
  7  " Form response variate: counts of numbers within specified bounds."
  8  CALCULATE S[1...7] = SUM(Random<=Limit[2...8] .AND. Random>Limit[1...7])
  9  VARIATE   [VALUES=S[1...7]] Count
 10  " Set up expression to calculate expected counts for a Normal variable."
 11  &         [VALUES=Limit[2...8]] L1
 12  &         [VALUES=Limit[1...7]] L2
 13  EXPRESSION [VALUE=P=50*(NORMAL((L1-Mean)/SD)-NORMAL((L2-Mean)/SD))]\
 14             Normal
 15  MODEL      [DISTRIBUTION=multinomial] Count; FITTED=P
 16  RCYCLE     Mean,SD; STEPLENGTH=0.02,*; LOWER=*,0.5; INITIAL=0,1
 17  FITNONLINEAR [CALCULATION=Normal]

Nonlinear regression analysis
=============================

 Response variate: Count
     Distribution: Multinomial
Nonlinear parameters: Mean, SD
  Model calculations: Normal


Summary of analysis
-------------------

                                        mean  deviance
Source        d.f.     deviance     deviance     ratio
Regression       2            *            *
Residual         4        1.904       0.4760
Total            6            *            *

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
Mean                 0.068        0.151
SD                   1.024        0.137

* MESSAGE: s.e.s are based on dispersion parameter with value 1
```

3.8.4 General nonlinear models

The earlier parts of this section have dealt with two methods of calculating the likelihood at each step of the iterative search: performing linear regression of the response variate on calculated explanatory variates, and directly comparing the response variate with a calculated variate of fitted values. A third method is to calculate the likelihood explicitly. You can also use this to minimize the value of a function that is not a likelihood at all. Remember, however, that the methods described earlier in this chapter actually maximize the likelihood function by minimizing the deviance, which is minus twice the log-likelihood ratio. (So, if you want to estimate standard errors for the parameters, you should specify deviances rather than likelihoods here too.)

To use the regression directives to minimize a function, you need to start with a `MODEL` statement that has no response variate, but where the `FUNCTION` option is set to a scalar. You then specify the parameters with the `RCYCLE` directive as before, and perform the minimization

with FITNONLINEAR, supplying an expression that calculates the function from the parameters and places the result into the scalar. Example 3.8.4a shows the minimization of an awkward two-dimensional test function.

Example 3.8.4a

```
   2  " Finding the minimum of a function of two parameters:
  -3    Rosenbrock's steep-sided valley. open '3-8-4.hpg';4;gr: device 4"
   4  EXPRESSION   Rbrock; VALUE=!e(F = 100*(P2-P1*P1)**2+(1-P1)**2)
   5  MODEL        [FUNCTION=F]
   6  RCYCLE       P1,P2; STEPLENGTH=0.01; INITIAL=-1.2,1
   7  FITNONLINEAR [PRINT=summary,estimates,correlation,monitoring; \
   8               CALCULATION=Rbrock]

Convergence monitoring
----------------------

Cycle Eval Move    Function value    Current parameters
    0    1    0         24.200000      -1.20000      1.00000
                           Steps     0.0100000     0.0100000
                           Steps    0.00622720     0.0160586
    1   10    0         4.7307251      -1.17501       1.38003
    2   23    5         4.1553333     -0.908300      0.753335
    3   32    0         3.2049376     -0.783431      0.598171
    4   45    5         2.7437562     -0.578825      0.284932
                           Steps    0.00911229     0.0109742
    5   54    0         2.1023790     -0.435217      0.168790
    6   63    0         1.9557949     -0.154202     -0.0551909
    7   72    0         1.1802438    -0.0853558    0.00254582
    8   85    5        0.94127635     0.0679015    -0.0223094
                           Steps     0.0186665    0.00535718
    9   94    0        0.66350789      0.213541     0.0243889
   10  103    0        0.45561812      0.359756      0.108045
   11  112    0        0.29300700      0.475594      0.212771
   12  121    0        0.18435710      0.607706      0.351853
                           Steps    0.00900248     0.0111081
   13  130    0        0.10277990      0.685032      0.463290
   14  139    0       0.067494726      0.822903      0.658161
   15  148    0       0.020597675      0.856950      0.733204
   16  157    0       0.013285223      0.961115      0.912893
                           Steps    0.00719726     0.0138942
   17  166    0     0.00099578333      0.968449      0.937839
   18  175    0     0.00012767864      0.989629      0.978917
                           Steps    0.00177577    0.00351961
   19  184    1     0.00010532730      0.989737      0.979579
   20  193    0     1.2554856E-06      0.999373      0.998653

Convergence in Newton-Raphson  loop at cycle 20.
   21  203    6     3.9717401E-07      0.999370      0.998740
                           Steps     0.0700523      0.140195
    1  213    0     3.9717401E-07      0.999370      0.998740


Results of optimization
=======================

Minimum function value
----------------------

  3.97174E-07


Estimates of parameters
-----------------------

Parameter         estimate       "s.e."
P1                   0.999        0.819
P2                    1.00         1.64
```

```
Scaled inverse of second derivatives
------------------------------------

Parameter     ref    scaled inverse of 2nd derivatives

P1              1    1.000
P2              2    0.998  1.000
                         1      2
```

The FUNCTION option of the MODEL statement defines the scalar to be F, and the expression Rbrock in the CALCULATION option of FITNONLINEAR sets F to the value of the function.

When you are minimizing a general function in this way, some of the output from FITNONLINEAR is different. Genstat ignores the accumulated and fittedvalues settings, and the deviance and summary settings display only the minimum function value. The correlation setting displays the inverse of the estimated matrix of second derivatives of the function with respect to the parameters, scaled by the diagonal values. Similarly, in place of the standard errors usually displayed by the estimates setting, Genstat prints the square roots of the diagonal values of twice the inverse of the second-derivative matrix. These can give a useful indication of the form of the function near the minimum. As indicated by their title in the output, if the function is a deviance you can interpret these as asymptotic standard errors and correlations (not scaled by an estimate of dispersion). For a general function, the "s.e." can be interpreted as the approximate change in a parameter required to increase the function by 1.0 starting from the minimum.

Genstat ignores the CONSTANT option of the FITNONLINEAR directive for general functions, and you must not set the parameter. Similarly, the WEIGHTS and OFFSET options of the MODEL directive are ignored, and the GROUPS option must not be set. The only parameters of the RKEEP directive that are available are ESTIMATES, SE, INVERSE, EXIT, GRADIENTS and GRID. The minimum value of the function is of course available in the scalar specified by the FUNCTION option of the MODEL directive.

You will usually want to inspect the shape of the function near the minimum. So next we form a grid of function values using the NGRIDLINES option of FITNONLINEAR; to save space in the output, we do not display the values with the option setting PRINT=grid, but just extract them with the GRID parameter of RKEEP, and display them with the DSURFACE directive (1:6.1). The picture is in Figure 3.8.4.

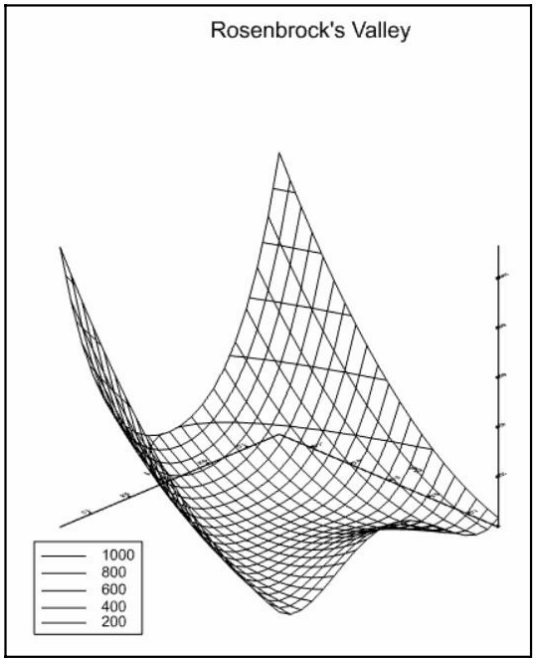


Figure 3.8.4

Example 3.8.4b

```
  9  " Draw a contour map of the function
 -10   for P1 in (-1.2,1.2), P2 in (0,1.2)."
  11  RCYCLE       P1,P2; LOWER=-1.4,-1.4; UPPER=1.4,1.4
  12  FITNONLINEAR [PRINT=*; NGRIDLINES=21; CALCULATION=Rbrock]
  13  RKEEP    GRID=Vgrid
  14  MATRIX   [ROWS=21; COLUMNS=21] Mgrid; VALUES=Vgrid
  15  XAXIS    3; TITLE='P1'; LOWER=-1.4; UPPER=1.4
  16  YAXIS    3; TITLE='P2'; LOWER=-1.4; UPPER=1.4
  17  DSURFACE [TITLE='Rosenbrock''s Valley'; WINDOW=3; AZIMUTH=45]\
```

```
18          Mgrid; PENFILL=0
```

If you have a function that is too complicated to be calculated by a list of Genstat expressions (for example its definition may need you to use directives or procedures as well), you can use the MINIMIZE, MIN1DIMENSION or SIMPLEX procedures. These are described in Part 3 of the *Genstat Reference Manual*.

3.9 Regression trees

3.9.1 Constructing a regression tree

BREGRESSION procedure

Constructs a regression tree (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (summary, details, indented, bracketed, labelleddiagram, numbereddiagram, graph, monitoring); default * i.e. none
Y = *variate*	Response variate for the regression
TREE = *tree*	Saves the tree that has been constructed
MSLIMIT = *scalar*	Limit on the mean square of the observations at a node at which to stop making splits; default 0
NSTOP = *scalar*	Specifies the number of observations at a node at which to stop making splits; default 1
OWNBSELECT = *string token*	Indicates whether or not your own version of the BSELECT procedure is to be used (yes, no); default no

Parameters

X = *variates* or *factors*	Independent variables available for constructing the tree
ORDERED = *string tokens*	Whether factor levels are ordered (yes, no); default no

A regression tree is a mechanism for predicting a response variable from a set of independent variables (see Chapter 8 of Breiman *et al.*). The tree is constructed using data on a set of observations. Their values for the response variable are specified (in a variate) using the Y option, and their values for the independent variables are specified (in a list of variates) using the X parameter. Factors may have either ordered or unordered levels, according to whether the corresponding value ORDERED parameter is set to yes or no. For example, a factor called Dose with levels 1, 1.5, 2 and 2.5 would usually be treated as having ordered levels, whereas levels labelled 'Morphine', 'Amidone', 'Phenadoxone' and 'Pethidine' of a factor called Drug would be regarded as unordered.

The construction process splits the observations into subsets, according to whether or not they are less than a particular value of one of the independent variates. The aim is to form subsets that have similar values for the response variate. The predicted value of the response variable for each node of the tree is the mean of its value for the subset of observations at that node. The *accuracy* of the node is the squared distance of the values of the response variate from their mean for the observations at the node, divided by the total number of observations. The potential splits at the node are assessed by their effect on the accuracy, that is the difference between the accuracy of the node and the sum of the accuracies of the two potential successor nodes. The node will become a terminal node if none of the splits provides any improvement in accuracy, or if the mean square of the observations at the node is less than or equal to a limit specified by

the MSLIMIT option (default 0), or if the number of observations at the node is less than or equal to the number specified by the NSTOP option (default 1).

The resulting tree can be saved using the TREE option. Details of the tree can be printed as selected by the PRINT option, with settings:

summary	prints a summary of the properties of the tree;
details	gives detailed information about the nodes of the tree;
bracketed	display as used to represent an identification key in "bracketed" form (printed node by node).
indented	display as used to represent an identification key in "indented" form (printed branch by branch);
labelleddiagram	diagrammatic display including the node labels;
numbereddiagram	diagrammatic display with the nodes labelled by their numbers;
graph	plots the tree using high-resolution graphics.
monitoring	prints information monitoring the construction process.

BREGRESSION stores the information required for printing as part of the tree. For variates and ordered factors, the labels are generally formed as "*identifier*<*p*" and "*identifier*>*p*", where *p* is the value chosen to partition the data for the variate concerned. Alternatively, if you have defined an "extra" text for the variate (using the EXTRA parameter of the VARIATE command), this will be used instead. The labels are then "*extra-text* < *p*" and "*extra-text* > *p*". The style is similar for unordered factors, but here the labels involve the operators .IN. and .NI. instead of < and >.

Example 3.9.1 uses BREGRESSION to construct a regression tree for some data relating water usage at a production plant (Draper & Smith 1981, page 352) during 17 months to the average temperature, the amount of production, the number of operating days and the number of employees. Notice that, as the MSLIMIT is at its default value of zero, the tree continues until there is a node for every distinct value of Wateruse (i.e. 17 nodes with residual degrees of freedom and sum of squares zero).

Example 3.9.1

```
  2  " Water usage data (Draper & Smith 1981, page 352)."
  3  READ  Temperature,Production,Operatingdays,Employees,Wateruse

    Identifier   Minimum      Mean   Maximum    Values   Missing
   Temperature     39.50     64.85     81.00        17         0
    Production     6.373     12.90     18.57        17         0
 Operatingdays     19.00     21.47     25.00        17         0
     Employees     129.0     181.8     206.0        17         0
      Wateruse     2.828     3.304     4.488        17         0

 21  " Form the regression tree."
 22  BREGRESSION [PRINT=summary; Y=Wateruse; TREE=Tree]\
 23              Employees,Operatingdays,Production,Temperature

Summary of regression tree: Tree
================================

Number of nodes: 33
Number of terminal nodes: 17
Residual sum of squares: 0
Residual degrees of freedom:  0
Residual mean square: *
Percentage variance accounted for: *
Variables in the tree: Production, Temperature, Employees, Operatingdays.
```

BREGRESSION calls procedure BCONSTRUCT (1:4.12.6) to form the tree. This uses a special-purpose procedure BSELECT, which is customized specifically to select splits for use in regression trees. You can use your own method of selection by providing your own BSELECT

and setting option OWNBSELECT=yes. In the standard version of BSELECT, the BASSESS directive (1:4.12.8) is used to assess the potential splits.

3.9.2 Displaying a regression tree

BRDISPLAY procedure

Displays a regression tree (R.W. Payne).

Option

PRINT = *string tokens*	Controls printed output (summary, details, indented, bracketed, labelleddiagram, numbereddiagram, graph); default * i.e. none

Parameter

TREE = *tree*	Tree to be displayed

Further output for a regression tree can be obtained with the BRDISPLAY procedure. The tree is specified by the TREE parameter, and the PRINT option selects the output (with settings that all operate as in the PRINT option of BREGRESSION).

Example 3.9.2 uses BRDISPLAY to print the tree from Example 3.9.1 in indented form. The first explanatory variable in the tree (at index 1) is Production. If the value is less than 17.63, the next task (at index 2) is to see whether or not the Temperature if less than 71.4 (index 2); if it is greater than 17.63, we reach a terminal node where water usage is predicted to be 4.488. Some further examples are in Example 3.9.3.

Example 3.9.2

```
  24  BRDISPLAY [PRINT=indented] Tree

1 Production<17.63 2
 2 Temperature<71.40 3
  3 Temperature<55.25 4
   4 Operatingdays<21.00 5
    5 Operatingdays<19.50 3.125
    5 Operatingdays>19.50 6
     6 Employees<188.5 3.286
     6 Employees>188.5 3.211
   4 Operatingdays>21.00 3.542
  3 Temperature>55.25 7
   7 Temperature<64.30 8
    8 Employees<192.0 9
     9 Employees<157.5 3.067
     9 Employees>157.5 3.060
    8 Employees>192.0 3.022
   7 Temperature>64.30 10
   10 Employees<147.0 2.828
   10 Employees>147.0 11
    11 Employees<172.5 2.891
    11 Employees>172.5 2.922
 2 Temperature>71.40 12
 12 Employees<170.5 2.994
 12 Employees>170.5 13
  13 Employees<192.0 14
   14 Employees<182.0 3.502
   14 Employees>182.0 15
    15 Employees<190.0 3.898
    15 Employees>190.0 3.950
  13 Employees>192.0 16
   16 Employees<196.5 3.082
   16 Employees>196.5 3.295
```

```
1 Production>17.63 4.488
```

3.9.3 Pruning a regression tree

Generally the construction of a regression tree will result in *over-fitting*, that is it will form a tree that keeps making splits beyond the point that can be justified statistically. The solution is to prune the tree to remove the uninformative sub-branches, and this can be performed using the BPRUNE procedure. It is best, if possible, to base the pruning on an independent set of data. The pruning uses the *accuracy* figures, which are stored with the tree. The *prediction* at each node is the mean of the observations that occur at the node. The *accuracy* of the node is the squared distance of the values of the response variate for the observations at the node from the prediction, divided by the total number of observations. The BRVALUES procedure can be used to calculate new accuracy and prediction values, from another data set.

BRVALUES procedure

Forms values for nodes of a regression tree (R.W. Payne).

Options

Y = *variate*	Values of the response variate for the new data set
TREE = *tree*	Tree for which predictions and accuracy values are to be formed
REPLACE = *string token*	Whether to replace the values stored in the tree (yes, no); default no
PREDICTION = *pointer*	New predictions for the nodes of the tree
ACCURACY = *pointer*	New accuracy values for the nodes of the tree
NOBSERVATIONS = *pointer*	New numbers of observarions for the nodes of the tree

Parameter

X = *variates*	Values of the x-variates for the new data set

The TREE option specifies the tree for which the values are to be formed. The Y option specifies the values of the response variate for the observations in the new data set, and the X parameter defines their values for the x-variates as used to construct the tree. You can set option REPLACE=yes to use the new values to replace those already stored in the tree. Alternatively, you can use the PREDICTION parameter to save the predictions, in a pointer. This has an element for each node of the tree (and with the same suffix as that node) pointing to a scalar storing the prediction for the node. Similarly, the ACCURACY parameter saves the accuracies, and the NOBSERVATIONS parameter saves the numbers of observations at each node. You can use these later to replace the prediction and accuracy values in the original tree by

```
CALCULATE Tree[]['accuracy'] = ACCURACY[]
&         Tree[]['prediction'] = PREDICTION[]
&         Tree[]['nobservations'] = NOBSERVATIONS[]
```

Alternatively, you may want to combine them first with other estimates, for example to form bootstrapped estimates.

The pruning is performed by the BPRUNE procedure

BPRUNE procedure

Prunes a tree using minimal cost complexity (R.W. Payne).

Option

PRINT = *string tokens*	Controls printed output (graph, table, monitoring);

default `tabl`

Parameters

`TREE` = *trees*	Trees to be pruned
`ACCURACY` = *pointers*	Accuracy values for the nodes of each tree; default is to use those stored with the tree
`NEWTREES` = *pointers*	Saves the trees generated during the pruning of each tree
`RTPRUNED` = *variates*	Accuracy of the pruned trees of each tree
`NTERMINAL` = *variates*	Number of terminal nodes in the pruned trees of each tree

The tree to be pruned is specified by the `TREE` parameter. `BPRUNE` assumes that there is an *accuracy* figure R(t) available for each node t of the tree. By default this is assumed to be stored with the tree itself, but you can specify other values using the `ACCURACY` parameter. This should be set to a pointer whose suffixes are the same as the numbers of the nodes in the tree, and whose elements are scalars storing the relevant accuracy values. The accuracy R(T) of the whole tree T is defined to be the sum of the accuracies of its terminal nodes.

`BPRUNE` uses the principle of minimal cost complexity (Breiman *et al*. 1984, Chapter 3) to produce a sequence of pruned trees. At each stage it prunes at the node which is the *weakest link*. Define R(T_t) to be the accuracy of the subtree with root at node t, and nterm(t) to be its number of terminal nodes. The weakest link is then the node for which

$$(R(t) - R(T_t)) / (nterm(t) - 1)$$

is a minimum. The pruned trees can be saved, in a pointer, using the `NEWTREES` parameter. Their accuracies can be saved (in a variate) using the `RTPRUNED` parameter, and their numbers of terminal nodes can be saved (also in a variate) using the `NTERMINAL` parameter.

Printed output is controlled by the `PRINT` option, with settings:

`graph`	plots `RTPRUNED` against `NTERMINAL`;
`table`	prints a table with `RTPRUNED` and `NTERMINAL`;
`monitoring`	provides monitoring information during the pruning.

The plot of `RTPRUNED` against `NTERMINAL` demonstates the trade-off between accuracy and complexity (number of terminal nodes). It should show an initial rapid decrease, followed by a long flat region, and then often a gradual increase. The aim is to select a tree that is accurate but not over-complex. One possibility is to take the tree at the point where the graph levels off. However, `RTPRUNED` contains only an estimate of the accuracy of the trees. So Breiman *et al*. (1984) recommend taking a tree a little above that (in fact at one standard error of `RTPRUNED` above the minimium point in the graph: see Chapters 3 and 11). In practice though a small amount of over-fitting should not be a problem, so the exact choice of pruned tree should not be crucial.

Example 3.9.3 prunes the tree from Example 3.9.1. There is no independent set of data available here, so the pruning is based on the accuracy values from the original data used to construct the tree. Examining the accuracies of the pruned trees (printed in the column headed `RT`, and plotted in Figure 3.9.3) suggests that tree 7 is the most appropriate choice. The `BCUT` directive (1:4.12.4) in line 164 replaces `Tree` with this tree, `Pruned[7]`, renumbering its nodes at the same time. `BRDISPLAY` then displays the new tree.

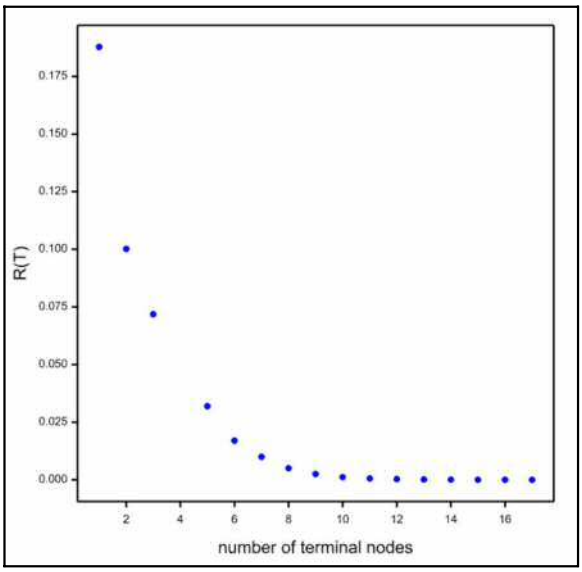


Figure 3.9.3

Example 3.9.3

```
  25  " Prune the tree."
  26  BPRUNE [PRINT=table,graph] Tree; NEWTREES=Pruned

Characteristics of the pruned trees
===================================


  Tree          RT   Number of
   no.              terminal
                       nodes
     1     0.00000          17
     2     0.00000          16
     3     0.00003          15
     4     0.00010          14
     5     0.00018          13
     6     0.00034          12
     7     0.00058          11
     8     0.00118          10
     9     0.00252           9
    10     0.00505           8
    11     0.00999           7
    12     0.01697           6
    13     0.03198           5
    14     0.07186           3
    15     0.10014           2
    16     0.18780           1

  27  " Use tree 7 - renumber nodes."
  28  BCUT [RENUMBER=yes] Pruned[7]; NEWTREE=Tree
  29  " Display the tree."
  30  BRDISPLAY [PRINT=summary,indented] Tree

Summary of regression tree: Tree
================================

Number of nodes: 21
Number of terminal nodes: 11
Residual sum of squares: 0.009926
Residual degrees of freedom:  6
Residual mean square: 0.001654
Percentage variance accounted for: 99.17
Variables in the tree: Production, Temperature, Employees, Operatingdays.
```

```
1 Production<17.63 2
 2 Temperature<71.40 3
  3 Temperature<55.25 4
   4 Operatingdays<21.00 5
    5 Operatingdays<19.50 3.125
    5 Operatingdays>19.50 3.248
   4 Operatingdays>21.00 3.542
  3 Temperature>55.25 6
   6 Temperature<64.30 3.050
   6 Temperature>64.30 2.880
 2 Temperature>71.40 7
  7 Employees<170.5 2.994
  7 Employees>170.5 8
   8 Employees<192.0 9
    9 Employees<182.0 3.502
    9 Employees>182.0 3.924
   8 Employees>192.0 10
   10 Employees<196.5 3.082
   10 Employees>196.5 3.295
1 Production>17.63 4.488
```

3.9.4 Predictions from a regression tree

BRPREDICT procedure

Makes predictions using a regression tree (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (prediction, transcript); if PRINT is unset in an interactive run BRPREDICT will ask what you want to print, in a batch run the default is pred
TREE = *tree*	Specifies the tree
PREDICTIONS = *variate*	Saves the prediction for the observations
TERMINALNODES = *pointer*	Saves the numbers of the terminal nodes from which each prediction was obtained
MVINCLUDE = *string token*	Whether to provide predictions for units with missing or unavailable values of the x-variables (explanatory); default expl

Parameters

X = *variates* or *factors*	Explanatory variables
VALUES = *scalars*, *variates* or *texts*	Values to use for the explanatory variables; if these are unset for any variable, its existing values are used

BRPREDICT makes predictions using a regression tree. The tree is specified by the TREE option. Alternatively, BRPREDICT will ask you for the identifier of the tree if you do not specify TREE when running interactively.

The x-values for the predictions can be specified in the variates or factors listed by the X parameter. These must have identical names (and levels) to those used originally to construct the tree. You can use the VALUES parameter to supply new values, if those stored in any of the variates or factors are unsuitable.

If you do not set X when running interactively, BRPREDICT will ask you to supply the relevant x-values in turn, as required by the tree. Otherwise, if an x-variable in the tree is not specified in the X parameter list, its values are assumed to be unavailable (i.e. missing).

By default, when the x-variable required at a node in the tree is unavailable or contains a

missing value, BRPREDICT will follow all the branches from that node, and average the predictions that they generate. You can set option MVINCLUDE=*, if you would prefer the prediction to be missing.

The PRINT option controls printed output, with settings:

prediction	prints the predictions obtained using the tree;
transcript	prints the x-values supplied in response to questions in an interactive run.

If you do not set PRINT in an interactive run, BRPREDICT will ask what you would like to print. In batch, the default is to print the predictions.

You can save the predictions, in a variate, using the PREDICTIONS option. The TERMINALNODES option allows you to save a pointer, with an element for each prediction, containing the numbers of the terminal nodes reached in the tree to provide the predictions. This will be a scalar if the prediction was derived from a single node, or a variate if it involved more than one (because several branches have been taken, as the result of a missing x-value).

Example 3.9.4 makes a prediction of water usage using the tree from Example 3.9.3.

Example 3.9.4

```
  31  " Predict water usage for a day with 150 employees, 15 operating days,
 -32    production 12.5 and temperature 65."
  33  VARIATE     Employees,Operatingdays,Production,Temperature;\
  34              VALUES=150,21,12.5,65
  35  BRPREDICT   [PRINT=prediction; TREE=Tree; PREDICTION=Prediction]\
  36              Employees,Operatingdays

Prediction:
3.332
```

3.9.5 Predictions from a regression tree

BRKEEP procedure

Saves information from a regression tree (R.W. Payne).

No options

Parameters

TREE = *trees*	Tree from which the information is to be saved
SUMMARY = *variates*	Saves summary information about each tree
XVARIABLES = *pointers*	Saves the identifiers of the x-variables in each tree

BRKEEP saves information about a regression tree, constructed by the BREGRESSION procedure. The tree can be saved using the TREE option of BREGRESSION, and is specified for BRKEEP using its TREE parameter.

The SUMMARY parameter saves a variate containing summary information: number of nodes, number of terminal nodes, residual sum of squares, residual degrees of freedom, residual mean square and percentage variance accounted for (in that order).

The XVARIABLES parameter saves a pointer containing the identifiers of the x-variables in the tree.

Example 3.9.5 saves and prints information about the tree from Example 3.9.3.

Example 3.9.5

```
  37  BRKEEP      Tree; SUMMARY=Summary; XVARIABLES=Xvariables
  38  PRINT       Summary & Xvariables
```

```
                                    Summary

                  Number of nodes       21.00
          Number of terminal nodes      11.00
           Residual sum of squares       0.01
       Residual degrees of freedom       6.00
              Residual mean square       0.00
 Percentage variance accounted for      99.17


   Xvariables
   Production
  Temperature
    Employees
Operatingdays
```

3.10 Quantile regression

Standard regression methods fit models to predict the mean of the probability distribution that generates the observations at each set of values of the explanatory variables. As you will have seen, earlier in this chapter, ordinary regression assumes a Normal distribution (see 3.1 - 3.4), while generalized linear models (3.5) allow for a wider class of distributions, including binomial, Poisson gamma etc. In quantile regression, no parametric probability distribution is assumed, but instead models are fitted to show how the explanatory variables affect the quantiles of the distribution. This allows the distribution to be studied in much more detail, and also provides estimates that are robust against outliers. See Koenker (2005) for more information.

The basic utilities for quantile regression are provided in Genstat by the `FRQUANTILES` directive and `RQOBJECTIVE` function, which are based on the algorithm of Koenker & D'Orey (1987). Procedures have been written to use these utilities to fit various types of quantile regression. Section 3.10.1 describes the `RQLINEAR` procedure, which fits quantile regressions for linear models. Nonlinear models and loess or spline models can be fitted by the similar procedures `RQNONLINEAR` and `RQSMOOTH` (see the *Genstat Reference Manual, Part 3 Procedures* for details).

3.10.1 The `RQLINEAR` procedure

`RQLINEAR` procedure

Fits and plots quantile regressions for linear models (D.B. Baird).

Options

`PRINT` = *list*	What to print (`model`, `estimates`, `summary`, `fittedvalues`, `correlations`, `wald`, `jointqtest`, `separateqtest`); default `mode`, `esti`, `summ`, `wald`
`PLOT` = *list*	What to plot (`rhistogram`, `phistograms`, `fittedvalues`, `estimates`, `bootestimates`); default `rhis`, `phis`, `fitt`
`TERMS` = *formula*	Terms to be fitted
`WEIGHTS` = *variate*	Weights for data values; default equally weighted
`CONSTANT` = *string token*	Whether to include a constant in the model (`omit`, `estimate`); default `esti`
`FACTORIAL` = *scalar*	Limit on number of factors or variates in a term; default 3.
`FITINDIVIDUALLY` = *string token*	Whether to fit the regression model one term at a time (`yes`, `no`); default `no`
`FULL` = *string token*	Whether to assign all possible parameters to factors and

	interactions (yes, no); default no
BMETHOD = *string token*	Bootstrap method (xy, weightedxy); default xy
NBOOT = *scalar*	Number of times to bootstrap data to estimate confidence limits; default 200
SEED = *scalar*	Seed for bootstrap randomization; default 0
CIPROBABILITY = *scalar*	Probability level for confidence interval; default 0.95
XPLOT = *variate*	Variate to plot fitted values against; default 1st variate in model

Parameters

Y = *variates*	Response variate
PRQUANTILES = *scalars* or *variates*	Proportions at which to calculate quantiles; default 0.5
RESIDUALS = *variates* or *pointers*	Residuals from regression for each quantile
FITTEDVALUES = *variates* or *pointers*	Fitted values from regression for each quantile
ESTIMATES = *variates* or *pointers*	Estimated coefficients of model terms for each quantile
SE = *variates* or *pointers*	Standard errors of the estimated coefficients for each quantile
VCOVARIANCE = *symmetric matrices* or *pointers*	Variance-covariance matrix of estimates for each quantile
DF = *scalars* or *variates*	Numbers of degrees of freedom fitted by the model
LOWER = *variates* or *pointers*	Lower confidence limit of coefficients for each quantile
UPPER = *variates* or *pointers*	Upper confidence limit of coefficients for each quantile
LOWFITTEDVALUES = *variates* or *pointers*	Lower confidence limit of fitted values for each quantile
UPPFITTEDVALUES = *variates* or *pointers*	Upper confidence limit of fitted values for each quantile
OBJECTIVE = *scalars* or *variates*	Optimal values of the objective function
EXIT = *scalars* or *variates*	Exit codes indicating whether the estimation was successful

RQLINEAR calculates and plots quantile regressions. The dependent variate is specified by the Y parameter. The proportions (between 0 and 1) for which the model is to be fitted are specified by the PRQUANTILES parameter, as a scalar is there is only one, or a variate if there are several. The default value for PRQUANTILES is 0.5, i.e. the median.

The model defining the explanatory terms is specified by the TERMS option, and can include variates, factors and polynomial terms, and interactions between them. RQLINEAR cannot fit LOESS or SPLINE models. The FACTORIAL, CONSTANT and FULL options control how the model is constructed, as in the ordinary regression commands (see e.g. FIT or TERMS). FACTORIAL option sets a limit on the number of factors and/or variates in each terms, CONSTANT option allows you to omit the constant term, and FULL controls how each term is parameterized.

Output is controlled by the PRINT option with settings:

model	the details of model that is being fitted;
summary	a summary of the fit;
estimates	the model estimates (and confidence limits, standard errors and t-values if bootstrapping is used);
fittedvalues	the residuals and fitted values from the model;
correlation	correlations between the estimates;
wald	Wald Statistic for each model term;
jointqtest	the significance of the joint changes in the model

parameters (excepting the intercept) between the quantiles; and

`separateqtest` the significance of the changes in the individual model parameters between the quantiles.

Correlations and Wald statistics are available only if bootstrapping is done. If option `FITINDIVIDUALLY=yes`, the model terms are added in one at a time, and the Wald statistics are given for for each step. Otherwise only an overall test of the full model versus the null model (i.e. just the constant) is provided. The settings `jointqtest` and `separateqtest` are relevant only if several quantiles have been requested by `PRQUANTILES`. These compare the differences between all the quantiles in a single test. So if you want to compare quantiles for two specific proportions, you should set `PRQUANTILES` to just those two values.

The `PLOT` option controls what plots are displayed, with settings

`rhistogram` histograms of residuals;

`phistograms` histograms of the bootstrap estimates for each parameter;

`fittedvalues` observed and fitted values plotted against the explanatory variate specified by the `XPLOT` option (if `XPLOT` is not set, the first explanatory variate is used);

`estimates` parameter estimates plotted against the quantiles;

`bootestimates` parameter estimates and bootstrap confidence limits plotted against quantiles (note this plot can be slow to produce).

For the fitted plot, the observed and fitted values can be plotted against a specific variate given by the option `XPLOT`, rather than just the default which is the first variate in the `TERMS` statement.

The `BMETHOD` option controls the method that is used to obtain standard errors and confidence limits by bootstrapping for the parameter estimates and fitted values. The `xy` setting re-samples the units with replacement; this is the default. Alternatively, the `weightedxy` setting uses all the units but with weights are generated from a exponential distribution with mean 1. Bootstrapping can be slow, you can set `BMETHOD=*` to stop any being done. The `NBOOT` option specifies the number of bootstrap samples that are taken, and the `CIPROBABILITY` option sets the size of the confidence limits. The `SEED` option defines the seed for the random numbers that are used to select the bootstrap samples. The default of zero continues the existing sequence of random numbers if any have already been used in the current Genstat job. If none have been used, Genstat picks a seed at random.

The results from the model fit can be saved in various parameters. The `ESTIMATES`, `FITTEDVALUES`, `RESIDUALS`, `LOWER`, `UPPER`, `SE`, `LOWFITTEDVALUES` and `UPPFITTEDVALUES` parameters save their results in variates if only one quantile has been defined, or in pointers to a set of variates (one for each quantile) if there were several. Similarly `VCOVARIANCE` saves a symmetric matrix, or a pointer to several symmetric matrices, while `DF`, `OBJECTIVE` and `EXIT` save either a scalar or a variate (with a value for each quantile). `EXIT` saves the value of the exit code from the estimation of each set of regression quantiles by the `FRQUANTILES` directive (which is used inside `RQLINEAR`): a value of zero indicates that the estimation was successful, a value of one means the solution is non-unique (this may not be a problem, as the returned solution will still be optimal), and a value of two means the algorithm has failed.

The example below fits quantile linear regresions to Engel's 1857 data of household food expenditure and household income.

Example 3.10.1

```
2  SPLOAD   '%GENDIR%/Examples/Engel.gsh'
```

```
Loading Spreadsheet File
------------------------

Catalogue of file D:\Gen15ed\Examples\Engel.gsh

Sheet Title:
Engel's 1857 data of household food expenditure and household income,
taken from 235 European working-class households.


Sheet Type: vector
  Index       Type    Nval   Name
      1    variate     235   Income
      2    variate     235   Food_Exp

   3  RQLINEAR [PRINT=#,separateqtest; PLOT=*; TERMS=Income;\
   4           BMETHOD=xy; NBOOT=200; SEED=412261] Food_Exp;\
   5           PRQUANTILES=!(0.1,0.25,0.5,0.75,0.9)

Quantile regression
===================

Response variate: Food_Exp
Fitted terms: Constant + Income

10% Quantile
============

Objective function = 3870
Model degrees of freedom = 1
Residual sum of squares = 6816507
Residual degrees of freedom = 233


Wald statistic
--------------

               Term    d.f.        v.r.     F prob.
          All Terms       1      82.893      <0.001


Parameter estimates
-------------------

             Estimate        s.e.    Lower CI    Upper CI   t-value    t-prob

 Constant      110.14       31.41       56.99      161.29     3.506    <0.001
   Income        0.40        0.04        0.34        0.48     9.105    <0.001

95% Bootstrap confidence intervals based on 200 resamplings.


25% Quantile
============

Objective function = 7082
Model degrees of freedom = 1
Residual sum of squares = 3970500
Residual degrees of freedom = 233


Wald statistic
--------------

               Term    d.f.        v.r.     F prob.
          All Terms       1     206.021      <0.001

Parameter estimates
-------------------
```

	Estimate	s.e.	Lower CI	Upper CI	t-value	t-prob
Constant	95.48	24.68	61.94	169.35	3.869	<0.001
Income	0.47	0.03	0.37	0.51	14.353	<0.001

95% Bootstrap confidence intervals based on 200 resamplings.

50% Quantile
============

Objective function = 8780
Model degrees of freedom = 1
Residual sum of squares = 3402600
Residual degrees of freedom = 233

Wald statistic

Term	d.f.	v.r.	F prob.
All Terms	1	219.452	<0.001

Parameter estimates

	Estimate	s.e.	Lower CI	Upper CI	t-value	t-prob
Constant	81.48	29.25	33.55	159.08	2.785	0.006
Income	0.56	0.04	0.46	0.62	14.814	<0.001

95% Bootstrap confidence intervals based on 200 resamplings.

75% Quantile
============

Objective function = 6529
Model degrees of freedom = 1
Residual sum of squares = 5809136
Residual degrees of freedom = 233

Wald statistic

Term	d.f.	v.r.	F prob.
All Terms	1	402.766	<0.001

Parameter estimates

	Estimate	s.e.	Lower CI	Upper CI	t-value	t-prob
Constant	62.40	25.62	16.277	128.90	2.436	0.016
Income	0.64	0.03	0.564	0.70	20.069	<0.001

95% Bootstrap confidence intervals based on 200 resamplings.

90% Quantile
============

Objective function = 3392
Model degrees of freedom = 1
Residual sum of squares = 8828531
Residual degrees of freedom = 233

Wald statistic

Term	d.f.	v.r.	F prob.
All Terms	1	683.152	<0.001

Parameter estimates

	Estimate	s.e.	Lower CI	Upper CI	t-value	t-prob
Constant	67.35	20.46	20.33	101.76	3.292	0.001
Income	0.69	0.03	0.63	0.74	26.137	<0.001

95% Bootstrap confidence intervals based on 200 resamplings.

Test of equality of parameters across quantiles

Parameter	d.f.	Residual d.f.	F value	F prob.
Constant	4	1171	0.756	0.554
Income	4	1171	11.466	<0.001

4 Analysis of variance and design of experiments

This chapter first describes the Genstat commands for analysis of variance. In Genstat *for Windows*, these analyses can all be specified through the Analysis of Variance menu. The description below, however, provides further insights into the models that are fitted and the output that is obtained. Then, in Sections 4.9 - 4.13, we describe the commands for designing experiments (which are used by the design and sample size menus of Genstat *for Windows*).

Usually the data will be from a designed experiment in which each treatment is applied to several units, such as plots of land, or animal or human subjects, or samples of material. Usually the treatments are allocated randomly, since the units might not be absolutely identical. This guards against any treatment systematically getting more than its fair share of the best units, which might cause it to appear to be better than the treatments on the less favourable units. It is also one form of justification for the statistical analysis. For a more detailed discussion of why randomization is important, see for example Chapter 5 of Cox (1958).

In the simplest type of investigation, the treatments do not have any particular structure. In a field experiment, for example, they may be several varieties of a crop; in an industrial experiment they could be different types of catalyst. In Genstat you represent treatments like these by a factor. The factor has a level for each treatment; the values of the factor indicate which treatment was applied to each unit.

More complicated are factorial experiments. Here there are several different types of treatment, each represented by a different factor. For example, in an investigation of animal diets, you might wish to vary the amounts both of protein and of carbohydrates; in a fertilizer trial, you might have different levels of both nitrogen and phosphorus. Then the set of treatments is the set of all combinations of the levels of the different factors. Thus if there were a levels of nitrogen and b of phosphorus, there would be $a \times b$ treatments altogether.

The advantage of factorial experiments is that you can look not only at the overall effects of each factor, but also at *interactions* which show how the effects of one factor differ according to other factors (4.1). The overall effects are often called *main effects* (though that does not mean that they have to be the main thing that you are interested in). An interaction would be, for example, nitrogen having a large effect in the absence of phosphorus, but only a small effect in its presence.

You specify which main effects, interactions and other treatment terms are to be included in the model using the `TREATMENTSTRUCTURE` directive (4.1.1). You can also do more sophisticated modelling of the effects of factors, by partitioning them (and their interactions) into polynomial or other contrasts (4.5): for example, the yield of a crop might increase linearly with the amount of nitrogen.

There can also be structure in the units themselves. In a simple experiment, they are unstructured: that is, they are assumed to come from a single homogeneous population. The treatments can then be allocated to the units at random, without the need to consider any other groupings of the units. This is called a *completely randomized design* (see 4.1). The analysis of experiments where the units do have an underlying structure is described in 4.2. For example, you might expect there to be less variation among animals from the same litter than among different litters. You specify the structure of the units by the `BLOCKSTRUCTURE` directive; if you omit to do this, Genstat assumes that the units are unstructured.

In an experiment, various measurements will be made to assess how the treatments affect the units. These may be made at the end of the experiment, or while it is still in progress. For example, in a field experiment on potatoes, you might be interested in the yield from each plot, the number of potatoes from each plot, estimates of the percentage areas of potato skin affected by particular diseases, and so on. Analysis of variance allows you to examine only one such measurement at a time. The value measured on each unit (or plot) should be entered into a

variate and analysed by the ANOVA directive (4.1.2).

Sometimes measurements are made before the experiment. For example, the initial blood pressures and other attributes of human subjects might be recorded before the treatments are given. You would want to allow for these baseline readings (or *covariates*) when analysing the effects of the treatments. You specify the variates that are to act as covariates using the COVARIATE directive (4.3.1). By default, the model is assumed to contain no covariates.

The analysis can cope with missing values, either in the variates to be analysed, or in the covariates (4.4). But no factor values should be missing.

So, to summarise, to perform an analysis of variance, you should first define the model to be fitted, using the directives:

BLOCKSTRUCTURE	defines the blocking structure of the design, and hence the strata and error terms (4.2.1)
COVARIATE	specifies a list of covariates for analysis of covariance (4.3.1)
TREATMENTSTRUCTURE	defines the treatment (or systematic) terms (4.1.1)

For unstructured designs with a single error term, BLOCKSTRUCTURE need not be specified, and COVARIATE is needed only for analysis of covariance. Section 4.3 gives a full description of analysis of covariance, and also describes the AFCOVARIATES procedure which allows you to define more complicated types of covariate models (4.3.2).

Once the model has been defined, the y-variates can be analysed using the ANOVA directive:

ANOVA	performs analysis of variance (4.1.2)

Then, after you have fitted the model, you can display or calculate further results, check the assumptions of the analysis, plot means and residuals, or store the results in data structures for use elsewhere in Genstat:

ADISPLAY	displays further output from analyses produced by ANOVA (4.1.3)
AFMEANS	forms tables of means classified by ANOVA treatment factors (4.1.5)
AGRAPH	plots tables of means from ANOVA (4.1.5)
APLOT	plots residuals from an ANOVA analysis (4.1.4)
AFIELDRESIDUALS	display residuals from a field experiment in field layout (4.1.4)
ABLUPS	calculates BLUPs for block terms in an ANOVA analysis (4.2.2)
ACHECK	checks the assumptions for an ANOVA analysis (4.1.6)
AKEEP	copies information from an ANOVA analysis into Genstat data structures (4.6.1)
APERMTEST	performs random permutation and exact tests for analysis of variance (4.1.7)
APOLYNOMIAL	calculates the equation for a polynomial contrast fitted by ANOVA (4.5.1)
ADPOLYNOMIAL	plots single-factor polynomial contrasts fitted by ANOVA (4.5.2)
ARESULTSUMMARY	provides a summary of results from an ANOVA analysis (4.1.3)
ASPREADSHEET	saves analysis of variance results in a spreadsheet (4.6.3)
ASTATUS	provides information about the settings of ANOVA models and variates (4.6.2)
AMCOMPARISON	performs pairwise multiple comparison tests for ANOVA means (4.1.9)

AMDUNNETT	forms Dunnett's simultaneous confidence interval around a control (4.1.10)
ACONFIDENCE	calculates simultaneous confidence intervals (4.1.8)
A%VARIANCE	calculates the percentage variance and sum of squares accounted for in the strata of an ANOVA analysis
FALIASTERMS	forms information about aliased model terms in analysis of variance

The designs that can be analysed by ANOVA are said to be *balanced* or, more accurately, to have the property of *first-order balance* defined by Wilkinson (1970) and James & Wilkinson (1971). A brief explanation of the property is given in 4.7, where the method of analysis is explained, but you do not need to understand this in order to use Genstat. Virtually all the standard designs can be analysed, including all the generally-balanced designs (Nelder 1965 a,b; Payne & Tobias 1992). Here are some examples:

(a) all orthogonal designs, whether with a single error term or with several: for example, completely randomized designs, randomized blocks, split plots, Latin and Graeco-Latin squares, split-split plots and fractional replicates;
(b) all designs with balanced confounding: for example, balanced incomplete blocks, balanced lattices and Youden squares;
(c) designs with partial balance, provided the pattern of balance can be specified by pseudo-factors (4.7.3).

Amongst the worked examples available on your computer are data files showing how to analyse all the worked examples in Cochran & Cox (1957); this should cover most of the designs that you are likely to encounter. Genstat itself detects whether or not your design is balanced, by a process known as the *dummy analysis* (4.7.5). So, if you are unsure about whether or not a particular design can be analysed, try it and see what happens. Unbalanced designs with a single error term can be analysed using procedures AUNBALANCED, AUDISPLAY and AUKEEP. The model is specified just as for ANOVA but the analysis uses the Genstat regression facilities (see Chapter 3). If you have only two treatment factors in an unbalanced design with a single error term, it may be more convenient to use A2WAY. Unbalanced designs with several error terms can be analysed by the REML directive (Chapter 5). However, if the additional random terms contain very little information about the treatments, it may be more convenient (and equally effective) to treat these as fixed nuisance terms, and use AUNBALANCED. Decisions like this can be made using the AOVANYHOW procedure. Also, procedures GLMM (3.5.10) and HGANALYSE (3.5.11) provide methods for fitting generalized linear models with additional random terms to model data from a stratified experiment.

AUNBALANCED	performs analysis of variance for unbalanced designs (4.8.1)
AUDISPLAY	produces further output for an unbalanced design (4.8.2)
AUGRAPH	plots tables of means from an unbalanced design (4.8.3)
AUKEEP	saves output from analysis of an unbalanced design (4.8.4)
AUPREDICT	forms predictions from an unbalanced design (4.8.5)
AUSPREADSHEET	Saves results from an analysis of an unbalanced design in a spreadsheet (4.8.6)
AUMCOMPARISON	performs pairwise multiple comparison tests for means from unbalanced designs
A2WAY	performs analysis of variance of a balanced or unbalanced design with up to two treatment factors (2.3.3)
A2DISPLAY	provides further output following an analysis of variance by A2WAY (2.3.3)
A2KEEP	copies information from an A2WAY analysis into Genstat data structures (2.3.3)

AOVANYHOW	performs analysis of variance using ANOVA, AUNBALANCED, A2WAY or REML as appropriate (4.8.7)
AOVDISPLAY	provides further output from an analysis by AOVANYHOW
AN1ADVICE	aims to give useful advice if a design that is thought to be balanced fails to be analysed by ANOVA (4.8.8)

There are several other directives and procedures that you may find useful during an analysis of variance. You can use the RESTRICT directive (4.4.1) to restrict the analysis to only a subset of the units. You can specify how many decimal places will be used in the output of tables of means, effects, contrasts and residuals by setting the DECIMALS parameter in the declaration of the variate to be analysed (2.1.2). Procedure VHOMOGENEITY can be used to check the homogeneity of variances, and procedure AREPMEASURES (8.1.3) can check the validity of the ordinary analysis of variance if you have repeated measurements. If ordinary anova cannot be used, alternatives are provided by procedures MANOVA (multivariate analysis of variance; 6.6.1), or by procedures ANTORDER and ANTTEST (antedependence structure; 8.1.5), or by modelling the covariance structure over time by REML (Chapter 5).

Other procedures relevant to analysis of variance, in the Procedure Library, include:

AMTIER	analyses a multitiered design specified by up to three model formulae (4.2.3)
AMTDISPLAY	displays further output for multitiered designs (4.2.3)
AMTKEEP	saves information from the analysis of a multitiered design (4.2.3)
VSPECTRALCHECK	forms the spectral components from the canonical components of a multitiered design, and constrains any negative spectral components to zero
ASCREEN	performs screening tests for designs with orthogonal block structure (4.7.6)
AONEWAY	provides one-way analysis of variance (2.3.2)
APAPADAKIS	analysis of variance with an added Papadakis covariate, formed from neighbouring residuals
A2WAY	performs analysis of variance of a balanced or unbalanced design with up to two treatment factors (2.3.3)
A2DISPLAY	provides further output from an A2WAY analysis (2.3.3)
A2KEEP	saves information from an A2WAY analysis (2.3.3)
ABIVARIATE	produces graphs and statistics for bivariate analysis of variance
ABOXCOX	estimates the power λ in a Box-Cox transformation, that maximizes the partial log-likelihood in ANOVA
ACANONICAL	determines the orthogonal decomposition of the sample space for a design, using an analysis of the canonical relationships between the projectors derived from two or more model formulae.
ACDISPLAY	provides further output from an analysis by ACANONICAL.
ACKEEP	saves information from an analysis by ACANONICAL.
AMMI	provides exploratory analysis of genotype × environment interactions
FMEGAENVIRONMENTS	forms mega-environments based on winning genotypes from an AMMI-2 model
STEEL	performs Steel's many-one rank test (2.6.3)
TEQUIVALENCE	performs equivalence, non-inferiority and non-superiority tests
AREPMEASURES	produces an analysis of variance for repeated

	measurements (8.1.3)
ARETRIEVE	retrieves an ANOVA save structure from an external file
ASTORE	stores an ANOVA save structure in an external file
AYPARALLEL	does the same analysis of variance for several y-variates, and collates the output
A2PLOT	plots effects from two-level designs with robust s.e. estimates
A2RDA	saves results from an analysis of variance in R data frames
AU2RDA	saves results from an unbalanced analysis of variance, by AUNBALANCED, in R data frames
CENSOR	pre-processes censored data before analysis by ANOVA
CINTERACTION	clusters rows and columns of a two-way interaction table
DIALLEL	analyses full and half diallel tables with parents
FRIEDMAN	performs Friedman's nonparametric analysis of variance (2.6.2)
LVARMODEL	analyses a field trial using the Linear Variance Neighbour model
MAANOVA	does a parallel analysis of variance of data from a single-channel microarray design
NLCONTRASTS	fits non-linear contrasts to quantitative factors in ANOVA
SED2ESE	calculates effective standard errors that give good approximate standard errors of differences
SEDLSI	calculates least significant intervals
LSIPLOT	plots least significant intervals, saved from SEDLSI
VHOMOGENEITY	tests homogeneity of variances
WSTATISTIC	calculates the Shapiro-Wilk test for Normality

Full details can be found in Part 3 of the *Genstat Reference Manual*.

Genstat has a comprehensive set of facilities for design of experiments ranging from procedures that allow you to select and generate a design from an extensive repertoire of possibilities, to directives and procedures that enable you to develop new designs and assess their properties. Collectively, these are known as the *Genstat Design System*. Many different design types are covered, each with a procedure that allows you to view and choose from the available possibilities. Other procedure allow designs and data forms to be displayed. There is also a general procedure DESIGN that can be used interactively to provide a single point of access to all the design types.

DESIGN	provides a menu-driven interface for selecting and generating experimental designs (4.9.1)
AGALPHA	forms alpha designs for up to 100 treatments (4.9.7)
AGBIB	generates balanced-incomplete-block designs (4.9.8)
AGBOXBEHNKEN	generates Box-Behnken designs (4.9.12)
AGCENTRALCOMPOSITE	generates central composite designs (4.9.11)
AGCROSSOVERLATIN	generates Latin squares balanced for carry-over effects (4.9.4)
AGCYCLIC	generates cyclic designs from standard generators (4.9.9)
AGDESIGN	generates generally-balanced designs – factorial designs with blocking, fractional factorial designs, Lattice squares etc. (4.9.3)
AGFACTORIAL	generates minimum aberration complete and fractional factorial designs (4.9.2)
AGFRACTION	generates fractional factorial designs
AGHIERARCHICAL	generates orthogonal hierarchical designs (4.9.1)

AGLATIN	generates mutually orthogonal Latin squares (4.9.4)
AGLOOP	generates loop designs e.g. for time-course microarray experiments (4.9.17)
AGMAINEFFECT	generates designs to estimate main effects of two-level factors (4.9.13)
AGNEIGHBOUR	generates neighbour-balanced designs (4.9.10)
AGNONORTHOGONALDESIGN	generates non-orthogonal multi-stratum designs
AGSPACEFILLINGDESIGN	generates space filling designs
AGQLATIN	generates complete and quasi-complete Latin squares (4.9.4)
AGREFERENCE	generates reference-level designs e.g. for microarray experiments (4.9.16)
AGSEMILATIN	generates semi-Latin squares (4.9.5)
AGSQLATTICE	generates square lattice and lattice square designs (4.9.6)
AGYOUDENSQUARE	generates a Youden square
PDESIGN	prints treatment combinations tabulated by the block factors (4.10.1)
DDESIGN	plots the plan of a design (4.10.2)
ADSPREADSHEET	puts the data and plan of an experimental design into a spreadsheet (4.10.3)

DESIGN and the AG... procedures (above) that it calls provide the Select Design facilities in Genstat *for Windows*, while the alternative Standard Design menu uses AGHIERARCHICAL, AGLATIN and AGSQLATTICE to generate completely randomized designs, randomized blocks, Latin and Graeco-Latin squares, split-plots, strip-plots (or criss-cross designs) and lattices.

There are also procedures that you can use to determine the sample size (i.e. replication) required for experiments that are to be analysed by analysis of variance, t-test or various non-parametric tests. You can also calculate the power (or probability of detection) for terms in analysis of variance or regression analyses.

APOWER	calculates the power (probability of detection) for terms in an analysis of variance (4.12.3)
ASAMPLESIZE	finds the replication (sample size) to detect a treatment effect or contrast (4.12.2)
RPOWER	calculates the power (probability of detection) for regression models (3.1.8)
ADETECTION	calculates the minimum size of effect or contrast detectable in an analysis of variance (4.12.4)
SBNTEST	calculates the sample size for binomial tests (4.12.5)
SPNTEST	calculates the sample size for Poisson tests (4.12.6)
SCORRELATION	calculates the sample size to detect specified correlations (4.12.10)
SLCONCORDANCE	calculates the sample size for Lin's concordance coefficient (4.12.11)
SMANNWHITNEY	calculates the sample size for the Mann-Whitney test (4.12.9)
SMCNEMAR	calculates the sample size for McNemar's test (4.12.8)
SPRECISION	calculates the sample size to obtain a specified precision
SSIGNTEST	calculates the sample size for a sign test (4.12.7)
STTEST	calculates the sample size for t-tests, including equivalence tests and tests for non-inferiority (4.12.1)

The design-generation procedures form and randomize the designs automatically, calling other

directives and procedures to perform the necessary tasks, so there is no need for you to be aware of any of the details. However, we give more information, below and in Sections 4.9 - 4.11 and 4.13, if you do want to study the process in more depth or to add new designs. Briefly, the Design System is based on a range of standard generators. Some of these, such as the Galois fields used to generate Latin squares or the Hadamard matrices needed for main-effect designs, can be formed when required – and so there is no limitation on the available designs. Repertoires of others, such as design keys, are stored in backing-store files which are scanned by the design generation procedures to form menus listing the available possibilities. Algorithms are available to form generators for new designs, and these can then be added to the design files to become an integral part of the system. Other design utilities include procedures for combining simple designs into more complicated arrangements, for constructing augmented designs, and for determining how many replicates are needed. There are also directives for constructing response-surface designs using the BLKL algorithm of Atkinson & Donev (1992) and for constructing doubly resolvable row-column designs. The relevant commands include the directives

AFRESPONSESURFACE	uses the BLKL algorithm to construct designs for estimating response surfaces (4.9.14)
AGRCRESOLVABLE	forms doubly resolvable row-column designs
GENERATE	generates values of factors in systematic order or as defined by a design key, or forms values of pseudo-factors (4.13.1)
RANDOMIZE	puts units of vectors into random order, or randomizes units of an experimental design (4.11.1)
FKEY	forms design keys for multi-stratum experimental designs, allowing for confounding and aliasing of treatments (4.13.6)
FPSEUDOFACTORS	determines patterns of confounding and aliasing from design keys, and extends the treatment formula to incorporate the necessary pseudo-factors (4.13.7)
SET2FORMULA	forms a model formula using structures supplied in a pointer (1:4.8.3)

and the procedures

AEFFICIENCY	calculates efficiency factors for experimental designs
AFALPHA	generates alpha designs (4.9.7)
AFAUGMENTED	forms an augmented design (4.13.5)
AFCARRYOVER	forms factors to represent carry-over effects in cross-over trials
AFCYCLIC	generates block and treatment factors for cyclic designs (4.9.9)
AFLABELS	forms a variate of unit labels for a design
AFNONLINEAR	forms D-optimal designs to estimate the parameters of a nonlinear or generalized linear model (4.9.15)
AFPREP	searches for an efficient partially-replicated design
AFUNITS	forms a factor to index the units of the final stratum of a design
AFRCRESOLVABLE	forms doubly resolvable row-column designs, with output
AKEY	generates values for treatment factors using the design key method (4.13.2)
AMERGE	merges extra units into an experimental design (4.13.3)
APRODUCT	forms a new experimental design from the product of two designs (4.13.4)

ARANDOMIZE	randomizes and prints an experimental design (4.11.2)
COVDESIGN	produces experimental designs efficient under analysis of covariance
FACDIVIDE	represents a factor by factorial combinations of a set of factors
FACPRODUCT	forms a factor with a level for every combination of other factors
FBASICCONTRASTS	forms the basic contrasts of a model term (4.13.8)
FCOMPLEMENT	forms the complement of an incomplete block design
FDESIGNFILE	forms a backing-store file of information for AGDESIGN
FHADAMARDMATRIX	forms Hadamard matrices
FOCCURRENCES	forms a "concurrence" matrix recording how often each pair of treatments occurs in the same block of a design
FPLOTNUMBER	forms plot numbers for a row-by-column design
FPROJECTIONMATRIX	forms a projection matrix for a set of model terms
XOEFFICIENCY	calculates the efficiency for estimating of effects in cross-over designs
XOPOWER	estimates the power of contrasts in cross-over designs

4.1 Designs with a single error term

Suppose that you have done an experiment to examine v different treatments, and that the value measured on the jth unit out of r receiving treatment i is y_{ij}. For each treatment i, we suppose that there is an underlying mean value of y that we wish to estimate; we shall write this as m_i. This will not be the value observed because there will be measurement error, there may be uncontrolled differences in the way the different units have been dealt with, and the units themselves may not be uniform. So y_{ij} is assumed to follow the linear model

$$y_{ij} = m_i + \varepsilon_{ij}$$

where ε_{ij}, termed the *residual* for the ijth unit, represents the difference between the true value m_i and the value actually observed. The residuals are assumed to be independently distributed: that is, the size of the residual on one unit is assumed to be unaffected by the residuals on other units. They are also assumed to have a zero mean and a constant variance (so the expected value for the ijth unit is m_i). For some of the properties of analysis of variance, it is necessary to assume also that the residuals each have a Normal distribution.

The process by which values for the parameters m_i are estimated from the observed measurements y_{ij} is known as *least squares*. The estimators $\hat{m}_i$ are chosen to minimize the sum of squares of the estimated residuals:

$$RSS = \sum_{i=1}^{v} \sum_{j=1}^{r} (y_{ij} - \hat{m}_i)^2$$

You can find details of this process in any standard statistical textbook. For a simple design like this one, the estimate of each mean, $\hat{m}_i$, is simply the average of the values observed on the units with treatment i. However this may not be so in more complicated experiments, for example where there is non-orthogonality (4.7) or where there are covariates (4.3). In such cases Genstat uses the term *mean* to denote the prediction of the mean value for a treatment, rather than its crude average, and we follow the same convention in this chapter.

Analysis of variance also estimates the uncertainty attached to the estimates of the parameters, allowing you to assess whether the treatments genuinely differ in their effects. In simple cases, this involves assessing whether the variation between the units with different treatments is genuinely greater than that between units with the same treatment. To help investigate this, a more common form of the linear model is

$$y_{ij} = \mu + e_i + \varepsilon_{ij}$$

where μ is known as the *grand mean*, and e_i as the effect of treatment i. So:

$$m_i = \mu + e_i$$

If the treatments do not differ, the effects (e_i, $i = 1 \ldots v$) will all be zero. To assess this we would fit first a model containing just the grand mean (and residuals), and then a model with the effects as well. The difference between the residual sums of squares of these two models measures whether the treatments differ: this difference is called the sum of squares due to treatments. Conventionally the different sums of squares are presented in a table known as the analysis-of-variance table.

The example below shows the analysis-of-variance table for a rather more complicated experiment, details of which can be found in Snedecor & Cochran (1980, page 305); further output is shown in 4.1.3 and 4.5. The experiment studies the effect of diet on the weight gains of rats. There were six treatments arising from two treatment factors: the source of protein (beef, pork or cereal), and its amount (high or low). The 60 rats that provided the experimental units were allocated at random into six groups of ten rats, one group for each treatment combination. The model to be fitted in the analysis contains three terms to explain the effects of the treatments: s_i (i = 1,2,3) the main effects of the source of protein (beef, pork or cereal); a_j (j = 1,2) the main effects of the amount of protein (high or low); and sa_{ij} the interaction between source and amount of protein.

$$y_{ijk} = \mu + s_i + a_j + sa_{ij} + \varepsilon_{ijk}$$

The parameters a_j make the same contribution to the model irrespective of the source of the protein received by the rat. So they represent the overall effects of the amount of protein. Similarly, the parameters s_i represent the overall effects of the source of protein. If the interaction effects were all zero, we would have a model in which the difference between high and low amounts of protein was the same whatever the source of the protein. Also, the difference between sources of protein would be identical whether at high or low amounts. So the parameters sa_{ij} indicate whether or not these two factors interact: whether we can determine the best source of protein without regard to its amount; likewise whether we can decide the best amount without considering the source. The estimates of the parameters are included in the output under the heading "Tables of effects" (4.1.3).

Genstat prints the analysis-of-variance table in the conventional form, which you can find in statistical textbooks: there is a line for each treatment term, a line for the residual, and a final "Total" line recording the total sum of squares after fitting the grand mean. The first column, "d.f." standing for *degrees of freedom*, records the number of extra independent parameters included when each term is added into the model; thus with the source of protein, there are three parameters (s_1, s_2, s_3) but, since the grand mean μ has already been fitted, they sum to zero and so the degrees of freedom are two. (A full explanation of this too can be found in statistical textbooks.) The second column "s.s." contains the sums of squares. The column "m.s.", standing for *mean square*, has sums of squares divided by numbers of degrees of freedom. You can assess whether a particular treatment term has had an effect by comparing its mean square with the residual mean square: if there has been an effect, then the mean square for the treatment term will be large compared to the residual. The column denoted "v.r." (for *variance ratio*) helps you make these comparisons: it contains the ratio of each treatment mean square to the residual mean square. If the residuals do indeed have independent Normal distributions with zero mean and equal variance, then each such ratio has an F distribution with t and r degrees of freedom, where t is the number of degrees of freedom of the treatment term and r is the number of degrees of freedom of the residual. The corresponding probabilities can be looked up in statistical tables, or you can ask Genstat to calculate them for you (see the column headed "F pr."), by setting option `FPROBABILITY=yes` in the `ANOVA` or `ADISPLAY` directives. However you should not interpret these probabilities too rigidly, as the assumptions are rarely more than approximately satisfied; for this reason, Genstat does not print probabilities less than 0.001, but will put "<.001"

instead. Also, you should not merely report that a term in an analysis is significant; you should also study its means or its effects to see what their biological (or economic) importance may be, whether their pattern can be explained scientifically, and so on.

Example 4.1

```
  2  " 3x2 factorial experiment (Snedecor & Cochran 1980, p.305)."
  3  UNITS [NVALUES=60]
  4  FACTOR [LABELS=!T(beef,cereal,pork); VALUES=(1...3)20] Source
  5  & [LABELS=!T(high,low); VALUES=3(1,2)10] Amount
  6  READ Gain

   Identifier   Minimum      Mean   Maximum    Values   Missing
         Gain     49.00     87.87     120.0        60         0

 17  TREATMENTSTRUCTURE Source*Amount
 18  ANOVA [PRINT=aovtable; FPROBABILITY=yes] Gain

Analysis of variance
====================

Variate: Gain

Source of variation     d.f.       s.s.       m.s.    v.r.  F pr.
Source                     2      266.5      133.3    0.62  0.541
Amount                     1     3168.3     3168.3   14.77  <.001
Source.Amount              2     1178.1      589.1    2.75  0.073
Residual                  54    11586.0      214.6
Total                     59    16198.9
```

Before you can do the analysis you must set up factors to define the treatment that was applied to each unit. Here there are two factors, for source and for amount of protein. Also you must form a variate containing the data values y_{ijk} that are to be analysed.

For the analysis of variance, you must first define the model to be fitted. Here we have a single error term ε_{ijk}: the units have no structure. Consequently you need not give a `BLOCKSTRUCTURE` statement (4.2.1) but can let it take its default value. If you have already defined some other structure (perhaps for an earlier analysis), you should cancel it by giving either a `BLOCKSTRUCTURE` statement with a null formula, or else one with a single factor indexing the units (4.2.1). Provided you have no covariates (4.3), the only statement that you need give is `TREATMENTSTRUCTURE`.

4.1.1 The `TREATMENTSTRUCTURE` directive

`TREATMENTSTRUCTURE` directive

Specifies the treatment terms to be fitted by subsequent `ANOVA` statements.

No options

Parameter

formula	Treatment formula, specifies the treatment model terms to be fitted by subsequent `ANOVA`s

The single unnamed parameter of the `TREATMENTSTRUCTURE` directive is a formula known as the *treatment formula*. Formulae (1.5.3) are composed of identifier lists and functions, separated by the operators:

```
.   +   *   /   //   -   -*   -/
```

In the formulae for analysis of variance, the identifier lists can only be of factors. Variates and

matrices can appear in the functions (to fit polynomials, for example); these are described in 4.5. Here we describe the first four operators, which are those that are used most often. The pseudo-factorial operator //, which occurs only in treatment formulae, is described in 4.7.3. The final three operators are for deletion. Full definitions of all the operators are in 1:1.6.3.

Genstat expands a formula into a series of model terms, linked by the operator plus (+). Each model term consists of one or more elements, separated from one another by the operator dot (.); in analysis of variance the elements are either factors or functions. You can always specify a formula in this expanded form: the other operators simply provide a more succinct way of writing long formulae. For the formulae defined by `TREATMENTSTRUCTURE` and by `BLOCKSTRUCTURE` (4.2.1), this expansion does not take place until the analysis is being done (by `ANOVA`). `TREATMENTSTRUCTURE` and `BLOCKSTRUCTURE` merely store the formulae in their original form. Consequently there are some syntactic errors that will not be found until the `ANOVA` statement. When Genstat does the expansion, the `FACTORIAL` option of `ANOVA` sets a limit on the number of elements in a model term from the treatment formula: any terms with more elements are deleted.

Each model term in the treatment formula corresponds to a treatment term in the linear model. The expanded version of the formula in line 17 of the example is

```
Source + Amount + Source.Amount
```

(So you could have specified this instead of `Source*Amount`.) Terms with a single factor represent main effects of the factor: for example `Source` corresponds to the main effects of the source of protein, s_i. Terms with several factors define higher-order effects: for example `Source.Amount` corresponds to the interaction effects between source and amount of protein, sa_{ij}. However the meaning of a higher-order term depends on the context: in general, it refers to all those joint effects of the factors in the term that have not been accounted for by preceding terms in the model. So `Source.Amount`, above, is an interaction because the main effects of source and amount have both been fitted already. But, in the formula

```
A + A.B
```

there are no main effects of `B`, merely a_i and ab_{ij}, so `A.B` denotes the fitting of different `B` effects for each level of `A`; these are usually called the *B-within-A* effects.

Any redundant terms in a formula are deleted. So, for example,

```
A + B + A
```

becomes

```
A + B
```

Also, as `A.B` is defined to include all the joint effects of `A` and `B` that are not yet accounted for, the formula

```
A.B + B
```

becomes just `A.B`, which already includes the `B` main effects. Thus the order in which you specify the terms is important.

The operators `*` and / are termed the crossing and nesting operators respectively. For example,

```
Source * Amount
```

defines the factors `Source` and `Amount` to have a crossed relationship: that is, we wish to examine the effects of each factor individually, and then their interaction. Models containing only crossing are often called factorial models. Another factorial model, but with three factors, is in 4.7.1: the formula is

```
N * K * D
```

which expands to

```
N + K + D + N.K + N.D + K.D + N.K.D
```

including not only two-factor interactions, like `N.K`, but also the three-factor interaction `N.K.D`.

In general, if L and M are two formulae, the definition (1:1.6.3) is that

```
L*M = L + M + L.M
```

Nesting (/) occurs most often in block formulae, which are specified by the BLOCKSTRUCTURE directive (4.2). To take an illustration from later in this chapter, for the example analysed in 4.3 the formula is

```
Blocks / Plots
```

indicating that plots are nested within blocks; so the interest is in block effects and the effects of plots within blocks (see 4.2.1). This is exactly what the operator / provides: the expanded form of the formula is

```
Blocks + Blocks.Plots
```

The general definition of the slash operator (1:1.6.3) is that

```
L/M = L + L.M
```

where *L* is a model term containing all the factors that occur in L. (The rationale for this is that if M is nested within all the terms in L, it must be nested within all the factors in L.) For example, if you expand the first operator in the formula

```
Blocks/Wplots/Subplots
```

used to specify a split-plot design (4.2.1), you obtain

```
(Blocks + Blocks.Wplots)/Subplots
```

This then expands to

```
Blocks + Blocks.Wplots + Blocks.Wplots.Subplots
```

(which, reassuringly, gives an identical list of terms to those obtained by expanding the second operator before the first operator).

An example of a treatment formula in which there is nesting is the factorial plus added control

```
Fumigant/(Dose*Type) = Fumigant + Fumigant.Dose
                     + Fumigant.Type + Fumigant.Dose.Type
```

in which the factorial combinations of dose and type occur within the 'fumigated' level of the factor Fumigant, as explained in 4.3.

The definition of the operator dot (.) with model formulae L and M is that L.M is the sum of all pairwise combinations of a term in L with a term in M. For example

```
(A + B.C).(D + E) = A.D + A.E + B.C.D + B.C.E
```

After expanding the operators dot (.), star (*) and slash (/), Genstat rearranges the list of model terms so that the numbers of factors in the terms are in increasing order. Where several terms contain the same numbers of factors, the terms are put into lexicographical order according to the order in which the factors first appeared in the formula. For example

```
(A + C.D + B + A.B) * E
```

expands to

```
A + C.D + B + A.B + E + A.E + C.D.E + B.E + A.B.E
```

which is reordered to

```
A + B + E + A.B + A.E + C.D + B.E + A.B.E + C.D.E
```

4.1.2 The ANOVA directive

Once you have defined the model, you can analyse the variates containing the data (the *y-variates*) using ANOVA. All the options and parameters are listed here, although some are relevant only to the more complicated designs and analyses described later in this chapter.

ANOVA directive

Analyses y-variates by analysis of variance according to the model defined by earlier BLOCKSTRUCTURE, COVARIATE and TREATMENTSTRUCTURE statements.

PRINT = *string tokens*	Output from the analyses of the y-variates, adjusted for any covariates (aovtable, information, covariates, effects, residuals, contrasts, means, cbeffects, cbmeans, stratumvariances, %cv, missingvalues); default aovt, info, cova, mean, miss
UPRINT = *string tokens*	Output from the unadjusted analyses of the y-variates (aovtable, information, effects, residuals, contrasts, means, cbeffects, cbmeans, stratumvariances, %cv, missingvalues); default * i.e. no printing
CPRINT = *string tokens*	Output from the analyses of the covariates, if any (aovtable, information, effects, residuals, contrasts, means, %cv, missingvalues); default * i.e. no printing
FACTORIAL = *scalar*	Limit on number of factors in a treatment term; default 3
CONTRASTS = *scalar*	Limit on the order of a contrast of a treatment term; default 4
DEVIATIONS = *scalar*	Limit on the number of factors in a treatment term for the deviations from its fitted contrasts to be retained in the model; default 9
PFACTORIAL = *scalar*	Limit on number of factors in printed tables of means or effects; default 9
PCONTRASTS = *scalar*	Limit on order of printed contrasts; default 9
PDEVIATIONS = *scalar*	Limit on number of factors in a treatment term whose deviations from the fitted contrasts are to be printed; default 9
FPROBABILITY = *string token*	Printing of probabilities for variance ratios (yes, no); default no
PSE = *string token*	Standard errors to be printed with tables of means, PSE=* requests s.e.'s to be omitted (differences, lsd, means); default diff
TWOLEVEL = *string token*	Representation of effects in 2^n experiments (responses, Yates, effects); default resp
DESIGN = *pointer*	Stores details of the design for use in subsequent analyses; default *
WEIGHTS = *variate*	Weights for each unit; default * i.e. all units with weight one
ORTHOGONAL = *string token*	Whether or not design to be assumed orthogonal (notassumed, assumed, compulsory); default nota
SEED = *scalar*	Seed for random numbers to generate dummy variate for determining the design; default 12345
MAXCYCLE = *scalar*	Maximum number of iterations for estimating missing values; default 20
TOLERANCES = *variate*	Allows you to redefine the tolerances for zero used by various parts of the algorithm
NOMESSAGE = *string tokens*	Which warning messages to suppress (nonorthogonal,

	residual); default *
LSDLEVEL = *scalar*	Significance level (%) to use in the calculation of least significant differences; default 5
EXIT = *scalar*	Saves an exit code indicating the properties of the design

Parameters

Y = *variates*	Variates to be analysed
RESIDUALS = *variates*	Variate to save residuals for each y variate
FITTEDVALUES = *variates*	Variate to save fitted values
SAVE = *identifiers*	Save details of each analysis for use in subsequent ADISPLAY or AKEEP statements

Before Genstat does any calculations with the y-variates, it does an initial investigation to acquire all the information that it needs for the analysis. Alternatively, you can supply this from an earlier analysis using the DESIGN option.

During this initial investigation Genstat first generates the model, excluding covariates (4.3), by expanding the block and treatment formulae into a list of model terms (4.1.1). For a design with a single error term, you do not have to define the block formula; its use in the definition of more complicated designs is described in 4.2.1. Genstat also finds out whether the treatment formula contains any functions and, if so, forms the contrasts that they define (4.5).

The treatment terms to be included in the model are controlled by the options FACTORIAL, CONTRASTS and DEVIATIONS. FACTORIAL sets a limit on the number of factors in a treatment term: terms containing more than that number are deleted. CONTRASTS and DEVIATIONS control the inclusion of contrasts, and of deviations from fitted contrasts (4.5). The maximum number of different factors that you can have in the block and treatment formulae is 1053, but special versions of Genstat can be formed for anyone that needs more than this!

Genstat then checks whether any of the y-variates is restricted (1:4.4.1). If several variates are restricted, they must all be restricted to the same set of units. Only these units are included in the analysis of each y-variate.

Next Genstat investigates the design: for example, it checks whether each term can be estimated, whether any are non-orthogonal (4.7), which error term is appropriate for each estimated treatment term if the model contains several, and indeed whether the design has the balance required for ANOVA to analyse it. This process, known as the *dummy analysis*, involves the analysis of a specially generated variate which contains random numbers from a Cauchy distribution. The starting value for their generation is set by the SEED option. The full details are described in 4.7.5 (but you do not have to understand how this works in order to use ANOVA).

The WEIGHT option allows you to specify a weight for each unit, to define a weighted analysis of variance. You might want to do this if, for example, different parts of the experiment have different variability; each weight would then be proportional to the reciprocal of the expected variance for the corresponding unit. However unless the weights are fairly systematic, for example to give proportional weighted replication (4.5), the design is unlikely to be balanced.

Genstat has a simplified version of the dummy analysis which you can use to save computing time if all the model terms are orthogonal and if, for every term, all the combinations of its factors were applied to the same number of units (4.7.5). A check is incorporated which will detect non-orthogonality except in particularly complicated designs where terms are aliased. If you set option ORTHOGONAL=assumed, Genstat does the simple version unless non-orthogonality is detected, whereupon it gives a warning message and then switches to the full version. The simplified version is done also if ORTHOGONAL=compulsory, but non-orthogonality now causes the analysis to stop altogether, with an error message; this is useful for checking for typing errors in the factor values when you know that the design should otherwise be orthogonal.

The TOLERANCES option can supply a variate with up to four values to define tolerances for zero in various parts of the analysis: the first is used to calculate the tolerance for the analysis of the y-variates (default 10^{-7}), the second is for the tolerance used in the dummy analysis (default 10^{-9}; see 4.7.5), the third is for the estimation of missing values (default 10^{-5}; see 4.4) and the fourth is for the estimation of stratum variances (default 10^{-5}; see 4.7.1).

You can use the DESIGN option to store the details of the model, the design and any restrictions of the units, so that Genstat need not recalculate them for future ANOVA statements. The setting of the option is automatically declared as a pointer if you have not declared it already. It points to several other structures which store information about different aspects of the analysis. The only other details that are required for future analyses are the values of the factors in the block and treatment formulae.

If you have not previously declared the design structure, or if it has no values, then the current statement derives and stores the necessary information. If the pointer does already have values, then these are used to do the analysis. In that case, of course, values of the factors in the block and treatment formulae must not have been changed since the design structure was formed. The current settings of options FACTORIAL, CONTRASTS, DEVIATIONS and WEIGHT are then ignored, as is any change in the restrictions on the y-variates. The DESIGN option is particularly useful with designs where there are many model terms or where there is non-orthogonality, as the dummy analysis may then be time-consuming.

The MAXCYCLE option, which sets a limit on the number of iterations for estimating missing values, is described in 4.4. The EXIT option is described in 4.7.5. The other ANOVA options control the printed output, and are described with the ADISPLAY directive (4.1.3).

The first parameter of ANOVA, Y, lists the variates whose values are to be analysed. Genstat examines them all and forms a list of units for which any of the y-variates or any covariate (4.3) has a missing value. These units are treated as missing in all the analyses. (This is necessary to avoid having to re-analyse covariates for each y-variate; analysis of covariance is described in 4.3.) However, if your y-variates have different missing units, you may prefer to analyse them with separate ANOVA statements, while saving details of the model and design with the DESIGN option to improve efficiency (see 4.4).

The RESIDUALS parameter allows you to specify a variate to save the estimated residuals from each analysis. Genstat will declare this variate for you if you have not done so already. In models where there are several error terms, only the final one is included. Others can be obtained using the AKEEP directive (4.6.1).

The fitted values from the analysis are defined to be the data values minus the estimated residuals. These too can be saved, using the FITTEDVALUES parameter. In models where there are several error terms, only the final error term is subtracted. If this is not what you want, you can use AKEEP (4.6.1) to save the full residuals, containing residual effects from all the error terms.

The last parameter, SAVE, allows you to save the complete details of the analysis in an *ANOVA save structure*. The ADISPLAY directive lets you use a save structure to produce further output (4.1.3). You can also use it in the AKEEP directive to put quantities calculated from the analysis into data structures which you can then use elsewhere in Genstat (4.6.1). Save structures are special compound structures (1:2.8), and Genstat declares them automatically. The save structure for the last y-variate analysed is stored automatically, and forms the default for ADISPLAY and AKEEP if you do not provide one explicitly.

Genstat still generates the model and does the dummy analysis even if a y-variate has no values, or if you specify a null entry in the Y list. You then get a *skeleton* analysis-of-variance table, which excludes sums of squares, mean squares and variance ratios; the only other output available is the information summary (4.1.3). You can save a design structure, but no save structure is formed. This is a good way of checking that a design can be analysed, before the experiment is carried out.

4.1.3 Output from ANOVA

This section describes the output from ANOVA, and the ADISPLAY directive that allows you to display further output without repeating the analysis. It also describes the ARESULTSUMMARY procedure that can be used to provide a summary of the results.

ADISPLAY directive

Displays further output from analyses produced by ANOVA.

Options

PRINT = *string tokens*	Output from the analyses of the y-variates, adjusted for any covariates (aovtable, information, covariates, effects, residuals, contrasts, means, cbeffects, cbmeans, stratumvariances, %cv, missingvalues); default * i.e. no printing
UPRINT = *string tokens*	Output from the unadjusted analyses of the y-variates (aovtable, information, effects, residuals, contrasts, means, cbeffects, cbmeans, stratumvariances, %cv, missingvalues); default * i.e. no printing
CPRINT = *string tokens*	Output from the analyses of the covariates, if any (aovtable, information, effects, residuals, contrasts, means, %cv, missingvalues); default * i.e. no printing
CHANNEL = *identifier*	Channel number of file, or identifier of a text to store output; default current output file
PFACTORIAL = *scalar*	Limit on number of factors in printed tables of means or effects; default 9
PCONTRASTS = *scalar*	Limit on order of printed contrasts; default 9
PDEVIATIONS = *scalar*	Limit on number of factors in a treatment term whose deviations from the fitted contrasts are to be printed; default 9
FPROBABILITY = *string token*	Printing of probabilities for variance ratios in the aov table (yes, no); default no
PSE = *string tokens*	Standard errors to be printed with tables of means, PSE=* requests s.e.'s to be omitted (differences, lsd, means); default diff
TWOLEVEL = *string token*	Representation of effects in 2^n experiments (responses, Yates, effects); default resp
NOMESSAGE = *string tokens*	Which warning messages to suppress (nonorthogonal, residual); default *
LSDLEVEL = *scalar*	Significance level (%) to use in the calculation of least significant differences; default 5

Parameter

identifiers	Save structure (from ANOVA) to provide details of each analysis from which information is to be displayed; if omitted, output is from the most recent ANOVA

The ADISPLAY directive allows you to display further output from one or more analyses of variance, without having to repeat all the calculations. You can store the information from each

analysis in a save structure, using ANOVA, and then specify the same structure in the SAVE parameter of ADISPLAY. Several save structures can be listed, corresponding to the analyses of several different variates. They need not all have been produced by the same ANOVA statement nor even be from the same design. Alternatively, if you just want to display output from the last y-variate that was analysed, you need not specify the SAVE parameter in either ANOVA or ADISPLAY: the save structure for the last y-variate analysed is saved automatically, and provides the default for ADISPLAY.

Apart from CHANNEL, all the options of ADISPLAY also occur with ANOVA. CHANNEL can be set to a scalar to divert the output to another output channel. Alternatively, it can specify the identifier of text data structure to store the output (and in fact an undeclared structure will be defined as a text, automatically).

The PRINT option selects which components of output are to be displayed. These are all illustrated in this chapter, as indicated in this list.

aovtable	analysis-of-variance table (4.1, 4.2.1, 4.3, 4.5 and 4.7)
information	information summary, giving details of aliasing and non-orthogonality (4.1.3 and 4.7.1) or of any large residuals (4.2.1 and 4.7.1)
covariates	estimated covariate regression coefficients (4.3.1)
effects	tables of estimated treatment parameters (4.1.3 and 4.7.1)
residuals	tables of estimated residuals (4.1.3 and 4.2.1)
contrasts	estimated contrasts of treatment effects (4.5)
means	tables of predicted means for treatment terms (4.1.3)
cbeffects	estimated effects of treatment terms combining information from all the strata in which each term is estimated (4.7.1)
cbmeans	predicted means for treatment terms combining information from all the strata in which each term is estimated (4.7.1 and 4.7.3)
stratumvariances	estimated variances of the units in each stratum (4.7.1)
%cv	coefficients of variation and standard errors of individual units (4.1.3 and 4.2)
missingvalues	estimates of missing values (4.4)

The default for PRINT with ADISPLAY is different from that with ANOVA. With ANOVA, the default gives the output that you will require most often from a full analysis: aovtable, information, covariates, means and missingvalues. You are most likely to use ADISPLAY when you are working interactively, to examine one component of output at a time, and it is not obvious that any one component will then be more popular than any other. So the default for ADISPLAY produces no output (that is, PRINT=*). This also means that you do not need to suppress the output explicitly when you are using UPRINT and CPRINT to examine components of output from analysis of covariance (4.3).

The settings information, covariates and missingvalues have a slightly different effect with ANOVA than with ADISPLAY. As they are part of the default specified for ANOVA, they will not produce any output unless there is something definite to report. With ADISPLAY you need to request them explicitly, so Genstat will always produce some sort of report. For example, there are no missing values with the variate Gain analysed earlier in this section, there are no covariates, and there is no aliasing or non-orthogonality. The information summary will also contain warning messages about any large residuals (see 4.2.1) unless the NOMESSAGES option has been set to residuals to exclude these: for example

```
ADISPLAY [PRINT=information; NOMESSAGES=residuals]
```

The criterion used to decide whether or not to report a residual is the same that used in regression

analysis (3.1.2). In this set of data there are none. The other setting, `nonorthogonality`, of the `NOMESSAGES` option suppresses the warning produced when there is orthogonality between treatment terms (4.7.4) or covariates (4.3.1).

Example 4.1.3a

```
  19  ADISPLAY [PRINT=information,covariates,missingvalues]

Information summary
===================

All terms orthogonal, none aliased.


Covariate regressions
=====================

No covariates


Missing values
==============

Variate: Gain

No missing values
```

If you had asked for these three pieces of information by `ANOVA`, you would not have obtained any output, since there is nothing positive to report.

The other default components produced by `ANOVA` are the analysis-of-variance table, shown earlier in this section, and the tables of means.

Example 4.1.3b

```
  20  ADISPLAY [PRINT=means]

Tables of means
===============

Variate: Gain

Grand mean  87.9

   Source     beef   cereal     pork
              89.6     84.9     89.1

   Amount     high      low
              95.1     80.6

   Source   Amount     high      low
     beef             100.0     79.2
   cereal              85.9     83.9
     pork              99.5     78.7


Standard errors of differences of means
---------------------------------------

Table               Source      Amount      Source
                                            Amount
rep.                    20          30          10
d.f.                    54          54          54
s.e.d.                4.63        3.78        6.55
```

A table of means is produced for each term in the treatment model. By using the PFACTORIAL option you can exclude tables for terms containing more than a specified number of factors; Genstat does not allow tables to have more than nine factors, so the default value of nine gives all the available tables.

The means are predicted mean values: estimated expected values for each combination of levels in the table, averaged over the levels of other factors. The table for each term is calculated by taking the table of estimated effects for the term and then adding in the estimated effects of all its margins. The grand mean is a margin, as is every term whose factors are a subset of those in the table. For example, the effects of source of protein have only the grand mean as a margin, and so the table of means for Source is calculated by adding the grand mean to each of the Source effects. Source.Amount has three margins; its table of means is formed by adding the grand mean and the main effects of Source and of Amount to the Source.Amount interaction effects. (You can verify this from the tables of effects printed in Example 4.1.3d, below.)

An assumption of analysis of variance is that the effects of each error term (or residuals) are independently distributed with zero mean and a common variance (see the initial part of 4.1); so they have predicted values of zero. Consequently, even if a term from the block formula (4.2.1) is a margin of a treatment term, its effects will not be included in the table of means. Similarly, if the deviations from fitted contrasts have been ascribed to error (4.5), these effects are also excluded; the table of means is then said to be *smoothed* (4.5).

Usually this process of prediction produces tables of means that are the same as the averages of the observed values: for example, in the common situation where the design is orthogonal and there are no covariates, the only further requirements for this to happen are that the term for the table must have no block terms as margins nor any of its deviations ascribed to error. In an analysis of covariance, the means are all adjusted to correspond to a common value, namely the grand mean of each covariate (4.3.1). Adjusted means are also produced when there is non-orthogonality: they are adjusted for the effects that are non-orthogonal to the term or to its margins (4.7.4).

Genstat has printed an s.e.d. for each table of means in Example 4.1.3b – that is, a standard error for assessing the difference between a pair of means within the table. These are provided by the default setting, differences, of the PSE option. The setting means (see Example 4.1.3c) gives e.s.e.'s, that is effective standard errors for the means which can be used for calculating standard errors for comparisons between means. In Example 4.1.3b, the means are uncorrelated and so the e.s.e.'s are the same as the standard errors of the means (used for comparing a mean with zero), but this may not be the case in a stratified design (4.2), nor if there are covariates (4.3). The lsd setting gives least significant differences (see Example 4.1.3c), or you can put PSE=* to suppress the standard errors altogether. More than one s.e.d., e.s.e. or l.s.d. will be given when some of the comparisons between the means in a table have different standard errors, as for example in split-plot designs (4.2.1).

Example 4.1.3c

```
  21  ADISPLAY [PRINT=means; PSE=means,lsd]

Tables of means
===============

Variate: Gain

Grand mean  87.9

   Source     beef   cereal     pork
              89.6     84.9     89.1

   Amount     high      low
              95.1     80.6
```

```
   Source   Amount      high       low
     beef              100.0      79.2
   cereal               85.9      83.9
     pork               99.5      78.7


Standard errors of means
------------------------

Table                Source       Amount       Source
                                                Amount
rep.                     20           30           10
d.f.                     54           54           54
e.s.e.                 3.28         2.67         4.63


Least significant differences of means (5% level)
-------------------------------------------------

Table                Source       Amount       Source
                                                Amount
rep.                     20           30           10
d.f.                     54           54           54
l.s.d.                 9.29         7.58        13.13
```

The l.s.d.'s are the standard errors of differences between means, multiplied by the t-statistic for the degrees of freedom of the standard error (see the d.f. line). For simple designs, as in Example 4.1, the degrees of freedom are merely the residual degrees of freedom. The situation for designs with several error terms, like the split-plot in Example 4.2.1a, is explained in Section 4.2.1. By default the t-statistic is for a 5% (two-sided) significance level, but this can be changed using the `LSDLEVEL` option.

The replication of the means in each table is also printed. In an unweighted analysis of variance, like that above, the replication is the number of units that received each combination of the treatments in the table. In a weighted analysis, the weighted replication (wt. rep.) is given: this is the sum of the weights of the units that received each treatment combination. If the replication (or weighted replication) is the same for every combination in the table, it is printed with the standard error; otherwise a table of replications is printed in parallel with the table of means, as illustrated in 4.3.

When the means have different replications, standard errors are printed for three types of comparison: between two means with the minimum replication, between two means with the maximum replication, and between a mean with minimum replication and one with maximum replication. But if, for example, there is only one mean with the minimum replication, the first type of comparison will not arise. If Genstat detects such situations, the appropriate s.e.d. is marked with an `X`. Note, however, that if you want standard errors for all possible comparisons, you can save these in a symmetric matrix using `AKEEP` (4.6.1), and then display them using the `PRINT` directive (1:3.2).

In stratified designs (4.2), there may be information on a treatment term in more than one stratum. The setting `means` uses only the effects from the lowest stratum in which the term is estimated (4.7.1). Alternatively, you can specify `cbmeans` to obtain means that combine information from all the strata in which the term or its margins are estimated. These will provide more accurate predictions. However, their distributional properties are not well understood, and so it is better to use effects or ordinary means for testing. Combined estimates of means are illustrated in 4.7.1 and 4.7.3, along with the combined estimates of effects (`cbeffects`) and estimated stratum variances (`stratumvariances`) from which they are calculated.

Example 4.1.3d

```
  21  ADISPLAY [PRINT=effects]

Tables of effects
=================

Variate: Gain

Source effects,  e.s.e. 3.28,  rep. 20

   Source     beef   cereal     pork
               1.7     -3.0      1.2

Amount response            -14.5,  s.e. 3.78,  rep. 30

Source.Amount effects,  e.s.e. 4.63,  rep. 10

   Source   Amount     high      low
     beef               3.1     -3.1
   cereal              -6.3      6.3
     pork               3.1     -3.1
```

Tables of effects are estimates of treatment parameters in the linear model (4.1). Although effects are used less often than means for summarizing the results of an experiment, they may be useful if you wish to study the model in more detail. The option `PFACTORIAL` applies to tables of effects in the same way as to tables of means. In this example, there are tables for the `Source` main effects and the `Source.Amount` interaction. (The `Amount` main effects are presented as a *response*, as we explain later.)

Each term is subject to constraints that are generated by the fitting of the terms that come before it in the linear model. The grand mean is fitted first of all. So the sum of the effects, each multiplied by its replication (or weighted replication), is zero within every table. The replication is printed in the header line of the table or, if the replications are unequal, with the table itself. Here the effects within all the tables are equally replicated, and you can check that their sum is zero within each table.

Similarly the table of `Source.Amount` interaction effects has zero row and column sums because the main effects of `Source` and `Amount` have been fitted first.

The header also specifies an e.s.e. or a range of e.s.e.'s for the effects in the table: e.s.e. stands for *effective standard error* – the adjective *effective* reminds you that it is appropriate only for comparisons that are unaffected by the constraints within the table. So the e.s.e. for `Source` is appropriate for obtaining an s.e.d. to assess differences between effects, but not for testing the sum of the effects, nor any individual effect, against zero.

To understand how the e.s.e. arises, we can consider the `Source` main effects. (If you do not want to know about this piece of theory, skip this paragraph.) These effects are estimated by

$$s_i = 1/20 \sum_{j=1}^{2} \sum_{k=1}^{10} y_{ijk} - 1/60 \sum_{i=1}^{3} \sum_{j=1}^{2} \sum_{k=1}^{10} y_{ijk}$$

and can be shown to have a variance of $\sigma^2(1/20 - 1/60)$ where 20 is the replication of the `Source` effects, and 60 is the total number of units. The second term in the formula (which is the estimate of the grand mean) is common to all the estimates, and it is because of this that pairs of effects have a non-zero covariance of $-\sigma^2/60$. The variance of the difference between two effects can be calculated by a familiar formula: it is the sum of the variances of the two effects minus twice their covariance, giving an s.e.d. of $\sqrt{(2\sigma^2/20)}$. However an easier way of deriving this s.e.d. is to notice that, when you subtract one estimate from the other, the second term cancels out to leave the difference between two sums of independent random variables, each

with variance $\sigma^2/20$. We can thus refer to each estimated effect as having an effective variance of $\sigma^2/20$ and an effective covariance of zero when calculating the variance of a comparison unaffected by the constraint. The general formula for the e.s.e. is:

e.s.e. = $\sqrt{(\sigma^2/((\text{weighted}) \text{ replication} \times \text{efficiency factor} \times \text{covariance efficiency factor}))}$

The efficiency factor is described in 4.7.1; for an orthogonal term its value is one. Likewise, the covariance efficiency factor is one when there are no covariates (4.3). The variance σ^2 is estimated by the residual mean square of the stratum where the effects are estimated. Strata are explained in 4.2. Here there is only one stratum and residual, so σ^2 is estimated by 214.6 and the e.s.e. is $\sqrt{(214.6/20)}$.

When a factor has only two levels, like `Amount` above, Genstat prints the difference between the two main effects. This difference is called a *response*. For interaction terms whose factors all have only two levels, there are two forms of response. The choice between them is controlled by the `TWOLEVEL` option. If you leave the default, `TWOLEVEL=response`, Genstat calculates the response for an interaction between two factors as the difference between the two main-effect responses, and so on; this is the form described in most textbooks. By putting `TWOLEVEL=Yates`, you can obtain the effects specified by Yates (1937), which are defined so that the standard error of the interaction effects remains the same as that of the main effects. Alternatively, you can put `TWOLEVEL=effects` if you prefer not to have responses, but to have the effects themselves, as for factors with more than two levels.

Example 4.1.3e

```
  23  ADISPLAY [PRINT=residuals]

Tables of residuals
===================

Variate: Gain

*Units* residuals,  s.e. 13.90,  rep. 1

  *units*        1        2        3        4        5        6        7
           -27.0     12.1     -5.5     10.8     23.1    -29.7      2.0

  *units*        8        9       10       11       12       13       14
           -11.9    -20.5     -3.2     11.1      3.3     18.0    -29.9

  *units*       15       16       17       18       19       20       21
            -3.5     10.8     13.1     -5.7      4.0     25.1     -1.5

  *units*       22       23       24       25       26       27       28
           -15.2     -3.9      7.3    -19.0      9.1      2.5      6.8

  *units*       29       30       31       32       33       34       35
            14.1      2.3      7.0      2.1      2.5    -28.2     -9.9

  *units*       36       37       38       39       40       41       42
            18.3      0.0     -3.9      8.5     -7.2     -9.9     27.3

  *units*       43       44       45       46       47       48       49
           -13.0     -8.9     -8.5     10.8    -16.9     -8.7     17.0

  *units*       50       51       52       53       54       55       56
             0.1     20.5     15.8      5.1    -17.7     11.0      6.1

  *units*       57       58       59       60
             5.5     -1.2    -25.9      3.3
```

Residuals correspond to the error parameters of the linear model (4.1). Here there is a single error term, and thus a single set of residuals. There is no block model (4.2.1) to define factors to index the units of the design, and so each estimated residual is printed with a unit number,

under the heading *units*. The header line shows the replication or weighted replication, and gives a standard error appropriate for comparing any residual with zero. If the replications or weighted replications were unequal, these would be printed in parallel with the residuals, and the range of standard errors would be printed, the lower value being appropriate for residuals with the maximum replication or weighted replication, and the upper value for those with the minimum replication or weighted replication.

Example 4.1.3f

```
  24  ADISPLAY [PRINT=%cv]

Stratum standard errors and coefficients of variation
=====================================================

Variate: Gain

   d.f.          s.e.          cv%
     54         14.65        16.7
```

The setting PRINT=%cv displays the residual number of degrees of freedom, the standard error of a single unit of the design and the *coefficient of variation* (cv%), which is the standard error of a single unit expressed as a percentage of the grand mean. The coefficient of variation is often used as an index of the variability when comparing several experiments on the yields of the same field crop. However it can be misleading, especially with transformed variables like the logarithm of yield, where the grand mean may even be zero, or with other variables that can take negative values. In designs with several error terms, the same information is presented for each stratum, as shown in 4.2.1. If the units in a stratum have unequal replication or weighted replication, there is no single standard error for a unit; so a missing value is printed instead.

The only component of output that we have not yet mentioned contains the estimates of treatment contrasts, which you can obtain by putting PRINT=contrasts. These are shown in 4.5, together with an explanation of how to control their printing by the options PCONTRASTS and PDEVIATIONS.

With analysis of covariance, you can also print output from the analyses of the covariates and from the analysis of the y-variate ignoring the covariates. This is controlled by options CPRINT and UPRINT respectively, as shown in 4.3.1.

The ARESULTSUMMARY procedure investigates an ANOVA analysis, to provide the information that would be useful for a report.

ARESULTSUMMARY procedure

Provides a summary of results from an ANOVA analysis (R.W. Payne).

Options

PRINT = *string tokens*	What to print (description, means, significant); default desc, mean, sign
PSE = *string tokens*	Standard errors to be printed with the means (sed, sedsummary, lsd, lsdsummary, dfmeans); default sed, dfme
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
SAVE = *ANOVA save structure*	Save structure for the analysis; default uses the save structure from the most recent ANOVA

By default, all the information is printed, but you can control this with the PRINT option, whose settings are:

description	prints the name of the y-variate, any covariates and the block and treatment models,
means	prints relevant tables of means, and
significant	lists the significant treatment terms.

The relevant tables of means are those that contain significant treatment effects. Also, each table contains all the significant effects involving any of its factors. In the example for the procedure, terms A, D, S and A.S are significant. Two tables of means are therefore presented, one classified by A and S, and the other by D. However, if the significant terms were A.S and D.S. there would be only one table, classified by factors A, D and S.

The PSE option controls the information provided with the tables of means:

sed	standard errors for differences between means,
sedsummary	summary of the standard errors for differences,
dfmeans	degrees of freedom for the standard errors of differences,
lsd	least significant differences between the means, and
lsdsummary	summary of the least significant differences.

The default is to print the standard errors of differences and their degrees of freedom. Note: if all the differences between means have the same standard error of difference, a summary is printed for the settings sed and lsd, instead of the full symmetric matrices of values.

The LSDLEVEL option specifies the significance level (%) to use in the calculation of least significant differences (default 5%).

Example 4.1.3g shows that the key information from the analysis in Section 4.1 is the fact that there is a significant effect of the amount of protein. This is reported by ARESULTSUMMARY, together with the tables of means for the AMOUNT factor.

Example 4.1.3g

```
  25  ARESULTSUMMARY

Results from analysis of variance
=================================

Variate: Gain
Treatment structure: Source*Amount
Factorial: 3


Significant treatment terms
---------------------------

Amount     <0.1%    (pr. <.001)


Predicted means for Amount
==========================


       Amount
         high       95.13
          low       80.60


Standard error of difference 3.782

Degrees of freedom for standard error of difference 54
```

4.1.4 Procedures for examining residuals

APLOT procedure

Plots residuals from an ANOVA analysis (R.W. Payne & A.D. Todd).

Options

RMETHOD = *string token*	Type of residuals to plot (simple, standardized); default simp
INDEX = *variate*	X-variate for an index plot; default !(1,2...)
STRATUM = *formula*	The stratum (or error term) whose residuals are to be plotted; the default is to plot the residuals from the final stratum
GRAPHICS = *string token*	What type of graphics to use (lineprinter, highresolution); default high
TITLE = *text*	Overall title for the plots; if unset, the identifier of the y-variate is used
SAVE = *ANOVA save structure*	Specifies the analysis from which the residuals and fitted values are to be taken; by default they are taken from the most recent ANOVA

Parameters

METHOD = *string tokens*	Type of residual plot (fittedvalues, normal, halfnormal, histogram, absresidual, index); default fitt, norm, half, hist
PEN = *scalars*, *variates* or *factors*	Pen(s) to use for each plot

Procedure APLOT provides six types of plots of residuals from an ANOVA analysis. These are selected using the METHOD parameter, with settings: fitted for residuals versus fitted values, normal for a Normal plot, halfnormal for a half-Normal plot, histogram for a histogram of residuals, absresidual for a plot of the absolute values of the residuals versus the fitted values, and index for a plot against an "index" variable (specified by the INDEX option). Up to four can be displayed at a time.

For a Normal plot, the Normal quantiles are calculated as follows:

$$q_i = \text{NED}((i-0.375) / (n+0.25))$$

while for a half-Normal plot they are given by

$$q_i = \text{NED}(0.5 + 0.5 \times (i-0.375) / (n+0.25))$$

The residuals and fitted values are accessed automatically from the structure specified by the SAVE option. If the SAVE option is not set, they are taken from the SAVE structure of the last y-variate to have been analysed by ANOVA. By default, simple residuals are plotted, but you can set option RMETHOD=standardized to plot standardized residuals instead. If, as in Section 4.2, your design has several strata (or error terms), you can set the STRATUM option to plot the residuals from one of the higher strata. The default is to plot the residuals from the final stratum.

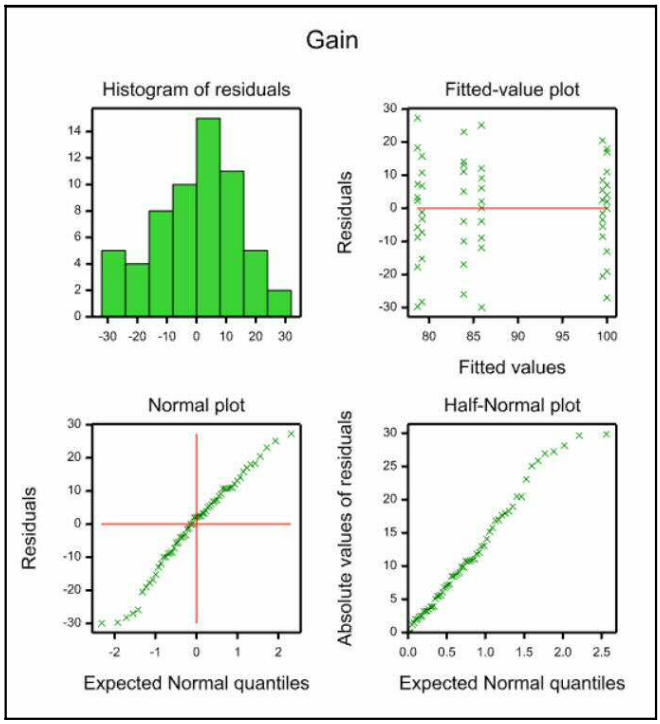


Figure 4.1.4

By default, high-resolution graphics are used. Line-printer graphics can be used by setting option GRAPHICS=lineprinter. With high resolution, the PEN parameter can be used to specify the graphics pen or pens to use for each plot. The TITLE option can supply an overall title. If this is not set, the identifier of the y-variate is used.

For example, typing the statement

```
APLOT fitted,normal,halfnormal,histogram
```

at the end of Example 4.1.3f would produce the plot in Figure 4.1.4.

If the data are from a field experiment, it may be interesting to study the spatial pattern of the residuals, for example to see if there are any systematic trends in fertility. This can be done using the AFIELDRESIDUALS procedure.

AFIELDRESIDUALS procedure

Display residuals in field layout (R.W. Payne & A.D.Todd).

Options

PRINT = *string tokens*	Controls output (contour, shade, table); default cont
GRAPHICS = *string token*	Type of graph (highresolution, lineprinter); default high
METHOD = *string token*	Type of residuals to take from the save structure when the RESIDUALS parameter is not specified (combined, finalstratum, standardizedfinal); default comb
MARGIN = *string token*	Whether to include margins in printed tables (yes, no); default no
YORIENTATION = *string token*	Y-axis orientation of the plot (reverse, normal); default norm
PENCONTOUR = *scalar*	Pen number to be used for the contours; default 1
PENFILL = *scalar* or *variate*	Pen number(s) defining how to fill the areas between

	contours; default 3
PENSHADE = *scalar* or *variate*	Pen(s) to use for the shade plot; default 3

Parameters

Y = *variates* or *factors*	Specifies the y-coordinates of the plots
X = *variates* or *factors*	Specifies the x-coordinates of the plots
RESIDUALS = *variates*	Residuals to be plotted; default is to take the residuals from the save structure specified by the SAVE option, or from the most recent ANOVA if that is unspecified
SAVE = *ANOVA, REML or regression save structures*	Save structure of the ANOVA, REML or regression analysis from which to take the residuals if the RESIDUALS parameter is not specified; default is to take the most recent ANOVA analysis
FIELDWIDTH = *scalars*	Field width for printing the residuals; default 12
DECIMALS = *scalars*	Number of decimal places to use when printing the residuals
TITLE = *texts*	Titles for the plots

The locations of the plots are defined by the Y and X parameters, specifying variates or factors containing their y- and x-coordinates respectively. The residuals can be supplied, in a variate, by the RESIDUALS parameter. If this is not set, the default is to take the residuals from the most recent ANOVA analysis. You can take the residuals from some other analysis of variance, or from a regression or REML analysis (see 3.1.1 and 5.3.1), by specifying its save structure using the SAVE parameter.

The METHOD option determines the type of residuals that are taken. The default setting combined gives residuals combining the residuals from all the strata or error terms in the analysis, as these include all the random variation; for information about analyses with several strata see Section 4.2. These are the residuals that would be saved using the CBRESIDUALS option of the AKEEP directive (4.6.1), or the use of the RESIDUALS option in VKEEP with option RMETHOD=all (5.9.1). Regression allows only a single error term, so combined is treated as the same as the next setting, finalstratum.

The setting finalstratum uses simple residuals from the final stratum or error term. These correspond to the RESIDUALS option of AKEEP with option RMETHOD=simple, or the RESIDUALS option of VKEEP with option RMETHOD=final, or the RESIDUALS parameter of RKEEP with option RMETHOD=simple.

The last setting, standardizedfinal, uses standardized residuals from the final stratum or error term. These correspond to the RESIDUALS option of AKEEP with option RMETHOD=standardized, or the RESIDUALS parameter of RKEEP with option RMETHOD=deviance. They are calculated using standard errors from procedure VFRESIDUALS for REML analyses (5.9.2).

Usually, the plots will all have different coordinates. However, if there are several plots with the same coordinates, mean residuals are calculated for each location. Thus for example, if you wanted only to look at the block and whole-plot residuals in a split-plot design (see 4.2.1), you could request combined residuals and then set identical coordinates for the (sub-) plots within each whole plot.

AFIELDRESIDUALS provides three forms of representation, selected using the PRINT option as follows:

table	prints the residuals in a table whose structure corresponds to the field layout,
contour	generates a contour plot if the plots are on a regular grid or

	a line graph if they are arranged in a single line, and
`shade`	can produce a shade plot for plots that are on a regular grid.

The `GRAPHICS` option determines the type of graphics that is used, with settings `highresolution` (the default) and `lineprinter`. No graph can be produced if the plots are in an irregular 2-dimensional arrangement. High-resolution contour plots require more than 3 rows and columns, and line-printer contour plots require more than 4 rows and columns. The way in which the lines are drawn in high-resolution contour plots is defined by the properties of the pen specified by the `PENCONTOUR` option, while the pen specified by the `PENFILL` parameter defines how to shade the areas between the contours. Their defaults are 1 and 3 respectively. Similarly, the pen or pens specified by the `PENSHADE` option control the colouring of the shade plot; the default is to use pen 3. For more information see the `DCONTOUR` and `DSHADE` directives (1:6.4.1 and 1:6.4.2).

The `MARGIN` option, with settings `no` (default) and `yes`, determines whether or not marginal summaries are included with the printed tables. The `FIELDWIDTH` and `DECIMALS` parameters can be used to specify the formats of the printed tables (as in the `PRINT` directive). The `TITLE` parameter can supply a title for the plots. If this is unset, a default title is formed.

The `YORIENTATION` option controls the orientation of the y-coordinates in the plots and tables. By default this is `normal`, so that they run upwards from the bottom of the page (as in a map).

The use of `AFIELDRESIDUALS` is shown in Example 4.2 and Figure 4.2.1b.

4.1.5 Displaying tables of means

`AGRAPH` procedure

Plots tables of means from `ANOVA` (R.W. Payne).

Options

`GRAPHICS` = *string token*	Type of graph (`highresolution`, `lineprinter`); default `high`
`METHOD` = *string token*	What to plot (`means`, `lines`, `data`, `barchart`, `splines`); default `mean`
`XFREPRESENTATION` = *string token*	How to label the *x*-axis (`levels`, `labels`); default `labels` uses the `XFACTOR` labels, if available
`PSE` = *string token*	What to plot to represent variation (`differences`, `lsd`, `means`, `allmeans`); default `diff`
`LSDLEVEL` = *scalar*	Significance level (%) to use for least significant differences; default 5
`DFSPLINE` = *scalar*	Number of degrees of freedom to use when `METHOD=splines`
`YTRANSFORM` = *string tokens*	Transformed scale for additional axis marks and labels to be plotted on the right-hand side of the y-axis (`identity`, `log`, `log10`, `logit`, `probit`, `cloglog`, `square`, `exp`, `exp10`, `ilogit`, `iprobit`, `icloglog`, `root`); default `iden` i.e. none
`PENYTRANSFORM` = *scalar*	Pen to use to plot the transformed axis marks and labels; default `*` selects a pen, and defines its properties, automatically
`KEYMETHOD` = *string token*	What to use for the key descriptions when `GROUPS` specifies more than one factor (`labels`,

	namesandlabels); default name
PLOTTITLEMETHOD = *string token*	What to use for the titles of the plots when TRELLISGROUPS specifies more than one factor (labels, namesandlabels); default name
PAGETITLEMETHOD = *string token*	What to use for the titles of the pages when PAGEGROUPS specifies more than one factor (labels, namesandlabels); default name
USEAXES = *string token*	Which aspects of the current axis definitions of window 1 to use (none, limits, marks, mpositions, nsubticks,); default none
SAVE = *ANOVA* or *regression save structure*	Save structure to provide the table of means; default uses the save structure from the most recent ANOVA

Parameters

XFACTOR = *factors*	Factor providing the *x*-values for each plot
GROUPS = *factors* or *pointers*	Factor or factors identifying groups of points in each plot; by default chosen automatically
TRELLISGROUPS = *factors* or *pointers*	Factor or factors specifying the different plots of a trellis plot of a multi-way table
PAGEGROUPS = *factors* or *pointers*	Factor or factors specifying plots to be displayed on different pages
NEWXLEVELS = *variates*	Values to be used for XFACTOR instead of its existing levels
TITLE = *texts*	Title for the graph; default defines a title automatically
YTITLE = *texts*	Title for the y-axis; default is to use the identifier of the y-variate, or to have no title if this is unnamed
XTITLE = *texts*	Title for the x-axis; default is to use the identifier of the XFACTOR
PENS = *variates*	Defines the pen to use to plot the points and/or line for each group defined by the GROUPS factors

AGRAPH plots tables of means from an ANOVA analysis. In its simplest form, the behaviour of AGRAPH depends on the model. If the treatment model contains only main effects, it plots the means for the first factor in the model. Otherwise it looks for the first treatment term involving two factors; it then plots the means with one of these factors as the x-axis, and the second as a grouping factor with levels identified by different plotting colours and symbols. By default, the means are from the most recent ANOVA. However, you can plot means from an earlier analysis, by using the SAVE option of AGRAPH to specify its save structure (saved using the SAVE parameter of the ANOVA command that performed the analysis).

Usually, each mean is represented by a point. However, with high-resolution plots, the METHOD option can be set to lines to draw lines between the points, or data to draw just the lines and then also plot the original data values, or barchart to plot the means as a barchart, or splines to plot the points together with a smooth spline to show the trend over each group of points. The DFSPLINE specifies the degrees of freedom for the splines; if this is not set, 2 d.f. are used when there are up to 10 points, 3 if there are 11 to 20, and 4 for 21 or more. The GRAPHICS option controls whether a high-resolution or a line-printer graph is plotted; by default GRAPHICS=high.

The PSE option specifies the type of error bar to be plotted with the means, with settings:

differences	average standard error of difference;

`lsd`	average least significant difference;
`means`	average effective standard error for the means;
`allmeans`	plots plus and minus the effective standard error around every mean.

The `LSDLEVEL` option sets the significance level (%) to use for the least significant differences (default 5). The `allmeans` setting is often unsuitable for plots other than barcharts when there are `GROUPS`, as the plus/minus e.s.e. bars may overlap each other.

You can define the table of means to plot explicitly, by specifying its classifying factors using the `XFACTOR`, `GROUPS`, `TRELLISGROUPS` and `PAGEGROUPS` parameters. The `XFACTOR` parameter defines the factor against whose levels the means are plotted. With a multi-way table, there will be a plot of means against the `XFACTOR` levels for every combination of levels of the other factors classifying the table. The `GROUPS` parameter specifies factors whose levels are to be included in a single window of the graph. For example, the statement below plots the means in Example 4.1, with `Amount` on the x-axis and a different line for each level of `Source`

```
AGRAPH Amount; Source
```

The resulting graph is shown in Figure 4.1.5a. Similarly Figure 4.1.5b shows a plot with the lines and the data, produced by

```
AGRAPH [METHOD=data] Amount; Source
```

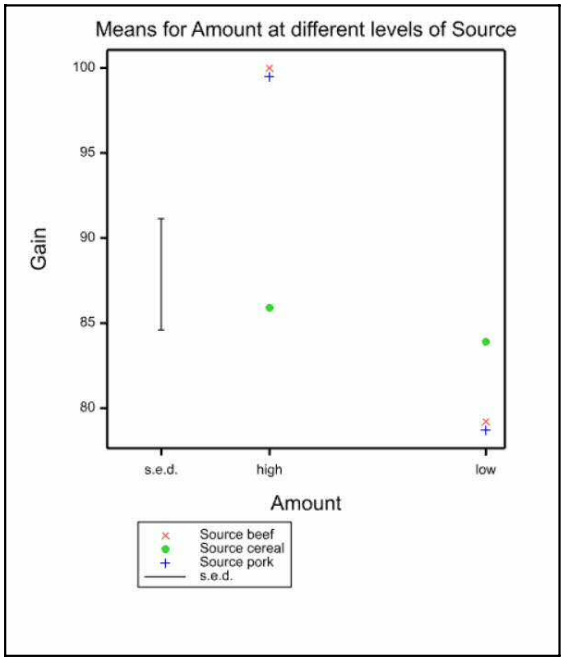


Figure 4.1.5a

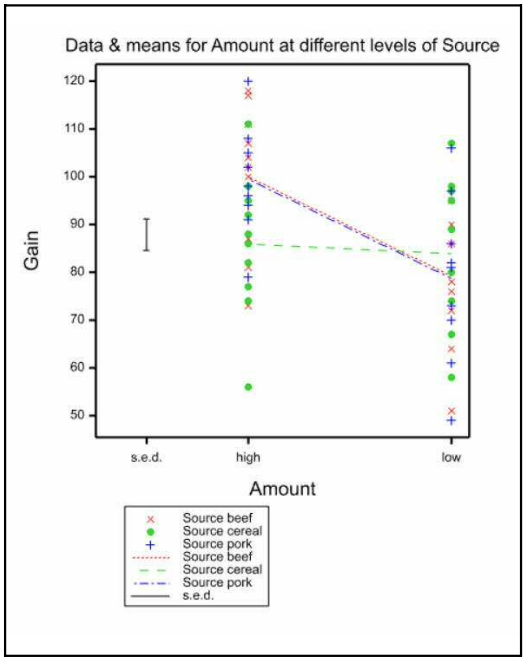


Figure 4.1.5b

You can set `GROUPS` to a pointer to specify several factors to define groups. For example

```
POINTER [VALUES=B,C] Groupfactors
AGRAPH [METHOD=line] XFACTOR=A; GROUPS=Groupfactors
```

to plot a line for every combination of the levels of factors `B` and `C`. Similarly, the `TRELLISGROUPS` option can specify one or more factors to define a trellis plot. For example,

```
AGRAPH [METHOD=line] XFACTOR=A; GROUPS=B; TRELLISGROUPS=C
```

will produce a plot for each level of `C`, in a trellis arrangement; each plot will again have factor `A` on the x-axis, and a line for each level of the factor `B`. Likewise, the `PAGEGROUPS` parameter can specify factors whose combinations of levels are to be plotted on different pages. So

```
AGRAPH [METHOD=line] XFACTOR=A; GROUPS=B; PAGEGROUPS=C
```

will produce a plot for each level of C, but now on separate pages. Multi-way tables can plotted even if the corresponding model term was not in the ANOVA analysis. For example you can plot a two-way table even if the analysis contained only the main effects of the two factors; however, the lines will then all be parallel and no standard errors or LSDs can be included.

The NEWXLEVELS parameter enables different levels to be supplied for XFACTOR if the existing levels are unsuitable. If XFACTOR has labels, these are used to label the x-axis unless you set option XFREPRESENTATION=levels.

The TITLE, YTITLE and XTITLE parameters can supply titles for the graph, the y-axis and the x-axis, respectively. The symbols, colours and line styles that are used in a high-resolution plot are usually set up by AGRAPH automatically. If you want to control these yourself, you should use the PEN directive to define a pen with your preferred symbol, colour and line style, for each of the groups defined by combinations of the GROUPS factors. The pen numbers should then be supplied to AGRAPH, in a variate with a value for each group, using the PENS parameter.

The YTRANSFORM option allows you to include additional axis markings, transformed onto another scale, on the right-hand side of the y-axis. Suppose, for example, suppose you have analysed a variate of percentages that have been transformed to logits. You might then set YTRANSFORM=ilogit (the inverse-logit transformation) to include markings in percentages alongside the logits. The settings are the same as those of the TRANSFORM parameter of AXIS, which is used to add the markings (1:6.9.7). You can control the colours of the transformed marks and labels, by defining a pen with the required properties, and specifying it with the PENYTRANSFORM option. Otherwise, the default is to plot them in blue.

When there is more than one GROUPS factor, the KEYMETHOD controls whether to use the factor names with their labels (or levels for factors with no labels) or just the labels (or levels) in the key descriptions. The default is to use the names and the labels (or levels). Similarly, the PLOTTITLEMETHOD specifies what to use for the titles of the plots when there is more then one TRELLISGROUPS factor, and the PAGETITLEMETHOD specifies what to use for the titles of the plots when there is more then one PAGEGROUPS factor. You can setKEYMETHOD=* to have no key at all.

The USEAXES option allows you to control various aspects of the axes. First you need to use the XAXIS and YAXIS directives to define them for window 1. Then specify which of the aspects of the axes in window 1 are to be used by DTABLE, by specifying USEAXES with the following settings:

limits	y- and x-axis limits (LOWER and UPPER parameters);
marks	location and labelling of the tick marks (MARKS, LABELS, LDIRECTION, LROTATION, DECIMALS, DREPRESENTATION, and VREPRESENTATION parameters);
mpositions	positions of the tick marks (MPOSITION parameter); and
nsubticks	number of subticks per interval (NSUBTICKS parameter).

By default none are used.

For compatibility with previous releases, AGRAPH allows you to plot predicted means from an analysis by the AUNBALANCED procedure (which uses the Genstat regression commands). However, procedure AUGRAPH (new in Release 13) is now recommended instead; see Section 4.8.3. Also, in Release 13, a new procedure DTABLE was included to plot a user-supplied table. Previously this could be done using the MEANS parameter of AGRAPH, which has now been withdrawn.

The AFMEANS procedure provides another way of printing tables of means. It has the advantage over ADISPLAY (4.1.3) that it can calculate and print predicted means for terms that were not in the original analysis: the means must be classified by treatment factors from the analysis, but the term defined by the full list of factors need not have been included in the

treatment model. So, for example, you can obtain an A × B table of means, even if the model contained only the A and B main effects. Alternatively, in a more realistic scenario, you may have significant A.B and B.C interactions, but no A.B.C interaction. You might then still want to present an A × B × C table means, even though you might not want to include an A.B.C interaction. You can also save the means, and their standard errors etc, in Genstat data structures for later use..

AFMEANS procedure

Forms tables of means classified by ANOVA treatment factors (R.W. Payne).

Options

PRINT = *string tokens*	What to print (means, sed, sedsummary, ese, lsd, lsdsummary); default mean, sed
MEANS = *table*	Saves means; default *
SED = *symmetric matrix*	Saves matrices of standard errors of differences between means; default *
ESE = *table*	Saves effective standard errors; default *
LSD = *symmetric matrix*	Saves least significant differences between means; default *
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
DFMEANS = *symmetric matrices*	Saves degrees of freedom for comparisons between every pair of entries in the table of means
EQFACTORS = *factors*	Factors whose levels are to be assumed to be equal within the comparisons between means, when calculating effective standard errors
SAVE = *ANOVA save structure*	Save structure to provide the table of means; default uses the save structure from the most recent ANOVA

Parameter

CLASSIFY = *vectors*	Factors to classify table of means (from those in the TREATMENTSTRUCTURE in the ANOVA analysis)

The factors classifying the table of means are specified by the CLASSIFY parameter. By default the means are formed for the most recent ANOVA, but you can use the SAVE option to supply the save structure from an earlier analysis.

Printed output is controlled by settings of the PRINT option:

means	means,
ese	effective standard errors of the means,
sed	standard errors for differences between the means,
sedsummary	summary of the standard errors for differences between the means,
dfmeans	degrees of freedom for the standard errors of differences between means,
lsd	least significant differences between the means, and
lsdsummary	summary of the least significant differences between the means.

The default is to print means and a summary of the standard errors of differences. The LSDLEVEL option specifies the significance level (%) to use in the calculation of least significant differences (default 5%). Note: if all the differences between means have the same standard error of difference, a summary is printed for the settings sed and lsd, instead of the full symmetric

matrix of values. The EQFACTORS option allows you to specify factors within the tables of means whose levels are assumed to be equal for the two means, when calculating effective standard errors.

The MEANS, SED, ESE, LSD and DFMEANS options allow the results to be saved in appropriate Genstat data structures.

4.1.6 Checking the assumptions

ACHECK procedure

Checks assumptions for an ANOVA analysis (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (tests, confirmation); default conf
ASSUMPTION = *string tokens*	Which assumptions to test (homogeneity, normality, stability); default homo, norm, stab
PROBABILITY = *scalar*	Critical value for the test probabilities to decide whether to generate warning messages; default 0.025
SAVE = *ANOVA save structure*	Specifies the analysis to be checked; by default this will be the most recent ANOVA

No parameters

Procedure ACHECK checks some of the assumptions for an analysis of variance that has been performed by the ANOVA directive. By default, the most recent ANOVA analysis is checked. However, you can check an earlier analysis, by using the SAVE option of ACHECK to specify its save structure (saved using the SAVE parameter of the earlier ANOVA command).

The assumptions to check are controlled by the ASSUMPTIONS option, with the following settings.

homogeneity	performs Levene tests to check whether the residual variance seems to be affected by any of the terms in the analysis. With stratified designs it will make similar checks for the residual variation in the higher strata (e.g. for the whole-plot variation in a split-plot design).
normality	performs a Shapiro-Wilk test to check for evidence that the residuals do not come from a Normal distribution.
stability	performs two Levene tests to check whether the residual variance differs according to the size of the response. The data are divided into three groups (small, intermediate and large) according to the sizes of their fitted values. The tests compare the variance of the residuals in the first (small) group with those in the third (large) group, and the variance of the second (intermediate) group with the variance of other two groups combined.

By default, they are all tested.

ACHECK produces warning messages if any of the tests generates a test probability less than or equal to the value specified by the PROBABILITY option. The default value is 0.025 (i.e. 2.5%), which is the same as the value used for the similar messages that may occur with the summary of analysis in regression(3.1.2). It is important to realise that the estimated residuals (from either regression or analysis of variance) will be correlated. The Levene and Shapiro-Wilk tests assume that the residuals are independent Normally-distributed observations. Their test

probabilities may therefore be too low – and generate too many significant results. So the use of a smaller critical probability value provides some protection against spurious messages. You can print the detailed test results by setting option `PRINT=tests`. (By default these are not printed.) The default `PRINT=confirmation` prints a confirmation if there are no problems.

Example 4.1.6 shows the tests for the data in Example 4.1. None of the tests is significant (i.e. there are no problems).

Example 4.1.6

```
  29  ACHECK [PRINT=tests]

Tests of assumptions for ANOVA
==============================

Variate: Gain

Levene tests for homogeneity of variance
========================================

Analysis of variance
====================

Variate: Absolute residuals

Source of variation     d.f.       s.s.       m.s.    v.r.  F pr.
Source                     2     0.2084     0.1042    0.28  0.756
Amount                     1     0.3149     0.3149    0.85  0.361
Source.Amount              2     0.3634     0.1817    0.49  0.616
Residual                  54    20.0437     0.3712
Total                     59    20.9304


Levene tests for stability of variance
======================================

                                    Test t-statistic        d.f.         pr.
               Small vs. large responses       1.022      41.763       0.312
Intermediate v.s. small & large responses       0.108      11.660       0.916


Shapiro-Wilk test for Normality
===============================

Data variate:     Residuals
Test statistic W: 0.9766
Probability:      0.303
```

4.1.7 Permutation and exact tests for analysis of variance

`APERMTEST` procedure

Does random permutation tests for analysis-of-variance tables (R.W. Payne).

Options

`PRINT` = *string tokens*	Controls printed output (`aovtable`, `critical`); default `aovt`
`PLOT` = *string*	What to plot (`histogram`); default `*`
`NTIMES` = *scalar*	Number of permutations to make; default 999
`EXCLUDE` = *factors*	Factors in the block model of the design whose levels are not to be randomized
`SEED` = *scalar*	Seed for the random number generator used to make the permutations; default 0 continues from the previous

	generation or (if none) initializes the seed automatically
AOVTABLE = *pointer*	Saves the aov-table, with permutation probabilities
CRITICAL = *pointer*	Saves the aov-table, with critical values
SAVE = *ANOVA save structure*	Save structure from the analysis of variance; default uses the save structure from the most recent ANOVA

Random permutation tests provide an alternative to using the F probabilities printed for variance ratios in an analysis-of-variance table in situations where the assumptions of the analysis are not satisfied. These assumptions can be assessed by studying the residual plots produced by APLOT (4.1.4), or by using the ACHECK procedure (4.1.6). In particular, the use of the F distribution to calculate the probabilities is based on the assumption that the residuals from each stratum have Normal distributions with equal variances, and so the histogram of residuals produced by APLOT should look reasonably close to the Normal, bell-shaped curve. Experience shows the analysis is robust to small departures from Normality. APERMTEST can be useful if the histogram looks very non-Normal (and you are unable to redefine the analysis as a generalized linear model; see FIT).

The simplest form of use is simply to specify the command

```
APERMTEST
```

straight after the ANOVA. APERMTEST recovers the necessary information about the analysis automatically, and performs 999 random permutations (made using a default seed). The probability for each variance ratio is then determined from its distribution over the randomly permuted datasets.

The NTIMES option of APERMTEST allows you to request another number of permutations, and the SEED option allows you to specify another seed. APERMTEST checks whether NTIMES is greater than the number of possible permutations available for the data set. If so, APERMTEST does an "exact" test instead, which uses the SETALLOCATIONS directive (1:4.3.4) to make each possible permutation once.

The information about the analysis is obtained from the save structure of the most recent ANOVA (which is stored automatically within Genstat). You can save the information from any analysis of variance explicitly using the SAVE parameter of ANOVA. You can then perform permutation tests for that analysis by using the save structure as the setting of the SAVE option of APERMTEST. The EXCLUDE option allows you to restrict the randomization so that one or more of the factors in the block model is not randomized. The most common instance where this is required is when one of the treatment factors involves time-order, which cannot be randomized.

Output is controlled by the PRINT option, with settings:

aovtable	for an analysis-of-variance table with the usual F probabilities replaced by those from the permutation test; and
critical	for a table giving critical values for each variance ratio.

These can be saved using the AOVTABLE and CRITICAL parameters.

Example 4.1.7 does permutation tests for the data in Example 4.1, which confirm the earlier conclusions. Notice that the seed has been set by default. To recreate the same analysis, we should set the SEED option to 508631.

Example 4.1.7

```
  30  APERMTEST

* MESSAGE: Default seed for random number generator used with value 508631
```

```
Analysis of variance
====================

Variate: Gain
Probabilities determined from 999 random permutations

Source of variation  d.f.       s.s.       m.s.    v.r.  prob.
Source                  2      266.5      133.3    0.62  0.525
Amount                  1     3168.3     3168.3   14.77  0.002
Source.Amount           2     1178.1      589.1    2.75  0.075
Residual               54    11586.0      214.6
Total                  59    16198.9
```

4.1.8 Simultaneous confidence intervals for means

ACONFIDENCE procedure

Calculates simultaneous confidence intervals for ANOVA means (D.M. Smith).

Options

PRINT = *string token*	Controls printed output (intervals); default inte
METHOD = *string token*	Type of interval (individual, smm, product, Bonferroni, Scheffe); default smm
FACTORIAL = *scalar*	Limit on the number of factors in each term; default 3
PROBABILITY = *scalar*	The required significance level; default 0.05
SAVE = *ANOVA save structure*	Save structure to provide the tables of means and associated information; default uses the save structure from the most recent ANOVA

Parameters

TERMS = *formula*	Treatment terms whose means are to be required
MEANS = *pointer* or *table*	Saves the means
LOWER = *pointer* or *table*	Saves the lower limits
UPPER = *pointer* or *table*	Saves the upper limits

ACONFIDENCE calculates sets of simultaneous confidence intervals i.e. intervals whose formation takes account of the number of intervals formed, and the fact that the intervals are (slightly) correlated because of the use of a common variance (see Hsu 1996 and Bechhofer, Santner & Goldsman 1995). The methodology implemented in the procedure closely follows that described in Section 1.3 of Hsu (1996).

The type of interval to be formed is specified by the METHOD option, with settings individual, smm (studentized maximum modulus), product (inequality), Bonferroni and Scheffe. The individual setting calculates the intervals as if they were independent, each with the input probability. The smm setting calculates the intervals as correlated, each with a probability adjusted for the multiplicity of intervals. The two settings product and Bonferroni calculate the intervals as independent, but with a probability adjusted for the multiplicity of intervals. These two settings produce very similar intervals although the Bonferroni intervals are always slightly larger. The final setting Scheffe calculates the intervals using pivoted F statistics; see Hsu (1996, Section 1.3.7). The default setting is smm because it produces exact simultaneous confidence intervals.

The TERMS parameter specifies a model formula to define the treatment terms whose means and confidence intervals are required. The means (and the necessary associated information) are usually taken from the most recent analysis of variance (performed by ANOVA), but you can set the SAVE option to a save structure from another ANOVA if you want to examine means from an

earlier analysis. As in ANOVA, the FACTORIAL option sets a limit on the number of factors in each term (default 3). Note: intervals cannot be formed for means whose effects are estimated in different strata.

The MEANS parameter can save the means. If the TERMS parameter specifies a single term, MEANS should be set to a table. If TERMS specifies several terms, you must supply a pointer which will then be set up to contain as many tables as there are terms. Similarly the LOWER parameter can save the lower bounds of the confidence intervals, and the UPPER parameter can save the upper bounds.

You can set option PRINT=* to suppress printing of the intervals; by default PRINT=intervals.

Example 4.1.8 produces simultaneous confidence intervals of various types for the Source means from Example 4.1.

Example 4.1.8

```
  31  ACONFIDENCE [METHOD=smm] Source

Studentized Maximum Modulus 95.0% confidence intervals
------------------------------------------------------


                  Mean       Lower       Upper
      Source
        beef      89.60       81.54       97.66
      cereal      84.90       76.84       92.96
        pork      89.10       81.04       97.16


  32  ACONFIDENCE [METHOD=individual] Source

Individual 95.0% confidence intervals
-------------------------------------


                  Mean       Lower       Upper
      Source
        beef      89.60       83.03       96.17
      cereal      84.90       78.33       91.47
        pork      89.10       82.53       95.67


  33  ACONFIDENCE [METHOD=product] Source

Product inequality 95.0% confidence intervals
---------------------------------------------


                  Mean       Lower       Upper
      Source
        beef      89.60       81.53       97.67
      cereal      84.90       76.83       92.97
        pork      89.10       81.03       97.17


  34  ACONFIDENCE [METHOD=bonferroni] Source

Bonferroni inequality 95.0% confidence intervals
------------------------------------------------


                  Mean       Lower       Upper
      Source
        beef      89.60       81.51       97.69
      cereal      84.90       76.81       92.99
        pork      89.10       81.01       97.19
```

```
  35  ACONFIDENCE [METHOD=scheffe] Source

Scheffe 95.0% confidence intervals
----------------------------------


                  Mean       Lower       Upper
       Source
         beef     89.60       80.15       99.05
       cereal     84.90       75.45       94.35
         pork     89.10       79.65       98.55
```

4.1.9 Multiple comparison tests

AMCOMPARISON procedure

Performs pairwise multiple-comparison tests for ANOVA means (D.M. Smith).

Options

PRINT = *string tokens*	Controls printed output (comparisons, critical, description, lines, letters, plot, mplot, pplot); default lett
METHOD = *string token*	Test to be performed (tukey, snk, regwmr, duncan, scheffe, fplsd, fulsd, bonferroni, sidak); default fpls
FACTORIAL = *scalar*	Limit on the number of factors in each term; default 3
DIRECTION = *string token*	How to sort means (ascending, descending); default asce
PROBABILITY = *scalar*	The required significance level; default 0.05
STUDENTIZE = *string token*	Whether to use the alternative LSD test where the Studentized Range statistic is used instead of Student's t (yes, no); default no
SAVE = *ANOVA save structure*	Save structure to provide the tables of means and associated information; default uses the save structure from the most recent ANOVA

Parameters

TERMS = *formula*	Treatment terms whose means are to be compared
MEANS = *pointer* or *variate*	Saves the (sorted) means
DIFFERENCES = *pointer* or *symmetric matrix*	Saves differences between the (sorted) means
LABELS = *pointer* or *text*	Saves labels for the (sorted) means
LETTERS = *pointer* or *text*	Saves letters indicating groups of means that do not differ significantly
SIGNIFICANCE = *pointer* or *symmetric matrix*	Indicators to show significant comparisons between (sorted) means
CIWIDTH = *pointer* or *symmetric matrix*	Saves the width of the confidence interval for the absolute differences between the (sorted) means

Multiple-comparison tests are designed to take account of the fact that there may be many possible comparisons between pairs of treatment means in an analysis of variance (with n treatments there are $n \times (n-1) / 2$). So, some researchers feel that their significance levels should

be adjusted to take account of all the tests that they might make – and this can be achieved by use of a multiple-comparison test. Conversely, it has been pointed out that multiple-comparisons are unnecessary if you have only a small number of comparisons to make – either because there are few treatments, or because you should have identified beforehand the comparisons that you feel are likely to be of interest. Also, they are inappropriate if the treatments have any sort of structure. For example, the levels of a treatment factor may represent different amounts of a substance like a fertiliser or a drug. It would then be more sensible to assess the treatment effect over all its levels by fitting some sort of trend (see Section 4.5 for information about polynomial and regression contrasts), and implausible to assume that only some of the amounts might have an effect. Alternatively, the treatments may have a factorial structure, and you should then be more interested in studying the main effects and interactions of the various factors (see 4.1). For further discussion of the issues see Nelder (1971), Maindonald & Cox (1984) and Perry (1986).

If, however, multiple-comparison tests are required, they can be obtained using procedure AMCOMPARISON. The methodology implemented in the procedure closely follows that described in Chapter 5 of Hsu (1996).

The TERMS parameter specifies a model formula to define the treatment terms whose means are to be compared. The means (and the necessary associated information) are usually taken from the most recent analysis of variance performed by ANOVA, but you can set the SAVE option to a save structure from another ANOVA if you want to examine means from an earlier analysis. As in ANOVA, the FACTORIAL option sets a limit on the number of factors in each term (default 3).

Printed output is controlled by the PRINT option, with settings:

comparisons	indicates the significance or non-significance of the comparison between each pair of means;
critical	gives critical values for the t-statistic for situations where these do not vary amongst the comparisons (i.e. for the Scheffe, Bonferroni and Sidak methods, as well as the Fisher LSD methods provided all the comparisons have the same number of residual degrees of freedom);
description	provides a description including information such as the experiment-wise and compartment-wise error rates;
lines	gives the means, with lines joining those that do not differ significantly;
letters	gives the means, with identical letters (a, b etc.) alongside those that do not differ significantly;
mplot	does a mean-mean scatter plot (synonym plot);
pplot	displays the probabilities in a shade plot.

By default PRINT=letters.

The means are usually sorted into ascending order, but you can set option DIRECTION=descending for descending order, or DIRECTION=* to leave them in their original order. Note, though, that the lines joining means with non-significant differences may then be broken.

If the standard errors for the differences between the means are unequal (as will happen, for example, if the means have unequal replication), the memberships of the groups defined by the lines or letters may be inconsistent. Suppose, for example, you have ordered means A, B and C. If the s.e.d. for A vs. C is large compared to those for A vs. B and B vs C, you might find that there is no significant difference between A and C, but there are significant differences between A and B, and between B and C. So treatments A and B and treatments B and C would be in different groups. However, treatments A and C (which are further apart) would be in the same group. This contradicts the idea behind multiple comparisons, where you expect that if means m_1 and m_2 are in the same group, than any mean between them should be in that group too. If AMCOMPARISON finds inconsistencies like this, it gives a diagnostic and suppresses the printing

of lines and letters (but not the other types of output).

The mean-mean scatter plot allows you to assess the confidence region for the difference between each pair of means visually. It has grid lines from both the x- and y-axis at the position of each mean, and a diagonal line at 45 degrees marking y=x. The confidence interval for each pair of means is plotted as a line at an angle of −45 degrees and centred on the intersection above the line y=x of the grid lines for the two means (so the y grid line is for the larger of the two means, and the x grid line is for the smaller mean). The difference between the means is significant if their confidence line does not intersect the line y=x. For more details, see Hsu (1996) pages 151-153.

The shade plot displays the probabilities in a symmetric matrix. The colour of each cell represents the probability for the difference between the means for the treatments in the corresponding row and column.

The type of test to be performed is specified by the `METHOD` option, with settings `Tukey`, `SNK` (Student-Newman-Keuls), `REGWMR` (Ryan/Einot-Gabriel/Welsch multiple range test), `Duncan`, `Scheffe`, `FPLSD` (Fisher's Protected Least Significant Difference), `FULSD` (Fisher's Unprotected Least Significant Difference), `Bonferroni` and `Sidak`. The `PROBABILITY` option allows the experiment-wise significance level for the intervals to be changed from the default 0.05 (e.g. to 0.01). The `STUDENTIZE` option can specify that the Fisher's protected or unprotected LSD tests should use the Studentized Range statistic rather than Student's t (for further information see Hsu 1996, page 139).

The `MEANS` parameter can save the means, sorted according to the `DIRECTION` option and omitting any that were non-estimable. If the `TERMS` parameter specifies a single term, `MEANS` should be set to a variate. If `TERMS` specifies several terms, you must supply a pointer which will then be set up to contain as many variates as there are terms. Similarly the `LABELS` parameter can save labels to identify the means, in either a text (for a single term) or in a pointer of texts (for several). Likewise the `LETTERS` parameter can save texts with the letters identifying means that do not differ significantly, and the `SIGNIFICANCE` parameter can save symmetric matrices containing ones or zeros according to whether the various comparisons were significant or non-significant. The `DIFFERENCES` parameter can save symmetric matrices containing the differences between the (sorted) means, and the `CIWIDTH` parameter can save symmetric matrices containing the widths of the confidence intervals for the differences.

We can obtain a Bonferroni multiple-comparison test for the `Source` means in Example 4.1.8 by

```
AMCOMPARISON [METHOD=bonferroni] Source
```

as shown in Example 4.1.9. (However, this is not really necessary, as there are only three treatments here!)

Example 4.1.9

```
Bonferroni test
===============

Source
------


             Mean
cereal      84.90 a
  pork      89.10 a
  beef      89.60 a
```

4.1.10 Simultaneous confidence limits around a control

AMDUNNETT procedure

Forms Dunnett's simultaneous confidence interval around a control (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (interval); default inte
METHOD = *string token*	Form of the alternative hypothesis (twosided, greaterthan, lessthan); default twos
CIPROBABILITY = *scalar*	Probability level for the confidence interval; default 0.95, i.e. a 95% confidence interval
LOWER = *scalar*	Saves the lower confidence limit
UPPER = *scalar*	Saves the upper confidence limit
SAVE = *ANOVA save structure*	Save structure to provide the means; default uses the save structure from the most recent ANOVA

Parameters

FACTOR = *factors*	Define the model term whose means are to be compared
CONTROL = *scalars* or *texts*	Scalar or single-valued text for each factor to identify which of the means of the term is the control; default uses the reference level of the FACTOR

Dunnett's test (Dunnett 1955, 1989) is useful when you want to compare several treatments with a control treatment, and use a critical value that controls the chance that any one comparison may be found significant when there are no true differences. (It is designed thus to take account of the fact that you are making multiple comparisons with the control.)

The FACTOR parameter lists the factors that define the treatment term whose means are to be compared. The means are usually taken from the most recent analysis of variance (performed by ANOVA), but you can set the SAVE option to a save structure from another ANOVA if you want to examine means from an earlier analysis. The CONTROL parameter specifies a list of scalars to identify the levels of the factors that correspond to the control, or you can use a string (or single-valued text) to identify the level of any factor that has labels. If CONTROL is unset, AMDUNNETT uses the reference level of the FACTOR.

The METHOD option defines the type of interval that is formed. By default AMDUNNETT forms a two-sided interval. If you set METHOD=lowerthan, a lower confidence interval is formed to assess the one-sided test of the null hypothesis that the treatment means are not lower than the control mean. Alternatively, you can set METHOD=greaterthan, to obtain an upper confidence interval to assess the one-sided test of the null hypothesis that the treatment means are not greater than the mean of the control.

The probability for the confidence interval is specified by the CIPROBABILITY option; the default 0.95 gives a 95% interval. The lower and upper values of the interval can be saved (in scalars) using the LOWER and UPPER options, respectively. By default the interval is printed, but this can be suppressed by setting option PRINT=*.

Example 4.1.10 continues Example 4.1.9, forming a simultaneous confidence interval around the cereal level of the Source treatment. Again this is not really necessary, as there are only three treatments, but the fact that the interval includes the means for both Beef and Pork confirms the conclusion already noted from Example 4.1, that there are no differences between the sources of protein.

Example 4.1.10

```
  37  AMDUNNETT  Source; CONTROL='cereal'

Dunnett's simultaneous two-sided confidence interval around control
-------------------------------------------------------------------

Source means.
Control level: cereal.

95% confidence interval: (74.38, 95.42)
```

Steel's many-one rank test (procedure STEEL), which provides a nonparametric alternative to Dunnett's test, is described in Subsection 2.6.3.

4.2 Designs with several error terms

The units in the designs covered in 4.1 had no structure: they were assumed to be from a single homogeneous population. The randomization was over the design as a whole, without taking account of any groupings of the units, and there was thus a single error term. Often, however, the population of units is not homogeneous. The rats used to study a set of diets might be grouped according to their litter. An agricultural experiment might involve several different fields, or parts of a field, all with different underlying levels of fertility. An industrial experiment might need to be conducted on several different days, with different batches of material. Or you might wish to impose a structure artificially, by trying to form sets of similar units (and perhaps also subsets) with the aim of decreasing the variability of the experiment.

This structure should then be reflected in the way that you do the randomization and apply the treatments. Some examples are described below. Others can be found in text books on design of experiments: for example, Cochran & Cox (1957), John (1971), John & Quenouille (1977) and Mead (1988).

4.2.1 The BLOCKSTRUCTURE directive

BLOCKSTRUCTURE directive

Defines the blocking structure of the design and hence the strata and the error terms.

No options

Parameter

formula	Block model (defines the strata or error terms for subsequent ANOVA statements)

The BLOCKSTRUCTURE directive specifies the underlying (or *blocking*) structure of the design that is to be analysed. Examples of its use are given below and in 4.3 and 4.7. For unstructured designs with a single error term you can omit this directive, as described in Section 4.1.

In many designs, the units are nested. The simplest is the randomized block design. Here the units are grouped into sets, known as *blocks*, the aim being that units in the same block should be more similar than those in different blocks. The allocation of the treatments is randomized independently within each block. The design thus has two sources of random variation: differences between blocks as a whole, and differences between the units within each block. An example is in 4.3, where the units are plots of land and the blocks are groupings of nearby plots. The block model is

```
Blocks/Plots
```

indicating that the plots are nested within blocks, and thus that there is no special similarity, for example, between the plot numbered 3 in block 1 and plot 3 of the other blocks. The expanded version of the formula is

```
Blocks + Blocks.Plots
```

giving terms for the differences between blocks as a whole, and the differences between the units within each block, as required.

In the simplest form of the randomized block design, there is a single treatment factor, each of whose levels occurs once in every block. More complicated arrangements are possible, but each treatment combination must still occur exactly the same number of times in every block. This means that any differences found between the blocks cannot be caused by differences between treatments. Thus the treatment terms are all estimated between the plots within the blocks. If the blocks have been chosen successfully, the variation within the blocks should be less than that between blocks, and so the treatment estimates will be less variable than if a completely randomized design had been used.

For the example in Section 4.3, the treatments have the structure

```
TREATMENTSTRUCTURE Fumigant/(Dose*Type)
```

If you look at the first analysis shown in Section 4.3, which ignores the covariate discussed later in that section, you can see that the analysis of variance is split into two components called *strata*. The `Blocks` stratum contains the sums of squares between blocks; this all arises from the variability between the blocks. The `Blocks.Plots` stratum contains the sum of squares for the plots within the blocks; this is partitioned into the sums of squares due to each of the treatment terms, and a residual against which these can be assessed.

Thus, you can deduce the block model from the structure of the units, which should correspond to the way in which the randomization has been done. Genstat expands the block model to form the list of *block* (or *error*) terms, each of which defines a stratum corresponding to one of the sources of variability in the design. Alternatively, if you prefer to deduce the error terms by some other means, as for example if you follow the philosophy of fixed and random effects, you can specify the block model to be the sum of these terms.

In the analysis, Genstat initially partitions the sums of squares according to the block model alone. This gives the total sum of squares for each of the strata. Then it partitions each stratum sum of squares into sums of squares for those treatment terms estimated in that stratum, and a residual which provides an estimate of variability against which these treatment sums of squares should be compared.

In the randomized block design, the treatments are estimated only in the final (bottom) stratum. You would thus get the same sums of squares if you omitted the `BLOCKSTRUCTURE` statement and put Blocks at the start of the treatment model. In the example, you would put

```
TREATMENTSTRUCTURE Blocks + Fumigant/(Dose*Type)
```

The effect would also be the same if you specified this treatment model and retained the block model, because any model term that occurs in both the block and treatment models is deleted from the block model. So `Blocks` would be deleted and there would then be a single stratum `Blocks.Plots`. You may prefer this specification as it gives an analysis of variance that looks more conventional. However the form in the example better reflects the structure of the design, as it correctly identifies `Blocks` as an error term. It also allows for the possibility of treatments being estimated between blocks, as in the balanced-incomplete-block design.

The simplest design in which the treatments are not all estimated in one stratum is the split-plot design. This again is a nested structure. It was originally devised for agricultural experiments where some of the factors can be applied to smaller plots of land than others. However, it also occurs in industrial experiments (for example Cox 1958, page 149), in medical experiments (Armitage 1974), and even in the study of cake mixtures (Cochran & Cox 1957,

page 299). A well-known example (Yates 1937, page 74; John 1971, page 99) is shown below. There are two treatment factors: three different varieties of oats (line 8), and four levels of nitrogen (line 9). Because of limitations on the machines for sowing seed, different varieties cannot conveniently be applied to plots as small as those that can be used for the different rates of fertilizer. So the design was set up in two stages. First of all, the blocks were each divided into three plots of the size required for the varieties, and the three varieties were randomly allocated to the plots within each block (exactly as in the randomized blocks design). Then each of these plots, or *whole-plots* as they are usually known, was split into four *sub-plots* (one for each rate of nitrogen), and the allocation of nitrogen was randomized independently within each whole-plot.

To specify the block structure for this design, three factors are required (lines 4 to 6): `Blocks` to indicate the block (1 to 6) to which each unit belongs, `Wplots` to indicate the whole-plot (numbered 1 to 3 within each block), and `Subplots` to identify the sub-plot (numbered 1 to 4 within each whole-plot). You can use the same whole-plot numbers in each block, since the block model (defined below) does not contain any main effect for whole-plots: that is, Genstat will not assume any special similarity between whole-plots with the same numbers. In fact it is best that you do use the same numbering, since otherwise the tables of residuals become very sparse and wasteful of space. In situations like this, it is often convenient to arrange the values of the factors in the block model in a systematic order, for example to reflect positions on the field. This makes patterns in the tables of residuals easier to see. The `GENERATE` directive (4.13.1) provides a convenient way of specifying their values (line 7).

The design has sub-plots nested within whole-plots, which are themselves nested within the blocks: that is,

```
BLOCKSTRUCTURE Blocks/Wplots/Subplots
```

The block model expands to

```
Blocks + Blocks.Wplots + Blocks.Wplots.Subplots
```

(see 4.1.1 and 1:1.6.3), giving strata for variation between blocks, between whole-plots within the blocks, and for sub-plots within the whole-plots (within blocks). The treatment model (line 24) specifies terms for the main effects of variety and of nitrogen, and for their interaction (4.1.1).

Just as in the randomized block design, the blocks all contain the same sets of treatments, and so no treatments are estimated in the `Blocks` stratum. But varieties, which were applied to whole-plots, are estimated in the `Blocks.Wplots` stratum; in conventional terminology this is called the stratum for whole-plots within blocks. The variance ratio for varieties is calculated by dividing the `Variety` mean square by the `Blocks.Wplots` residual mean square. It is easy to see that this is the correct thing to do. When we look to see whether the varieties differ we are really trying to answer the question: "Do the yields from the three sets of whole-plots, on the first of which the variety Victory was grown, on the second Golden rain, and on the third Marvellous, differ by more than the amount that we would expect for any three randomly chosen sets of whole-plots?". Technically, variety is said to be *confounded* with whole plots. The terms for `Nitrogen`, which was applied to sub-plots, and for the `Variety.Nitrogen` interaction are both estimated in the stratum for sub-plots within whole-plots (`Blocks.Wplots.Subplots`).

Variance ratios are also produced for block terms, provided there is an appropriate term lower in the hierarchy of strata with which to compare them. Here `Blocks` can be compared with `Blocks.Wplots`, and `Blocks.Wplots` with `Blocks.Wplots.Subplots`. Thus, for example, the variance ratio of 5.28 for `Blocks` indicates that the blocks of land in this experiment are indeed more variable than the plots within each block. However, F probabilities are not produced for variance ratios of block terms. Conversely, in the block formula for replicated Latin squares, discussed later in this section,

```
Squares / (Rows * Columns)
```

which expands to

```
Squares + Squares.Rows + Squares.Columns
+ Squares.Rows.Columns
```

the term `Squares` could equally well be compared with either `Squares.Rows` or `Squares.Columns`. The ratio of most interest would depend on the exact layout of the trial; for example, if the squares were alongside each other, it might be interesting to see whether the squares were more variable than columns within squares. Genstat has no information about layout, so it leaves you to make these comparisons yourself.

Example 4.2.1a

```
  2  " Split-plot design (Yates 1937, p.74; also John 1971, p.99)."
  3  UNITS [NVALUES=72]
  4  FACTOR [LEVELS=6] Blocks
  5  & [LEVELS=3] Wplots
  6  & [LEVELS=4] Subplots
  7  GENERATE Blocks,Wplots,Subplots
  8  FACTOR [LABELS=!T(Victory,'Golden rain',Marvellous)] Variety
  9  & [LABELS=!T('0 cwt','0.2 cwt','0.4 cwt','0.6 cwt')] Nitrogen
 10  VARIATE Yield; EXTRA=' of oats'
 11  READ [SERIAL=yes] Nitrogen,Variety,Yield

   Identifier   Minimum      Mean   Maximum    Values   Missing
        Yield     53.00     104.0     174.0        72         0

   Identifier    Values   Missing    Levels
     Nitrogen        72         0         4
      Variety        72         0         3

 24  TREATMENTSTRUCTURE Variety*Nitrogen
 25  BLOCKSTRUCTURE Blocks/Wplots/Subplots
 26  ANOVA  [FPROBABILITY=yes; PSE=differences,lsd] Yield

Analysis of variance
====================

Variate: Yield of oats

Source of variation     d.f.       s.s.       m.s.    v.r.  F pr.

Blocks stratum             5    15875.3     3175.1    5.28

Blocks.Wplots stratum
Variety                    2     1786.4      893.2    1.49  0.272
Residual                  10     6013.3      601.3    3.40

Blocks.Wplots.Subplots stratum
Nitrogen                   3    20020.5     6673.5   37.69  <.001
Variety.Nitrogen           6      321.8       53.6    0.30  0.932
Residual                  45     7968.8      177.1

Total                     71    51985.9


* MESSAGE: the following units have large residuals.

Blocks 1            31.4   s.e. 14.8


Tables of means
===============

Variate: Yield of oats

Grand mean  104.0

  Variety     Victory Golden rain  Marvellous
                 97.6       104.5       109.8
```

```
 Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
              79.4     98.9    114.2    123.4

     Variety Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
     Victory              71.5     89.7    110.8    118.5
 Golden rain              80.0     98.5    114.7    124.8
  Marvellous              86.7    108.5    117.2    126.8


Standard errors of differences of means
---------------------------------------

Table              Variety    Nitrogen     Variety
                                           Nitrogen
rep.                    24          18           6
s.e.d.                7.08        4.44        9.72
d.f.                    10          45       30.23
Except when comparing means with the same level(s) of
Variety                                       7.68
d.f.                                            45


Least significant differences of means (5% level)
--------------------------------------------------

Table              Variety    Nitrogen     Variety
                                           Nitrogen
rep.                    24          18           6
l.s.d.                15.77        8.93       19.83
d.f.                    10          45       30.23

Except when comparing means with the same level(s) of
Variety                                       15.47
d.f.                                            45
```

This shows the default output from `ANOVA`, but with the addition of F probabilities in the analysis-of-variance table, and least significant differences as well as standard errors of differences. Notice that a separate s.e.d. (and l.s.d.) is given for comparisons between means in the variety × nitrogen table when both means are for the same variety. To see why this is necessary, consider how you might calculate the difference between two of the means, using the original data. One way would be to look at each block to find the pairs of sub-plots with these two treatment combinations, and then to calculate the sum of the differences between the values recorded on each pair. If the means are both for the same variety, each pair of sub-plots will be within the same whole-plot; when you take the differences any whole-plot variation then cancels out, to give a smaller s.e.d. The degrees of freedom for the s.e.d. between means with the same variety is 45, which is the residual degrees of freedom for the `Blocks.Wplots.Subplots` stratum. The other comparisons involve both whole plot and sub-plot variation. For comparisons like these, approximate numbers of degrees of freedom are estimated using Satterthwaite's method; these lie between the minimum of the residual degrees of freedom in any of the strata where effects contributing to the table are estimated, and the sum of the residual degrees of freedom in those strata.

Example 4.2.1a also illustrates the messages that are printed about large residuals. Checking is done for the residuals of every stratum, and the criterion used is the same that used in regression analysis (3.1.2). Here there are no large residuals in either the `Blocks.Wplots.Subplots` or the `Block.Wplots` strata, but the residual for block 1 is 31.4 compared to its standard error of 14.8. In this instance, the message can be taken as confirming the success of the choice of blocks: that is, that the yields of the plots in block 1 are consistently higher than those in other blocks. Large residuals in the lower strata might indicate aberrant values, or outliers.

The second section of output first plots the means, as shown in Figure 4.2.1a, and prints the tables of residuals and estimated treatment effects from each stratum, followed by the

coefficients of variation. It then uses procedure AFIELDRESIDUALS (4.1.4) to plot the residuals in field layout (Figure 4.2.1b). These are *combined* residuals (incorporating the block and whole-plot residuals as well as the sub-plot residuals), so they should show the fertility trends in the field.

Example 4.2.1b

```
  27  AGRAPH [METHOD=lines]
  28  ADISPLAY [PRINT=effects,residuals,%cv]

Tables of effects and residuals
===============================

Variate: Yield of oats

Blocks stratum
--------------

Blocks residuals,  s.e. 14.85,  rep. 12

   Blocks        1        2        3        4        5        6
             31.4     -5.8      3.3    -13.1     -8.1     -7.7

Blocks.Wplots stratum
---------------------

Variety effects,  e.s.e. 5.01,  rep. 24

  Variety     Victory Golden rain  Marvellous
               -6.3         0.5         5.8

Blocks.Wplots residuals,  s.e. 9.14,  rep. 4

   Blocks   Wplots        1        2        3
        1             -11.4     14.0     -2.6
        2              -9.0     -0.3      9.3
        3               5.5      8.2    -13.7
        4             -11.5      4.1      7.4
        5              -9.7     -7.1     16.8
        6              -0.4     -6.5      6.9

Blocks.Wplots.Subplots stratum
------------------------------

Nitrogen effects,  e.s.e. 3.14,  rep. 18

 Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
            -24.6     -5.1     10.2     19.4

Variety.Nitrogen effects,  e.s.e. 5.43,  rep. 6

     Variety Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
     Victory              -1.5     -2.9      3.0      1.5
 Golden rain               0.1     -0.9     -0.1      0.9
  Marvellous               1.5      3.8     -2.9     -2.4

Blocks.Wplots.Subplots residuals,  s.e. 10.52,  rep. 1

   Blocks   Wplots Subplots        1        2        3        4
        1        1                9.2    -19.1     11.5     -1.6
                 2               -5.9     -5.0     10.1      0.8
                 3                8.2    -13.2     17.6    -12.6
        2        1                1.6     -1.9     -4.7      5.0
                 2                9.6      8.6      5.5    -23.7
                 3                1.0    -19.5     13.8      4.7
        3        1                0.8      2.6     15.4    -18.8
                 2                5.7      4.0     -7.6     -2.1
                 3               -0.1     -8.1     11.7     -3.5
        4        1               16.4     -3.3      0.9    -14.0
                 2                9.0    -11.7     10.2     -7.5
                 3                6.0     -5.2      3.1     -3.9
```

```
        5       1            -11.1     -2.2     -7.9     21.2
                2             16.3    -17.4     11.6    -10.5
                3              6.1    -11.5     11.8     -6.4
        6       1             15.3    -10.4      2.6     -7.5
                2             -6.6    -14.4     23.2     -2.2
                3             11.1     -8.7      2.6     -5.0

Stratum standard errors and coefficients of variation
=====================================================

Variate: Yield of oats

Stratum                    d.f.          s.e.         cv%

Blocks                        5         16.27        15.6
Blocks.Wplots                10         12.26        11.8
Blocks.Wplots.Subplots       45         13.31        12.8

  29  VARIATE [VALUES=2(1...18)2] Row
  30  & [VALUES=(1,2)18,(3,4)18] Column
  31  AFIELDRESIDUALS Y=Row; X=Column
```

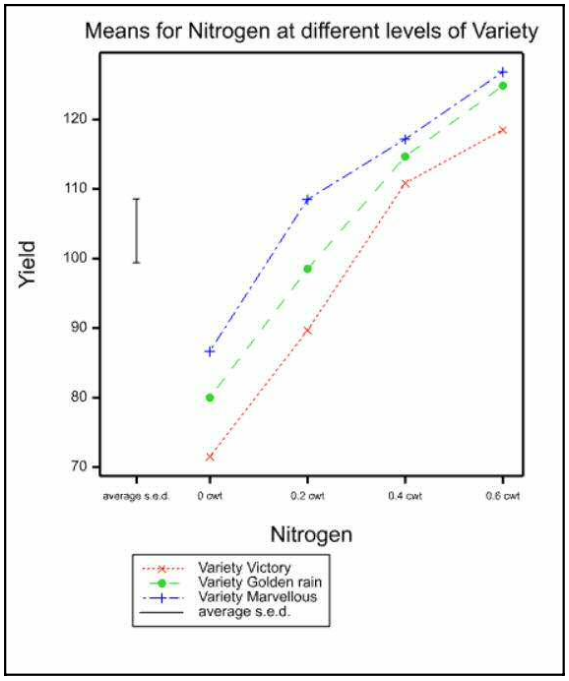


Figure 4.2.1a

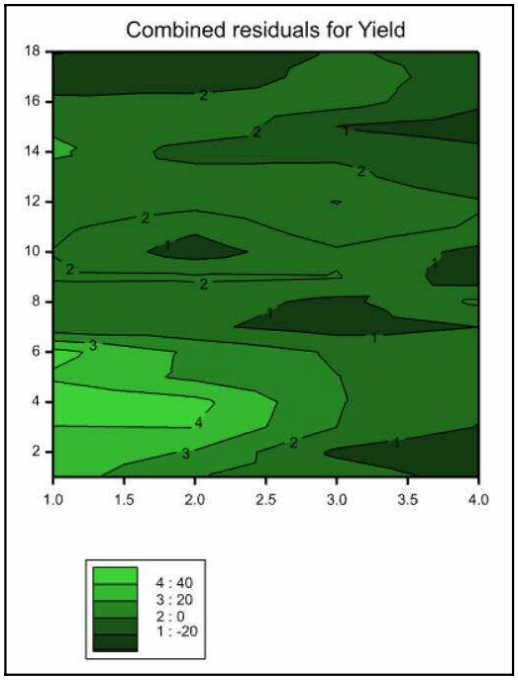


Figure 4.2.1b

There are some designs where the units have a crossed instead of a nested structure. A simple example is the Latin square. This was devised for agricultural experiments to cater for situations where there are fertility trends both along and across the field, but it can be used whenever there are two independent ways of grouping the units: for example time of testing and batch of material, or the litter of the rat and its order by weight within the litter. In field experiments, the plots are arranged in a square, with blocking factors called `Rows` and `Columns`. These each have the same number of levels as there are treatments. Values of the single treatment factor are arranged so that each level occurs once in each row and once in each column. The block structure has rows crossed with columns: that is,

```
BLOCKSTRUCTURE Rows*Columns
( = Rows + Columns + Rows.Columns )
```

The treatments are estimated only in the Rows.Columns stratum. Removing variation between rows and between columns should make these estimates less variable. We do not include output from a Latin square, but recommend that you try an example from one of the books listed earlier in this section.

More complicated designs can involve both crossing and nesting, for example:

```
BLOCKSTRUCTURE Squares/(Rows*Columns)
(= Squares + Squares.Rows + Squares.Columns +
Squares.Rows.Columns)
```

which is used for replicated Latin squares (John 1971, page 114), quasi-Latin squares (Cochran & Cox 1957, pages 317-324; John & Quenouille 1977, pages 146-152) and lattice squares (Cochran & Cox 1957, pages 483-506; John & Quenouille 1977, page 192). Another example is

```
BLOCKSTRUCTURE (Rows*Columns)/Subplots
(= Rows + Columns + Rows.Columns + Rows.Columns.Subplots )
```

which is for a Latin square with the plots split into sub-plots (Kempthorne 1952, page 378).

If the factors in the block formula do not provide a unique index for every unit of the experiment, the terms in the block model will not account for all the variation. Genstat must then define a final stratum to contain the variation between the sets of units whose levels are the same for each block factor. At the end of the block model, Genstat therefore sets up an extra term containing all the block factors, together with an extra "factor", denoted *units*, which numbers the units within each set. So, for the randomized block design, you could put just

```
BLOCKSTRUCTURE Blocks
```

which would then become

```
BLOCKSTRUCTURE Blocks + Blocks.*units*
```

Likewise, for the split-plot design,

```
BLOCKSTRUCTURE Blocks/Wplots
```

would become

```
BLOCKSTRUCTURE Blocks/Wplots + Blocks.Wplots.*units*
```

Consequently, if you define no block structure at all, Genstat assumes

```
BLOCKSTRUCTURE *units*
```

giving a single source of variation representing random differences between the units; this defines a completely randomized design, as in 4.1. However, you may prefer to define a more meaningful labelling of the units, for example

```
BLOCKSTRUCTURE Rat
```

The factor Rat would be very easy to set up; it simply contains the numbers 1, 2, onwards. To produce a factor equivalent to *units* in more complicated situations, you can use procedure AFUNITS. For example

```
AFUNITS [BLOCKSTRUCTURE=Blocks/Wplots] Splot
```

to generate a factor Splots to index the units within Blocks and Wplots.

4.2.2 The ABLUPS procedure

ABLUPS procedure

Calculates BLUPs for block terms in an ANOVA analysis (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (blups); default blup
PTERMS = *formula*	Specifies the block terms whose BLUPs are to be

	printed; default is to print them all
PSE = *string tokens*	Types of standard errors to be printed with the BLUPs (differences, alldifferences, blups, allblups); default diff, blup
SAVE = *identifier*	Save structure for the ANOVA analysis; default is to take the most recent ANOVA analysis

Parameters

TERMS = *formula*	Block terms whose BLUPs etc are to be saved
BLUPS = *table* or *pointer* to *tables*	Saves the BLUPs
SEBLUPS = *table* or *pointer* to *tables*	Standard errors for the BLUPs of each term
SEDMEANS = *symmetric matrix* or *pointer* to *symmetric matrices*	Standard errors of differences between the BLUPs of each term

The ABLUPS procedure can be used to calculate best linear unbiased predictors (BLUPs) for block terms. These differ from the ordinary ANOVA residuals in that they are *predictors* rather than estimates of the random effects; see 5.3.3 They usually have the property of *shrinkage*, i.e. they are biased towards zero. As a result they are more likely to represent future observations of the same terms.

This is illustrated in Example 4.2.2, which shows the BLUPs for the block terms in Example 4.2.1a. Notice that their absolute values are smaller than the residuals printed in Example 4.2.1b.

Example 4.2.2

```
  32  ABLUPS

BLUPS for block terms
---------------------


       Blocks
            1      25.422
            2      -4.706
            3       2.657
            4     -10.583
            5      -6.530
            6      -6.260


Standard error: 8.342

Standard error of differences: 9.013


       Wplots           1           2           3
       Blocks
            1      -3.854      14.077       2.348
            2      -7.116      -1.001       5.789
            3       4.299       6.209      -9.194
            4      -9.848       1.117       3.498
            5      -7.916      -6.064      10.751
            6      -1.316      -5.637       3.858


Standard error: 7.782
```

```
Average standard error of differences: 10.73
Minimum standard error of differences: 9.35
Maximum standard error of differences: 11.31
```

By default, the BLUPs are from most recent ANOVA analysis. However, you can use an earlier analysis, by using the SAVE option of ABLUPS to specify its save structure (saved using the SAVE parameter of the earlier ANOVA command).

The BLUPs are usually printed. However, this can be suppressed by setting option PRINT=*. The PTERMS option can be used to specify the block terms whose BLUPs are to be printed. The default is to print the BLUPs for all the block terms.

The PSE option specifies which standard errors are printed, with the following settings.

differences	prints a summary of the standard errors of differences between pairs of BLUPs,
alldifferences	prints all the standard errors of differences between pairs of BLUPs,
blups	prints a summary of the standard errors of the BLUPs, and
allblups	prints all the standard errors of the BLUPs.

By default PSE=differences,blups.

The parameters of ABLUPS can save the BLUPs and standard errors. The TERMS parameter specifies the block terms whose BLUPs or standard errors are to be saved. The BLUPS parameter saves tables of BLUPs, the SEMEANS parameter saves tables containing their standard errors, and the SEDMEANS parameter saves symmetric matrices containing standard errors of differences between pairs of BLUPs. If you have a single term, you can supply a table or symmetric matrix for each of these parameters, as appropriate. However, if you have several terms, you must supply a pointer which will then be set up to contain as many tables or symmetric matrices as there are TERMS. A fault is given if the pointer has been defined already with a different number of elements to the number of TERMS.

4.2.3 Multitiered designs

The earlier part of this section has shown how one model formula is required to specify an analysis of variance when there are several error terms. The underlying structure of the data (which indicates the error terms for the analysis) is defined by a model formula specified by the BLOCKSTRUCTURE directive (4.2.1), while the treatment terms to be fitted in the analysis are defined in a model formula specified by the TREATMENTSTRUCTURE directive (4.1.1). However, experiments that involve multiple randomizations (Brien & Payne 1999, Brien & Bailey 2006), such as two-phase experiments, may require more than two model formulae to define their analysis correctly.

For example, Brien (1983) considered a two-phase experiment set up to evaluate a set of wines. These are evaluated at a tasting where several tasters are given the wines over a number of sittings. One wine is presented to each taster at a sitting, and each wine is evaluated only once by each taster. The order of presentation of the wines is randomized for each taster. The basic observational unit is a glass of wine presented to a particular taster in the tasting phase. These have a structure of tasters/sittings. If this phase represented the whole experiment, tasters/sittings would be the block formula, and the treatment formula would be the factor wines. So we would have

```
BLOCKSTRUCTURE tasters/sittings
TREATMENTSTRUCTURE wines
```

Now suppose that the wines were produced from a field experiment and, in fact, that each one was produced from one of the plots of a randomized-block design. The second model formula would then be blocks/plots, and the final formula would be treatments (the factor identifying the treatments applied in the field). Designs like this can be analysed by procedure

AMTIER.

AMTIER procedure

Analyses a multitiered design by an analysis of variance specified by up to three model formulae (C.J. Brien & R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output from the analysis (aovtable, aovpseudotable, design, effects, fittedvalues); default aovt
F1 = *formula*	First model formula
F2 = *formula*	Second model formula
F3 = *formula*	Third model formula
FACTORIAL = *scalar*	Limit on the number of factors in a model term
F2BALANCETYPE = *string token*	Type of balance required for F2 (orthogonal, firstorder); default orth
F3BALANCETYPE = *string token*	Type of balance required for F3 (orthogonal, firstorder); default orth
PSEUDOTERMS = *formula structures*	Specifies pseudo-terms for terms in the F1, F2 or F3 formulae
DESIGN = *tree*	Saves or specifies details of the design and analysis
SEED = *scalar*	Seed for random numbers to generate dummy variate for determining the design; default 13579
TOLERANCE = *variate*	Tolerance for zero sweeps in dummy and y-variate analyses
DPRINT = *string tokens*	Controls debug output (setup, analysis, dummyanalysis); default * i.e. none

Parameters

Y = *variates*	Each of these contains the data values for an analysis
RESIDUALS = *variates*	Saves the residuals from each analysis
FITTEDVALUES = *variates*	Saves the fitted values from each analysis
SAVE = *pointers*	Save structure for each analysis (to use in AMTDISPLAY)

The three model formulae are specified by the options F1, F2 and F3. For the example in Brien (1983), the statement would be

```
AMTIER [F1=tasters/sittings; F2=blocks/plots;\
       F3=treatments] Y
```

The Y parameter specifies the response variate. Residuals and fitted values can be saved by the RESIDUALS and FITTEDVALUES parameters, respectively. The SAVE parameter can save a pointer containing the full details of the analysis. This can be used as input to the AMTDISPLAY procedure to obtain further output, or to the AMTKEEP procedure to save information into Genstat data structures.

The FACTORIAL option sets a limit on the number of factor in the model terms generated from the formulae. The F2BALANCETYPE and F3BALANCETYPE options control whether the terms from the second and third model formulae are allowed to be first-order balanced rather than orthogonal (see 4.7.2). The default is that the terms are required to be orthogonal. It is emphasized that this applies only to terms from the same model formula. Even if the terms from a model formula are required to be orthogonal, they may still only be structure balanced in relation to terms from other formulae. However, if terms from any model formula are non-

orthogonal, then the experiment is not structure balanced, and so sums of squares for sources differ depending on their order in the model formula. The PSEUDOTERMS option allows you to specify a list of formula structures defining pseudo-terms for some of the terms in the formulae (see 4.7.3). Each pseudoterm formula is of the form

```
group_term // pseudoterms_formula
```

All pseudo-terms must be defined explicitly as none are generated, for example from relations between the group term and other factors. Furthermore, all marginal terms to a pseudoterm need to be included in its formula, irrespective of whether they themselves are pseudoterms. Those that are not pseudo-terms need to occur in one of the three main model formulae and will not be included in the analysis sequence again as a result of their appearance in the pseudo-term formula. The pseudo-terms are placed immediately before the group term in the analysis sequence. Any repetitions of pseudo-terms are removed.

The DESIGN option can save a tree structure representing the design and analysis. You can then specify this as the design in a subsequent AMTIER statement, to avoid having to go through the process of determining the design structure with another response variate from the same experiment. The design structure is determined by a similar dummy analysis process as in the standard ANOVA directive. The TOLERANCE option specifies a variate with two values. The first defines the tolerance multiplier for zero sweeps in the dummy analysis and the second defines the multiplier for use in the analysis of the y-variates.

Printed output is controlled by the PRINT option with settings:

aovtable	to print the analysis-of-variance table,
aovpseudotable	to print the analysis-of-variance table with lines for all the pseudo-terms (generated by pseudo-factors) given explicitly,
design	to display the structure of the design,
effects	to print tables of effects and residuals, and
fittedvalues	to print a table with the y-variate, fitted valued and residuals.

The DPRINT option controls debug output, with settings:

setup	for information from the set-up stage,
analysis	for information from the analysis of the y-variates, and
dummyanalysis	for information from the dummy analysis.

Example 4.2.3 shows the analysis of the example in Brien (1983). Notice that the analysis-of-variance table has three depths instead of the more usual two. The terms blocks and blocks.plots (from the block structure of the field experiment) are estimated within the term tasters.sittings of the tasting experiment, and the term treatments is estimated within the term blocks.plots.

Example 4.2.3

```
 2  FACTOR [NVALUES=60; LEVELS=5] tasters
 3  &      [LEVELS=12] sittings
 4  &      [LEVELS=3] blocks
 5  &      [LEVELS=4] plots,treatments
 6  READ   tasters,sittings,blocks,plots,treatments

  Identifier    Values   Missing    Levels
     tasters        60         0         5
    sittings        60         0        12
      blocks        60         0         3
       plots        60         0         4
  treatments        60         0         4

19  AMTIER [PRINT=aov; F1=tasters/sittings; F2=blocks/plots; F3=treatments]
```

```
Analysis of variance
====================

Source                          d.f.   e.f.
tasters                            4  1.000
tasters.sittings
  blocks                           2  1.000
  blocks.plots
    treatments                     3  1.000
    Residual                       6  1.000
  Residual                        44  1.000
Total                             59  1.000
```

The AMTDISPLAY procedure allows you to obtain further output, and the AMTKEEP procedure allows you to save information into Genstat data structures.

AMTDISPLAY procedure

Displays further output for multitiered experiments analysed by AMTIER (C.J. Brien & R.W. Payne).

Option

PRINT = *string tokens*	Controls printed output from the analysis (aovtable, aovpseudotable, design, effects, fittedvalues); default * i.e. none

Parameter

SAVE = *pointers*	Save structure for each analysis (saved from AMTIER); if this is not set the output is from the most recent AMTIER analysis

---

AMTKEEP procedure

Saves information from the analysis of a multitiered design by AMTIER (C.J. Brien & R.W. Payne).

Options

RESIDUALS = *variate*	Saves the residuals
FITTEDVALUES = *variate*	Saves the fitted values
AOVTABLE = *pointer*	Saves the analysis-of-variance table
SKELETON = *string token*	Whether to save only the skeleton analysis-of-variance table (yes, no); default no
PSEUDOLINES = *string token*	Whether to include lines for pseudo-terms in the analysis-of-variance table (yes, no); default no
OMITMISSINGLINES = *string token*	Whether to omit lines of the analysis-of-variance table that contain only missing values (yes, no); default no
SAVE = *pointer*	Save structure for the analysis; if this is not set, information is saved from the most recent AMTIER analysis

No parameters

4.3 Analysis of covariance

You can do analysis of covariance for any of the designs that can be analysed by ANOVA (4.1). As well as defining the block and treatment models (4.1.1 and 4.2.1), you must also define the covariates. You can either specify a list of variates to act as covariates using the COVARIATE directive (4.3.1), or define more complicated covariate models using the AFCOVARIATES procedure (4.3.2). Then you can do the analysis by ANOVA (4.1.2), get further output by ADISPLAY (4.1.3), and save information by AKEEP (4.6.1), all exactly as in an ordinary analysis of variance.

The example used in this section illustrates the treatment structure of a factorial arrangement of several types of treatment, as well as a control. This structure of *factorial plus added control* can be useful when you wish to examine several ways of modifying a preparation, and also wish to see what would happen if you applied nothing at all. This experiment was done at Rothamsted in 1935 to study soil fumigants for decreasing the numbers of nematodes (or eelworms as they were then known). Further details are given in Cochran & Cox (1957, pages 45-46), although there the data are analysed untransformed. There were four types of fumigant, each of which was applied in either a single or a double dose. A randomized block design was used, with four blocks of twelve plots. In each block, four plots were untreated (to act as controls), and there was one plot for each dose of each type of fumigant. This first section of output analyses the logarithm of the numbers of nematode cysts counted in a sample of 400 grammes of soil, taken at the end of the experiment.

Example 4.3

```
  2  "Example of a factorial + added control and analysis of covariance
 -3    (Cochran & Cox 1957, p.46). A log transformation has been used,
 -4    and unit 43 has a missing value in the y-variate."
  5  UNITS [NVALUES=48]
  6  FACTOR [LEVELS=4] Blocks
  7  & [LEVELS=12] Plots
  8  FACTOR [LEVELS=5; LABELS=!T(None,CN,CS,CM,CK)] Type
  9  & [LEVELS=3; LABELS=!T(None,Single,Double)] Dose
 10  & [LEVELS=2; LABELS=!T('Not fumigated',Fumigated)] Fumigant
 11  GENERATE Blocks,Plots
 12  READ Dose,Type,Initnem,Finalnem

   Identifier   Minimum      Mean   Maximum    Values   Missing
      Initnem     9.000     128.5     283.0        48         0
    Finalnem     80.00     311.7     708.0        48         1

   Identifier    Values   Missing    Levels
         Dose        48         0         3
         Type        48         0         5

 25  CALCULATE Fumigant = NEWLEVELS(Dose; !(1,2,2))
 26  & Initnem,Finalnem = LOG(Initnem,Finalnem)
 27  BLOCKSTRUCTURE Blocks/Plots
 28  TREATMENTSTRUCTURE Fumigant/(Dose*Type)
 29  ANOVA [FPROBABILITY=yes] Finalnem

Analysis of variance
====================

Variate: Finalnem

Source of variation     d.f.(m.v.)      s.s.       m.s.    v.r.  F pr.

Blocks stratum              3          4.0295     1.3432    7.24

Blocks.Plots stratum
Fumigant                    1          0.6918     0.6918    3.73  0.062
Fumigant.Dose               1          0.0650     0.0650    0.35  0.558
Fumigant.Type               3          0.6656     0.2219    1.20  0.325
Fumigant.Dose.Type          3          0.1212     0.0404    0.22  0.883
```

```
Residual                  35(1)       6.4898     0.1854

Total                     46(1)      11.7582


Tables of means
===============

Variate: Finalnem

Grand mean  5.618

 Fumigant Not fumigated    Fumigated
                 5.788        5.533
     rep.           16           32

      Fumigant     Dose    None  Single  Double
 Not fumigated             5.788
     Fumigated                    5.488   5.578

      Fumigant     Type    None      CN      CS      CM      CK
 Not fumigated             5.788
                   rep.      16
     Fumigated                    5.529   5.370   5.763   5.470
                   rep.          8       8       8       8

      Fumigant     Dose    Type    None      CN      CS      CM      CK
 Not fumigated     None            5.788
                           rep.      16
     Fumigated   Single                  5.483   5.280   5.818   5.371
                           rep.          4       4       4       4
                 Double                  5.575   5.461   5.707   5.570
                           rep.          4       4       4       4


Standard errors of differences of means
---------------------------------------

Table             Fumigant    Fumigant    Fumigant    Fumigant
                                  Dose        Type        Dose
                                                          Type
rep.               unequal          16     unequal     unequal
d.f.                    35          35          35          35
s.e.d.                                      0.2153      0.3045  min.rep
                    0.1318      0.1522      0.1865      0.2407  max-min
                                            0.1522X     0.1522X max.rep

(No comparisons in categories where s.e.d. marked with an X)
(Not adjusted for missing values)


Missing values
==============

Variate: Finalnem

 Unit  estimate
   43     5.071

Max. no. iterations 3
```

The block model for this design (line 27) is discussed in 4.2.1. The treatment model requires three factors (lines 8 to 10): `Fumigant` indicates whether or not the plot has been fumigated with any type of fumigant at all, `Type` indicates the type of fumigant (if any), and `Dose` indicates how much was used. If you examine the table of means classified by `Fumigant`, `Dose` and `Type`, you can see that `Dose` and `Type` have a crossed structure within the `'fumigated'` level of `Fumigant`. This suggests a treatment model

```
Fumigant/(Dose*Type)
```

which expands to

```
Fumigant + Fumigant.Dose + Fumigant.Type
+ Fumigant.Dose.Type
```

As explained in 4.1.1, a term like `Fumigant.Dose` represents all the joint effects of these two factors, after eliminating any terms that precede it in the model. The main effect `Fumigant` removes the difference between no fumigant and any positive dose (either single or double). So `Fumigant.Dose` represents the difference between a single and a double dose. Similarly, `Fumigant.Type` represents differences between types of fumigant, and `Fumigant.Dose.Type` represents the interaction between dose and type of fumigant. Notice that one of the units has a missing value; this aspect of the analysis is explained in Section 4.4.

The numbers of nematodes were also sampled at the start of the experiment, before any treatments were applied. This gives extra information about the plots, which we can incorporate into the analysis by using the original numbers as a covariate. We have transformed the initial numbers to logarithms, in the same way as the final numbers; so the model to be fitted assumes that the final numbers are related to some power of the original numbers.

You can use covariates to incorporate any quantitative information about the units into the model. In field experiments there may often be linear trends in fertility. These can be estimated and removed by fitting a covariate of the position of the plot along the direction of the trend. For a quadratic trend, you would also include a covariate containing the squares of the positions. In experiments on animals, you may wish to use measurements such as the original weight. However the assumption is always that the y-variate is linearly related to the covariates.

After you have defined variates to contain the measurements that are to act as covariates and done any transformations that may be required, you list them in the `COVARIATE` directive.

4.3.1 The `COVARIATE` directive

`COVARIATE` directive

Specifies covariates for use in subsequent `ANOVA` statements.

No options

Parameter

variates or *pointers*	Covariates

Covariates are incorporated into the model as terms for a linear regression. Genstat fits the covariates, together with the treatments, in each stratum. This should explain some of the variability of the units in the stratum, and so decrease the stratum residual mean square.

In the simplest form of the `COVARIATE` directive, its (unnamed) parameter just contains a list of the variates that are to be used as covariates. Alternatively, you can group some of the variates into pointers. The analysis-of-variance table will then contain a line for each group instead of the individual covariates in that group (see below).

Each treatment combination will have been applied to units whose mean value for each covariate differs from that of other treatment combinations; so even in the absence of any treatment effects, the y-values recorded for the different combinations would not be identical. A further effect of the analysis is to adjust the treatment estimates for the covariates, to correct for this. The adjustment causes some loss of efficiency in the treatment estimation. The remaining efficiency is measured by the *covariance efficiency factor*, shown for each treatment term in the "cov. ef." column of the analysis-of-variance table. The values are in the range zero to one. A value of zero indicates that the treatment contrasts are completely correlated with the covariates: after the covariates have been fitted there is no information left about the treatments.

A value of one indicates that the covariates and the treatment term are orthogonal. Usually the values will be around 0.8 to 0.9. The covariance efficiency factor is analogous to the efficiency factor printed for non-orthogonal treatment terms (see 4.7.1); details of its derivation can be found in Payne & Tobias (1992).

A low value of the covariance efficiency factor for a treatment can be taken as a warning: either the measurements used as covariates have been affected by the treatments, which may occur when the measurements on covariates are taken after instead of before the experiment (see for example Cochran & Cox 1957, page 90); or the random allocation of treatments has been unfortunate in that some treatments are on units with generally low values of the covariates while others are on generally high ones. (Note, that if you are forming a design with covariates and know their values beforehand, you can use procedure `COVDESIGN` to perform a restricted randomization that aims to give covariance efficiency factors close to 1 for the treatment terms; see Part 3 of the *Genstat Reference Manual*.)

For a residual line in the analysis of variance, the value in the "cov. ef." column measures how much the covariates have improved the precision of the experiment. This is calculated by dividing the residual mean square in the unadjusted analysis (which excludes the covariates) by its value in the adjusted analysis.

The covariance efficiency factor is used by Genstat in the calculation of standard errors for tables of effects, as shown by the formula in 4.1.3. So, if you want to calculate the net effect of the analysis of covariance on the precision of the estimated effects of a treatment term, you should multiply the covariance efficiency factor of the term by the value printed in the residual line of the stratum where the term is estimated. Where a term has more than one degree of freedom, the adjustment given by the covariance efficiency factor is an average over all the comparisons between the effects of the term. However this adjustment should not differ by much from those required for any particular comparison unless the randomization has been especially unfortunate. For Fumigant in the example, the calculation is 0.99×2.35. So the e.s.e. of the `Fumigant` effects from the adjusted analysis is less than that from the unadjusted analysis by a factor of $\sqrt{2.3}$.

In the example we have printed tables of means, but no tables of effects. However, since the table of means for `Fumigant` is calculated merely by adding the grand mean to each entry in its table of effects (4.1.3), the same factor also applies to the s.e.d. of the `Fumigant` means. For a table of means classified by several factors, Genstat combines the covariance efficiency factors of the effects from which the means are calculated (4.1.3) into a harmonic mean, weighted according to the numbers of degrees of freedom of each term: for example $4/(1/0.99 + 3/0.92)$ for `Fumigant.Type`.

The adjusted analysis-of-variance table has an extra line in the analysis of each stratum, giving the sum of squares due to the covariates. This is the extra sum of squares that is removed by the covariates after eliminating all that can be ascribed to the treatments. It lets you assess whether there is any evidence that the covariates are required in the model. If there are several covariates Genstat will also print their individual contributions to that sum of squares, giving first the sum of squares that can be explained by the first covariate in the `COVARIATE` list, then the extra sum of squares that can be accounted for by fitting the second covariate, and so on. However, if some of the covariates were grouped together into a pointer in the `COVARIATE` list, their contributions will be pooled into a single line.

The line for each treatment term in the analysis-of-variance table contains the sum of squares eliminating the covariates. It indicates whether there is evidence of any effects of that term, after taking account of the differences in the values of the covariates on the units to which each treatment was applied.

As explained in 4.7.4, when an analysis of variance contains non-orthogonal components, the total sum of squares is given by adding the sum of squares for component 1 ignoring component 2 to that for component 2 eliminating component 1, and so on. Here, however, the sums of

squares are for covariates eliminating the treatment terms, and for each treatment term eliminating the covariates. So you will find that the values in the s.s. column of the analysis-of-variance table do not add up to the total.

Example 4.3.1

```
  30  COVARIATE Initnem
  31  ANOVA [PRINT=aovtable,covariates,means] Finalnem

Analysis of variance (adjusted for covariate)
=============================================

Variate: Finalnem
Covariate: Initnem

Source of variation     d.f.(m.v.)      s.s.       m.s.    v.r. cov.ef.  F pr.

Blocks stratum
Covariate                    1         3.35292    3.35292    9.91          0.088
Residual                     2         0.67657    0.33828    4.29    3.97

Blocks.Plots stratum
Fumigant                     1         0.89557    0.89557   11.36    0.99  0.002
Fumigant.Dose                1         0.00179    0.00179    0.02    0.98  0.881
Fumigant.Type                3         1.41718    0.47239    5.99    0.92  0.002
Fumigant.Dose.Type           3         0.11913    0.03971    0.50    0.99  0.682
Covariate                    1         3.81015    3.81015   48.34          <.001
Residual                  34(1)        2.67969    0.07881            2.35

Total                     46(1)       11.75815


Covariate regressions
=====================

Variate: Finalnem

Covariate               coefficient        s.e.
Blocks stratum
Initnem                        0.48       0.153

Blocks.Plots stratum
Initnem                       0.522      0.0751

Combined estimates
Initnem                       0.512      0.0651


Tables of means (adjusted for covariate)
========================================

Variate: Finalnem
Covariate: Initnem

Grand mean  5.618

 Fumigant Not fumigated    Fumigated
                 5.818        5.518
     rep.           16           32

      Fumigant     Dose    None  Single  Double
 Not fumigated            5.818
     Fumigated                    5.520   5.515

      Fumigant     Type    None      CN      CS      CM      CK
 Not fumigated            5.818
                  rep.      16
     Fumigated                    5.783   5.357   5.692   5.239
                  rep.           8       8       8       8
```

```
     Fumigant      Dose     Type     None       CN       CS       CM       CK
 Not fumigated     None             5.818
                           rep.       16
     Fumigated   Single                    5.703    5.401    5.768    5.209
                           rep.                4        4        4        4
                 Double                    5.864    5.313    5.616    5.269
                           rep.                4        4        4        4


Standard errors of differences of means
---------------------------------------

Table              Fumigant     Fumigant     Fumigant     Fumigant
                                    Dose         Type         Dose
                                                              Type
rep.                unequal          16      unequal      unequal
d.f.                     34          34           34           34
s.e.d.                                        0.1449       0.2024  min.rep
                    0.0862      0.0999       0.1255       0.1600  max-min
                                             0.1025X      0.1012X max.rep

(No comparisons in categories where s.e.d. marked with an X)
(Not adjusted for missing values)
```

The method that Genstat uses for analysis of covariance essentially reproduces the method that you would use if you were doing the calculations by hand. First of all, it analyses each covariate according to the block and treatment models. You can print information from these analyses using the CPRINT option of either ANOVA or ADISPLAY. As ADISPLAY (4.1.3) does not constrain you to list save structures that were all produced by the same ANOVA, CPRINT will produce information about the covariate analyses from every save structure that you list; duplicate information will thus be produced if several of the save structures are for analyses involving the same covariates. The output from CPRINT, particularly the analysis-of-variance table, gives you another way of assessing the relationship between treatments and covariates: a large variance ratio for a treatment term in the analysis of one of the covariates would indicate either that the treatment had affected the covariate or that the randomization had been unfortunate (as discussed in the description of cov. ef. above).

Genstat then analyses each y-variate in turn. First of all it does the usual analysis ignoring the covariates. You can control output from this unadjusted analysis by the UPRINT option of ANOVA and ADISPLAY. (So the whole of the output given for the example could have been produced by a single ANOVA statement.) Then the covariates are fitted by linear regression and the full, adjusted, analysis is calculated. Output from the adjusted analysis is controlled by the PRINT option of ANOVA and ADISPLAY. This option has an extra setting, which is not available for UPRINT and CPRINT: PRINT=covariates prints the regression coefficients of the covariates as estimated in each stratum.

4.3.2 The ACOVARIATES procedure

AFCOVARIATES procedure

Defines covariates from a model formula for ANOVA (R.W. Payne).

Options

COVARIATES = *pointer*	Saves the covariates
COVGROUPS = *pointer*	Saves the pointers defined to contain the covariates formed for each term in TERMS
FACTORIAL = *scalar*	Limit on number of factors in the model terms formed from TERMS; default 3

Parameters

TERMS = *formula*	Model terms from which to define covariates

The COVARIATE directive (4.3.1) covers only the simple situation where you have a list of variates that you want to use as covariates, and does not allow for more complicated situations. For example you might want to fit a different covariate regression coefficient within each block of a randomized-block experiment, or to use the covariate to fit the effects of terms in an unbalanced design.

The AFCOVARIATES procedure therefore provides an alternative to the COVARIATE directive, to allow you to specify a model formulae to define the terms to be fitted as covariates in the analysis. The model formula is specified by the TERMS parameter, using the same conventions as for example in the Genstat regression commands (see 3.3). The dummy variables that are generated to represent the model terms in the formula use the same parameterization as the regression commands (3.3.2).

So, for example, you can see whether you need to fit a different regression coefficient for the variate Initnem within each block in Example 4.3.1 by specifying

```
AFCOVARIATES Initnem + Blocks.Initnem
```

As shown in Example 4.3.2, within the Covariates section of the analysis of variance there will then be a line Initnem representing the overall covariate regression, and another term Blocks.Initnem to assess whether a different regression is needed within each block. This appears only in the Blocks.Plots stratum as it can only be estimated within blocks. In the example, this has an F probability of 0.553 showing that in this case a common regression coefficient is all that is needed.

Example 4.3.2

```
  32  AFCOVARIATES Initnem + Blocks.Initnem
  33  ANOVA [PRINT=aovtable] Finalnem

* MESSAGE: the sums of squares for individual covariates are sequential; each
one is for the covariate concerned eliminating previous covariates (as well as
treatments) and ignoring the later ones.

Analysis of variance (adjusted for covariates)
==============================================

Variate: Finalnem
Covariates: Initnem, Initnem.Blocks 2, Initnem.Blocks 3, Initnem.Blocks 4

Source of variation     d.f.(m.v.)      s.s.       m.s.    v.r. cov.ef.  F pr.

Blocks stratum
Covariates                   3         4.02949    1.34316
  Initnem                    1         3.35292    3.35292

Blocks.Plots stratum
Fumigant                     1         0.91945    0.91945   11.37    0.97  0.002
Fumigant.Dose                1         0.00084    0.00084    0.01    0.93  0.920
Fumigant.Type                3         1.39191    0.46397    5.74    0.91  0.003
Fumigant.Dose.Type           3         0.11952    0.03984    0.49    0.88  0.690
Covariates                   4         3.98241    0.99560   12.31          <.001
  Initnem                    1         3.81015    3.81015   47.11          <.001
  Initnem.Blocks             3         0.17226    0.05742    0.71          0.553
Residual                  31(1)        2.50744    0.08089            2.29

Total                     46(1)       11.75815
```

The COVARIATES option allows you to supply a pointer to store the covariates that are calculated. Otherwise they will be unnamed, and thus usable only in the subsequent ANOVA

analyses. The covariates are grouped into a pointer for each model term specified by TERMS. The COVGROUPS option allows you to supply a pointer to store these pointers. Otherwise they too will be unnamed, and thus usable only in the ANOVA analyses. Each covariate is each defined with an extra text, using the EXTRA parameter of the VARIATE directive (1:2.3.1), to indicate the parameter that it represents. Also the IPRINT option of VARIATE is set to extra, so that this extra text will be used in output instead of the identifier of the covariate itself. Similarly, the COVGROUPS pointers are given extra texts indicating the model term that each one represents.

The FACTORIAL option sets a limit on the number of factors or variates in each of the terms formed from the TERMS formula. Any term containing more than that limit is deleted.

4.4 Missing values

Values from some of the units of an experiment may occasionally fail to be recorded. A laboratory animal may become ill or die during the experiment for reasons unconnected with the treatments. A human subject may withdraw from a clinical trial before it is complete. A plot in a field experiment may become flooded and fail to produce any plants. A value may need to be regarded as missing if a mistake has been made in its recording, or in the way in which the unit was managed during the experiment.

To obtain the exact analysis in such circumstances these units should be excluded, but that would lose the properties such as balance for which the experiment was designed. Consequently techniques have been devised by which missing values are entered for these units, and then estimated during the analysis. The estimates can be printed using the missingvalues setting of the PRINT, CPRINT or UPRINT options of ANOVA or ADISPLAY. Example 4.4 uses the ADISPLAY directive to print the missing value estimated in the analysis of covariance in Example 4.3.2.

Example 4.4

```
  34  ADISPLAY [PRINT=missingvalues]

Missing values (adjusted for covariates)
========================================

Variate: Finalnem
Covariates: Initnem, Initnem.Blocks 2, Initnem.Blocks 3, Initnem.Blocks 4

 Unit  estimate
   43     5.632

Max. no. iterations 3
```

You can have missing values in the y-variates or the covariates, but not in the block or treatment factors: that is, you should at least know where each unitsmissing unit belongs according to the factors of the block model, and what treatments it was scheduled to receive. Genstat regards a unit as missing for all the y-variates listed in an ANOVA statement if it is missing for any one of them, or if it is missing for a covariate. This is because the analysis of covariance requires a missing value in either the y-variate or a covariate to be set missing throughout (Wilkinson 1957); forming the complete list over all the y-variates avoids having to re-analyse the covariates for each y-variate. If you have units where some but not all of the y-variates have missing values, you may prefer to analyse each y-variate separately: for example

```
FOR Y=Weight,Age,Height
  ANOVA [DESIGN=Dsave] Y
ENDFOR
```

instead of

```
ANOVA Weight,Age,Height
```

Use of the DESIGN option (4.1.2) avoids Genstat having to redetermine the structure of the design for each analysis.

Genstat uses the method of Healy & Westmacott (1956). This estimates the missing values by an iterative approach in which they are initially set to the grand mean, then the analysis is repeated with the estimate for each missing unit adjusted each time to set its residual to zero. Genstat also employs the modification discussed by Preece (1971) which over-adjusts each residual to accelerate convergence, but this is discontinued if divergence results instead. Missing cells can occur in higher strata, for example if all the sub-plots in a whole-plot are missing. These missing effects are estimated by a similar iteration of the analysis within the stratum. Likewise missing treatment effects are estimated by minimizing the sum of squares of the treatment term concerned. There is a limit on the number of iterations; by default it is 40, but this can be changed by the MAXCYCLE option of ANOVA. Genstat decides that the process has converged when the residual sum of squares from the previous iteration exceeds the current residual sum of squares by less than 10^{-5} times the current residual sum of squares. This value of 10^{-5} can be changed using the third value of the variate in the TOLERANCES option of ANOVA. Genstat prints the maximum number of iterations required in any of the strata of the design, along with the estimates of the missing values. Convergence is usually fairly rapid: for the example above, only three iterations were required.

In the analysis of variance, as shown in the example in 4.3, the numbers of degrees of freedom are decreased to take account of the missing units and effects; the number subtracted is shown in brackets. The analysis of variance is only approximate. The residual sums of squares are correct (to within the tolerance of convergence) but the treatment sums of squares will be larger than their correct value. (As a result, the sums of squares in the analysis-of-variance table will no longer sum to the total.) If there are few missing values, this increase is unlikely to be large. The estimated effects and means are correct but the calculation of the standard errors does not take account of the missing units. So some standard errors will be too small. For further details, see for example Cochran & Cox (1957, pages 80-82).

If the model has only one error term, you can obtain the exact analysis using regression (Chapter 3). Alternatively you could use the method of Bartlett (1937), in which a dummy covariate is specified for each missing value with minus one in the missing unit and zero elsewhere. The missing units in the y-variates should be set to zero; the regression coefficients of the covariates then estimate the missing values.

4.5 Contrasts between treatments

Sometimes there may be comparisons between the levels of a treatment factor that you particularly wish to assess. With the three sources of protein in 4.1, you might wish to see whether the animal sources (beef and pork) were uniformly better than the cereal source, or you might suspect that the type of meat made little difference and so wish to compare beef with pork. These comparisons are examples of *contrasts*. They are specified by defining a coefficient for each level of the factor. The estimated value of the contrast is then obtained by taking the sum of the coefficients each multiplied by the appropriate effect. For example the comparison contrasts between the sources of protein are defined by coefficients:

```
                                Source: beef    cereal     pork
Contrast: animal versus cereal           0.5      -1.0      0.5
          beef versus pork               1.0       0.0     -1.0
```

To compare beef with pork you subtract one effect from the other; while for animal versus cereal sources, you subtract the effect of cereal from the mean of the effects of the animal sources. As shown by this example, to represent a comparison between the levels of the factor, the sum of the coefficients must be zero. These particular contrasts are also orthogonal: they represent

independent comparisons between the effects. This is shown by the fact that the sum of the pairwise products of the coefficients, weighted according to the replication of the levels of the factor (here 20), is zero: 0.5×1.0×20 + (−1.0)×0.0×20 + 0.5×(−1.0)×20. However, comparison contrasts need not always be orthogonal (see Example 4.5c).

With factors whose levels represent the application of different amounts of some substance like a fertilizer or a drug, you may wish to model the relationship between the effect and the amount. For example, with the nitrogen fertilizer in Section 4.2, you might wish to see if the yield of oats increases linearly with the amount of fertilizer; you might also include a quadratic term to check for curvature in the response. You can assess these by fitting polynomial contrasts. Genstat also allows you to define other regression contrasts.

To specify contrasts, you put a function of the factor of interest into the treatment formula, instead of the factor itself. Comparisons between factor levels are specified by the `COMPARISON` function. This has three arguments: the first specifies the factor amongst whose effects the comparisons are being made; the second specifies the number of comparisons that are to be fitted; and the third provides a matrix with a column for each level of the factor, and a row for each comparison specifying the coefficients that define that comparison. In line 39 of Example 4.5a, which continues the analyses started in Example 4.1, the factor is `Source` and the matrix is `Compare` (see lines 37 and 38).

Example 4.5a

```
  38  MATRIX [ROWS=!T('animal vs cereal','beef vs pork'); COLUMNS=3; \
  39     VALUES=0.5,-1,0.5,1,0,-1] Compare
  40  TREATMENTSTRUCTURE COMPARISON(Source; 2; Compare) * Amount
  41  ANOVA [PRINT=aov,contrasts; FPROBABILITY=yes] Gain

Analysis of variance
====================

Variate: Gain

Source of variation     d.f.       s.s.       m.s.    v.r.  F pr.
Source                     2      266.5      133.3    0.62  0.541
  animal vs cereal         1      264.0      264.0    1.23  0.272
  beef vs pork             1        2.5        2.5    0.01  0.914
Amount                     1     3168.3     3168.3   14.77  <.001
Source.Amount              2     1178.1      589.1    2.75  0.073
  animal vs cereal.Amount  1     1178.1     1178.1    5.49  0.023
  beef vs pork.Amount      1        0.0        0.0    0.00  1.000
Residual                  54    11586.0      214.6
Total                     59    16198.9


Tables of contrasts
===================

Variate: Gain

Source contrasts
----------------

animal vs cereal      4.5,  s.e. 4.01,  ss.div. 13.3

beef vs pork          0.5,  s.e. 4.63,  ss.div. 10.0

Source.Amount contrasts
-----------------------

animal vs cereal.Amount,  e.s.e. 5.67,  ss.div. 6.67

   Amount     high      low
               9.4     -9.4

beef vs pork.Amount,  e.s.e. 6.55,  ss.div. 5.00
```

```
   Amount      high       low
                0.0       0.0
```

In the analysis-of-variance table, the line for the main effect of `Source` is now accompanied by two additional lines (indented to show that they relate to `Source`) giving the degrees of freedom, sums of squares etc. for the two comparisons. Notice that they are labelled by the names given to the rows of the contrast matrix `Compare` in line 37. If you do not label the rows, the contrasts are labelled `Comp1`, `Comp2`, and so on. The interaction between `Source` and `Amount` is also accompanied by extra lines showing how the comparisons are affected by the amount of protein. For example, the line "`animal vs cereal.Amount`", allows you to examine whether there is any evidence that the difference between animal and cereal sources of protein varies according to the amount of protein fed to the rats or, equivalently, whether there is evidence that the response to the amount of protein varies according to whether the protein is from cereal or animals. Here it appears that this first comparison does have an interaction with amount, but that the second comparison (beef vs. pork) does not.

The estimates of the contrasts can be printed by including `contrasts` in the settings of the `PRINT` option of `ANOVA` or `ADISPLAY`. So, you can check the values in Example 4.5a by referring back to Examples 4.1.3b or 4.1.3d: for example the estimate of the beef vs. pork contrast is indeed the difference between the effects of beef and pork (1.7 – 1.2).

Polynomial contrasts are assessed by fitting orthogonal polynomials. The quadratic contrast then represents the effect of adding a quadratic term into a linear polynomial, the cubic represents the effect of adding a cubic term into a quadratic polynomial, and so on (see for example: John 1971, page 50; John & Quenouille 1977, pages 33-36). The coefficients of the orthogonal polynomials to examine the linear and quadratic effects of the `Nitrogen` factor in Section 4.2 are

```
                        Nitrogen:  0.0   0.2   0.4   0.6
     Contrast:  linear            -0.3  -0.1   0.1   0.3
                quadratic          0.4  -0.4  -0.4   0.4
```

Polynomial contrasts are specified by the `POL` function. This again has three arguments: the first specifies the factor, the second is a number or a scalar giving the order of polynomial to be fitted (1 for linear, 2 for quadratic, 3 for cubic and 4 for quartic), and the third is a variate specifying numerical values for each level of the factor. Genstat calculates the orthogonal polynomials for you. In the `Nitrogen` example, the levels are equally spaced and in ascending order of magnitude, but this need not be so. You can omit the third argument if the levels already declared with the factor are suitable. For `Nitrogen`, the declaration (line 9 in the output shown in 4.2.1) specified only labels, and so the levels are the defaults 1 to 4. The variate `Nitlev` is defined to supply the correct values (line 33).

Example 4.5b

```
  33  VARIATE [VALUES=0,0.2,0.4,0.6] Nitlev
  34  TREATMENTSTRUCTURE POL(Nitrogen; 2; Nitlev) * Variety
  35  ANOVA [PRINT=aovtable,contrasts] Yield

Analysis of variance
====================

Variate: Yield of oats

Source of variation     d.f.       s.s.       m.s.    v.r.

Blocks stratum             5    15875.3     3175.1    5.28

Blocks.Wplots stratum
Variety                    2     1786.4      893.2    1.49
Residual                  10     6013.3      601.3    3.40
```

```
Blocks.Wplots.Subplots stratum
Nitrogen                    3    20020.5     6673.5   37.69
  Lin                       1    19536.4    19536.4  110.32
  Quad                      1      480.5      480.5    2.71
  Deviations                1        3.6        3.6    0.02
Nitrogen.Variety            6      321.8       53.6    0.30
  Lin.Variety               2      168.3       84.2    0.48
  Quad.Variety              2       11.1        5.5    0.03
  Deviations                2      142.3       71.2    0.40
Residual                   45     7968.8      177.1

Total                      71    51985.9


Tables of contrasts
===================

Variate: Yield of oats

Blocks.Wplots.Subplots stratum
------------------------------

Nitrogen contrasts
------------------

Lin      73.7,  s.e. 7.01,  ss.div. 3.60

Quad     -65.,  s.e. 39.2,  ss.div. 0.115

Deviations,  e.s.e. 3.14,  ss.div. 18.0

 Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
               0.1     -0.3      0.3     -0.1

Nitrogen.Variety contrasts
--------------------------

Lin.Variety,  e.s.e. 12.1,  ss.div. 1.20

  Variety     Victory Golden rain  Marvellous
                   7.          2.         -9.

Quad.Variety,  e.s.e. 67.9,  ss.div. 0.0384

  Variety     Victory Golden rain  Marvellous
                  -1.         12.        -11.

Deviations,  e.s.e. 5.43,  ss.div. 6.00

 Nitrogen  Variety     Victory Golden rain  Marvellous
    0 cwt                  0.7         0.1        -0.8
  0.2 cwt                 -2.2        -0.3         2.4
  0.4 cwt                  2.2         0.2        -2.4
  0.6 cwt                 -0.7        -0.1         0.8
```

In the analysis of variance, the sum of squares for `Nitrogen` is partitioned into the amount that can be explained by a linear relationship of the yields with nitrogen (the line marked `Lin`), the extra amount that can be explained if the relationship is quadratic (the line `Quad`), and the amount represented by deviations from a quadratic polynomial. A cubic term would be labelled as `Cub`, and a quartic as `Quart`. You are not allowed to fit more than fourth-order polynomials.

The interaction of nitrogen and variety is also partitioned: `Lin.Variety` lets you assess the effect of fitting three different linear relationships, one for each variety, instead of a single overall linear contrast; `Quad.Variety` represents three different quadratic contrasts; and `Deviations` represents deviations from these three quadratic polynomials.

The estimated values of the contrasts are again printed in the section headed "Tables of contrasts". The table of estimated contrasts for `Quad.Variety`, for example, gives the differences between the overall contrast of −65 for `Quad` and the contrasts fitted for the three

varieties separately. So the estimated contrast for Golden rain is −65 + 12 = −53.

The "ss. div" value that accompanies the estimated contrasts is analogous to the replication in a table of effects: it is the divisor used when calculating the estimated values of the contrasts. This is useful mainly where there is a range of e.s.e.'s for a table of contrasts: the contrasts with the smallest values of the ss. div. are those with the largest e.s.e., and vice versa. The ss. div. of each estimated contrast is the sum of squares of the values of the orthogonal polynomial (or other x-variable) used to calculate the contrast, weighted according to the replication (or weighted replication in a weighted analysis of variance). The formula for the e.s.e. is similar to that for tables of effects (4.1.3):

e.s.e. = $\sqrt{(\sigma^2 / (\text{ss. div.} * \text{efficiency factor} * \text{covariance efficiency factor}))}$

The variance σ^2 is estimated from the residual mean square of the stratum (4.2) where the contrasts are estimated. The efficiency factor (4.7.1) has the value one for terms that are orthogonal, like those in this design. The covariance efficiency factor (4.3.1) equals one when there are no covariates.

The third contrast function, REG, allows you to specify regression contrasts other than polynomials. The first argument again specifies the factor, and the second is a number or scalar giving the number of contrasts to be fitted, which can be from one up to the number of degrees of freedom of the factor. The third argument is a matrix whose rows supply the x-variates for the regression contrasts. The matrix has a column for each level of the factor and a row for each contrast specifying the coefficients of the corresponding x-variate, similar to that for the COMPARISONS function. However, Genstat orthogonalizes the x-variates for REG; so the sum of squares, and the estimate, for the second contrast represent the improvement from fitting the second contrast after the first has already been fitted, and so on. If you use a text to label the rows of the matrix, Genstat will use it to annotate the output. Otherwise the contrasts are labelled Reg1 to Reg7.

Where a term has two or more factors partitioned into contrasts, Genstat will fit interactions between the contrasts. For example Lin.Lin looks at the linear change in the linear component of each factor with the other. With two REG functions, terms like Reg1.Reg1 or Reg2.Reg1 will appear whose interpretation will depend on exactly what comparisons you have defined. If the partitioning of a factor has a component for deviations, there will also be terms like Dev.Lin, which represents the interaction between the deviations component of the first factor and the linear part of the second factor. You can suppress the fitting of these interactions by using the function POLND instead of POL, or REGND instead of REG. For example, putting POLND(A; 1) instead of POL(A; 1) ensures that no interactions will be fitted between other contrasts and the Dev component of A.

The CONTRASTS option in the ANOVA directive (4.1.2) places a limit on the order of contrast to be fitted. For a term involving a single factor, the orders of successive terms run from one upwards, with the deviations term (if any) numbered highest. So for Nitrogen in the example above, the orders are Lin 1, Quad 2 and Deviations 3; while for Source they are "animal vs cereal" 1, "beef vs pork" 2. In interactions between contrasts, the order is the sum of the orders of the component parts, so Lin.Lin has order 2, Quad.Lin has order 3, Reg1.Quad has order 3, Reg1.Reg3 has order 4, and so on. Where the component is a factor, it contributes one to the sum, so Lin.Variety has order 2. The default value for CONTRASTS is 4. Option PCONTRASTS sets a limit on the order of the contrasts that are printed by either ANOVA or ADISPLAY (4.1.3); its default value is 9.

In Example 4.5c, we illustrate interactions between comparison and polynomial contrasts. This time we want to compare each of the varieties with the first variety, Victory. So we have the matrix

```
                Variety:  Victory  Golden rain Marvellous
Comparison:
Golden rain versus Victory     -1            1          0
Marvellous versus Victory      -1            0          1
```

These represent comparisons amongst the variety effects because the coefficients of each comparison sum to zero (and ANOVA will give a fatal diagnostic if this is not true). However, they are not orthogonal: the sum of the pairwise products of their coefficients is one (and not zero as would be required for orthogonality). If we were to fit these contrasts using REG function they would be orthogonalized to become

```
                    Variety:  Victory  Golden rain Marvellous
     Comparison:
     Golden rain versus Victory    -1            1          0
     Marvellous versus Victory   -0.5         -0.5          1
```

which would not give what we want! Notice that we need to set option CONTRASTS=5 in the ANOVA statement in line 40, as "Marvellous versus Victory.Dev" has order 2+3 = 5. The variety by nitrogen interaction is now accompanied by interactions between the contrasts: "Golden rain versus Victory.Lin" assesses how the linear effects of nitrogen differ between Golden rain and Victory, "Golden rain versus Victory.Quad" assesses how the quadratic effects of nitrogen differ between Golden rain and Victory, "Golden rain versus Victory.Dev" assesses how the deviations from the two quadratic polynomials of nitrogen differ between Golden rain and Victory, and so on.

Example 4.5c

```
  36  TEXT [VALUES='Golden rain versus Victory','Marvellous versus Victory']\
  37       Compname
  38  MATRIX [ROWS=Compname; COLUMNS=3; VALUES=-1,1,0, -1,0,1] Victcomp
  39  TREATMENTS COMPARISON(Variety;2;Victcomp) * POL(Nitrogen; 2; Nitlev)
  40  ANOVA [PRINT=aovtable,contrasts; CONTRASTS=5] Yield

Analysis of variance
====================

Variate: Yield of oats

Source of variation     d.f.       s.s.       m.s.    v.r.

Blocks stratum             5    15875.3     3175.1    5.28

Blocks.Wplots stratum
Variety                    2     1786.4      893.2    1.49
  Golden rain versus Victory
                           1      567.2      567.2    0.94
  Marvellous versus Victory
                           1     1776.3     1776.3    2.95
Residual                  10     6013.3      601.3    3.40

Blocks.Wplots.Subplots stratum
Nitrogen                   3    20020.5     6673.5   37.69
  Lin                      1    19536.4    19536.4  110.32
  Quad                     1      480.5      480.5    2.71
  Deviations               1        3.6        3.6    0.02
Variety.Nitrogen           6      321.8       53.6    0.30
  Golden rain versus Victory.Lin
                           1       19.8       19.8    0.11
  Marvellous versus Victory.Lin
                           1      163.3      163.3    0.92
  Golden rain versus Victory.Quad
                           1        3.5        3.5    0.02
  Marvellous versus Victory.Quad
                           1        2.1        2.1    0.01
Residual                  45     7968.8      177.1

Total                     71    51985.9


Tables of contrasts
===================
```

```
Variate: Yield of oats

Blocks.Wplots stratum
---------------------

Variety contrasts
-----------------

Golden rain versus Victory      6.9,  s.e. 7.08,  ss.div. 12.0

Marvellous versus Victory      12.2,  s.e. 7.08,  ss.div. 12.0

Blocks.Wplots.Subplots stratum
------------------------------

Nitrogen contrasts
------------------

Lin      73.7,  s.e. 7.01,  ss.div. 3.60

Quad     -65.,  s.e. 39.2,  ss.div. 0.115

Deviations,  e.s.e. 3.14,  ss.div. 18.0

 Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
               0.1     -0.3      0.3     -0.1

Variety.Nitrogen contrasts
--------------------------

Golden rain versus Victory.Lin       -6.,  s.e. 17.2,  ss.div. 0.600

Marvellous versus Victory.Lin       -16.,  s.e. 17.2,  ss.div. 0.600

Golden rain versus Victory.Quad      14.,  s.e. 96.0,  ss.div. 0.0192

Marvellous versus Victory.Quad      -10.,  s.e. 96.0,  ss.div. 0.0192
```

If your design has few or no degrees of freedom for the residual, you may wish to regard the deviations from some of the fitted contrasts as error components, and assign them to the residual of the stratum where they occur. You can do this by the DEVIATIONS option of ANOVA (4.1.2); its value sets a limit on the number of factors in the terms whose deviations are to be retained in the model. For example, by putting DEVIATIONS=1, the deviations from the contrasts fitted to all terms except main effects will be assigned to error. The option PDEVIATIONS in ANOVA or ADISPLAY (4.1.3) similarly controls the printing of deviations: putting PDEVIATIONS=0, for example, would ensure that no deviations are printed. When deviations have been assigned to error, they will not be included in the calculation of tables of means (4.1.3), which will then be labelled "smoothed". However the associated standard errors of the means are not adjusted for the smoothing.

There are limitations on the models and designs for which Genstat can fit contrasts. In a factorial model, each interaction that is partitioned into contrasts must have equal or proportional replication (or proportional weighted replication in a weighted analysis of variance). Otherwise Genstat gives an error. Here is an example of proportional replication for two factors A and B, giving the numbers of replications for each combination of their levels.

```
                B:    1   2   3        Total over B
                A:
                1     4   8  12             24
                2     2   4   6             12
     Total over A:    6  12  18             36
```

The fraction of the replication in each cell is the product of the fractions in the marginal total cells: for example the cell for level 1 of A and level 3 of B has 12/36 (= 1/3) of the total replication; the product of the marginal totals for these levels is also 1/3, being 24/36 × 18/36.

An exception to this rule occurs in nested models like the factorial with added control which were discussed in 4.3.1. The table below shows what the replication of the factors Fumigant, Dose and Type would be if, for illustration, there were also a triple level of dose.

```
          Fumigant:  not fumigated          fumigated
             Type:       none  CN  CS  CM  CK  none  CN  CS  CM  CK
      Dose:
      none              16    -   -   -   -     -   -   -   -   -
    single               -    -   -   -   -     -   4   4   4   4
    double               -    -   -   -   -     -   4   4   4   4
    triple               -    -   -   -   -     -   4   4   4   4
```

The treatment model has

```
Fumigant/(Dose*Type)
= Fumigant + Fumigant.Dose + Fumigant.Type
+ Fumigant.Dose.Type
```

None of the higher-order terms (such as Fumigant.Dose) has either equal or proportional replication. However, within the 'fumigated' level of Fumigant, there is equal replication. So Genstat can fit any contrast of the nested factors (Type and Dose) provided the level 'none' is excluded. For example, you could estimate linear and quadratic contrasts of Dose using only the non-zero doses by using the REG function:

```
MATRIX [ROWS=2; COLUMNS=4; VALUES= 0, -1,  0,   1 \
                                   0,  1, -2,   1 ] Quadcon
TREATMENTSTRUCTURE Fumigant / (REG(Dose;2;Quadcon) * Type)
```

But the rows of Quadcon must be specified in orthogonal form. Otherwise the automatic orthogonalization, using the overall replication of Dose, would produce contrasts involving 'none'.

A further limitation is that contrasts cannot be fitted to terms that involve pseudo-factors (4.7.3). In such situations, the specification of the contrasts is ignored by Genstat.

In nested models, no coherent meaning can be given to contrasts between levels of one of the nested factors if the factor within which it is nested is also partitioned into contrasts. So, for example, the specification

```
POL(A; 1) / POL(B; 2)
```

would generate an error.

The contrasts described above, that can be fitted directly by ANOVA, are all linear in their coefficients. Procedure NLCONTRASTS in the Genstat Procedure Library extends this to enable nonlinear contrasts to be fitted to the effects of a quantitative factor and its interaction with another factor. Full details can be found in the Part 3 of the *Genstat Reference Manual*.

4.5.1 The APOLYNOMIAL procedure

APOLYNOMIAL procedure

Forms equations for a polynomial contrast fitted by ANOVA (R.W. Payne).

Options

PRINT = *string token*	Whether to print the equation of the polynomial (equation); default equa
SAVE = *ANOVA save structure*	Save structure (from ANOVA) to provide details of the analysis from which the equations are to be formed; default uses the save structure from the most recent ANOVA

Parameters

TERMS = *formula*	Model terms whose polynomial equations are required
COEFFICIENTS = *pointers*	Saves the coefficients of each polynomial

The estimates of the polynomial contrasts do not (directly) give you the coefficients of the polynomial that has been fitted. The polynomial coefficients can, however, be obtained using procedure APOLYNOMIAL.

The TERMS parameter specifies the treatment terms whose equations are required. Each term must contain no more than one factor with a polynomial function (POL or POLND), and no factors with regression or comparison functions (REG, REGND or COMPARISON); otherwise it is ignored. If TERMS is not set, APOLYNOMIAL takes the full treatment model.

APOLYNOMIAL usually prints the equation, but you can set option PRINT=* to suppress this. The COEFFICIENTS parameter can supply a pointer to save the coefficients of the equations. The pointer will contain a pointer for each term. These are given suffixes 0 upwards, corresponding to the powers of the factor in each polynomial.

By default, the equation is formed for the contrasts estimated in the most recent analysis performed by ANOVA, but the SAVE option can be used to supply the save structure from an earlier analysis to use instead.

APOLYNOMIAL is illustrated in Example 4.5.1, which refits the polynomial contrasts in Example 4.5b, and then calculates their equations.

Example 4.5.1

```
  40  TREATMENTSTRUCTURE POL(Nitrogen; 2; Nitlev) * Variety
  41  ANOVA [PRINT=*] Yield
  42  APOLYNOMIAL Nitrogen + Variety.Nitrogen

Equation of the polynomial for Nitrogen
---------------------------------------

79.29 + 112.4 * Nitrogen - 64.58 * Nitrogen**2


Equations of the polynomials for Nitrogen.Variety
-------------------------------------------------

      Variety
      Victory 70.67 + 120.5 * Nitrogen - 65.62 * Nitrogen**2
  Golden rain 79.82 + 106.6 * Nitrogen - 52.08 * Nitrogen**2
   Marvellous 87.38 + 110.2 * Nitrogen - 76.04 * Nitrogen**2
```

4.5.2 The ADPOLYNOMIAL procedure

ADPOLYNOMIAL procedure

Plots single-factor polynomial contrasts fitted by ANOVA (R.W. Payne).

Option

SAVE = *ANOVA save structure*	Save structure (from ANOVA) to provide details of the analysis from which the polynomials are to be plotted; default uses the save structure from the most recent ANOVA

Parameters

XFACTOR = *factors*	Factor over which the polynomial contrasts have been formed

GROUPS = *factors* or *pointers*	Factor(s) for which different polynomial coefficients should be plotted in the same graph
TRELLISGROUPS = *factors* or *pointers*	Factor or factors for which different polynomial coefficients should be plotted in a trellis plot
TITLE = *texts*	Title for the graph; default defines a title automatically
YTITLE = *texts*	Title for the y-axis; default ' '
XTITLE = *texts*	Title for the x-axis; default is to use the identifier of the XFACTOR
PENS = *variates*	Defines the pen to use to plot the points and/or line for each group defined by the GROUPS factors

ADPOLYNOMIAL plots polynomials fitted in analyses by the ANOVA directive. It also plots the corresponding means so that you can see how well the polynomials fit. By default, the polynomials are plotted from the most recent analysis performed by ANOVA, but the SAVE option can be used to supply the save structure from an earlier analysis to use instead.

The XFACTOR parameter specifies the factor over whose effects the polynomial contrasts have been fitted. If the analysis contains interactions between the XFACTOR and other factors, you can plot the polynomials for all the combinations of levels of these other factors by setting the GROUPS and TRELLISGROUPS parameters. If only GROUPS is specified, all the polynomials are plotted in a single graph. Alternatively, you can set the TRELLISGROUPS parameter to one or more of the factors to produce a trellis plot; there is then a graph for each of the combination of levels of the trellis factors (and each of these graphs plots the polynomials for every level of the group factors, at the relevant levels of the trellis factors). You should set GROUPS or TRELLISGROUPS to the factor if there is only one factor, or to a pointer containing all the factors if there are several.

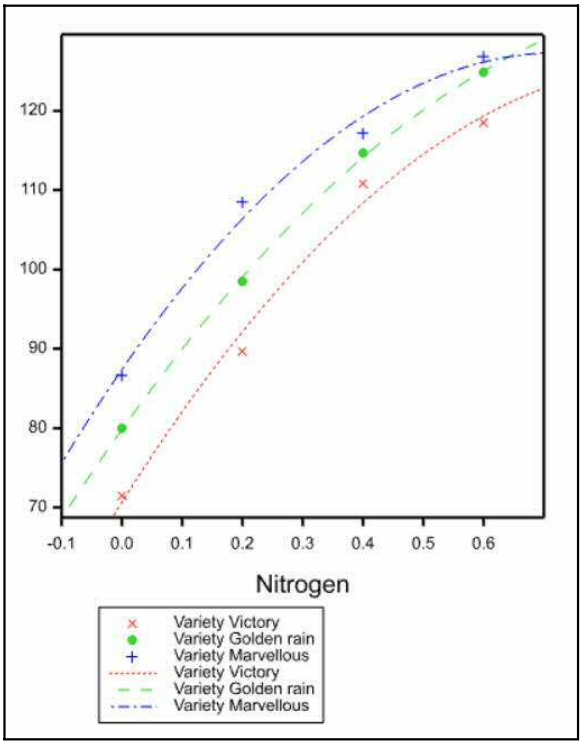


Figure 4.5.2

The TITLE, YTITLE and XTITLE parameters can supply titles for the graph, the y-axis and the x-axis, respectively. The symbols, colours and line styles that are used in a high-resolution plot are usually set up by ADPOLYNOMIAL automatically. If you want to control these yourself, you should use the PEN directive to define a pen with your preferred symbol, colour and line style, for each of the groups defined by combinations of the GROUPS factors. The pen numbers should then be supplied to ADPOLYNOMIAL, in a variate with a value for each group, using the PENS parameter.

Figure 4.5.2 shows a plot the polynomials fitted in Examples 4.5b and 4.5.1, produced by the command

```
ADPOLYNOMIAL Nitrogen; GROUPS=Variety
```

4.6 Saving information from an analysis of variance

Most of the quantities calculated during an analysis of variance can be saved in data structures within Genstat. This allows you to write analyses where the analysis of variance itself is only a component part. One example is the multivariate analysis of variance (6.6.1). Alternatively, you may wish to save components of the output (such as tables of means) for plotting, or for printing in the form required for a publication.

You can save variates containing residuals for the final error term of the model, using the RESIDUALS parameter of ANOVA (4.1.2). The FITTEDVALUES parameter similarly allows you to save the fitted values. Other components of the output can be saved using AKEEP (4.6.1). ASTATUS (4.6.2) can save details of the models defined for the analysis, and ASPREADSHEET (4.6.3) allows you to save the complete output from an analysis into a spreadsheet.

4.6.1 The AKEEP directive

AKEEP directive

Copies information from an ANOVA analysis into Genstat data structures.

Options

FACTORIAL = *scalar*	Limit on number of factors in a model term; default 3
STRATUM = *formula*	Model term of the lowest stratum to be searched for effects; default * implies the lowest stratum
SUPPRESSHIGHER = *string token*	Whether to suppress the searching of higher strata if a term is not found in STRATUM (yes, no); default no
TWOLEVEL = *string token*	Representation of effects in 2^n experiments (responses, Yates, effects); default resp
RESIDUALS = *variate*	To save residuals from the final stratum (as in the RESIDUALS parameter of ANOVA)
FITTEDVALUES = *variate*	To save fitted values (data values or missing value estimates, minus the residuals from the final stratum – as in the FITTEDVALUES parameter of ANOVA)
CBRESIDUALS = *variate*	To save the sum of the residuals from all the strata
CBCREGRESSION = *variate*	To save the estimates of the covariate regression coefficients, combining information from all the strata
CBCVCOVARIANCE = *symmetric matrix*	Saves the variance-covariance matrix of the combined estimates of the covariate regression coefficients
TREATMENTSTRUCTURE = *formula structure*	To save the treatment formula used for the analysis
BLOCKSTRUCTURE = *formula structure*	To save the block formula used for the analysis
AFACTORIAL = *scalar*	To save the setting of the FACTORIAL option used in the ANOVA command that performed the analysis
WEIGHTS = *variate*	To save the weights used in the analysis
YVARIATE = *dummy*	Dummy to be set to the y-variate of the analysis
LSDLEVEL = *scalar*	Significance level (%) to use in the calculation of least significant differences; default 5
AOVTABLE = *pointer*	To save the analysis-of-variance table as a pointer with a variate or text for each column (source, d.f., s.s., m.s. etc)
EQFACTORS = *factors*	Factors whose levels are to be assumed to be equal

	within the comparisons between means calculated for SEMEANS
RMETHOD = *string token*	Type of residuals to form if parameter RESIDUALS is set (simple, standardized); default simp
EXIT = *scalar*	Saves an exit code indicating the properties of the design
SAVE = *identifier*	Defines the Save structure (from ANOVA) that provides details of the analysis; default * gives that from the most recent ANOVA

Parameters

TERMS = *formula*	Model terms for which information is required
MEANS = *tables*	Table to store means for each term (available for treatment terms only)
SEMEANS = *tables*	Table of effective standard errors for the means, usable for calculating standard errors for differences between means in the table, at equal levels of the factors specified by the EQFACTORS option
SEDMEANS = *symmetric matrices*	Standard errors for comparisons between every pair of entries in the table of means
VCMEANS = *symmetric matrices*	Variances and covariances of means
EFFECTS = *tables* or *scalars*	Table or scalar (for terms with 1 d.f. when TWOLEVEL=responses or Yates) to store effects (for treatment terms only)
PARTIALEFFECTS = *tables*	Table or scalar (for terms with 1 d.f. when TWOLEVEL=responses or Yates) to store partial effects (for treatment terms only)
REPLICATIONS = *tables* or *scalars*	Table to store replications or scalar if they are all equal
RESIDUALS = *tables*	Table to store residuals (for block terms only)
DF = *scalars*	Number of degrees of freedom for each term
LSDMEANS = *symmetric matrices*	Least significant differences of means
DFMEANS = *symmetric matrices*	Degrees of freedom for comparisons between every pair of entries in the table of means
SS = *scalars*	Sum of squares for each term
EFFICIENCY = *scalars*	Efficiency factor for each term
VARIANCE = *scalars*	Unit variance for the effects of each term
RTERM = *formula structures*	Residual terms: for a treatment term this saves the lowest stratum where the term is estimated (down to the stratum specified by the STRATUM option); for a block term it saves all the strata to which it would be appropriate to compare the term
CEFFICIENCY = *scalars*	Covariance efficiency factor for each term
CREGRESSION = *variates*	Estimated regression coefficients for the covariates in the specified stratum
CVCOVARIANCE = *symmetric matrix*	Variance-covariance matrix of the covariate regression
CSSP = *symmetric matrices*	Covariate sums of squares and products in the specified stratum
CONTRASTS = *pointers*	Estimates for the fitted contrasts of each treatment term, stored in a pointer to scalars or tables; units of the pointer are labelled by the contrast name (as used in the

	analysis-of-variance table)
XCONTRASTS = *pointers*	X-variates used to fit contrasts, as orthogonalized by ANOVA, stored in a pointer to tables; units of the pointer are labelled as for CONTRASTS
SECONTRASTS = *pointers*	Standard errors for estimated contrasts, stored in a pointer to scalars or tables; units of the pointer are labelled as for CONTRASTS
DFCONTRASTS = *pointers*	Degrees of freedom for estimated contrasts, stored in a pointer to scalars; units of the pointer are labelled as for CONTRASTS
CBMEANS = *tables*	Table to store estimates of the means, combining information from all the strata (for treatment terms only)
SECBMEANS = *tables*	Table of standard errors for the combined means, usable for calculating standard errors for differences between means in the table, at equal levels of the factors specified by the EQFACTORS option
SEDCBMEANS = *symmetric matrices*	Standard errors for comparisons between every pair of entries in the table of combined means
VCCBMEANS = *symmetric matrices*	Variances and covariances of combined means
LSDCBMEANS = *symmetric matrices*	Least significant differences of combined means
DFCBMEANS = *symmetric matrices*	Effective degrees of freedom for comparisons between every pair of entries in the table of combined means
CBEFFECTS = *tables* or *scalars*	Table or scalar (for terms with 1 d.f. when TWOLEVEL=responses or Yates) to store estimates of the effects, combining information from all the strata (for treatment terms only)
CBVARIANCE = *scalars*	Unit variance for the combined estimates of the effects of each term
DFCEFFECTS = *scalars*	Effective degrees of freedom for the combined estimates of the effects of each term
CBCEFFICIENCY = *scalars*	Covariance efficiency factor for the combined estimates of each term
STRATUMVARIANCE = *scalars*	Estimates of the stratum variances (for block terms only)
COMPONENT = *scalars*	Stratum variance components (for block terms only)
STATUS = *scalars*	Status code describing how the term is estimated (together with its marginal terms, if the term is a treatment term)

AKEEP allows you to copy components of the output from an analysis of variance into standard Genstat data structures. You can save the information from the analysis in a save structure, using the SAVE option of ANOVA (4.1.2) and then specify the same structure in the SAVE option of AKEEP. Alternatively, Genstat automatically stores the save structure from the last y-variate that has been analysed, and this is used as a default by AKEEP if you do not specify a save structure explicitly.

Several options are provided to save information about the analysis as a whole. The RESIDUALS and FITTEDVALUES options allow variates to be specified to store the residuals and fitted values, respectively. The residuals, like those saved by the RESIDUALS parameter of ANOVA, are taken only from the final stratum. The RMETHOD option controls whether these are simple residuals (like those printed by ANOVA – the default) or whether they are standardized according to their estimated variances. As an alternative, the CBRESIDUALS option saves residuals that incorporate the variability from all the strata. With an orthogonal design, these are

simply the sum of the residuals from every stratum. For a non-orthogonal design, they are the data values minus the combined estimates of the treatment effects (4.7.1). Likewise, the CBCREGRESSION option allows you to save estimates of covariate regression coefficients that combine information from all the strata, and the CBCVCOVARIANCE option can save their variances and covariances. (The estimates and their variances and covariances from each individual stratum can be saved using the CREGRESSION and CVCOVARIANCE parameters, as described below.)

The TREATMENTSTRUCTURE, BLOCKSTRUCTURE and WEIGHTS options save the treatment and block formulae, and the weights variate (if any) that were used to specify the analysis. The AFACTORIAL option can save the value used for the FACTORIAL option in the ANOVA comamnd that did the analysis, and the YVARIATE option can be set to a dummy to point to the variate that was analysed (i.e. the variate defined by the Y parameter of ANOVA; see 4.1.2). Information about the properties of the design can be saved using the EXIT option, which is described in 4.7.5.

The AOVTABLE option saves the analysis-of-variance table, as a pointer with a variate or a text for each column of the table. The pointer elements are labelled with the column labels of the table, and the variates contain missing values where the table has blanks. These can be printed as blanks by setting option MISSING=' ' in the PRINT directive. Example 4.6a saves and prints the analysis-of-variance table from Examples 4.5b and 4.5.1.

Example 4.6a

```
  45  AKEEP       [AOVTABLE=a]
  46  PRINT       [MISSING=' '] a[1...6]; JUSTIFICATION=left,5(right);\
  47              FIELD=20,5(10); DECIMALS=0,0,1,1,2,2; SKIP=0,5(1)

a['Source']           a['d.f.']  a['s.s.']  a['m.s.']  a['v.r.'] a['F pr.']
Blocks stratum                5    15875.3     3175.1       5.28
Blocks.Wplots stratum
Variety                       2     1786.4      893.2       1.49       0.27
Residual                     10     6013.3      601.3       3.40
Blocks.Wplots.Subplots stratum
Nitrogen                      3    20020.5     6673.5      37.69       0.00
  Lin                         1    19536.4    19536.4     110.32       0.00
  Quad                        1      480.5      480.5       2.71       0.11
  Deviations                  1        3.6        3.6       0.02       0.89
Nitrogen.Variety              6      321.7       53.6       0.30       0.93
  Lin.Variety                 2      168.3       84.2       0.48       0.62
  Quad.Variety                2       11.1        5.5       0.03       0.97
  Deviations                  2      142.3       71.2       0.40       0.67
Residual                     45     7968.8      177.1
Total                        71    51985.9
```

The parameters of AKEEP save information about particular model terms in the analysis. The TERMS parameter specifies a model formula, which Genstat expands to form the series of model terms about which you wish to save information. As in ANOVA (4.1.2), the FACTORIAL option sets a limit on the number of factors in each term. Any term containing more than that limit is deleted. The subsequent parameters allow you to specify identifiers of data structures to store various components of information for each of the terms that you have specified. If there are components that are not required for some of the terms, you should insert a missing identifier (*) at that point of the list. For example

```
AKEEP Source + Amount + Source.Amount; MEANS=*,*,Meangain;\
  SS=Ssource,Samount,Ssbya; VARIANCE=Vsource,*,*
```

sets up a table Meangain containing the source by amount table of means; it forms scalars Ssource, Samount and Ssbya to hold the sums of squares for Source, Amount and Source.Amount respectively, and scalar Vsource to store the unit variance for the effects of Source.

The structures to hold the information are defined automatically, so you need not declare them in advance. If you have declared any of the tables already, its classification set will be redefined, if necessary, to match the factors in the table that you wish to store. Thus `Meangain` here would be redefined to be classified by the factors `Source` and `Amount`, if it had previously been declared with some other set of classifying factors. Sizes of variates and symmetric matrices will also be redefined if necessary.

Most of the components are self-explanatory. Tables of means and effects are described in 4.1.2; these are relevant only for treatment terms. Standard errors for a table of means can be saved using the `SEMEANS` parameter. For some designs, such as split-plots, different standard errors are needed for the means according to which pair of means is to be compared. The `EQFACTORS` option allows you to specify factors within the tables of means whose levels are assumed to be equal for the two means. Alternatively, the `SEDMEANS` parameter can save a symmetric matrix containing a standard error of difference for each pair of means, the `VCMEANS` parameter can save a symmetric matrix with the variances and covariances for the means, and the `LSDMEANS` parameter can save a symmetric matrix containing least significant differences. The `LSDLEVEL` option specifies the significance level to use; default 5(%). The `DFMEANS` parameter saves a symmetric matrix with the degrees of freedom for comparing each pair of means. The rows and columns of these matrices are labelled by the factor name and level (or label if available) of the mean concerned.

Note: the `AFMEANS` procedure (4.1.5) provides an alternative way of saving predicted means and their standard errors etc. It has the advantage over `AKEEP` that the term need not have been included in the analysis. So, for example, you can obtain an `A` × `B` table of means, even if the model contained only the `A` and `B` main effects.

Partial effects (which are also available only for treatment terms) differ from the usual effects, presented by Genstat, only when there is non-orthogonality. The usual effects of a treatment term are estimated after eliminating the terms that precede it in the model (4.1.1), whereas the partial effects are those that would be estimated after eliminating the subsequent treatment terms as well (4.7.4). The `TWOLEVEL` option controls what it stored for terms whose factors all have only two levels. The settings `response` (the default) or `Yates` generate a scalar response, as described in 4.1.3; whereas `TWOLEVELS=effects` produces a table of effects. Replication tables are described in 4.1.3 and appear in the example in 4.3. The replications must be stored in a table if the values are unequal. For equal replications you can supply either a scalar or a table, but if the saving structure has not been declared `AKEEP` will define it as a scalar. Tables of residuals, available for block terms, are illustrated in 4.1.3 and 4.2.1. The `RMETHOD` option controls whether or not they are standardized.

Example 4.6b saves and prints tables of means for `Variety`, `Nitrogen` and `Variety.Nitrogen`, that were discussed in Section 4.2.1 (see Example 4.2.1a).

Example 4.6b

```
48  AKEEP       Nitrogen + Variety + Nitrogen.Variety;\
49              MEANS=Nmean,Vmean,NVmean; SEMEANS=Nsem,Vsem,NVsem
50  &           [EQFACTORS=Variety] Nitrogen.Variety; SEMEANS=NVsem_same_V
51  PRINT       Nmean,Nsem; FIELD=8

               Nmean    Nsem
   Nitrogen
      0 cwt     79.4   3.137
    0.2 cwt     98.9   3.137
    0.4 cwt    114.2   3.137
    0.6 cwt    123.4   3.137

52  &           Vmean,Vsem; FIELD=8
```

```
              Vmean    Vsem
    Variety
    Victory    97.6   5.006
Golden rain   104.5   5.006
 Marvellous   109.8   5.006

53  &           NVmean,NVsem,NVsem_same_V; FIELD=14

    Variety       Victory
                   NVmean         NVsem  NVsem_same_V
   Nitrogen
      0 cwt          71.5         6.870         5.433
    0.2 cwt          89.7         6.870         5.433
    0.4 cwt         110.8         6.870         5.433
    0.6 cwt         118.5         6.870         5.433


    Variety   Golden rain
                   NVmean         NVsem  NVsem_same_V
   Nitrogen
      0 cwt          80.0         6.870         5.433
    0.2 cwt          98.5         6.870         5.433
    0.4 cwt         114.7         6.870         5.433
    0.6 cwt         124.8         6.870         5.433


    Variety    Marvellous
                   NVmean         NVsem  NVsem_same_V
   Nitrogen
      0 cwt          86.7         6.870         5.433
    0.2 cwt         108.5         6.870         5.433
    0.4 cwt         117.2         6.870         5.433
    0.6 cwt         126.8         6.870         5.433
```

Four components can be saved in scalars: sums of squares (4.1), numbers of degrees of freedom (4.1), efficiency factors (4.7.1) and unit variances. The unit variance of a treatment term is the residual mean square of the stratum where the term is estimated, divided by its efficiency factor and covariance efficiency factor. Thus you can calculate the estimated variance of any of the effects of the term by dividing its unit variance by the replication of the effect (4.1.3).

For a treatment term, the `RTERM` parameter can be used to save a formula containing the model term corresponding to the lowest stratum in which it is estimated (down to and including any stratum defined by the `STRATUM` option). This can then be used as the setting of the `TERMS` parameter of a subsequent `AKEEP` statement to obtain further information about the stratum, for example its number of residual degrees of freedom (see Example 4.1.8). For a block term, `RTERM` saves all the strata to which it would be appropriate to compare the term. So, with a block structure of

```
Blocks/Plots/Subplots
```

the command

```
AKEEP Blocks + Blocks.Plots; RTERM=Rb,Rbp
```

would define `Rb` as the formula `!f(Blocks.Plots)`, and `Rbp` as the formula `!f(Blocks.Plots,Subplots)`. Alternatively, with a block structure of

```
Reps/(Rows*Columns)
```

the command

```
AKEEP Reps; RTERM=Rr
```

would define `Rr` as the formula `!f(Reps.Rows + Reps.Blocks)`.

There are three parameters that allow you to save information about the covariates (4.3). To save the regression coefficients estimated in a particular stratum, you should specify the model term of the stratum with the `TERMS` parameter and a variate with the `CREGRESSION` parameter. Genstat defines the variate to have a length equal to the number of covariates, and stores the

estimated regression coefficients of the covariates in the order in which they were listed in the COVARIATE statement (4.3.1). For the example in 4.3.1, you could put

```
AKEEP Blocks.Plots; CREGRESSION=B
```

to save the regression coefficient estimated for the covariate in the Blocks.Plots stratum; B will be declared implicitly as a variate of length one, as there was only one covariate. The CVCOVARIANCE parameter saves the variances and covariances of the estimated covariate regression coefficients, in a symmetric matrix. The CSSP parameter allows you to obtain sums of squares and products between the covariates for the specified model term. These are arranged in a symmetric matrix. The value in row *i* on the diagonal is the sum of squares for the term in the analysis of variance that has as its y-variate the *i*th covariate listed in the COVARIATE statement. The value in row *i* and column *j* is the cross-product between the effects estimated for the term in the analysis of variance of covariate *i* and those estimated for the same term in the analysis of covariate *j*.

Four parameters save information about contrasts (4.5). For each treatment term there will generally be several contrasts, so the information is stored in pointers with one element for each contrast. Example 4.6c shows how to save the estimates, the x-variates, the standard errors and the degrees of freedom for the contrasts of Nitrogen and of Variety.Nitrogen fitted in Example 4.6a. The structure Ncontr, for example, is defined as a pointer with three elements, labelled 'Lin', 'Quad' and 'Deviations': Ncontr['Lin'] (that is Ncontr[1]) is a scalar containing the estimated linear contrast of Nitrogen; Ncontr['Quad'] similarly contains the estimated quadratic contrast; while Ncontr['Deviations'] is a one-way table, classified by Nitrogen, containing the deviations from the fitted quadratic polynomial. Lines 56-63 of the program print the information for each contrast, to show the structure of each identifier and what it stores.

Example 4.6c

```
  54  AKEEP Nitrogen+Nitrogen.Variety; XCONTRASTS=Nxvar,NVxvar; \
  55    CONTRASTS=Ncontr,NVcontr; SECONTRASTS=Nse,NVse; DFCONTRASTS=Ndf,NVdf
  56  PRINT Ncontr[1],Nse[1],Ndf[1]; FIELD=14

 Ncontr['Lin']    Nse['Lin']    Ndf['Lin']
         73.67         7.014         1.000

  57  PRINT Ncontr[2],Nse[2],Ndf[2]; FIELD=14

Ncontr['Quad']   Nse['Quad']   Ndf['Quad']
        -64.58         39.21         1.000

  58  PRINT Ncontr[3],Nse[3]; FIELD=22 & Ndf[3]; FIELD=22

              Ncontr['Deviations']     Nse['Deviations']
    Nitrogen
       0 cwt                0.1000                 3.137
     0.2 cwt               -0.3000                 3.137
     0.4 cwt                0.3000                 3.137
     0.6 cwt               -0.1000                 3.137


    Ndf['Deviations']
                1.000

  59  PRINT Nxvar[]; FIELD=14

              Nxvar['Lin'] Nxvar['Quad'] Nxvar['Deviations']
    Nitrogen
       0 cwt       -0.3000       0.04000         1.000
     0.2 cwt       -0.1000      -0.04000         1.000
     0.4 cwt        0.1000      -0.04000         1.000
     0.6 cwt        0.3000       0.04000         1.000

  60  PRINT NVcontr[1],NVse[1]; FIELD=24 & NVdf[1]; FIELD=24
```

```
            NVcontr['Lin.Variety']     NVse['Lin.Variety']
    Variety
    Victory                  7.417                   12.15
Golden rain                  1.667                   12.15
 Marvellous                 -9.083                   12.15


   NVdf['Lin.Variety']
                2.000

61  PRINT NVcontr[2],NVse[2]; FIELD=24 & NVdf[2]; FIELD=24

           NVcontr['Quad.Variety']    NVse['Quad.Variety']
    Variety
    Victory                 -1.042                   67.91
Golden rain                 12.500                   67.91
 Marvellous                -11.458                   67.91


  NVdf['Quad.Variety']
                2.000

62  PRINT NVcontr[3],NVse[3]; FIELD=24 & NVdf[3]; FIELD=24

    Variety                 Victory
            NVcontr['Deviations']      NVse['Deviations']
   Nitrogen
      0 cwt                  0.725                   5.433
    0.2 cwt                 -2.175                   5.433
    0.4 cwt                  2.175                   5.433
    0.6 cwt                 -0.725                   5.433


    Variety             Golden rain
            NVcontr['Deviations']      NVse['Deviations']
   Nitrogen
      0 cwt                  0.083                   5.433
    0.2 cwt                 -0.250                   5.433
    0.4 cwt                  0.250                   5.433
    0.6 cwt                 -0.083                   5.433


    Variety              Marvellous
            NVcontr['Deviations']      NVse['Deviations']
   Nitrogen
      0 cwt                 -0.808                   5.433
    0.2 cwt                  2.425                   5.433
    0.4 cwt                 -2.425                   5.433
    0.6 cwt                  0.808                   5.433


    NVdf['Deviations']
                2.000

63  PRINT Nxvar[1,2]; FIELD=14 & NVxvar[3]; FIELD=14

            Nxvar['Lin'] Nxvar['Quad']
   Nitrogen
      0 cwt      -0.3000       0.04000
    0.2 cwt      -0.1000      -0.04000
    0.4 cwt       0.1000      -0.04000
    0.6 cwt       0.3000       0.04000


          NVxvar['Deviations']
    Variety       Victory   Golden rain    Marvellous
   Nitrogen
      0 cwt         1.000         1.000         1.000
    0.2 cwt         1.000         1.000         1.000
    0.4 cwt         1.000         1.000         1.000
    0.6 cwt         1.000         1.000         1.000
```

The CBMEANS, CBSEMEANS, CBSEDMEANS, LSDCBMEANS, VCCBMEANS, DFCBMEANS, CBEFFECTS, CBVARIANCE, DFCEFFECTS, CBCEFFICIENCY and STRATUMVARIANCES parameters save details of estimates that combine information from all the strata of the design, and the COMPONENT parameter saves the stratum variance components. These are explained in 4.7.1.

In designs where there is partial confounding, and treatment terms are estimated in more than one stratum (4.7.1), options STRATUM and SUPPRESSHIGHER allow you to specify the strata from which the information is to be taken. This is relevant to tables of effects and partial effects, sums of squares, efficiency factors, unit variances, sums of squares and products between covariates, and information about contrasts. By default, Genstat searches all the strata, and takes the information from the lowest of the strata where the term is estimated. If you set the STRATUM option, only strata down to the specified stratum are searched. By setting SUPPRESSHIGHER=yes, you can restrict the search to only that stratum. For Example 4.7.1a,

```
AKEEP [STRATUM=Blocks] K.D; EFFECTS=EffKD;\
  EFFICIENCY=EfacKD
```

would take the effects estimated for K.D in the Blocks stratum, and put them into the table EffKD, and it would put their efficiency factor into the scalar EfacKD.

You cannot save tables of means if you have excluded any stratum from the search. Likewise, tables of residuals and residual sums of squares cannot be saved for any of the excluded strata. If a term is not estimated in any of the strata that are searched, the corresponding data structures are filled with missing values.

The STATUS parameter saves an integer code that describes the type of term, and how it is estimated. If the term is a treatment term, the code also gives information about how its marginal terms are estimated. (For example, the interaction term A.B has the main effects A and B as margins.)

1	the term is a treatment term; the term itself and all of its margins are orthogonal, and are estimated in the same stratum.
2	the term is a treatment term; the term itself and all of its margins have the same efficiency factor, and are estimated in the same stratum.
3	the term is a treatment term; the term and its margins have different efficiency factors, but are all estimated in the same stratum.
4	the term is a treatment term; the term itself and all of its margins are orthogonal, but are estimated in different strata.
5	the term is a treatment term; the term itself and all of its margins have the same efficiency factor, but are estimated in different strata.
6	the term is a treatment term; the term and its margins have different efficiency factors and are all estimated in different strata.
0	the term is a treatment term; and term itself or one of its margins is aliased.
−1	the term is an orthogonal block term.
−2	the term is a non-orthogonal block term.
*	the term was not in either the block or treatment model but all of its factors occurred somewhere in the analysis (AKEEP gives a fault if the term contains factors that did not occur anywhere in the analysis); all other parameters

are then ignored for that term.

As explained in Section 4.2.1, Genstat will set up an extra "factor" denoted `*Units*` if the block formula does not specify the final stratum explicitly. `AKEEP` allows you to refer to this "factor", if necessary, by putting the string `'*Units*'` (or `'*units*'` or `'*UNITS*'`) in the `TERMS` formula. Thus, to save the residual sum of squares in Example 4.1 you could put

```
AKEEP '*Units*'; SS=RatRSS
```

4.6.2 The ASTATUS procedure

The `ASTATUS` procedure provides an alternative to `AKEEP` (4.6.1) for accessing details of the models defined for `ANOVA`. It is particularly useful if you want to check models that have been defined automatically, for example by the Genstat design procedures (4.9).

ASTATUS procedure

Provides information about the settings of `ANOVA` models and variates (R.W. Payne).

Option

`PRINT` = *string tokens*	Controls printed output (`y`, `model`, `weights`); default `mode`

Parameters

`Y` = *pointers*	Pointer of length 1 to save the identifier of the y-variate of the most recent `ANOVA` or that used to form `INSAVE`
`TREATMENTSTRUCTURE` = *formula structures*	Saves the current setting of `TREATMENTSTRUCTURE` or the setting used to form `INSAVE`
`BLOCKSTRUCTURE` = *formula structures*	Saves the current setting of `BLOCKSTRUCTURE` or the setting used to form `INSAVE`
`COVARIATE` = *pointers*	Saves the current `COVARIATE` setting or the setting used to form `INSAVE`
`DESIGN` = *pointers*	Pointer of length 1 to save the design structure in the most recent `ANOVA` or the one used to form `INSAVE`
`WEIGHTS` = *pointers*	Pointer of length 1 to save the identifier of the variate of weights (if any) in the most recent `ANOVA` or that used to form `INSAVE`
`SAVE` = *asave* structures	Saves the save structure from the most recent `ANOVA`
`INSAVE` = *asave* structures	Provides a save structure from which to save `Y`, `TREATMENTSTRUCTURE`, `BLOCKSTRUCTURE` and `COVARIATE`; default `*` uses the current settings

---

`ASTATUS` allows information to be printed and saved about the model settings and other information involved in an `ANOVA` analysis.

By default `ASTATUS` prints the current settings defined by the directives `TREATMENTSTRUCTURE`, `BLOCKSTRUCTURE` and `COVARIATE`. This is governed by the default setting, `model`, of the `PRINT` option. The `y` setting prints the name of the y-variate from the most recent `ANOVA`, and the `weights` setting prints the identifier of the variate of weights (if any). Alternatively, if the `INSAVE` parameter is set to the save structure from an `ANOVA` analysis, the y-variate, weights and model settings will be those used to form the save structure.

If the `INSAVE` parameter is not set, the `Y` parameter can be used to save the identifier of the y-variate most recently analysed by `ANOVA`, in a pointer of length one. The

TREATMENTSTRUCTURE parameter saves the current setting defined by the TREATMENTSTRUCTURE directive (in a formula structure), and the BLOCKSTRUCTURE parameter similarly saves the current setting defined by the BLOCKSTRUCTURE directive. The COVARIATE parameter saves the current setting defined by the COVARIATE directive (in a pointer). The DESIGN parameter can save the design structure, which contains the information for the analysis, in a pointer of length one. Finally, the WEIGHTS parameter can save the identifier of the variate of weights in the most recent ANOVA, in a pointer of length one; the pointer is not formed if this was an unweighted analysis.

Alternatively, if INSAVE is set to an ANOVA save structure, the parameters Y, TREATMENTSTRUCTURE, BLOCKSTRUCTURE, COVARIATE, DESIGN and WEIGHTS save the settings used to form INSAVE.

The SAVE parameter saves the save structure from the most recent ANOVA (regardless of the setting of INSAVE).

Example 4.6d continues Example 4.6c, showing the models defined in the earlier parts of the analysis (see Examples 4.2.1a and 4.5b).

Example 4.6d

```
  64  ASTATUS

Treatment structure: POL(Nitrogen; 2; Nitlev)*Variety
Block structure: Blocks/Wplots/Subplots
Covariates: not set
Factorial: 3
```

4.6.3 The ASPREADSHEET procedure

ASPREADSHEET procedure

Saves results from an analysis of variance in a spreadsheet (R.W. Payne).

Options

MEANS = *pointer*	Pointer to tables to contain the treatment means; default means
SEMEANS = *pointer*	Pointer to tables to contain the effective standard errors of treatment means; default ese
SEDMEANS = *pointer*	Pointer to matrices to contain standard errors of differences of treatment means; default sed
EFFECTS = *pointer*	Pointer to tables to contain the treatment effects; default effects
REPLICATIONS = *pointer*	Pointer to tables of treatment replications; default replication
RESIDUALS = *variate*	Variate to save the residuals in the fittedvalues page; default residuals
FITTEDVALUES = *variate*	Variate to save the fitted values in the fittedvalues page; default fittedvalues
AOVTABLE = *pointer*	Pointer to a text and variates containing the information in the analysis-of-variance table; default aovtable
COVINFORMATION = *pointer*	Pointer to a text and variates containing the information about the estimated covariate regression coefficients; default cov
MVINFORMATION = *pointer*	Pointer to a text and variates containing the information the about estimated missing values; default missing

EQFACTORS = factors	Factors whose levels are to be assumed to be equal within the comparisons between means, when calculating effective standard errors
RMETHOD = string token	Type of residuals to form (simple, standardized); default simp
LSDMEANS = *pointer*	Pointer to matrices to contain least significant differences for means
LSDLEVEL = *scalar*	Significance level (as a percentage) for the least significant differences; default 5
SPREADSHEET = *string tokens*	What to include in the spreadsheet (aovtable, covariates, effects, means, semeans, sedmeans, lsdmeans, replications, fittedvalues, missingvalues); default aovt, cova, mean, sedm, repl, fitt, miss
OUTFILENAME = *text*	Name of Genstat workbook file (.gwb) or Excel (.xls or .xlsx) file to create
SAVE = *ANOVA save structure*	Specifies which analysis to save; default * i.e. most recent one

No parameters

ASPREADSHEET puts results from an analysis of variance into a spreadsheet. By default the results are from the most recent ANOVA, but you use the SAVE option to specify the save structure from some other analysis.

The SPREADSHEET option specifies which pages of the spreadsheet to form, with settings:

aovtable	analysis of variance table,
covariates	estimated covariate regression coefficients and their standard errors (if any covariates in the analysis),
effects	tables of treatment effects,
means	tables of treatment means,
semeans	tables of effective standard errors of treatment means,
sedmeans	symmetric matrices of standard errors of differences of treatment means,
lsdmeans	matrices of least significant differences of treatment means,
replications	replication tables of treatment terms,
fittedvalues	y-variate, fitted values and residuals,
missingvalues	estimates for missing values (if any).

By default, SPREADSHEET = aovt, cova, mean, sedm, repl, fitt, miss.

To help avoid clashes between the columns of the spreadsheets if you want to save results from more than one analysis, the parameters MEANS, SEMEANS, SEDMEANS, LSDMEANS, EFFECTS, REPLICATIONS, RESIDUALS, FITTEDVALUES, AOVTABLE, COVINFORMATION and MVINFORMATION allow you to specify identifiers for the columns (or sets of columns) that will store the corresponding results in the current spreadsheet.

The EQFACTORS option allows you to specify factors within the tables of means whose levels are assumed to be equal for the two means, when calculating effective standard errors.

The RMETHOD option controls whether the residuals are simple residuals (like those printed by ANOVA – the default) or whether they are standardized according to their variances.

You can save the data in either a Genstat workbook (.gwb) or an Excel spreadsheet (.xls or .xlsx), by setting the OUTFILENAME option to the name of the file to create. If the name is specified without a suffix, '.gwb' is added (so that a Genstat workbook is saved). If

OUTFILENAME is not specified, the data are put into a spreadsheet opened inside Genstat.

So, you could save the analysis-of-variance table, means and standard errors of differences of means in an Excel spreadsheet called Oatsresults.xlsx by giving the command

```
ASPREADSHEET [SPREADSHEET=aovtable,means,sedmeans;\
              OUTFILE='Oatsresults.xlsx]
```

4.7 Non-orthogonality and balance

So far, all the examples in this chapter have all been orthogonal. Each treatment term has been estimated in only one stratum. Any confounding between block and treatment terms has been complete: for example, in the split-plot design in 4.2.1, differences between varieties were completely confounded with whole-plots, and so were estimated only in that stratum.

The ANOVA directive can also analyse designs where there is partial confounding or where there is non-orthogonality, provided there is still the necessary property of balance. These concepts are discussed in this section.

4.7.1 Efficiency factors

The example below is of a design where there is partial confounding. Full details are given by Yates (1937, page 21) and by John (1971, page 135). This is an experiment to study the effects of three factors N, K and D on the yields of King Edward potatoes. The factor levels were as follows.

N: sulphate of ammonia at rates of 0 and 0.45 cwt per acre
K: sulphate of potash at rates of 0 and 1.12 cwt per acre
D: dung at rates of 0 and 8 tons per acre

The treatment formula (line 22) is

```
N * K * D  =  N + K + D + N.K + N.D + K.D + N.K.D
```

There were eight treatment combinations, but the blocks each had only four plots. Consequently some of the treatment terms needed to be confounded between blocks. This was done by confounding N.K.D between blocks 1 and 2, N.K between blocks 3 and 4, N.D between blocks 5 and 6, and K.D between blocks 7 and 8. There was thus only partial confounding: the interaction terms could be estimated within some of the blocks but not others. To illustrate how this was done, we can consider N.K: this represents the difference in the effect of N according to the level of K (and vice versa). Representing the treatment combinations as triplets of letters, giving respectively the level of N (- or n), K (- or k) and D (- or d), this can be written as

```
{ ('n--' + 'n-d') - ('---' + '--d') }
 - { ('nk-' + 'nkd') - ('-k-' + '-kd') }
= ('n--' + 'n-d' + '-k-' + '-kd')
 - ('---' + '--d' + 'nk-' + 'nkd')
```

The combinations in the first pair of brackets all occur in block 3, while those in the second pair all occur in block 4. Thus within blocks 3 and 4 there is no information on N.K; but information is available within the other 6 blocks. Thus N.K is estimated with efficiency 6/8 (= 0.75) in the Blocks.Plots stratum. The difference between the mean of the yields of the plots in block 3 and those in block 4 also provides an estimate of N.K; this represents the remaining 1/4 of the efficiency available for estimating N.K.

If a term is orthogonal, its efficiency factor equals one: the term is estimated with full efficiency in the stratum concerned. The efficiency factors of non-orthogonal terms are listed in the Information Summary obtained by setting option PRINT=information in either ANOVA or ADISPLAY (4.1.3). Terms that are aliased with earlier terms in the model (and so cannot be estimated) are also listed: these have zero efficiency factors. You can obtain details of the model terms with which they are aliased, using the ALIAS procedure.

The efficiency factors are not always so easy to derive and interpret as here: the original

definition by Yates (1936) was for the balanced incomplete-block design. But they always represent the proportion of the information available to estimate a term.

Example 4.7.1a

```
  2  " Partially confounded factorial (Yates 1937, p.21; John 1971, p.135)."
  3  UNITS [NVALUES=32]
  4  FACTOR [LEVELS=8] Blocks
  5  & [LEVELS=4] Plots
  6  & [LEVELS=2; LABELS=!T(_,n)] N
  7  & [LABELS=!T(_,k)] K
  8  & [LABELS=!T(_,d)] D
  9  GENERATE Blocks,Plots
 10  READ [PRINT=data,errors] N,K,D; FREPRESENTATION=labels

 11   _ _ _    n k _    n _ d    _ k d       n _ _    _ k _    _ _ d    n k d
 12   n _ _    _ k _    n _ d    _ k d       _ _ _    _ _ d    n k _    n k d
 13   n _ _    _ _ d    n k _    _ k d       _ _ _    _ k _    n _ d    n k d
 14    _ k _   _ _ d    n k _    n _ d       _ _ _    n _ _    _ k d    n k d  :
 15  VARIATE Yield
 16  READ Yield

   Identifier   Minimum      Mean   Maximum    Values   Missing
        Yield     87.00     291.6     471.0        32         0

 21  BLOCKSTRUCTURE Blocks/Plots
 22  TREATMENTSTRUCTURE N * K * D
 23  ANOVA Yield

Analysis of variance
====================

Variate: Yield

Source of variation     d.f.       s.s.       m.s.    v.r.

Blocks stratum
N.K                        1      780.1      780.1    3.02
N.D                        1      276.1      276.1    1.07
K.D                        1     2556.1     2556.1    9.91
N.K.D                      1      112.5      112.5    0.44
Residual                   3      774.1      258.0    0.81

Blocks.Plots stratum
N                          1     3465.3     3465.3   10.86
K                          1   161170.0   161170.0  505.21
D                          1   278817.8   278817.8  873.99
N.K                        1       28.2       28.2    0.09
N.D                        1     1802.7     1802.7    5.65
K.D                        1    11528.2    11528.2   36.14
N.K.D                      1       45.4       45.4    0.14
Residual                  17     5423.3      319.0

Total                     31   466779.7


Information summary
===================

Model term                 e.f.  non-orthogonal terms

Blocks stratum
  N.K                   0.250
  N.D                   0.250
  K.D                   0.250
  N.K.D                 0.250

Blocks.Plots stratum
  N.K                   0.750  Blocks
  N.D                   0.750  Blocks
  K.D                   0.750  Blocks
```

```
  N.K.D                 0.750  Blocks

* MESSAGE: the following units have large residuals.

Blocks 6     Plots 4           28.2   approx. s.e. 13.0


Tables of means
===============

Variate: Yield

Grand mean  291.6

        N                 n
             281.2    302.0

        K                 k
             220.6    362.6

        D                 d
             198.2    384.9

        N        K                k
        _             211.3    351.1
        n             229.9    374.1

        N        D                d
        _             196.5    365.9
        n             200.0    404.0

        K        D                d
        _             105.4    335.9
        k             291.1    434.0

                 K        _               k
        N        D             d                  d
        _             106.1    316.5    286.9    415.2
        n             104.6    355.3    295.3    452.8


Standard errors of differences of means
---------------------------------------

Table                    N           K           D           N
                                                             K
rep.                    16          16          16           8
d.f.                    17          17          17          17
s.e.d.                6.31        6.31        6.31        8.93
Except when comparing means with the same level(s) of
N                                                        9.65
K                                                        9.65

Table                    N           K           N
                         D           D           K
                                                 D
rep.                     8           8           4
d.f.                    17          17          17
s.e.d.                8.93        8.93       13.15
Except when comparing means with the same level(s) of
N                     9.65                   13.64
K                                 9.65       13.64
D                     9.65        9.65       13.64
N.K                                          14.12
N.D                                          14.12
K.D                                          14.12
```

(Notice in the output that the underline symbol has been used instead of minus for the zero level, to avoid having to put quotes around the labels when they are read in lines 11 to 14.)

As we explained in 4.1.3, the means produced by setting PRINT=means in ANOVA or ADISPLAY take the effects of each term only from the lowest stratum where it is estimated. Thus it would estimate N.K for example only from the Blocks.Plots stratum. The different efficiency factors for the component terms of the two-way and three-way tables of means in the example lead to different standard errors for some comparisons. For example, the s.e.d. for the N.K.D table is 13.15 when comparing means with different levels of all three factors, it is 13.64 if the level of one of the factors is identical for both means, and it is 14.12 if two of the factors are at identical levels.

The effects from the lowest stratum are usually those that are estimated most precisely; the lower strata generally have smaller mean squares and, in most designs, terms will have higher efficiency factors in the lower strata. Moreover, under the usual assumptions of Normality of residuals, differences between the means can be tested by the usual t-statistics. Nevertheless, for prediction you will often want to present means and effects that combine the information about each term from all the strata where it is estimated. Provided the design possesses the condition of *first-order balance* that is required for it to be analysed by Genstat (see 4.7.2), and provided there is no non-orthogonality between treatment terms, you can use the PRINT settings cbeffects and cbmeans to print combined estimates of the effects and the means respectively. (The design is then a *generally-balanced design*; see Payne & Tobias 1992).

Example 4.7.1b

```
  24  ADISPLAY [PRINT=effects,cbeffects,cbmeans]

Tables of effects
=================

Variate: Yield

Blocks stratum
--------------

N.K response                39.5,  s.e. 22.72,  rep. 8

N.D response               -23.5,  s.e. 22.72,  rep. 8

K.D response               -71.5,  s.e. 22.72,  rep. 8

N.K.D response             -30.0,  s.e. 45.43,  rep. 4


Blocks.Plots stratum
--------------------

N response                  20.8,  s.e. 6.31,  rep. 16

K response                 141.9,  s.e. 6.31,  rep. 16

D response                 186.7,  s.e. 6.31,  rep. 16

N.K response                 4.3,  s.e. 14.58,  rep. 8

N.D response                34.7,  s.e. 14.58,  rep. 8

K.D response               -87.7,  s.e. 14.58,  rep. 8

N.K.D response             -11.0,  s.e. 29.17,  rep. 4


Tables of combined effects
==========================

Variate: Yield

N response                  20.8,  s.e. 6.36,  rep. 16,
                                   effective d.f. 17.90
```

```
K response                141.9,  s.e. 6.36,  rep. 16,
                                  effective d.f. 17.90

D response                186.7,  s.e. 6.36,  rep. 16,
                                  effective d.f. 17.90

N.K response               12.3,  s.e. 12.94,  rep. 8,
                                  effective d.f. 23.89

N.D response               21.6,  s.e. 12.94,  rep. 8,
                                  effective d.f. 23.89

K.D response              -84.0,  s.e. 12.94,  rep. 8,
                                  effective d.f. 23.89

N.K.D response            -15.3,  s.e. 25.88,  rep. 4,
                                  effective d.f. 23.89


Tables of combined means
========================

Variate: Yield

        N                 n
             281.2        302.0

        K                 k
             220.6        362.6

        D                 d
             198.2        384.9

        N         K                 k
                  213.3        349.1
        n         228.0        376.0

        N         D                 d
                  193.2        369.1
        n         203.3        400.7

        K         D                 d
                  106.3        335.0
        k         290.2        434.9

                  K                   _                   k
        N         D                   d                   d
                  106.2        320.3        280.2        417.9
        n         106.3        349.6        300.2        451.9


Standard errors of differences of combined means
------------------------------------------------

Table                    N           K           D           N
                                                             K
rep.                    16          16          16           8
s.e.d.                6.36        6.36        6.36        9.00
effective d.f.       17.90       17.90       17.90       17.90
Except when comparing means with the same level(s) of
N                                                         9.08
effective d.f.                                           21.76
K                                                         9.08
effective d.f.                                           21.76
Table                    N           K           N
                         D           D           K
                                                 D
rep.                     8           8           4
s.e.d.                9.00        9.00       12.78
effective d.f.       17.90       17.90       19.94
Except when comparing means with the same level(s) of
N                     9.08                    12.83
effective d.f.       21.76                    21.76
```

```
K                                    9.08       12.83
effective d.f.                      21.76       21.76
D                        9.08        9.08       12.83
effective d.f.          21.76       21.76       21.76
N.K                                             12.89
effective d.f.                                  23.14
N.D                                             12.89
effective d.f.                                  23.14
K.D                                             12.89
effective d.f.                                  23.14
```

The combined estimates of the effects of any treatment term take the form of a weighted average of the estimates from each of the strata, where the weight for any particular stratum is given by the efficiency factor of the term in that stratum, divided by the variance of the units of the stratum. One common method of estimating the stratum variances simply uses the residual mean squares. However, this method does not make use of all the available information - the differences between the various estimates of each treatment effect also contain information about variability. Moreover, there may sometimes be strata with no residual degrees of freedom, as in the square lattice shown in 4.7.3. Thus, a rather more powerful algorithm is used (Payne & Tobias 1992). This is equivalent to the use of residual maximum likelihood (REML) but, for the generally-balanced designs on which it operates, is very much more efficient particularly in its use of workspace (Payne & Welham 1990). The estimated stratum variances, together with the effective degrees of freedom and the variance components of the strata, can be printed by setting `PRINT=stratumvariance`. The effective degrees of freedom of the combined effects and means are calculated from the effective degrees of freedom of the stratum variances using an algorithm based on Satterthwaite's method (see Payne 2004).

Example 4.7.1c

```
  25  ADISPLAY [PRINT=stratumvariances]

Estimated stratum variances
===========================

Variate: Yield

Stratum                      variance  effective d.f.  variance component

Blocks                         371.56           6.099               11.87
Blocks.Plots                   324.10          17.901              324.10
```

4.7.2 Balance

The condition of first-order balance required for a design and its specification to be analysable by the `ANOVA` directive is explained algorithmically by Wilkinson (1970) and mathematically by James & Wilkinson (1971) and Payne & Tobias (1992). Essentially it is that the contrasts of each term should all have a single efficiency factor, wherever the term is estimated. In the example in 4.7.1, all the terms have only one degree of freedom, and so represent only one contrast. There is thus no difficulty in verifying that the design is balanced.

Suppose instead that the treatment combinations were represented by a single factor `T` with eight levels:

```
FACTOR [LABELS=!T('---','--d','-k-','-kd',\
                  'n--','n-d','nk-','nkd')] T
```

The main effect of `T` would not be balanced: the comparison of levels

```
'---' '--d' '-k-' '-kd'
```

with

```
'n--' 'n-d' 'nk-' 'nkd'
```

has efficiency factor one in the `Blocks.Plots` stratum and zero in the `Blocks` stratum (this contrast is equivalent to the main effect of `N` in the original specification); but the comparison of levels

```
'n--' 'n-d' '-k-' '-kd'
```

with

```
'---' '--d' 'nk-' 'nkd'
```

has efficiency 0.25 in the `Blocks` stratum and 0.75 in the `Blocks.Plots` stratum (this is equivalent to `N.K` in the original specification). Thus the main effect of `T` is not balanced, since in the `Block.Plots` stratum some of its contrasts have efficiency factor one, while others have efficiency factor 0.75. Genstat can detect this imbalance and will give you an error diagnostic: see later in this section.

For the design to have been balanced for `T`, a further three pairs of blocks would be required. By confounding the comparison corresponding to the main effect of `N` between the first pair of extra blocks, that for `K` between the second pair, and that for `D` between the third pair, all the contrasts of `T` would be estimated within twelve of the (now) fourteen blocks, and confounded in the other two. The extended design would thus be balanced - as you may wish to verify!

To analyse the original design with a single treatment term `T`, a more complicated specification is required involving pseudo-factors.

4.7.3 Pseudo-factors

Unbalanced designs with a single error term can be analysed using the `AUNBALANCED` procedure (Section 4.8.1), and those with several error terms can be analysed by `REML`. Alternatively, you may be able to use the pseudo-factorial operator `//` to partition an unbalanced treatment term into pseudo-terms, which are each balanced – and thus retain the more comprehensive output available from `ANOVA`. In our example, there is a factor `T`, some of whose contrasts have efficiency one in the `Blocks.Plots` stratum and zero elsewhere, while others have efficiency 0.25 in the `Blocks` stratum and 0.75 in the `Blocks.Plots` stratum. If instead of

```
TREATMENTSTRUCTURE T
```

we specify

```
TREATMENTSTRUCTURE T // (N + K + D + N.K + N.D + K.D)
```

the terms within the brackets that follow the operator `//` are linked to the term `T` as pseudo-terms. (Without the brackets, only the term immediately after `//` would be linked to `T`.) When the time comes for `T` to be fitted, the pseudo-terms `N`, `K`, `D`, `N.K`, `N.D` and `K.D` are fitted first. All the contrasts wholly estimated in the `Blocks.Plots` stratum are thus removed (by `N`, `K` and `D`), as well as some of the other contrasts. The remaining contrasts (denoted by `T` in the information summary) are all estimated with efficiency 0.25 between blocks and 0.75 within blocks. Thus all the pseudo-terms are balanced: those specified explicitly (`N`, `K`, `D`, `N.K`, `N.D` and `K.D`), and the final pseudo-term which represents the contrasts not accounted for by `N`, `K`, `D`, `N.K`, `N.D` and `K.D`. So by using the pseudo-factors, the design becomes analysable. In this example all the pseudo-terms represent single degrees of freedom - the final pseudo-term corresponds to the contrast represented earlier by `N.K.D` - but later we give an example where the pseudo-terms each have several degrees of freedom.

The sums of squares of the pseudo-terms are automatically combined to form the sum of squares for `T` in the analysis-of-variance table. Similarly the effects are all added together to form the table of means for `T`.

Example 4.7.3a

```
  26  FACTOR [LABELS=!T('_ _ _','_ _ d','_ k _','_ k d', \
  27                    'n _ _','n _ d','n k _','n k d')] T
  28  READ [PRINT=data,error] T; FREPRESENTATION=labels

  29  '_ _ _' 'n k _' 'n _ d' '_ k d'     'n _ _' '_ k _' '_ _ d' 'n k d'
  30  'n _ _' '_ k _' 'n _ d' '_ k d'     '_ _ _' '_ _ d' 'n k _' 'n k d'
  31  'n _ _' '_ _ d' 'n k _' '_ k d'     '_ _ _' '_ k _' 'n _ d' 'n k d'
  32  '_ k _' '_ _ d' 'n k _' 'n _ d'     '_ _ _' 'n _ _' '_ k d' 'n k d' :
  33  TREATMENTSTRUCTURE T // (N + K + D + N.K + N.D + K.D)
  34  ANOVA Yield
```

```
Analysis of variance
====================

Variate: Yield

Source of variation     d.f.       s.s.       m.s.    v.r.

Blocks stratum
T                          4     3724.9      931.2    3.61
Residual                   3      774.1      258.0    0.81

Blocks.Plots stratum
T                          7   456857.5    65265.4  204.58
Residual                  17     5423.3      319.0

Total                     31   466779.7


Information summary
===================

Model term                e.f.  non-orthogonal terms

Blocks stratum
  N.K                    0.250
  N.D                    0.250
  K.D                    0.250
  T                      0.250

Blocks.Plots stratum
  N.K                    0.750  Blocks
  N.D                    0.750  Blocks
  K.D                    0.750  Blocks
  T                      0.750  Blocks


* MESSAGE: the following units have large residuals.

Blocks 6     Plots 4           28.2   approx. s.e. 13.0


Tables of means
===============

Variate: Yield

Grand mean  291.6

        T   _ _ _   _ _ d   _ k _   _ k d   n _ _   n _ d   n k _   n k d
        N       1       1       1       1       2       2       2       2
        K       1       1       2       2       1       1       2       2
        D       1       2       1       2       1       2       1       2
            106.1   316.5   286.9   415.2   104.6   355.2   295.3   452.8
```

```
Standard errors of differences of means
---------------------------------------

Table                     T
rep.                      4
d.f.                     17
s.e.d.                13.15
Except when comparing means with the same level(s) of
N                     13.64
K                     13.64
D                     13.64
N.K                   14.12
N.D                   14.12
K.D                   14.12
```

The basic idea, then, is to use each pseudo-term to pick out a set of contrasts whose efficiency factors are all the same, wherever they are estimated. This should be reasonably straightforward, provided you understand how your design has been constructed. Pseudo-factors are set up automatically by the Genstat design procedures (4.9), and can also be formed by the GENERATE and FPSEUDOFACTORS directives (4.13.1 and 4.13.7). A further example is given below, but first we demonstrate that Genstat can indeed detect an unbalanced design. If we do not include the pseudo-factors, the design would be unbalanced. The error message correctly identifies T as the unbalanced term.

Example 4.7.3b

```
 35  TREATMENTSTRUCTURE T
 36  ANOVA Yield


******** Fault, code AN 1, statement 1 on line 36

Command: ANOVA Yield
Design unbalanced - cannot be analysed by ANOVA.
Model term T (non-orthogonal to term Blocks) is unbalanced,
in the Blocks.Plots stratum.
```

The traditional example for pseudo-factors is the partially balanced lattice. This has a single treatment factor, with number of levels equal to the square of some integer, *k*. To form the design, this factor is arbitrarily represented as the factorial combinations of two pseudo-factors, below called A and B, each with *k* levels. For further details see Yates (1937) or Kempthorne (1952). The example below is a simple lattice, taken from Cochran & Cox (1957, page 406). Here the treatment factor, Variety, has 25 levels. The correspondence between levels of Variety and the two pseudo-factors is:

```
       B:   1   2   3   4   5
     A:
     1      1   2   3   4   5
     2      6   7   8   9  10
     3     11  12  13  14  15
     4     16  17  18  19  20
     5     21  22  23  24  25
```

The simple lattice has two replicates, each with *k* blocks of *k* plots: the block model is

```
Rep/Block/Plot = Rep + Rep.Block + Rep.Block.Plot
```

The main effect of A is confounded with the blocks in the first replicate: block 1 has the five levels of Variety that correspond to level 1 of A, block 2 has those with level 2, and so on. Similarly, B is confounded with the blocks of the second replicate. Thus A and B are each confounded with blocks in one out of the two replicates. So they have efficiency 0.5 in the

Rep.Block (or blocks-within-replicates) stratum, and 0.5 in the Rep.Block.Plot (or plots-within-blocks) stratum. The treatment model is

```
Variety//(A + B)
```

The partially confounded parts of Variety are specified by the two pseudo-terms, A and B, and will be fitted first. The remaining contrasts of Variety correspond to the interaction between A and B, which is all estimated in the Rep.Block.Plot stratum. This final pseudo-term is thus also balanced, so the design can be analysed. The analysis-of-variance table in Example 4.7.3c differs from that presented by Cochran & Cox (1957); they do not present the treatment sums of squares between and within blocks, but merely a sum of squares unadjusted for blocks. Example 4.7.3c also prints the table of means combining information from both the Rep.Block and the Rep.Block.Plot strata (4.7.1), and the stratum variances and variance components.

Example 4.7.3c

```
  2  " 5x5 Simple lattice (Cochran & Cox 1957, p.406)."
  3  UNITS [NVALUES=50]
  4  FACTOR [LEVELS=2] Rep
  5  & [LEVELS=5] Block,Plot,A,B
  6  & [LEVELS=25; VALUES=(1...25),(1,6...21),(2,7...22), \
  7     (3,8...23),(4,9...24),(5,10...25)] Variety
  8  GENERATE Rep,Block,Plot
  9  & [TREATMENTS=Variety; REPLICATES=Rep; BLOCKS=Block] A,B
 10  READ Yield

   Identifier   Minimum      Mean   Maximum    Values   Missing
        Yield     4.000     13.62     30.00        50         0

 13  BLOCKSTRUCTURE Rep/Block/Plot
 14  TREATMENTSTRUCTURE Variety//(A+B)
 15  ANOVA [PRINT=aovtable,cbmeans,stratumvariances] Yield

Analysis of variance
====================

Variate: Yield

Source of variation     d.f.       s.s.       m.s.    v.r.

Rep stratum                1     212.18     212.18

Rep.Block stratum
Variety                    8     350.00      43.75

Rep.Block.Plot stratum
Variety                   24     711.12      29.63    2.17
Residual                  16     218.48      13.65

Total                     49    1491.78


Tables of combined means
========================

Variate: Yield

 Variety        1        2        3        4        5        6        7
       A        1        1        1        1        1        2        2
       B        1        2        3        4        5        1        2
           19.07    16.97    14.65    14.77    12.85    13.17     9.07

 Variety        8        9       10       11       12       13       14
       A        2        2        2        3        3        3        3
       B        3        4        5        1        2        3        4
            6.75     8.37     8.45    23.55    12.46    12.63    20.75
```

```
  Variety       15       16       17       18       19       20       21
        A        3        4        4        4        4        4        5
        B        5        1        2        3        4        5        1
            19.33    12.62    10.53    10.70     7.32    11.40    11.63

  Variety       22       23       24       25
        A        5        5        5        5
        B        2        3        4        5
            18.53    12.20    17.33    15.40


Standard errors of differences of combined means
------------------------------------------------

Table              Variety
rep.                     2
s.e.d.               4.234
effective d.f.       18.88
Except when comparing means with the same level(s) of
A                    3.974
effective d.f.       18.01
B                    3.974
effective d.f.       18.01


Estimated stratum variances
===========================

Variate: Yield

Stratum                      variance  effective d.f.  variance component

Rep                           212.180           1.000               4.015
Rep.Block                     111.805           7.129              19.630
Rep.Block.Plot                 13.655          16.871              13.655
```

Example 4.7.3c also illustrates how to use the GENERATE directive (line 8) to form the values of pseudo-factors; the details are explained in 4.13.1.

4.7.4 Non-orthogonality between treatment terms

The examples earlier in this section illustrate non-orthogonality between treatment and block terms. Balanced designs can also occur where the non-orthogonality is between treatment terms. However the interpretation of the analysis requires more care; indeed there may be information that Genstat is unable to calculate. (Similar difficulties occur in ordinary regression with observational data, see Chapter 8: usually the explanatory variables will not be orthogonal to each other and so their sums of squares, and thus the importance that may be ascribed to them, will depend on the order in which they are fitted.)

Suppose that the treatment model is

```
A + B + C
```

that B is non-orthogonal to A, and that C is non-orthogonal to both A and B. Genstat fits the model sequentially. Thus the sum of squares produced for A is for A ignoring B and C: no account is taken of these two factors, which are still to be fitted. With B, A has already been fitted and thus eliminated, whereas C has not. So the sum of squares produced for B is for B eliminating A and ignoring C. The sum of squares for C, which is fitted last, is eliminating both A and B.

Each sum of squares can be expressed as the difference between the residual sums of squares before and after fitting a particular term. So the sums of squares that are presented by Genstat will automatically add to the total sum of squares. Examining these enables you to check whether any of the terms in the model has an effect. However, to be sure that there is an effect of A, for example, that cannot be explained by B and C requires the sum of squares for A eliminating B and C. To obtain this you could redefine the treatment model as either

```
B + C + A
```

or

```
C + B + A
```

but the design would not necessarily be balanced according to these specifications.

Similarly, the effects estimated for each term are eliminating those terms fitted before it, and ignoring those that are still to be fitted. *Partial effects*, defined as the effects of a term eliminating all the other treatment terms, are calculated during the analysis and can be obtain using `AKEEP` (4.6.1).

A table of means for `A.B`, if this were in the model, would require the effects for `A` eliminating `B`, those for `B` eliminating `A`, and those for the interaction `A.B`. However, with the treatment model `A + B + C`, the necessary effects for `A` are not available. Consequently, no means are presented for terms that contain mutually non-orthogonal margins (like `A` and `B` for the table `A.B`).

A maximum of 10 mutually non-orthogonal terms is allowed. For example, term `T[10]` may be non-orthogonal to `T[9]`, which is non-orthogonal to `T[8]`, and so on down to term `T[2]`, which is non-orthogonal to term `T[1]`; but to include an extra term `T[11]` in the sequence would exceed the limit. This limit should be sufficient for any designed experiment. Data with many non-orthogonal terms are, in any case, analysed more efficiently by the regression directives described in Chapter 8.

Note that, if the terms `A`, `B` and `C` here had been orthogonal, the sum of squares and effects obtained for any one of them would remain the same irrespective of which of the other two terms had been fitted. For example, the sum of squares for `A` ignoring `B` and `C` would be identical to that for `A` eliminating `B` and `C`. Thus each of these three terms could be assessed independently, without regard to the other two. If two terms are far from orthogonal, you may find that the effects of either term ignoring the other are significant, but that neither set of effects is significant when the other term is eliminated. Deciding which of the terms are important may then be very difficult, and you may have to recommend that another experiment be done. This illustrates that orthogonality between treatment terms is not merely a convenience for making the computations more efficient: it also greatly simplifies the interpretation of the results.

4.7.5 The method of analysis

In this subsection we briefly describe the algorithm that is used to do the analysis of variance. However, for most purposes you will not need this information.

The model formulae defined by the `BLOCKSTRUCTURE` and `TREATMENTSTRUCTURE` are interpreted by an extension of the algorithm of Rogers (1973); further details are given by Wilkinson & Rogers (1973) and Payne (1990).

The method used to do the analysis is described in detail by Payne & Wilkinson (1977), Wilkinson (1970) and Payne & Tobias (1992). It operates on a working vector which initially contains the data values, and finally contains the residuals. The terms in the model are fitted by a series of *sweep* operations. Each sweep estimates the effects of a term, and then subtracts them from the current working vector, which then becomes the working vector for the next sweep. The first sweep is for the grand mean. The block terms are fitted next, to give an initial partitioning into strata. Then the treatments are fitted within each stratum.

If a term is orthogonal, its estimated effects are simply the corresponding table of means calculated from the current working vector. If the term is non-orthogonal to any of the terms already fitted, some of the information about the term is unavailable, and its effects are the totals calculated from the current vector, divided by its replication and efficiency factor. For the term to be balanced, the information still available must be the same for all the contrasts between the effects of the term, so that there is a single efficiency factor for all the contrasts. If the term is orthogonal, the efficiency factor is one. A zero efficiency factor indicates that the term is

completely aliased with earlier terms in the model, and so cannot be estimated.

A sweep for a non-orthogonal term reintroduces effects for the terms to which it is non-orthogonal. Before sweeping for the next term in the model, these effects are removed by a sequence of *re-analysis* sweeps for the terms concerned. If any term in the re-analysis sequence is itself non-orthogonal, it must itself be followed by its own re-analysis sequence, and so on. Genstat allows for re-analysis sequences to be nested only ten deep, which is why there is the limit of ten mutually non-orthogonal terms (4.7.4).

When there are several strata, the analysis of each one is introduced by a special sweep known as a *pivot*, in which the value in each unit of the working vector is replaced by the corresponding effect calculated for the block term of the stratum. During the analysis of a stratum, the re-analysis sweeps for its own block term take the form of recalculating the effects and repeating the pivot.

Procedure ASWEEP, which can perform all these types of sweep, is provided in the Procedure Library for those who wish to study the process further.

The algorithm, unlike multiple regression algorithms, does not distinguish between the individual contrasts of each term (unless you partition it up into pseudo-terms: 4.7.3). This makes the computations more efficient, but it means that only balanced terms can be fitted.

The design can be analysed if all the terms in the model are balanced: that is if they each have a single efficiency factor for their effects, in any stratum where they are estimated. The design is then said to have *first-order balance* with respect to the specified model (Wilkinson 1970, James & Wilkinson 1971, Payne & Tobias 1992): for a brief description, see 4.7.2.

A further consequence of the way in which the effects of each terms are all fitted together is that, if any part of a term is present in a stratum, Genstat must assume that all its effects can be estimated there. Thus if a term is only partially estimable in a stratum (due to partial aliasing or to partial confounding), the degrees of freedom will be incorrect. In such situations Genstat prints a warning diagnostic. To obtain an analysis with the correct numbers of degrees of freedom you should use pseudo-factors (4.7.3) to identify the parts of a term that are estimated in the different strata.

Genstat determines the structure of the design by a process known as the *dummy analysis* (4.1.2). This is similar to the analysis of the data, but involves extra sweeps to detect whether each term can be estimated in a particular stratum, and to determine its efficiency factor there. In these sweeps, a near-zero sum of squares is taken to indicate that the term cannot be estimated. However the test cannot be against an exact value of zero, because computer calculations always involve errors of round-off. Thus Genstat tests against a number slightly larger than zero; this zero limit is calculated as the total sum of squares in the working variate (after removing the grand mean) multiplied by the first element of the variate specified in the TOLERANCE option of ANOVA (4.1.2). By default, this first element contains the value 10^{-7}. A similar limit checks for zero sums of squares in the analysis of the data, but here the multiplier is given in the second element of the TOLERANCE variate; the default value is 10^{-9}.

The working vector for the dummy analysis contains random values from a Cauchy distribution. The starting value for their generation is set by the SEED option of ANOVA (4.1.2). Thus if you have doubts about a particular dummy analysis, for example if you think that a term is incorrectly listed as aliased, you can change the starting value and repeat the analysis with a different working vector.

A simpler and quicker form of the dummy analysis is available for designs that are orthogonal, and for which all the effects of each term have equal replication. (An orthogonal design is one in which each term has efficiency factor either zero or one in each stratum.) This incorporates a check which will detect any non-orthogonality, unless the design is particularly complicated and terms are aliased. The ORTHOGONAL option of ANOVA (4.1.2) allows you to specify whether non-orthogonality should cause Genstat to switch to the full dummy analysis, or to terminate the analysis with an error diagnostic.

You can use the EXIT option of ANOVA or AKEEP to save an "exit code" summarizing the properties of the design as determined by the dummy analysis:

0	design orthogonal;
1	design has general balance (blocks terms mutually orthogonal, treatment terms mutually orthogonal, some treatment terms non-orthogonal to the block terms);
2	blocks terms mutually orthogonal, treatment terms non-orthogonal;
3	block terms non-orthogonal, treatment terms orthogonal;
4	block terms non-orthogonal, treatment terms non-orthogonal;
*	design unbalanced (ANOVA failed to analyse it).

The final code, *, occurs only with ANOVA. AKEEP will be unavailable if ANOVA has failed.

4.7.6 Screening tests for unbalanced designs

ASCREEN procedure

Performs screening tests for designs with orthogonal block structure (R.W. Payne).

Options

PRINT = *string tokens*	Which tests to print (conditional, marginal, efficiency); default cond, marg
FACTORIAL = *scalar*	Limit on the number of factors in each treatment term; default 3
EXCLUDEHIGHER = *string token*	Whether to exclude higher-order interactions in the initial model for the conditional test of each term (yes, no); default no
FORCED = *formula*	Terms that must be included (together with any covariates) in the initial models for every term; default * i.e. none

Parameter

Y = *variates*	Variates to be analysed

ASCREEN can be used to assess the treatment terms in an analysis of variance when the design is unbalanced but its error terms that are all orthogonal to one another. This includes any design with a hierarchical block structure, for example

```
Blocks / Plots
```

or

```
Replicates / Wholeplots / Subplots
```

ASCREEN thus provides a way of testing treatment terms in designs that cannot be analysed by ANOVA. Once ASCREEN has been used to decided which terms need to be included in the treatment model, the treatment effects and means can be estimated using REML (Chapter 5).

Before using ASCREEN, the block and treatment models for the design must be defined by the BLOCKSTRUCTURE and TREATMENTSTRUCTURE directives, in exactly the same way as for an analysis by ANOVA. As in ANOVA, the FACTORIAL option sets a limit on number of factors in each treatment term (default 3). You can also define covariates using the COVARIATE directive. The y-variate is specified by the Y parameter of ASCREEN.

ASCREEN forms marginal and conditional tests for the treatment terms like those produced by the RSCREEN procedure (3.2.9). These are produced for the analysis of each stratum of the design

(i.e. for the variation associated with each error term). The PRINT option has settings conditional and marginal to control which tests are produced if there is more than one error term; by default both are printed. However, if there is only one error term, ASCREEN uses procedure RSCREEN, which always prints both. There is also a setting, efficiency, which prints the minimum, maximum and harmonic mean efficiency factor of the terms in each of the strata if there is more than one. These efficiency factors show the amount of information available to construct the marginal test for each of the terms in the strata where it can be estimated. The harmonic mean is presented, rather than an ordinary average, as this corresponds to the average variance of differences amongst the effects of the term (remember that the variance is proportional to the reciprocal of the efficiency factor).

In a marginal test, each term is assessed by adding it to the simplest possible model. So, with a treatment model of

```
A + B + C + D + A.B + A.C + A.D + B.C + C.D + A.B.C + A.B.D +
A.C.D + B.C.D + A.B.C.D
```

the main effect of A is added it to the null model, while the interaction term A.B is added to a model containing only the main effects of A and B.

In a conditional test, each term is added to the most complex possible model. So the main effect A is added to an initial model excluding any term that has A as one of its margins. A is a margin of any term that contains A as one of its factors. So the terms to exclude for A are A.B, A.C, A.D, A.B.C, A.B.D, A.C.D and A.B.C.D. Similarly the interaction A.B is added to a model excluding any term that has A.B as a margin; i.e. any term that contains A and B amongst its factors. So A.B.C, A.B.D and A.B.C.D are excluded with A.B. The other terms to be included in the initial model depend on the setting of the EXCLUDEHIGHER option. With the default setting of no, all other terms are included in the initial model. So, the initial model for A would be

```
B + C + D + B.C + C.D + B.C.D
```

Alternatively, if EXCLUDEHIGHER=yes, the initial model contains only terms with no more factors than the term being tested. So, the initial model for A would be

```
B + C + D
```

The FORCED option allows you to specify a model formula with terms that must be included in the initial model for the conditional and marginal tests of every treatment term. The forced model automatically includes any covariates.

Example 4.7.6 continues the analysis of Example 4.7.1, reinstating the original treatment formula. As the treatments are orthogonal to each other, the first ASCREEN analysis generates the marginal and conditional variance ratios are identical (and are the same as those in the analysis-of-variance table in Example 4.7.1a). Only blocks 1, 3, 5 and 7 are used for the second analysis, so that the treatments become mutually non-orthogonal. As there are no residual degrees of freedom in the Blocks stratum, no tests are made.

Example 4.7.6

```
 37  TREATMENTSTRUCTURE N * K * D
 38  ASCREEN  Yield

Screening tests for designs with orthogonal block structure
===========================================================

Y-variate: Yield

Blocks stratum
--------------

Term   Marginal v.r.  d.f.    pr. Conditional v.r.  d.f.    pr.
N.K             3.02     1  0.180             3.02     1  0.180
N.D             1.07     1  0.377             1.07     1  0.377
```

```
K.D             9.91     1  0.051             9.91     1  0.051
N.K.D           0.44     1  0.556             0.44     1  0.556

Residual sum of squares: 774.1
Residual degrees of freedom: 3

Blocks.Plots stratum
--------------------

Term   Marginal v.r.  d.f.    pr. Conditional v.r.  d.f.    pr.
N             10.86     1  0.004            10.86     1  0.004
K            505.21     1 <0.001           505.21     1 <0.001
D            873.99     1 <0.001           873.99     1 <0.001
N.K            0.09     1  0.770             0.09     1  0.770
N.D            5.65     1  0.029             5.65     1  0.029
K.D           36.14     1 <0.001            36.14     1 <0.001
N.K.D          0.14     1  0.711             0.14     1  0.711

Residual sum of squares: 5423
Residual degrees of freedom: 17

  39  " With only blocks 1, 3, 5 & 7, the treatments become non-orthogonal."
  40  RESTRICT Yield; Blocks.IN.!(1,3,5,7)
  41  ASCREEN  Yield

Screening tests for designs with orthogonal block structure
===========================================================

Y-variate: Yield

Blocks stratum
--------------

Residual sum of squares: 0
Residual degrees of freedom: 0

Blocks.Plots stratum
--------------------

Term   Marginal v.r.  d.f.    pr. Conditional v.r.  d.f.    pr.
N             28.96     1  0.002             3.81     1  0.099
K             95.80     1 <0.001           244.20     1 <0.001
D            293.13     1 <0.001           381.44     1 <0.001
N.K           33.69     1  0.001             0.07     1  0.801
N.D            0.34     1  0.583             2.22     1  0.187
K.D           43.20     1 <0.001            10.41     1  0.018

Residual sum of squares: 1716
Residual degrees of freedom: 6
```

4.8 Unbalanced designs

The ANOVA directive analyses only balanced designs or, more accurately, only designs with *first-order balance*, as explained in Section 4.7.2. However, you do not need to master this as ANOVA itself detects when a design is unbalanced, and gives a failure diagnostic. If this happens with a design with a single error term, you can analyse it instead using the AUNBALANCED procedure which carries out analysis of variance using the Genstat regression facilities.

Unbalanced designs with several error terms should be analysed using the commands for REML analysis of linear mixed models (Chapter 5). However, if the additional random terms contain very little information about the treatments, it may be more convenient (and equally effective) to treat these as fixed nuisance terms, and use AUNBALANCED. Decisions like this can be made using the AOVANYHOW procedure, described in Section 4.8.7. Finally, if your design is detected as being unbalanced but you feel that is should be balanced, you can use the AN1ADVICE procedure to see if this may have been caused by an error in the data (4.8.8).

4.8.1 The AUNBALANCED procedure

AUNBALANCED procedure

Performs analysis of variance for unbalanced designs (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output from the analysis (aovtable, effects, means, residuals, screen, %cv); default aovt, mean
FACTORIAL = *scalar*	Limit on number of factors in a treatment term; default 3
PFACTORIAL = *scalar*	Limit on number of factors in printed tables of predicted means; default 3
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality, vertical, df, inflation); default * i.e. none
FPROBABILITY = *string token*	Printing of probabilities for variance ratios in the analysis-of-variance table (yes, no); default no
TPROBABILITY = *string token*	Printing of probabilities for t-tests of effects (yes, no); default no
PLOT = *string tokens*	Which residual plots to provide (fittedvalues, normal, halfnormal, histogram); default * i.e. none
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
WEIGHTS = *variate*	Weights for each unit; default * i.e. all units with weight one
PSE = *string tokens*	Types of standard errors to be printed with the predicted means (differences, alldifferences, lsd, alllsd, means, ese); default diff
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
RMETHOD = *string token*	Type of residuals to plot (simple, standardized); default simp

Parameters

Y = *variates*	Data values to be analysed
RESIDUALS = *variates*	Variate to save the residuals from each analysis
FITTEDVALUES = *variates*	Variate to save the fitted values from each analysis
SAVE = *identifiers*	To save details of each analysis to use subsequently with the AUDISPLAY procedure

The use of AUNBALANCED is similar to ANOVA (4.1.2). The treatment terms to be fitted must be specified, before calling the procedure, by the TREATMENTSTRUCTURE directive (4.1.1). You can specify covariates to be included in the analysis, using the COVARIATE directive (4.3). AUNBALANCED will also take account of any blocking structure specified by the BLOCKSTRUCTURE directive (4.2), but it does not use this to generate a "stratified analysis" with several error terms like those produced by ANOVA, but merely treats the blocking terms as "nuisance" terms to be removed in the analysis before assessing the treatment terms; see Example 4.8.1b.

The parameters of the procedure are identical to those of ANOVA. The variates to be analysed are specified by the Y parameter. If the Y variate is restricted, only the units not excluded by the restriction will be analysed. Residuals and fitted values can be saved using the RESIDUALS and FITTEDVALUES parameters respectively. Finally, the SAVE parameter allows details of the analysis to be saved so that further output can be obtained using the AUDISPLAY procedure. (Note that this is a regression save structure, not an ANOVA structure, so it cannot be used with the directives ADISPLAY or AKEEP.)

Printed output is controlled by the PRINT option, with settings: aovtable to print the analysis-of-variance table, effects to print the effects (as estimated by Genstat regression; see Section 3.3.3), means to print tables of predicted means with standard errors, residuals to print residuals and fitted values, screen to print "screening" tests for treatment terms, and %cv to print the coefficient of variation. The default is to print the analysis-of-variance table and tables of means.

The FACTORIAL option, as in ANOVA, sets a limit on the number of factors that a higher-order term, such as an interaction, can contain; any terms with more factors are deleted from the analysis. Similarly, the PFACTORIAL option limits the number of factors in terms for which predicted means are printed. The WEIGHTS option allows a variate of weights to be specified for a weighted analysis of variance. Probabilities can be printed for variance ratios by setting option FPROBABILITY=yes, and probabilities for t-tests of effects by setting option TPROBABILITY=yes. The NOMESSAGE option allows various warning messages (produced by the FIT directive) to be suppressed, and the PLOT option allows various residual plots to be requested: fittedvalues for a plot of residuals against fitted values, normal for a Normal plot, halfnormal for a half Normal plot, and histogram for a histogram of residuals. By default, simple residuals are plotted, but you can set option RMETHOD=standardized to plot standardized residuals instead.

Tables of means are calculated using the PREDICT directive (see Section 3.3.4). These are illustrated in Example 4.8.2 below. The first step (A) of the calculation forms the full table of predictions, classified by every factor in the model. The second step (B) averages the full table over the factors that do not occur in the table of means. The COMBINATIONS option specifies which cells of the full table are to be formed in Step A. The default setting, estimable, fills in all the cells other than those that involve parameters that cannot be estimated, for example because of aliasing. Alternatively, setting COMBINATIONS=present excludes the cells for factor combinations that do not occur in the data. The ADJUSTMENT option then defines how the averaging is done in Step B. The default setting, marginal, forms a table of marginal weights for each factor, containing the proportion of observations with each of its levels; the full table of weights is then formed from the product of the marginal tables. The setting equal weights all the combinations equally. Finally, the setting observed uses the WEIGHTS option of PREDICT to weight each factor combination according to its own individual replication in the data.

The PSE option controls the types of standard errors that are produced to accompany the tables of means, with settings:

differences	summary of standard errors for differences between pairs of means;
alldifferences	standard errors for differences between all pairs of means;
lsd	summary of least significant differences between pairs of means;
alllsd	least significant differences between all pairs of means;
means	standard errors of the means (relevant for comparing them with zero);
ese	approximate effective standard errors – these are formed by procedure SED2ESE with the aim of allowing good

approximations to the standard errors for differences to be calculated by the usual formula of $sed_{i,j}$ = SQRT(ese_i^2 + ese_j^2).

The default is `differences`. The `LSDLEVEL` option sets the significance level (as a percentage) for the least significant differences.

Example 4.8.1a analyses results from an experiment to study the effects of factors `A`, `B` and `C` on the yield `Y` of a production process. The intention was originally to run the experiment in two separate days, and to have two observations of each treatment combination on each day. However, due to time constraints, there were several combinations (chosen at random) in each of the days that could only be performed once. As a result of this unequal replication the design is unbalanced and, if we attempt to analyse it by `ANOVA`, we obtain a fault message reporting that the design is unbalanced (see the start of Example 4.8.1a). So instead, in line 6, we use `AUNBALANCED`.

Example 4.8.1a

```
   2  FILEREAD [PRINT=summary; NAME='product.dat'] Day,A,B,C,Y;\
   3    FGROUP=4(yes),no

Summary
-------

The file product.dat is assumed to contain 5 structure(s), with one value for
each structure on each record.

The file contains 67 values for each of the following structures:

  Identifier      Type   Missing
         Day    factor         0
           A    factor         0
           B    factor         0
           C    factor         0
           Y   variate         0


   4  TREATMENTSTRUCTURE Day+A*B*C
   5  ANOVA Y

******** Fault 1, code AN 1, statement 1 on line 5

Command: ANOVA Y
Design unbalanced - cannot be analysed by ANOVA
Model term A.B (non-orthogonal to term Day) is unbalanced.

   6  AUNBALANCED [PRINT=aov; FPROBABILITY=yes] Y

Analysis of an unbalanced design using Genstat regression
=========================================================

Variate: Y


Accumulated analysis of variance
--------------------------------

Change                      d.f.         s.s.         m.s.      v.r.  F pr.
+ Day                          1        914.0        914.0      3.67  0.061
+ A                            2       1706.8        853.4      3.42  0.041
+ B                            2        418.8        209.4      0.84  0.438
+ C                            1       1065.9       1065.9      4.28  0.044
+ A.B                          4       1166.0        291.5      1.17  0.336
+ A.C                          2       2456.7       1228.3      4.93  0.011
+ B.C                          2        284.4        142.2      0.57  0.569
+ A.B.C                        4       1397.4        349.4      1.40  0.248
Residual                      48      11960.4        249.2
```

```
Total                          66      21370.4        323.8
```

AUNBALANCED uses the Genstat regression directives to fit the model, and so it produces an accumulated analysis-of-variance, like those in Sections 3.2 and 3.3, indicating the order in which the terms were fitted. The order here is determined by the order in the TREATMENTSTRUCTURE directive. We have specified the term Day first there because this is a *nuisance* term, reflecting random variability which we want to eliminate before we assess the treatments. The +A line then gives the (main) effect of A after eliminating Day. The +B line gives the main effect of B, eliminating Day and A, and so on. Each line in the table presents the effect of a particular term, eliminating the terms in the lines above, but ignoring the terms in the lines below. (This is technically true also for the analysis-of-variance tables produced by ANOVA, but generally the treatment terms in balanced designs are orthogonal, and the order of fitting does not matter; see Section 4.7.4 for more details.) Here the treatment terms are non-orthogonal, and we may obtain different sums of squares if they are fitted in a different order: for example if we change the order of the treatment factors to be C*A*B, the sums of squares for A, B and C will be 1699.1, 429.4 and 1063.0 respectively (see Example 4.8.1b). These results would lead to the same conclusions to those from Example 4.8.1a (namely that there are main effects of A and C, and an A by C interaction), but in a design with a greater degree of non-orthogonality you would be well advised to investigate several orderings.

Alternatively, AUNBALANCED can print screening tests (produced using the procedure RSCREEN, described in Section 3.2.9) which are based on the two most relevant orderings for each term. Marginal tests assess the effect of adding each term to the simplest possible model: so, here Day, A, B and C are added to a model that contains no other terms, A.B is added to a model containing only A and B (as an interaction cannot be fitted before its main effects), and so on. Conditional tests assess the effect of adding each term to the fullest possible model (i.e. a model containing all terms other than those to which the term is marginal): so, for example, A is added to a model containing Day, B, C and B.C (that is, all the terms except the interactions involving A).

If there are block terms or covariates, these are fitted (in that order) before the treatment terms. Any block terms and covariates are also included in the models to which terms are added for the marginal tests (as well as in those for the conditional tests). As already mentioned, Day here is a nuisance term. By specifying this in the BLOCKSTRUCTURE statement in line 7 of Example 4.8.1b, we ensure that it is removed before any testing of the treatment terms. No screening tests are now done for Day, and the marginal tests for A, B and C now assess the effect of adding these terms to a model that contains only Day, the marginal test for A.B assesses the effect of adding A.B to a model that contains Day, A and B, and so on. Example 4.8.1b also shows the effect of specifying the treatment terms in a different order, C*A*B instead of A*B*C.

Example 4.8.1b

```
   7  BLOCKSTRUCTURE Day
   8  TREATMENTSTRUCTURE C*A*B
   9  AUNBALANCED [PRINT=aov,screen; FPROBABILITY=yes] Y

Screening of terms in an unbalanced design
==========================================

Variate: Y

Marginal and conditional test statistics and degrees of freedom
---------------------------------------------------------------

    degrees of freedom for denominator (full model): 48
```

```
 term     mtest  mdf     ctest  cdf
    C      4.27    1      4.78    1
    A      3.42    2      3.47    2
    B      0.76    2      0.84    2

 term     mtest  mdf     ctest  cdf
  C.A      5.25    2      4.81    2
  C.B      0.71    2      0.57    2
  A.B      1.04    4      1.00    4

 term     mtest  mdf     ctest  cdf
C.A.B      1.40    4      1.40    4


P-values of marginal and conditional tests
------------------------------------------

 term     mprob     cprob
    C     0.044     0.034
    A     0.041     0.039
    B     0.474     0.439

 term     mprob     cprob
  C.A     0.009     0.013
  C.B     0.498     0.569
  A.B     0.395     0.415

 term     mprob     cprob
C.A.B     0.248     0.248


Analysis of an unbalanced design using Genstat regression
=========================================================

Variate: Y


Accumulated analysis of variance
--------------------------------

Change                       d.f.         s.s.         m.s.      v.r.  F pr.
+ Day                           1        914.0        914.0      3.67  0.061
+ C                             1       1063.0       1063.0      4.27  0.044
+ A                             2       1699.1        849.6      3.41  0.041
+ B                             2        429.4        214.7      0.86  0.429
+ C.A                           2       2605.7       1302.9      5.23  0.009
+ C.B                           2        301.9        151.0      0.61  0.550
+ A.B                           4        999.4        249.9      1.00  0.415
+ C.A.B                         4       1397.4        349.4      1.40  0.248
Residual                       48      11960.4        249.2

Total                          66      21370.4        323.8
```

4.8.2 The AUDISPLAY procedure

AUDISPLAY procedure

Produces further output for an unbalanced design (after AUNBALANCED) (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output from the analysis (aovtable, effects, means, residuals, %cv); default aovt, mean
PFACTORIAL = *scalar*	Limit on number of factors in printed tables of predicted means; default 3
FPROBABILITY = *string token*	Printing of probabilities for variance ratios in the analysis-of-variance table (yes, no); default no

TPROBABILITY = *string token*	Printing of probabilities for t-tests of effects (yes, no); default no
PLOT = *string tokens*	Which residual plots to provide (fittedvalues, normal, halfnormal, histogram); default * i.e. none
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
PSE = *string tokens*	Types of standard errors to be printed with the predicted means (differences, alldifferences, lsd, alllsd, means, ese); default diff
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
RMETHOD = *string token*	Type of residuals to plot (simple, standardized); default simp
PMEANTERMS = *formula*	Treatment terms for which predicted means are to be printed; default * implies all the treatment terms

Parameter

SAVE = *identifiers*	Save structure (from AUNBALANCED) containing details of the analysis for which further output is required; if omitted, output is from the most recent use of AUNBALANCED

Procedure AUDISPLAY can be used to produce further output for an unbalanced design. It has options PRINT, FPROBABILITY, TPROBABILITY, COMBINATIONS, ADJUSTMENT, PSE, LSDLEVEL and RMETHOD like those of AUNBALANCED (4.8.1), except that no screening tests are available. It has a SAVE parameter, like that of ADISPLAY, which can be set to the save structure from the analysis for which further output is required, but the structure here is a *regression* save structure, not an ANOVA save structure. If SAVE is not set, output will be produced for the most recent analysis from AUNBALANCED; however, none of the Genstat regression directives (MODEL, TERMS, FIT, ADD, DROP and so on) must then have been used in the interim. Also there is an option PMEANTERMS, which can be used to specify the treatment terms for which predicted means are to be printed; by default, they are printed for all the treatment terms (subject, of course, to the PFACTORIAL option).

In Example 4.8.2, which continues Example 4.8.1b, we use AUDISPLAY to print tables of means, all the standard errors of differences and approximate effective standard errors. The discrepancies between the true standard errors of differences and those calculated from the approximate effective standard errors are very small for this example – the maximum is only 0.25%. You can produce multiple comparisons for means from unbalanced analyses by using the AUMCOMPARISON procedure; this has similar options and parameters to the AMCOMPARISON procedure, described in 4.1.9.

Example 4.8.2

```
  10  AUDISPLAY [PRINT=means; PSE=differences,alldifferences,ese]

Analysis of an unbalanced design using Genstat regression
=========================================================

Variate: Y
```

```
Predictions from regression model
---------------------------------

Response variate: Y

              Prediction
            C
            1       110.6
            2       102.4

Standard error of differences between predicted means        3.903


Approximate effective standard errors
-------------------------------------

    C
    1       3.903
    2           *


Discrepancy between sed and value calculated from ese's
-------------------------------------------------------


Maximum discrepancy              *
Maximum % discrepancy            *


Predictions from regression model
---------------------------------

Response variate: Y

              Prediction
            A
            1       113.2
            2       101.2
            3       105.3


Approximate effective standard errors
-------------------------------------

    A
    1       3.391
    2       3.224
    3       3.550


Discrepancy between sed and value calculated from ese's
-------------------------------------------------------

Maximum discrepancy              0
Maximum % discrepancy         0.00


Standard errors of differences between pairs of predicted means
---------------------------------------------------------------

Rows and columns are labelled by the labels/levels of the factors:

        A  1   1           *
        A  2   2       4.679           *
        A  3   3       4.909       4.796           *
                           1           2           3

Minimum standard error of differences        4.679
Average standard error of differences        4.795
Maximum standard error of differences        4.909
```

```
Predictions from regression model
---------------------------------

Response variate: Y

             Prediction
          B
          1       103.2
          2       108.1
          3       108.3


Approximate effective standard errors
-------------------------------------

    B
    1       3.228
    2       3.450
    3       3.475


Discrepancy between sed and value calculated from ese's
-------------------------------------------------------

Maximum discrepancy             0
Maximum % discrepancy        0.00


Standard errors of differences between pairs of predicted means
---------------------------------------------------------------

Rows and columns are labelled by the labels/levels of the factors:

       B  1   1           *
       B  2   2       4.724           *
       B  3   3       4.742       4.896           *
                          1           2           3

Minimum standard error of differences        4.724
Average standard error of differences        4.788
Maximum standard error of differences        4.896


Predictions from regression model
---------------------------------

Response variate: Y

             Prediction
          A           1           2           3
          C
          1       125.9       101.7       104.6
          2       100.9       100.7       105.9


Approximate effective standard errors
-------------------------------------

    A           1           2           3
    C
    1       4.786       4.563       5.248
    2       4.786       4.563       4.790


Discrepancy between sed and value calculated from ese's
-------------------------------------------------------

Maximum discrepancy       0.005482
Maximum % discrepancy         0.08
```

```
Standard errors of differences between pairs of predicted means
---------------------------------------------------------------

Rows and columns are labelled by the labels/levels of the factors:

   C 1  A 1   1        *
   C 1  A 2   2    6.614        *
   C 1  A 3   3    7.103    6.955        *
   C 2  A 1   4    6.763    6.614    7.103        *
   C 2  A 2   5    6.614    6.454    6.955    6.614
   C 2  A 3   6    6.775    6.614    7.103    6.775    6.614        *
                       1        2        3        4        5        6

Minimum standard error of differences        6.454
Average standard error of differences        6.778
Maximum standard error of differences        7.103


Predictions from regression model
---------------------------------

Response variate: Y

             Prediction
          B           1           2           3
          C
          1       110.2       111.9       109.7
          2        96.5       104.5       106.9


Approximate effective standard errors
-------------------------------------

    B           1           2           3
    C
    1       4.564       5.191       4.808
    2       4.564       4.564       5.011


Discrepancy between sed and value calculated from ese's
-------------------------------------------------------

Maximum discrepancy      0.0001776
Maximum % discrepancy         0.00


Standard errors of differences between pairs of predicted means
---------------------------------------------------------------

Rows and columns are labelled by the labels/levels of the factors:

   C 1  B 1   1        *
   C 1  B 2   2    6.912        *
   C 1  B 3   3    6.629    7.075        *
   C 2  B 1   4    6.454    6.912    6.629        *
   C 2  B 2   5    6.454    6.912    6.629    6.454
   C 2  B 3   6    6.778    7.215    6.944    6.778    6.778        *
                       1        2        3        4        5        6

Minimum standard error of differences        6.454
Average standard error of differences        6.770
Maximum standard error of differences        7.215


Predictions from regression model
---------------------------------

Response variate: Y
```

```
                Prediction
          B           1           2           3
          A
          1       115.2       112.3       111.8
          2        97.9        99.9       106.4
          3        96.7       113.2       106.8


Approximate effective standard errors
-------------------------------------

    B           1           2           3
    A
    1       5.581       5.581       6.482
    2       5.581       5.581       5.581
    3       5.581       6.801       6.055


Discrepancy between sed and value calculated from ese's
-------------------------------------------------------

Maximum discrepancy        0.01798
Maximum % discrepancy         0.20


Standard errors of differences between pairs of predicted means
---------------------------------------------------------------

Rows and columns are labelled by the labels/levels of the factors:

   A 1  B 1   1         *
   A 1  B 2   2     7.894         *
   A 1  B 3   3     8.551     8.551         *
   A 2  B 1   4     7.894     7.894     8.551         *
   A 2  B 2   5     7.894     7.894     8.551     7.894         *
   A 2  B 3   6     7.894     7.894     8.551     7.894     7.894
   A 3  B 1   7     7.894     7.894     8.551     7.894     7.894
   A 3  B 2   8     8.799     8.799     9.393     8.799     8.799
   A 3  B 3   9     8.232     8.232     8.888     8.232     8.232
                        1         2         3         4         5


   A 2  B 3   6     *
   A 3  B 1   7     7.894         *
   A 3  B 2   8     8.799     8.799         *
   A 3  B 3   9     8.232     8.232     9.104         *
                        6         7         8         9


Minimum standard error of differences        7.894
Average standard error of differences        8.313
Maximum standard error of differences        9.393


Predictions from regression model
---------------------------------

Response variate: Y

                           Prediction
                       B           1           2           3
          C            A
          1            1       136.1       124.1       116.2
                       2       102.1       101.8       101.3
                       3        92.3       110.6       112.6
          2            1        95.1       100.8       107.6
                       2        93.8        98.1       111.3
                       3       101.1       115.8       101.2


Approximate effective standard errors
-------------------------------------
```

B		1	2	3
C	A			
1	1	7.892	7.892	9.136
	2	7.892	7.892	7.892
	3	7.892	11.162	7.892
2	1	7.892	7.892	9.136
	2	7.892	7.892	7.892
	3	7.892	7.892	9.142

Discrepancy between sed and value calculated from ese's

Maximum discrepancy 0.03227
Maximum % discrepancy 0.25

Standard errors of differences between pairs of predicted means

Rows and columns are labelled by the labels/levels of the factors:

C 1	A 1	B 1	1	*			
C 1	A 1	B 2	2	11.16	*		
C 1	A 1	B 3	3	12.07	12.07	*	
C 1	A 2	B 1	4	11.16	11.16	12.07	*
C 1	A 2	B 2	5	11.16	11.16	12.07	11.16
C 1	A 2	B 3	6	11.16	11.16	12.07	11.16
C 1	A 3	B 1	7	11.16	11.16	12.07	11.16
C 1	A 3	B 2	8	13.67	13.67	14.42	13.67
C 1	A 3	B 3	9	11.16	11.16	12.07	11.16
C 2	A 1	B 1	10	11.16	11.16	12.07	11.16
C 2	A 1	B 2	11	11.16	11.16	12.07	11.16
C 2	A 1	B 3	12	12.07	12.07	12.89	12.07
C 2	A 2	B 1	13	11.16	11.16	12.07	11.16
C 2	A 2	B 2	14	11.16	11.16	12.07	11.16
C 2	A 2	B 3	15	11.16	11.16	12.07	11.16
C 2	A 3	B 1	16	11.16	11.16	12.07	11.16
C 2	A 3	B 2	17	11.16	11.16	12.07	11.16
C 2	A 3	B 3	18	12.07	12.07	12.95	12.07
				1	2	3	4

C 1	A 2	B 2	5	*			
C 1	A 2	B 3	6	11.16	*		
C 1	A 3	B 1	7	11.16	11.16	*	
C 1	A 3	B 2	8	13.67	13.67	13.67	*
C 1	A 3	B 3	9	11.16	11.16	11.16	13.67
C 2	A 1	B 1	10	11.16	11.16	11.16	13.67
C 2	A 1	B 2	11	11.16	11.16	11.16	13.67
C 2	A 1	B 3	12	12.07	12.07	12.07	14.42
C 2	A 2	B 1	13	11.16	11.16	11.16	13.67
C 2	A 2	B 2	14	11.16	11.16	11.16	13.67
C 2	A 2	B 3	15	11.16	11.16	11.16	13.67
C 2	A 3	B 1	16	11.16	11.16	11.16	13.67
C 2	A 3	B 2	17	11.16	11.16	11.16	13.67
C 2	A 3	B 3	18	12.07	12.07	12.07	14.42
				5	6	7	8

C 1	A 3	B 3	9	*			
C 2	A 1	B 1	10	11.16	*		
C 2	A 1	B 2	11	11.16	11.16	*	
C 2	A 1	B 3	12	12.07	12.07	12.07	*
C 2	A 2	B 1	13	11.16	11.16	11.16	12.07
C 2	A 2	B 2	14	11.16	11.16	11.16	12.07
C 2	A 2	B 3	15	11.16	11.16	11.16	12.07
C 2	A 3	B 1	16	11.16	11.16	11.16	12.07
C 2	A 3	B 2	17	11.16	11.16	11.16	12.07
C 2	A 3	B 3	18	12.07	12.07	12.07	12.95
				9	10	11	12

```
   C 2  A 2  B 1   13        *
   C 2  A 2  B 2   14    11.16        *
   C 2  A 2  B 3   15    11.16    11.16        *
   C 2  A 3  B 1   16    11.16    11.16    11.16        *
   C 2  A 3  B 2   17    11.16    11.16    11.16    11.16
   C 2  A 3  B 3   18    12.07    12.07    12.07    12.07
                            13       14       15       16


   C 2  A 3  B 2   17        *
   C 2  A 3  B 3   18    12.07        *
                            17       18

Minimum standard error of differences        11.16
Average standard error of differences        11.74
Maximum standard error of differences        14.42
```

4.8.3 The AUGRAPH procedure

AUGRAPH procedure

Plots tables of means from AUNBALANCED (R.W. Payne).

Options

GRAPHICS = *string token*	Type of graph (highresolution, lineprinter); default high
METHOD = *string token*	What to plot (means, lines, data, barchart, splines); default mean
XFREPRESENTATION = *string token*	How to label the *x*-axis (levels, labels); default labels uses the XFACTOR labels, if available
PSE = *string token*	What to plot to represent variation (differences, lsd, means, allmeans); default diff
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
LSDLEVEL = *scalar*	Significance level (%) to use for least significant differences; default 5
DFSPLINE = *scalar*	Number of degrees of freedom to use when METHOD=splines
YTRANSFORM = *string tokens*	Transformed scale for additional axis marks and labels to be plotted on the right-hand side of the y-axis (identity, log, log10, logit, probit, cloglog, square, exp, exp10, ilogit, iprobit, icloglog, root); default iden i.e. none
PENYTRANSFORM = *scalar*	Pen to use to plot the transformed axis marks and labels; default * selects a pen, and defines its properties, automatically
KEYMETHOD = *string token*	What to use for the key descriptions when GROUPS specifies more than one factor (labels, namesandlabels); default name
PLOTTITLEMETHOD = *string token*	What to use for the titles of the plots when TRELLISGROUPS specifies more than one factor (labels, namesandlabels); default name
PAGETITLEMETHOD = *string token*	What to use for the titles of the pages when PAGEGROUPS specifies more than one factor (labels,

	namesandlabels); default name
USEAXES = *string token*	Which aspects of the current axis definitions of window 1 to use (none, limits, marks, mpositions, nsubticks,); default none
SAVE = *regression save structure*	Save structure to provide the table of means; default uses the save structure from the most recent AUNBALANCED analysis (provided no other regression analysis has been done in the interim)

Parameters

XFACTOR = *factors*	Factor providing the *x*-values for each plot
GROUPS = *factors* or *pointers*	Factor or factors identifying groups of points in each plot; by default chosen automatically
TRELLISGROUPS = *factors* or *pointers*	Factor or factors specifying the different plots of a trellis plot of a multi-way table
PAGEGROUPS = *factors* or *pointers*	Factor or factors specifying plots to be displayed on different pages
NEWXLEVELS = *variates*	Values to be used for XFACTOR instead of its existing levels
TITLE = *texts*	Title for the graph; default defines a title automatically
YTITLE = *texts*	Title for the y-axis; default is to use the identifier of the y-variate, or to have no title if this is unnamed
XTITLE = *texts*	Title for the x-axis; default is to use the identifier of the XFACTOR
PENS = *variates*	Defines the pen to use to plot the points and/or line for each group defined by the GROUPS factors

AUGRAPH plots tables of predicted means from an analysis by AUNBALANCED. The SAVE option can be set to the save structure from the analysis from which the means should be taken. If SAVE is not set, the means will be from the most recent analysis by AUNBALANCED; however, none of the Genstat regression directives (MODEL, TERMS, FIT, ADD, DROP and so on) must then have been used in the interim.

In its simplest form, the behaviour of AUGRAPH depends on the model. If the treatment model contains only main effects, it plots the means for the first factor in the model. Otherwise it looks for the first treatment term involving two factors; it then plots the means with one of these factors as the x-axis, and the second as a grouping factor with levels identified by different plotting colours and symbols. The means are predicted by the AUKEEP procedure using the averaging and adjustment methods specified by the COMBINATIONS and ADJUSTMENT options; see AUKEEP (4.8.4) for details.

Usually, each mean is represented by a point. However, with high-resolution plots, the METHOD option can be set to lines to draw lines between the points, or data to draw just the lines and then also plot the original data values, or barchart to plot the means as a barchart, or splines to plot the points together with a smooth spline to show the trend over each group of points. The DFSPLINE specifies the degrees of freedom for the splines; if this is not set, 2 d.f. are used when there are up to 10 points, 3 if there are 11 to 20, and 4 for 21 or more. The GRAPHICS option controls whether a high-resolution or a line-printer graph is plotted; by default GRAPHICS=high.

The PSE option specifies the type of error bar to be plotted with the means, with settings:

differences	average standard error of difference;
lsd	average least significant difference;
means	average effective standard error for the means;

`allmeans` plots plus and minus the effective standard error around every mean.

The `LSDLEVEL` option sets the significance level (%) to use for the least significant differences (default 5). The `allmeans` setting is often unsuitable for plots other than barcharts when there are `GROUPS`, as the plus/minus e.s.e. bars may overlap each other.

You can define the table of means to plot explicitly, by specifying its classifying factors using the `XFACTOR`, `GROUPS`, `TRELLISGROUPS` and `PAGEGROUPS` parameters.

The `XFACTOR` parameter defines the factor against whose levels the means are plotted. With a multi-way table, there will be a plot of means against the `XFACTOR` levels for every combination of levels of the factors specified by the `GROUPS`, `TRELLISGROUPS` and `PAGEGROUPS` parameters. The `GROUPS` parameter specifies factors whose levels are to be included in a single window of the graph. So, for example, if you specify

```
AUGRAPH [METHOD=line] XFACTOR=A; GROUPS=B
```

`AUGRAPH` will produce plot the means in a single window with factor `A` on the x-axis, and a line for each level of the factor `B`. You can set `GROUPS` to a pointer to specify several factors to define groups. For example

```
POINTER [VALUES=B,C] Groupfactors
AUGRAPH [METHOD=line] XFACTOR=A;
```

plots a line for every combination of the levels of the factors `B` and `C`.

The `TRELLISGROUPS` option can specify one or more factors to define a trellis plot.

Figure 4.8.3 shows a plot of the means from Example 4.8.1a, generated by the command

```
AUGRAPH [METHOD=line] XFACTOR=A; GROUPS=B; TRELLISGROUPS=C
```

This produces a plot for each level of `C`, in a trellis arrangement; each plot has factor `A` on the x-axis, and a line for each level of the factor `B`.

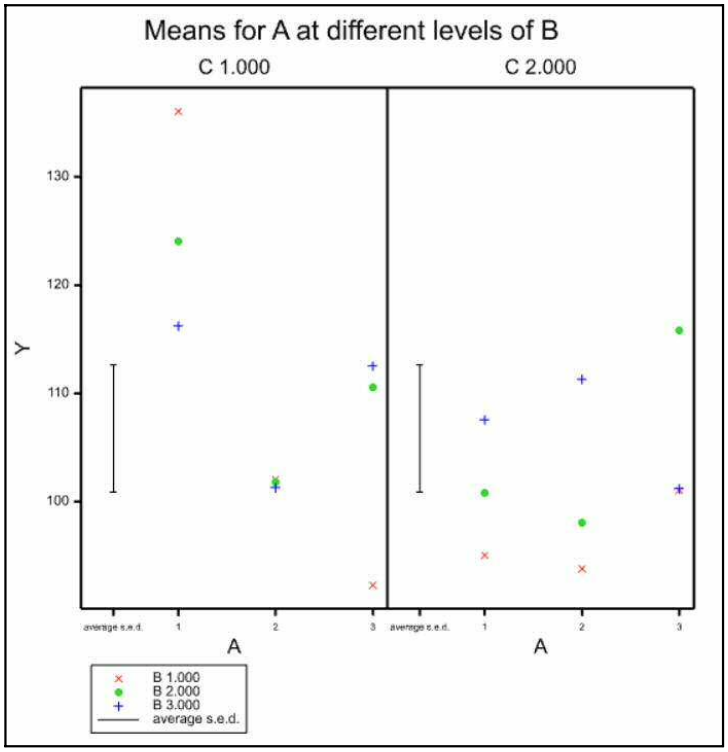


Figure 4.8.3

Similarly, the `PAGEGROUPS` parameter can specify factors whose combinations of levels are to be plotted on different pages. So

```
AUGRAPH [METHOD=line] XFACTOR=A; GROUPS=B; PAGEGROUPS=C
```

will produce a plot for each level of C, but now on separate pages. Multi-way tables can plotted even if the corresponding model term was not in the ANOVA analysis. For example you can plot a two-way table even if the analysis contained only the main effects of the two factors; however, the lines will then all be parallel and no standard errors or LSDs can be included.

The NEWXLEVELS parameter enables different levels to be supplied for XFACTOR if the existing levels are unsuitable. If XFACTOR has labels, these are used to label the x-axis unless you set option XFREPRESENTATION=levels.

The TITLE, YTITLE and XTITLE parameters can supply titles for the graph, the y-axis and the x-axis, respectively. The symbols, colours and line styles that are used in a high-resolution plot are usually set up by AUGRAPH automatically. If you want to control these yourself, you should use the PEN directive to define a pen with your preferred symbol, colour and line style, for each of the groups defined by combinations of the GROUPS factors. The pen numbers should then be supplied to AUGRAPH, in a variate with a value for each group, using the PENS parameter.

The YTRANSFORM option allows you to include additional axis markings, transformed onto another scale, on the right-hand side of the y-axis. Suppose, for example, suppose you have analysed a variate of percentages that have been transformed to logits. You might then set YTRANSFORM=ilogit (the inverse-logit transformation) to include markings in percentages alongside the logits. The settings are the same as those of the TRANSFORM parameter of AXIS,which is used to add the markings (1:6.9.7). You can control the colours of the transformed marks and labels, by defining a pen with the required properties, and specifying it with the PENYTRANSFORM option. Otherwise, the default is to plot them in blue.

When there is more than one GROUPS factor, the KEYMETHOD controls whether to use the factor names with their labels (or levels for factors with no labels) or just the labels (or levels) in the key descriptions. The default is to use the names and the labels (or levels). Similarly, the PLOTTITLEMETHOD specifies what to use for the titles of the plots when there is more then one TRELLISGROUPS factor, and the PAGETITLEMETHOD specifies what to use for the titles of the plots when there is more then one PAGEGROUPS factor. You can setKEYMETHOD=* to have no key at all.

The USEAXES option allows you to control various aspects of the axes. First you need to use the XAXIS and YAXIS directives to define them for window 1. Then specify which of the aspects of the axes in window 1 are to be used by DTABLE, by specifying USEAXES with the following settings:

limits	y- and x-axis limits (LOWER and UPPER parameters);
marks	location and labelling of the tick marks (MARKS, LABELS, LDIRECTION, LROTATION, DECIMALS, DREPRESENTATION, and VREPRESENTATION parameters);
mpositions	positions of the tick marks (MPOSITION parameter); and
nsubticks	number of subticks per interval (NSUBTICKS parameter).

By default none are used.

4.8.4 The AUKEEP procedure

AUKEEP procedure

Saves output from analysis of an unbalanced design (by AUNBALANCED) (R.W. Payne).

Options

FACTORIAL = *scalar*	Limit on number of factors in the model terms generated from the TERMS parameter; default 3

RESIDUALS = *variate*	To save residuals from the analysis
FITTEDVALUES = *variate*	To save fitted values
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
LSDLEVEL = *scalar*	Significance level (as a percentage) for the least significant differences
RMETHOD = *string token*	Type of residuals to form if the RESIDUALS option is set (simple, standardized); default simp
SAVE = *identifier*	Save structure (from AUNBALANCED) containing details of the analysis for which further output is required; if omitted, output is from the most recent use of AUNBALANCED

Parameters

TERMS = *formula*	Model terms for which information is required
MEANS = *table* or *pointer* to *tables*	Predicted means for each term
SEMEANS = *table* or *pointer* to *tables*	Standard errors of the means for each term
SEDMEANS = *symmetric matrix* or *pointer* to *symmetric matrices*	Standard errors of differences between means
ESEMEANS = *table* or *pointer* to *tables*	Approximate effective standard errors of the means: these are formed by procedure SED2ESE with the aim of allowing good approximations to the standard errors for differences to be calculated by the usual formula $sed_{i,j} = \sqrt{(ese_i^2 + ese_j^2)}$
LSD = *symmetric matrix* or *pointer* to *symmetric matrices*	Least significant differences

You can save output for the analysis of variance of an unbalanced design using procedure AUKEEP.

The RESIDUALS and FITTEDVALUES options allow variates to be specified to store the residuals and fitted values, respectively. The RMETHOD option controls whether simple or standardized residuals are saved; by default RMETHOD=simple.

The SAVE option can be set to the save structure from the analysis from which output is to be saved. If SAVE is not set, output will be produced for the most recent analysis from AUNBALANCED; however, none of the Genstat regression directives (MODEL, TERMS, FIT, ADD, DROP and so on) must then have been used in the interim. The COMBINATIONS, ADJUSTMENT and LSDLEVEL options operate as in AUNBALANCED.

The parameters of AUKEEP save information about particular model terms in the analysis. With the TERMS parameter you specify a model formula, which Genstat expands to form the series of model terms about which you wish to save information. As in AUNBALANCED, the FACTORIAL option sets a limit on the number of factors in each term. Any term containing more than that limit is deleted. The subsequent parameters allow you to specify identifiers of data structures to store various components of information for each of the terms that you have specified. The MEANS parameter saves tables of predicted means, the SEMEANS parameter saves tables of standard errors for the means, the SEDMEANS parameter saves symmetric matrices of standard errors of differences, the ESEMEANS parameter saves tables of approximate effective standard errors, and the LSD parameter saves symmetric matrices of least significant differences. If you

have a single term, you can supply a table or symmetric matrix for each of these parameters, as appropriate. However, if you have several terms, you must supply a pointer which will then be set up to contain as many tables or symmetric matrices as there are terms. The LSDLEVEL option sets the significance level (as a percentage) for the least significant differences.

So, following Example 4.8.2, we could save the A by C means in table ACmeans and their standard errors of differences in symmetric matrix ABsed, by

```
AUKEEP A.C; MEANS=ABmeans; SEDMEANS=ACsed
```

4.8.5 The AUPREDICT procedure

AUPREDICT procedure

Forms predictions from an unbalanced design (after AUNBALANCED) (R.W. Payne).

Options

PRINT = *string tokens*	What to print (description, predictions, se, sed, sedsummary, ese, lsd, lsdsummary, vcovariance); default pred, sed
MODEL = *formula*	Model to use to calculate the predictions; default * i.e. full model fitted by AUNBALANCED
FACTORIAL = *scalar*	Limit on number of factors or variates in each term specified by MODEL; default 3
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
PREDICTIONS = *tables* or *scalars*	Saves predictions; default *
SE = *tables* or *scalars*	Saves standard errors of predictions; default *
SED = *symmetric matrices*	Saves matrices of standard errors of differences between predictions; default *
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
VCOVARIANCE = *symmetric matrices*	Saves variance-covariance matrices of predictions; default *
SAVE = *identifier*	Save structure (from AUNBALANCED) containing details of the analysis for which predictions are required; if omitted, output is from the most recent use of AUNBALANCED

Parameters

CLASSIFY = *vectors*	Variates and/or factors to classify table of predictions
LEVELS = *variates* or *scalars*	To specify values of variates, levels of factors

AUPREDICT can produce predicted means following an analysis of variance by AUNBALANCED of an unbalanced design. The predictions are calculated using the PREDICT directive (see Section 3.3.4). The first step (A) of the calculation forms the full table of predictions, classified by every factor in the model. The second step (B) averages the full table over the factors that do not occur in the table of means. The COMBINATIONS option specifies which cells of the full table are to be formed in Step A. The default setting, estimable, fills in all the cells other than those that involve parameters that cannot be estimated, for example because of aliasing.

Alternatively, setting COMBINATIONS=present excludes the cells for factor combinations that do not occur in the data. The ADJUSTMENT option then defines how the averaging is done in Step B. The default setting, marginal, forms a table of marginal weights for each factor, containing the proportion of observations with each of its levels; the full table of weights is then formed from the product of the marginal tables. The setting equal weights all the combinations equally. Finally, the setting observed uses the WEIGHTS option of PREDICT to weight each factor combination according to its own individual replication in the data.

Printed output, which extends the output available from PREDICT, is controlled by settings of the PRINT option:

description	standardization policies used when forming the predictions,
predictions	predictions,
se	predictions and standard errors,
sed	standard errors for differences between the predictions,
sedsummary	summary of the standard errors for differences between the predictions,
lsd	least significant differences between the predictions,
lsdsummary	summary of the least significant differences between the predictions,
ese	approximate effective standard errors – these are formed by procedure SED2ESE with the aim of allowing good approximations to the standard errors for differences to be calculated by the usual formula of $sed_{i,j} = \sqrt{(ese_i^2 + ese_j^2)}$, and
vcovariance	variance and covariances of the predictions.

The default is to print predictions and a summary of the standard errors of differences. The standard errors (and sed's) are relevant for the predictions when considered as means of those data that have been analysed, with the means formed according to the averaging policy defined by the options of PREDICT. The word *prediction* is used because these are predictions of what the means would have been if the factor levels been replicated differently in the data; see Lane & Nelder (1982) for more details. The LSDLEVEL option specifies the significance level (%) to use in the calculation of least significant differences (default 5%).

Another extension in AUPREDICT is that you can produce predictions using a smaller model than the full model that has been fitted by AUNBALANCED. This can be useful if the full model contains many parameters. A substantial amount of time and computer workspace may then be needed to calculate the predictions and standard errors. Very large models may even exceed the capacity of some PCs. The model is specified by the MODEL option. The FACTORIAL option sets a limit on number of factors or variates in each term specified by MODEL; default 3.

You might choose to omit a term from the full model when forming a particular table of predictions if the term is orthogonal to all the terms involved in the table. For example, you might omit the term blocks when forming an A-by-B table of predictions if each combination of levels of the factors A and B is replicated the same number of times in every block. The justification is that an orthogonal term cannot affect the size of any of the differences between predictions. Different weighting of the levels of the orthogonal term may affect the overall mean of the predictions, but this is usually unimportant. If you omit the term, it is though you had included it with weightings based on the observed replication of its levels in the data set – and in any well-designed data set these should provide a satisfactory outcome. You might also omit a term if it is nearly orthogonal to the terms involved in the table, and you are happy to ignore its effect on the predictions. In Example 4.8.4, we produce an A by C table of predictions from the analysis in Example 4.8.1b, but ignoring the interaction A.B.C.

Example 4.8.5

```
  11  AUPREDICT [PRINT=predictions,sed; MODEL=Day+A*B*C-A.B.C] A,C

Predictions from regression model
---------------------------------

Response variate: Y

              Prediction
            C          1          2
            A
            1      126.1      100.6
            2      101.7      100.8
            3      105.3      106.4


Standard errors of differences between pairs of predicted means
---------------------------------------------------------------

Rows and columns are labelled by the labels/levels of the factors:


   A 1 C 1   1        *
   A 1 C 2   2    6.741        *
   A 2 C 1   3    6.608    6.605        *
   A 2 C 2   4    6.605    6.608    6.448        *
   A 3 C 1   5    7.031    7.001    6.871    6.870        *
   A 3 C 2   6    6.769    6.759    6.605    6.608    7.050        *
                     1        2        3        4        5        6
```

The PREDICTIONS, SE, SED, ESE, LSD and VCOVARIANCE options allow the results of the prediction to be save in appropriate Genstat data structures.

The SAVE option allows you to specify save structure from the analysis for which further output is required. If SAVE is not set, output will be produced for the most recent analysis from AUNBALANCED; however, none of the Genstat regression directives (MODEL, TERMS, FIT, ADD, DROP and so on) must then have been used in the interim.

4.8.6 The AUSPREADSHEET procedure

AUSPREADSHEET procedure

Saves results from an analysis of an unbalanced design (by AUNBALANCED) in a spreadsheet (R.W. Payne).

Options

MEANS = *pointer*	Pointer to tables to contain the treatment means; default means
SEMEANS = *pointer*	Pointer to tables to contain the standard errors of treatment means; default sem
SEDMEANS = *pointer*	Pointer to matrices to contain standard errors of differences of treatment means; default sed
ESEMEANS = *pointer*	Pointer to matrices to contain effective standard errors of treatment means; default ese
EFFECTS = *pointer*	Pointer to contain the estimated effects, their standard errors, t-statistics and probabilities; default effects
REPLICATIONS = *pointer*	Pointer to tables of treatment replications; default replication

RESIDUALS = *variate*	Variate to save the residuals in the fittedvalues page; default residuals
FITTEDVALUES = *variate*	Variate to save the fitted values in the fittedvalues page; default fittedvalues
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
AOVTABLE = *pointer*	Pointer to a text and variates containing the information in the analysis-of-variance table; default aovtable
RMETHOD = *string token*	Type of residuals to form (simple, standardized); default simp
LSDMEANS = *pointer*	Pointer to matrices to contain least significant differences for means
LSDLEVEL = *scalar*	Significance level (as a percentage) for the least significant differences; default 5
SPREADSHEET = *string tokens*	What to include in the spreadsheet (aovtable, effects, means, semeans, sedmeans, esemeans, lsdmeans, replications, fittedvalues); default aovt, mean, sedm, repl, fitt
OUTFILENAME = *text*	Name of Genstat workbook file (.gwb) or Excel (.xls or .xlsx) file to create
SAVE = *identifier*	Save structure (from AUNBALANCED) containing details of the analysis for which further output is required; if omitted, output is from the most recent use of AUNBALANCED

No parameters

AUSPREADSHEET puts results from the analysis of an unbalanced design into a spreadsheet. By default the results are from the most recent analysis by AUNBALANCED, but you use the SAVE option to specify the save structure from some other analysis.

The SPREADSHEET option specifies which pages of the spreadsheet to form, with settings:

aovtable	analysis of variance table,
effects	estimates of effects, with their standard errors, t-statistics and probabilities,
means	tables of treatment means,
semeans	tables of standard errors of treatment means,
sedmeans	symmetric matrices of standard errors of differences of treatment means,
esemeans	tables of effective standard errors of treatment means,
lsdmeans	matrices of least significant differences of treatment means,
replications	replication tables of treatment terms,
fittedvalues	y-variate, fitted values and residuals.

By default, SPREADSHEET = aovt, mean, sedm, repl, fitt.

Tables of means are obtained from the AUKEEP procedure, with the COMBINATIONS and ADJUSTMENT options operating as described in 4.8.5.

To help avoid clashes between the columns of the spreadsheets if you want to save results from more than one analysis, the parameters MEANS, SEMEANS, SEDMEANS, ESEMEANS, LSDMEANS, EFFECTS, REPLICATIONS, RESIDUALS, FITTEDVALUES and AOVTABLE allow you

to specify identifiers for the columns (or sets of columns) that will store the corresponding results in the current spreadsheet.

You can save the data in either a Genstat workbook (.gwb) or an Excel spreadsheet (.xls or .xlsx), by setting the OUTFILENAME option to the name of the file to create. If the name is specified without a suffix, '.gwb' is added (so that a Genstat workbook is saved). If OUTFILENAME is not specified, the data are put into a spreadsheet opened inside Genstat.

So, you could save the analysis-of-variance table, means and standard errors of differences of means in an Excel spreadsheet called Product.xlsx by giving the command

```
AUSPREADSHEET [SPREADSHEET=aovtable,means,sedmeans;\
              OUTFILE='Product.xlsx]
```

4.8.7 The AOVANYHOW procedure

AOVANYHOW procedure

Performs analysis of variance using ANOVA, regression or REML as appropriate (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output from the analysis (aovtable, information, means, residuals); default aovt, info, mean
METHOD = *string token*	Whether to complete the analysis or just form a recommendation (analyse, recommend); default anal
FACTORIAL = *scalar*	Limit on number of factors in a treatment term; default 3
FPROBABILITY = *string token*	Printing of probabilities for variance ratios in the analysis-of-variance table (yes, no); default no
PLOT = *string tokens*	Which residual plots to provide (fittedvalues, normal, halfnormal, histogram); default * i.e. none
COMBINATIONS = *string token*	Factor combinations for which to form predicted means (present, estimable); default esti
ADJUSTMENT = *string token*	Type of adjustment to be made when predicting means (marginal, equal, observed); default marg
WEIGHTS = *variate*	Weights for each unit; default * i.e. all units with weight one
PSE = *string tokens*	Types of standard errors to be printed with the predicted means (differences, alldifferences, lsd, alllsd, means; default diff
LSDLEVEL = *scalar*	Significance level (%) for least significant differences; default 5
EFLOSS = *scalar*	Maximum loss of efficiency occurring on any treatment contrast if the analysis is done by regression
EFLIMIT = *scalar*	Limit on the loss of efficiency for the analysis to be done by regression; default 0.1
EXIT = *scalar*	Exit code indicating the recommended method of analysis

Parameters

Y = *variates*	Data values to be analysed
RESIDUALS = *variates*	Variate to save the residuals from each analysis
FITTEDVALUES = *variates*	Variate to save the fitted values from each analysis

SAVE = *identifiers*	To save details of each analysis to use subsequently with the AOVDISPLAY procedure

AOVANYHOW assesses a data set to select the most appropriate method for analysis of variance. If the design is orthogonal or balanced it uses the ANOVA directive (4.1.2). Otherwise, if there is no blocking in the design (i.e. there is only one random term) it uses the Genstat regression facilities through either AUNBALANCED (4.8.1) or A2WAY (2.3.3). Finally, if there are additional random terms, it looks to see if these contain any useful information about the treatments in order to choose between regression and REML (5.1.3). The EFLIMIT option sets a limit on the amount of information that may be lost on any of the treatment contrasts if the analysis to be done by regression instead of REML; the default of 0.1 implies that no more than 10% of the information on any contrast may be estimated between the random terms.

The method of use is similar to that for ANOVA. The treatment terms to be fitted must be specified, before calling the procedure, by the TREATMENTSTRUCTURE directive. Similarly, any covariates must be indicated by the COVARIATE directive. Any blocking structure must be specified by the BLOCKSTRUCTURE directive.

The parameters of the procedure are identical to those of ANOVA. The variates to be analysed are specified by the Y parameter. If the Y variate or any of the factors or covariates is restricted, only the units not excluded by the restriction will be analysed. Residuals and fitted values can be saved using the RESIDUALS and FITTEDVALUES parameters respectively. Finally, the SAVE parameter allows details of the analysis to be saved so that further output can be obtained using the AOVDISPLAY procedure.

Printed output is controlled by the PRINT option. The settings are limited to those that can produce analogous output from any of the analysis methods:

aovtable	analysis-of-variance table from ANOVA or regression, or Wald and F tests for fixed effects from REML,
information	design type, efficiency factors and name of the command used for the analysis,
means	tables of (predicted) means, and
residuals	residuals (fitted values are printed too for analyses by regression or REML).

Probabilities can be printed for variance ratios by setting option FPROBABILITY=yes.

The SAVE parameter allows you to save a pointer containing information about the analysis. You can use this as the input for the SAVE parameter of the AOVDISPLAY procedure to print (or reprint) any of the information provided by the PRINT option above. AOVDISPLAY has options PRINT, FPROBABILITY, PLOT, COMBINATIONS, ADJUSTMENT, PSE, LSDLEVEL, EFLOSS and EXIT, and a parameter SAVE that operate in the same way as those of AOVANYHOW. Alternatively, the first element of the SAVE pointer is the save structure from the command that was used for the analysis. So, if you use this with the display commands associated with that analysis command, you can display the more specialized output from the command (for example, variance components from REML).

Tables of means from regression and REML are calculated using the directives PREDICT (3.3.4) and VPREDICT (5.5.1), respectively. The first step (A) of their calculations forms the full table of predictions, classified by every factor in the model. The second step (B) averages the full table over the factors that do not occur in the table of means. The COMBINATIONS option specifies which cells of the full table are to be formed in Step A. The default setting, estimable, fills in all the cells other than those that involve parameters that cannot be estimated, for example because of aliasing. Alternatively, setting COMBINATIONS=present excludes the cells for factor combinations that do not occur in the data. The ADJUSTMENT option then defines how the averaging is done in Step B. The default setting, marginal, forms a table of marginal weights for each factor, containing the proportion of observations with each of its

levels; the full table of weights is then formed from the product of the marginal tables. The setting `equal` weights all the combinations equally. Finally, for regression analyses, the setting `observed` uses the `WEIGHTS` option of `PREDICT` to weight each factor combination according to its own individual replication in the data.

The `PSE` option controls the types of standard errors that are produced to accompany the tables of means, with settings:

`differences`	summary of standard errors for differences between pairs of means;
`alldifferences`	standard errors for differences between all pairs of means;
`lsd`	summary of least significant differences between pairs of means;
`alllsd`	least significant differences between all pairs of means;
`means`	effective standard errors for analyses by `ANOVA`, or approximate effective standard errors for analyses by regression or `REML` – these are formed by procedure `SED2ESE` with the aim of allowing good approximations to the standard errors for differences to be calculated by the usual formula of $sed_{i,j} = \sqrt{(\ ese_i^2 + ese_j^2\ )}$.

The default is `differences`. The `LSDLEVEL` option sets the significance level (as a percentage) for the least significant differences.

The `PLOT` option allows various residual plots to be requested: `fittedvalues` for a plot of residuals against fitted values, `normal` for a Normal plot, `halfnormal` for a half Normal plot, and `histogram` for a histogram of residuals.

The `FACTORIAL` option sets a limit on the number of factors that a higher-order term, such as an interaction, can contain; any terms with more factors are deleted from the analysis. The `WEIGHTS` option allows a variate of weights to be specified for a weighted analysis of variance.

You can save a scalar indicating the recommended method of analysis by using the `EXIT` option. The scalar can take values with the following meanings.

0. The design is orthogonal. Analyse by `ANOVA` (4.1.2).
1. The design is balanced. Analyse by `ANOVA` (4.1.2).
2. The design unbalanced. It has 1 or 2 treatment factors and no blocking. Analyse by `A2WAY` (2.3.3).
3. The design unbalanced and has 1 or 2 treatment factors. No more than a proportion defined by the `EFLIMIT` option of the information on any treatment contrast is estimated between block terms. Analyse by `A2WAY` (2.3.3).
4. The design unbalanced, and there are either weights or more than 2 treatment factors. There is no blocking. Analyse by `AUNBALANCED` (4.8.1).
5. The design is unbalanced, and there either are weights or more than 2 treatment factors. No more than a proportion defined by the `EFLIMIT` option of the information on any treatment contrast is estimated between block terms. Analyse by `AUNBALANCED` (4.8.1).
6. The design unbalanced with several block (i.e. random) terms. Analyse by `REML` (5.1.3).

The `EFLOSS` option can save the maximum loss of efficiency that would occur on any treatment contrast if the analysis is done by regression.

You can set option `METHOD=recommend` to request that `AOVANYHOW` will just form a recommendation for the command to be used if the analysis cannot be done by `ANOVA`. The only available `PRINT` option is then `information`, which tells you which command is recommended. You can still use the `EXIT` and `EFLOSS` options, but residuals and fitted values will be saved (by the `RESIDUALS` and `FITTEDVALUES` parameters) if the analysis should be done by `ANOVA`.

Example 4.8.7 shows that less than 1% of the available treatment information was lost by using `AUNBALANCED` instead of `REML` to analyse the data in Example 4.8.1a.

Example 4.8.7

```
  12  AOVANYHOW [PRINT=information] Y

Analysis of variance by ANOVA, REML or regression
=================================================


Information summary
-------------------

Design is unbalanced. It does not have a one- or two-way treatment structure,
and no more than 0.801% of information on any treatment contrast is estimated
between block terms. Analyse by AUNBALANCED.
```

4.8.8 The AN1ADVICE procedure

AN1ADVICE procedure

Aims to give useful advice if a design that is thought to be balanced fails to be analysed by ANOVA (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (advice, suspects); default advi
FACTORIAL = *scalar*	Limit on number of factors in a treatment term; default 3
METHOD = *string tokens*	Method to use to predict the correct pattern of replication (median, mode, proportional); default mode
WEIGHTS = *variate*	Weights for the analysis; default * i.e. all units have weight one
SUSPECTS = *variate*	Saves the numbers of the units whose factor values are suspected to be incorrect

Parameter

Y = *variates*	Data values to be analysed (this is needed only if the analysis is to take place on a restricted set of units)

As already mentioned, the ANOVA directive analyses "balanced" designs, and will itself detect whether or not a design can be analysed. So if you are not sure about a particular design, you can run it through ANOVA and see whether it succeeds or fails with an "AN 1" diagnostic (see Example 4.8.1a). Sometimes the design will genuinely be unbalanced, but on other occasions it may be that errors have been made in entering the data. So the aim of AN1ADVICE is to give useful advice if you find that a set of data that you had expected to be balanced fails to be analysed by ANOVA.

The use of AN1ADVICE is very similar to ANOVA (4.1.2). You must first define the model that is to be fitted in the analysis, using the TREATMENTSTRUCTURE and BLOCKSTRUCTURE directives (see 4.11 and 4.2.1). As in ANOVA, the treatment terms to be included in the model are controlled by the FACTORIAL option, and the WEIGHT option can specify weights for a weighted analysis of variance.

AN1ADVICE has a parameter Y to specify the variate whose values are being analysed. However, this is required only if you are analysing a subset of the units. (You would then have used the RESTRICT directive, directly or through a menu, to restrict Y to the units concerned.)

In a balanced design, the joint replications of sets of factors in the design will usually have a

systematic pattern. Often there will be equal replication. Then, for example, if you look at the replication table for any pair of factors, it will contain a single value (the number of times each pair of their levels occurs in the design). Alternatively, the replications may have a proportional pattern. For example, you may have a "control" level of one of the factors with perhaps twice as many replicates as the other, "test", levels. Then, in every replication table involving that factor, the cells for the "control" level will have values twice as large as those in the corresponding "test" cells. So AN1ADVICE examines the factors in the model terms that ANOVA has found to be unbalanced, and examines their replications to try to identify cells whose values seem to be too small or too large.

The METHOD option controls how AN1ADVICE works out what the replication in each table ought to be. The default setting, mode, assumes that the values should all be equal, and that the non-zero value that occurs most often in the table is the correct one. The setting median is similar except that the right value is assumed to be the median of the non-zero values. Finally, the proportional setting estimates the correct values for each table by assuming that the replication has a proportional pattern.

The PRINT option controls the printed output, with settings:

advice	prints advice including replication tables of the factors that seem to be incorrect, highlighting the cells that seem to be too small or too large, and
suspects	prints the units with the combinations of factor levels that seem to occur too often in the design.

The default is PRINT=advice. The list of suspect units can also be saved, in a variate, using the SUSPECTS option.

If you believe that the design should be balanced, you may find that the factor values (or weights) of some of suspect units have been entered incorrectly. Alternatively, you may find that some units with the factor combinations whose replication has been highlighted as too low have been accidentally omitted from the data. If these mistakes can be corrected, the design may become balanced. Alternatively, if you cannot find any mistakes in the data, you will need to use AOVANYHOW (4.8.7), AUNBALANCED (4.8.1) or REML (5.3.1) instead.

Example 4.8.8 looks again at the data in Example 4.8.7a, and finds that the lack of balance is associated with unequal replication of factor B over days.

Example 4.8.8

```
  16  AN1ADVICE  [PRINT=advice]

Advice about the cause of an unbalanced design
==============================================

The term B is unbalanced i.e. its contrasts do not all have the same efficiency
factor. It is non-orthogonal to the term Day, so it is the "adjustment" for Day
that is causing its efficiency factors to be unequal.

In a balanced design, the joint replications of the factors will usually have
an "even" pattern. The table below shows the replications of the unevenly
replicated factors in the two terms, highlighting those that differ from the
most common replication.

          Day           1              2
            B
            1          12 *?*         12 *?*
            2          11             11
            3          11             10 *?*


If you believe that the design should be balanced, you may find that the factor
values of some of the units with the factor combinations highlighted above have
been entered incorrectly or accidently omitted from the data. If these mistakes
```

```
can be corrected, the design may become balanced. Alternatively, if you cannot
find any mistakes in the data, you will need to use regression or REML instead
of ANOVA.
```

4.9 Selecting and generating an experimental design

Genstat has a comprehensive set of directives and procedures for design of experiments. In the first part of this section we describe the procedures that allow you to select and generate a design. The final part describes the AFRESPONSESURFACE which uses the BLKL algorithm of Atkinson & Donev (1992) to construct designs for estimating response surfaces (4.9.14). Later sections describe the facilities for displaying designs (4.10), randomization (4.11), determining sample sizes (4.12), generating factors or pseudo-factors (4.13), adding additional plots to a design or combining designs (4.13), and constructing new design keys (4.13). Collectively, these directives and procedures are known as the *Genstat Design System*. The procedures cover many different types of design. These are listed below, together with the name (in brackets) of the procedure that you can use to select and generate designs of that type.

- Orthogonal hierarchical designs – designs such as randomized blocks, split-plots, split-split-plots, &c. (AGHIERARCHICAL)
- Complete factorial designs (with interactions confounded with blocks) – these are available for treatments that all have the same number of levels k, where k is a prime number or a power of a prime number. The design will be a minimum-aberration design. To explain this, we first define the resolution of a design as the largest integer r such that no interaction term with r factors is confounded with blocks. The aberration of the design is the number of interaction terms with r+1 factors that are confounded. A minimum aberration design is defined as a design with the smallest aberration out of the designs with the highest available resolution. So, essentially this selects the best design by minimizing the number of interactions with the minimum number of factors that are confounded. (AGFACTORIAL)
- Fractional factorial designs (with blocking if required) – these are formed by taking one block of a minimum-aberration factorial design. If required, the resulting fractional factorial can be further dividing into its own blocks. (AGFACTORIAL)
- Factorial designs from a repertoire (with confounding) – these have several treatment factors and a single blocking factor (giving strata for blocks and plots within blocks). The blocks are too small to contain a complete replicate of the treatment combinations and so various interaction are confounded with blocks. (AGDESIGN)
- Fractional factorial designs from a repertoire (with blocking) – again there are several treatment factors but the design does not contain every treatment combination and so some interactions are aliased; there can also be a blocking factor and some interactions will then be confounded with blocks. (AGFRACTION)
- Latin squares – designs are available for any number of treatments (subject to workspace limitations) also, where feasible, more than one orthogonal treatment factor can be generated to form Graeco-Latin squares etc. (AGLATIN).
- Latin squares balanced for carry-over effects – these are relevant when the same plots or subjects are treated during several successive time periods, and there is interest both in the direct effect of a treatment during the period in which it is applied and its carry-over (or "residual") effect during later periods (AGCROSSOVERLATIN).
- Complete and quasi-complete Latin squares – Latin squares designed to guard against interference between plots; a complete Latin square is a Latin square in which each ordered pair of treatments appears exactly once within the rows of the square, and exactly once within the columns; a quasi-complete Latin has similar properties, but here each unordered pair occurs exactly twice within the rows, and exactly twice within the columns (AGQLATIN).

Semi-Latin squares – $n \times n$ Latin squares whose individual plots are split into k sub-plots to cater for a treatment factor with $n \times k$ levels; three types are available Trojan squares, interleaving Latin squares and inflated Latin squares (AGSEMILATIN).

Youden squares – $n \times k$ designs where the treatments occur once in every row, and each pair of treatments occurs the same number of times in the columns. The simplest design is formed by deleting the final row of a Latin square (AGYOUDENSQUARE)

Lattice designs – designs for a single treatment factor with number of levels that is the square of some integer k. The design has replicates, each containing k blocks of k plots, and different treatment contrasts can be confounded with blocks in each replicate. (AGSQLATTICE)

Lattice squares – these are similar to lattices except that the blocking structure with the replicates has rows crossed with columns; again different treatment contrasts can be confounded with the rows and columns in each replicate. (AGSQLATTICE)

Alpha designs – these again have a single treatment factor but there is no constraint on the number of levels; the blocking structure has replicates and blocks within replicates; see Patterson & Williams (1976). (AGALPHA)

Balanced-incomplete-block designs – designs where the experimental units are grouped into blocks such that every pair of treatments occurs in an equal number of blocks. All comparisons between treatments are thus made with equal accuracy, so the design is balanced and, in particular, can be analysed by ANOVA. (AGBIB)

Cyclic designs – these are designs with a single blocking factor which defines blocks that are too small to contain every treatment. Usually there is a single treatment factor, but you can also generate the cyclic superimposed designs of Hall & Williams (1973) in which there are two treatment factors and the treatment structure fits only the main effects. An alternative refinement (Davis & Hall 1969) has a crossed blocking structure generally taken to represent Subjects*Time. (AGCYCLIC)

Neighbour-balanced designs – designs that allow an adjustments to be made for the effect that a treatment may have on adjacent plots. (AGNEIGHBOUR).

Central composite designs – used to study multi-dimensional response surfaces. (AGCENTRALCOMPOSITE)

Box-Behnken designs – used to study multi-dimensional response surfaces. (AGBOXBEHNKEN)

Plackett Burman (main effect) designs – for estimating main effects of factors with two levels, using a minimum number of experimental units; see Plackett & Burman (1946). (AGMAINEFFECT)

Loop designs – for use e.g. in time-course microarray experiments (AGLOOP)

Reference-level designs – for use e.g. in two-colour microarray experiments, (AGREFERENCE)

The procedures are very convenient to use interactively. Genstat then guides you through the process by asking questions first to select the design, then to give details such as the names of the factors, and so on. If you wish to avoid some of the question-and-answer process, the procedures all have options and parameters to supply the information otherwise obtained by the various questions and, provided you supply *all* the required information, they can also be used in batch.

To save you remembering the names of the individual procedures, there is also a general procedure DESIGN that can be used interactively to provide a single point of access to all the design types. DESIGN has an option called STATEMENT which allows you to save a Genstat text structure containing a command to use the relevant subsidiary procedure, and setting all the options and parameters required to recreate the design. DESIGN is called if you chose Select Design in Genstat *for Windows*, when it generates a pop-up menu. In other implementations, the question takes a more "conversational" form, as shown in Example 4.9.1a. The same is true for all the design questions which, in fact, are generated within the procedures using the QUESTION procedure. The alternative Standard Design menu of Genstat *for Windows* uses AGHIERARCHICAL, AGLATIN and AGLATTICE to generate completely randomized designs, randomized blocks, Latin and Graeco-Latin squares, split-plots, strip-plots (or criss-cross designs) and lattices. There are menus to generate factorial designs in blocks and fractional factorial designs, which use AGFACTORIAL.

The procedures mentioned above generate and randomize the designs automatically, calling other directives and procedures to perform the necessary tasks, and there is no need for you to be aware of any of the details. However, we give more information during this section in case you want to study the process in more depth or to add new designs. The design system is based on a range of standard generators. Some of these, such as the Galois fields used to generate Latin squares or the Hadamard matrices required for main-effect designs, can be formed by Genstat when required – and so there is no limitation on the available designs. There is also no limitation on the orthogonal hierarchical designs, which are constructed directly. Repertoires of other generators, such as design keys, are stored in backing-store files which are scanned by the design generation procedures to form menus listing the available possibilities. Algorithms are available to form new design keys (4.13.6), and these can then be added to the design files to become an integral part of the system. Table 4.9 lists the various generators, the design types that they can construct, and the associated procedures and directives.

Table 4.9: Directives and procedures for constructing designs in Genstat

Generator	Designs	Generation	Selection	Assembly (and construction)
design key	Factorial, Lattice sq.	AKEY (GENERATE)	AGDESIGN	FDESIGNFILE (FKEY and FPSEUDOFACTOR)
	Fractional		AGFRACTION	
Galois field	Latin square	AGLATIN		
	Semi-Latin square	AGSEMILATIN		
	Youden square	AGYOUDENSQUARE		
	Square Lattice	AGSQLATTICE		
Terraced group	Complete Latin sq.	AGQLATIN		
Alpha array	Alpha designs	AFALPHA	AGALPHA	
Initial block	Cyclic design	AFCYCLIC	AGCYCLIC	
Total cycles	Balanced neighbour	AGNEIGHBOUR		
Hadamard matrix	Balanced-incomplete-block	AGBIB		
	Box Behnken	AGBOXBEHNKEN		
	Plackett Burman	AGMAINEFFECT		
	Loop	AGLOOP		
	Reference level	AGREFERENCELEVEL		
	Central Composite	AGCENTRALCOMPOSITE		
Any type			DESIGN	

4.9.1 Orthogonal hierarchical designs

AGHIERARCHICAL procedure

Generates orthogonal hierarchical designs (R.W. Payne).

Options

PRINT = *string token* — Controls whether or not to print a plan of the design (design); if unset in an interactive run AGHIERARCHICAL will ask whether the design is to be printed, in a batch run the default is not to print the design

ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (no, yes); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run
SEED = *scalar*	Seed to be used to randomize the design; a negative value implies no randomization
STATEMENT = *text*	Saves a command to recreate the design (useful if the design information has been specified in response to questions from AGHIERARCHICAL)
EXCLUDELEVELS = *scalars*	Levels of the first block factor to exclude during randomization

Parameters

BLOCKFACTORS = *factors*	Specifies the identifier for the block factor used to index the units of each stratum (or level of the hierarchy)
TREATMENTFACTORS = *factors* or *pointers*	Specifies the identifier of the treatment factor or factors applied to the units of each stratum
LEVELS = *scalars* or *pointers*	Number of levels for the treatment factors in each stratum; if required, a pointer can contain an extra scalar to specify replication

AGHIERARCHICAL can generate any hierarchical equally-replicated factorial design. This category covers many popular designs, including completely randomized designs (i.e. designs with no blocking; see Section 4.1), randomized complete block designs (4.2.1 and 4.3), split-plots (4.2.1), split-split-plots, and so on. The designs can have any number of block and treatment factors, and the factors can have any number of levels. It can be used either interactively or in batch. If you are running Genstat interactively, you can simply issue the command

```
AGHIERARCHICAL
```

with no options or parameters. It will then ask questions to define the necessary details of the design. If, however, you wish to recreate the same design later, the STATEMENT option allows you to save a Genstat text structure containing a command specifying the same information. Yhe options and parameters of the procedure allow you to avoid the questions or to use the procedure when running Genstat in batch.

The units of each stratum (or level of the hierarchy) are identified by a block factor: for example Replicates, Blocks, Plots, Subplots, Subjects &c. These can be supplied by the BLOCKFACTORS parameter. The TREATMENTFACTORS parameter defines factors for the treatments applied to the units of the strata, and LEVELS defines the levels of treatments and replication of block factors. For example, in Example 4.9.1a,

```
AGHIERARCHICAL [PRINT=design; ANALYSE=yes; SEED=392384] \
  Blocks,Plots; *,A; 3,5
```

defines a randomized block design generated with three blocks, and a single treatment factor A (applied to the plots) with five levels.

Example 4.9.1a

```
2  AGHIERARCHICAL [PRINT=design; ANALYSE=yes; SEED=392384]\
3    Blocks,Plots; *,A; 3,5
```

```
Treatments on each unit of the design
=====================================

 Blocks   1   2   3
  Plots
      1   4   2   3
      2   3   1   1
      3   1   4   5
      4   2   5   2
      5   5   3   4

Treatment factor: A.


Analysis of variance
====================

Source of variation     d.f.

Blocks stratum             2

Blocks.Plots stratum
A                          4
Residual                   8

Total                     14
```

If there are several factors in a stratum, the identifiers should be placed into a pointer. For example,

```
AGHIERARCHICAL Blocks,Plots; *,!p(A,B); 3,2
```

for a randomized block design with two treatment factors, `A` and `B`, both with two levels. Similarly, if the factors in a stratum have different numbers of levels, the `LEVELS` parameter may contain pointers. For example

```
AGHIERARCHICAL [PRINT=design; ANALYSE=yes; SEED=581386]\
  Blocks,Plots; TREATMENTFACTORS=*,!p(Type,Amount);\
  LEVELS=3,!p(2,3)
```

defines the randomized block design in Example 4.9.1b, where `Type` has two levels and `Amount` has three.

`AGHIERARCHICAL` not only defines the values of the factors, it also contains `BLOCKSTRUCTURE` and `TREATMENTSTRUCTURE` statements to define the structure of the design. We can see what structure has been defined for the design, using the `ASTATUS` procedure (4.6.2).

Example 4.9.1b

```
  4  AGHIERARCHICAL [PRINT=design; ANALYSE=yes; SEED=581386]\
  5    Blocks,Plots; TREATMENTFACTORS=*,!p(Type,Amount);\
  6    LEVELS=3,!p(2,3)

Treatment combinations on each unit of the design
=================================================

 Blocks   1     2     3
  Plots
      1   2 2   1 2   1 2
      2   1 1   1 3   1 1
      3   1 2   1 1   1 3
      4   2 3   2 1   2 2
      5   1 3   2 3   2 3
      6   2 1   2 2   2 1

Treatment factors are listed in the order: Type, Amount.
```

```
Analysis of variance
====================

Source of variation     d.f.

Blocks stratum             2

Blocks.Plots stratum
Type                       1
Amount                     2
Type.Amount                2
Residual                  10

Total                     17

   7  ASTATUS

Treatment structure: Type*Amount
Block structure: Blocks/Plots
Covariates: not set
```

The pointer can contain an extra element to indicate that there is to be replication (as well as treatments) in a stratum:

```
AGHIERARCHICAL Blocks,Plots;\
  TREATMENTFACTORS=*,!p(Type,Amount);\
  LEVELS=3,!p(2,3,4)
```

indicates that there are to be four replicates of the `Type` and `Amount` combinations on the plots of each block. (There is no way of requesting this sort of replication, other than by including a "dummy" treatment factor to be generated and then ignored, if you are using `AGHIERARCHICAL` interactively.)

The `SEED` option allows you to specify a seed to randomize the design. In a batch run, this has a default of –1, to suppress randomization. The `PRINT` option can be set to `design` to print the plan of the design. By default, if you are running Genstat in batch, the plan is not printed. Similarly the `ANALYSE` option governs whether or not `AGHIERARCHICAL` produces a skeleton analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). You can use the `EXCLUDELEVELS` parameter to specify levels of the first block factor that you do not wish to randomize. (This can be useful in "demonstration experiments", when the treatments may need to be kept in a systematic order in some parts of the trial, but it is not a good idea in more normal situations.)

The treatment combinations are generated with equal replication, but you can use "dummy" factors together with the `NEWLEVELS` function to define designs where some treatment factor levels have additional replication. The factor `Amount` in Example 4.9.1c has two plots in each block with level 1, and one plot each for levels 2 and 3. This is achieved by first generating a factor (here called `Adum`) with four levels, and then using the `NEWLEVELS` function (line 12) to map levels 1 and 2 of `Adum` to level 1 of `Amount`, level 3 of `Adum` to level 2 of `Amount`, and level 4 of `Adum` to level 3 of `Amount`. Notice also how the option setting `MODIFY=yes` is used in the `FACTOR` statement in line 13 to add labels to the definition of the factor `Type`. Alternatively, you can use the `AMERGE` procedure (4.13.3) to add extra treatment levels such as controls, as shown in Example 4.13.3.

Example 4.9.1c

```
 8  AGHIERARCHICAL [PRINT=*; ANALYSE=no; SEED=-1]\
 9    Blocks,Plots; TREATMENTFACTORS=*,!p(Type,Adum);\
10    LEVELS=3,!p(2,4)
11  FACTOR [LEVELS=3] Amount
12  CALCULATE Amount = NEWLEVELS(Adum; !(1,1,2,3))
13  FACTOR [LABELS=!t(standard,test); MODIFY=yes] Type
14  PRINT Blocks,Plots,Type,Amount
```

```
    Blocks       Plots        Type      Amount
         1           1    standard           1
         1           2    standard           1
         1           3    standard           2
         1           4    standard           3
         1           5        test           1
         1           6        test           1
         1           7        test           2
         1           8        test           3
         2           1    standard           1
         2           2    standard           1
         2           3    standard           2
         2           4    standard           3
         2           5        test           1
         2           6        test           1
         2           7        test           2
         2           8        test           3
         3           1    standard           1
         3           2    standard           1
         3           3    standard           2
         3           4    standard           3
         3           5        test           1
         3           6        test           1
         3           7        test           2
         3           8        test           3

15  TABULATE [PRINT=counts; CLASSIFICATION=Type,Amount]

                 Count
     Amount          1          2          3
       Type
   standard          6          3          3
       test          6          3          3
```

4.9.2 Complete and fractional factorial designs

AGFACTORIAL procedure

Generates minimum aberration block or fractional factorial designs (P.J. Laycock, P.J. Rowley & R.W. Payne).

Options

PRINT = *string token*	Controls whether or not to print a plan of the design (design); if unset in an interactive run AGFACTORIAL will ask whether the design is to be printed, in a batch run the default is not to print the design
ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (yes, no); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run
FACTORIAL = *scalar*	Limit on number of factors in treatments terms in the analysis of variance; default 3

Parameters

LEVELS = *scalars*, *variates* or *texts*	Levels for the treatment factors in each design
NTREATMENTFACTORS = *scalars*	Number of treatment factors
NUNITS = *scalars*	Number of units per block
NFRACTIONBLOCK = *scalars*	Defines the number of the block to use to define a fractional factorial, or can be set to zero to take a block at random; if unset in an interactive run AGFACTORIAL

	will ask whether to form a fractional factorial design, in a batch run the default is to form the full (block) design
NSUBUNITS = *scalars*	Number of units in each sub-block
SEED = *scalars*	Seed to be used to randomize each design; a negative value implies no randomization
TREATMENTFACTORS = *pointers*	Specifies identifiers for the treatment factors
BLOCKS = *factors*	Identifier for the block factor
SUBBLOCKS = *factors*	Identifier for the sub-block factor
PSEUDOFACTORS = *pointers*	Specifies identifiers for pseudo-factors
UNITLABELS = *variates*	Specifies the identifier of a variate to store a unique numerical label for each unit in the design
NDESIGN = *scalars*	Saves or defines the design number
NSUBDESIGN = *scalars*	Saves or defines the sub-design number
STATEMENT = *texts*	Saves a command to recreate each design (useful if the design information has been specified in response to questions from AGFACTORIAL)

AGFACTORIAL generates efficient block or fractional factorial designs using the minimum aberration algorithm of Laycock & Rowley (1995), implemented in the AFMINABERRATION directive (4.13.9). It also sets the block and treatment formulae (using the BLOCKSTRUCTURE and TREATMENTSTRUCTURE directives), and generates any pseudo-factors needed to analyse the design using the ANOVA directive.

To explain minimum aberration for a block design, we start by defining the resolution of a design as the largest integer r such that no interaction term with r factors is confounded with blocks. The aberration of the design is the number of interaction terms with r+1 factors that are confounded. A minimum aberration design is defined as a design with the smallest aberration out of the designs with the highest available resolution. So, essentially this minimizes the number of interactions with the minimum number of factors that are confounded. The definition for a fractional factorial design is essentially the same. The fractional factorial is constructed by taking only one block from the block design, and the terms that were confounded with blocks in the block design become aliased in the fractional factorial.

AGFACTORIAL can be used either in batch or interactively. In an interactive run, it obtains the information necessary to select and define the design by asking questions. You need set the parameters only if you wish to anticipate some of the questions, or if you wish to use AGFACTORIAL in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The LEVELS parameter defines the number of levels of the treatment factors, either as a scalar or by providing a text or variate with the required number of levels, to use for the LEVELS option of the FACTOR directive. This must be a prime number (e.g. 2, 3, 5, 7, 11) or a power of a prime number (e.g. 4, 8, 9). The number of treatment factors is specified by the NTREATMENTFACTOR parameter. The number of the units in each block (or, equivalently, the number of units in a fractional factorial) is specified by the NUNITS parameter; this must be a power of the number of levels. The NFRACTIONBLOCK parameter allows you to form a fractional factorial, either by setting it to the number of the block to take, or by setting it to zero to take a block at random; if you set NFRACTIONBLOCK to a scalar containing a missing value, AGFACTORIAL forms a block design. You can define blocks for a fractional factorial (or, equivalently, sub-blocks for a block design) by defining their size using NSUBUNITS parameter; this too must be a power of the number of levels.

The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is –1, to suppress randomization. If you do not set SEED when running

interactively AGFACTORIAL will ask for a seed, and again a negative value suppresses any randomization.

The TREATMENTFACTORS parameter can specify a pointer to supply identifiers for the treatment factors in the design. For example, if there are two factors you could define their identifiers to be A and B by forming the pointer Tf (say) with the statement

```
POINTER [VALUES=A,B] Tf
```

and then setting TREATMENTFACTORS=Tf. Alternatively, and more succinctly, you could put TREATMENTFACTORS=!p(A,B), where !p(A,B) is an unnamed pointer containing the required two identifiers. The BLOCKS and SUBBLOCKS parameters allow you to specify identifiers for the block and sub-block factors. Designs where the treatment factors have more than two levels may require pseudo-factors to be defined in order for them to be analysed by ANOVA. The PSEUDOFACTORS parameter can specify a pointer to supply their identifiers. If the treatment, block or sub-block factors and any necessary pseudo-factors are not specified in a batch run, AGFACTORIAL will use identifiers that are local within the procedure and thus lost at the end of the procedure. If you are running interactively, AGFACTORIAL will ask you to provide identifiers, and these will remain available after AGFACTORIAL has finished running.

The UNITLABELS parameter can specify a variate to store a unique number to label each of the units in the design. In the first block, the variate contains the numbers one up to the number of units per block. The second block contains these numbers plus the smallest power of ten greater than the number of units per block, the third block contains the numbers plus twice this power of ten, and so on.

The PRINT option can be set to design to print the plan of the design, and summary to print a summary of the design properties. By default, if you are running Genstat in batch, these are not printed. If you do not set PRINT when running interactively, AGFACTORIAL will ask whether or not you wish to print them. Similarly the ANALYSE option governs whether or not AGFACTORIAL produces a skeleton analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). Again AGFACTORIAL assumes that this is not required if ANALYSE is unset in a batch run, and asks whether it is required if ANALYSE is unset in an interactive run. The FACTORIAL option sets a limit on the number of factors in the treatment terms in the analysis of variance; by default, this is three.

The NDESIGN parameter can save a unique *design number* for the design, and the NSUBDESIGN can save a unique number for the sub-design of the design (as defined by Laycock & Rowley 1995). You can input these with NDESIGN and NSUBDESIGN later, along with the same settings for LEVELS, NTREATMENTFACTORS, NUNITS and NSUBUNITS, to generate the design factors again without repeating the design search.

Example 4.9.2 uses AGFACTORIAL first to form a complete factorial with six factors, each with two levels, in four blocks of size 16. It then forms a fractional factorial, again with six factors with two levels, but now in four blocks of size eight (so this is a ½ fraction).

Example 4.9.2

```
  2  "2x2x2x2x2x2 design in 4 blocks of size 16"
  3  AGFACTORIAL [PRINT=design; ANALYSE=yes; FACTORIAL=4]\
  4              2; NTREATMENTFACTORS=6; NUNITS=16; NSUBUNITS=!s(*);\
  5              NFRACTIONBLOCK=!s(*); TREATMENTFACTORS=!p(A,B,C,D,E,F);\
  6              BLOCKS=Blocks; UNITLABELS=Labels; SEED=-1

Treatment combinations on each unit of the design
=================================================

  Blocks   1             2             3             4
 _units_
       1   1 1 1 1 1 1   1 1 1 2 1 2   1 1 1 2 1 1   1 1 1 1 1 2
       2   1 1 1 1 2 2   1 1 1 2 2 1   1 1 1 2 2 2   1 1 1 1 2 1
       3   1 1 2 2 1 1   1 1 2 1 1 2   1 1 2 1 1 1   1 1 2 2 1 2
       4   1 1 2 2 2 2   1 1 2 1 2 1   1 1 2 1 2 2   1 1 2 2 2 1
       5   1 2 1 2 1 2   1 2 1 1 1 1   1 2 1 1 1 2   1 2 1 2 1 1
       6   1 2 1 2 2 1   1 2 1 1 2 2   1 2 1 1 2 1   1 2 1 2 2 2
       7   1 2 2 1 1 2   1 2 2 2 1 1   1 2 2 2 1 2   1 2 2 1 1 1
       8   1 2 2 1 2 1   1 2 2 2 2 2   1 2 2 2 2 1   1 2 2 1 2 2
       9   2 1 1 2 1 2   2 1 1 1 1 1   2 1 1 1 1 2   2 1 1 2 1 1
      10   2 1 1 2 2 1   2 1 1 1 2 2   2 1 1 1 2 1   2 1 1 2 2 2
      11   2 1 2 1 1 2   2 1 2 2 1 1   2 1 2 2 1 2   2 1 2 1 1 1
      12   2 1 2 1 2 1   2 1 2 2 2 2   2 1 2 2 2 1   2 1 2 1 2 2
      13   2 2 1 1 1 1   2 2 1 2 1 2   2 2 1 2 1 1   2 2 1 1 1 2
      14   2 2 1 1 2 2   2 2 1 2 2 1   2 2 1 2 2 2   2 2 1 1 2 1
      15   2 2 2 2 1 1   2 2 2 1 1 2   2 2 2 1 1 1   2 2 2 2 1 2
      16   2 2 2 2 2 2   2 2 2 1 2 1   2 2 2 1 2 2   2 2 2 2 2 1

Treatment factors are listed in the order: A, B, C, D, E, F.


Analysis of variance
====================

Source of variation     d.f.

Blocks stratum
A.B.C.D                    1
A.B.E.F                    1
C.D.E.F                    1

Blocks.*Units* stratum
A                          1
B                          1
C                          1
D                          1
E                          1
F                          1
A.B                        1
A.C                        1
B.C                        1
A.D                        1
B.D                        1
C.D                        1
A.E                        1
B.E                        1
C.E                        1
D.E                        1
A.F                        1
B.F                        1
C.F                        1
D.F                        1
E.F                        1
A.B.C                      1
A.B.D                      1
A.C.D                      1
B.C.D                      1
A.B.E                      1
```

```
A.C.E                      1
B.C.E                      1
A.D.E                      1
B.D.E                      1
C.D.E                      1
A.B.F                      1
A.C.F                      1
B.C.F                      1
A.D.F                      1
B.D.F                      1
C.D.F                      1
A.E.F                      1
B.E.F                      1
C.E.F                      1
D.E.F                      1
A.B.C.E                    1
A.B.D.E                    1
A.C.D.E                    1
B.C.D.E                    1
A.B.C.F                    1
A.B.D.F                    1
A.C.D.F                    1
B.C.D.F                    1
A.C.E.F                    1
B.C.E.F                    1
A.D.E.F                    1
B.D.E.F                    1
Residual                   7

Total                     63


  7  "1/2 fraction of a 2x2x2x2x2x2 design in 4 blocks of size 8"
  8  AGFACTORIAL [PRINT=design; ANALYSE=yes; FACTORIAL=2]\
  9              2; NTREATMENTFACTORS=6; NUNITS=32;\
 10              NFRACTIONBLOCK=1; NSUBUNITS=8; SUBBLOCKS=block;\
 11              TREATMENTFACTORS=!p(a,b,c,d,e,f); UNITLABELS=label; SEED=-1

Treatment combinations on each unit of the design
=================================================


   block   1             2             3             4
 _units_
       1   1 1 1 1 1 1   1 1 1 1 2 2   1 1 1 2 1 2   1 1 1 2 2 1
       2   1 1 2 2 1 1   1 1 2 2 2 2   1 1 2 1 1 2   1 1 2 1 2 1
       3   1 2 1 2 2 2   1 2 1 2 1 1   1 2 1 1 2 1   1 2 1 1 1 2
       4   1 2 2 1 2 2   1 2 2 1 1 1   1 2 2 2 2 1   1 2 2 2 1 2
       5   2 1 1 2 2 2   2 1 1 2 1 1   2 1 1 1 2 1   2 1 1 1 1 2
       6   2 1 2 1 2 2   2 1 2 1 1 1   2 1 2 2 2 1   2 1 2 2 1 2
       7   2 2 1 1 1 1   2 2 1 1 2 2   2 2 1 2 1 2   2 2 1 2 2 1
       8   2 2 2 2 1 1   2 2 2 2 2 2   2 2 2 1 1 2   2 2 2 1 2 1


Treatment factors are listed in the order: a, b, c, d, e, f.


Analysis of variance
====================

Source of variation     d.f.

block stratum
e.f                        1
Residual                   2

block.*Units* stratum
a                          1
b                          1
c                          1
d                          1
e                          1
f                          1
```

```
a.b                    1
a.c                    1
b.c                    1
a.d                    1
b.d                    1
c.d                    1
a.e                    1
b.e                    1
c.e                    1
d.e                    1
a.f                    1
b.f                    1
c.f                    1
d.f                    1
Residual               8

Total                 31
```

4.9.3 Factorial designs with confounding

AGDESIGN procedure

Generates generally balanced designs (R.W. Payne).

Options

PRINT = *string token*	Controls whether or not to print a plan of the design and whether to print a catalogue of the designs in the subfile (design, catalogue); if unset in an interactive run AGDESIGN will ask whether the design is to be printed, in a batch run the default is not to print anything
ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (no, yes); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run
FILENAME = *text*	Name of the backing store file containing the design information; default uses the standard design file
SUBFILE = *identifier*	Subfile of the backing store file to be used

Parameters

DESIGN = *variates*	Contains codes to indicate the choice of design
TREATMENTFACTORS = *pointers*	Specifies identifiers for the treatment factors
BLOCKFACTORS = *pointers*	Specifies identifiers for the block factors
PSEUDOFACTORS = *pointers*	Specifies identifiers for any pseudo-factors
REPLICATEFACTOR = *factors*	Specifies the identifier of the factor to represent the replicates (if any) in each design
UNITLABELS = *variates*	Specifies the identifier of a variate to store a unique numerical label for each plot in the design
SEED = *scalars*	Seed to be used to randomize each design; a negative value implies no randomization
STATEMENT = *texts*	Saves a command to recreate each design (useful if the design information has been specified in response to questions from AGDESIGN)

These designs are generated by procedure AGDESIGN using design keys, selected from a stored repertoire. You do not need to know the details of how this is done, nor of where the keys are

stored, nor or which designs are available. The keys are accessed automatically, together with the other information required to form the design, and AGDESIGN generates a menu listing the choices. This is illustrated in Example 4.9.3, which shows the questions and answers (printed in bold font) to select and form a design for three treatment factors, A, B and C, each at three levels. In Genstat *for Windows* the questions are the same, but they appear in pop-up menus.

The design has four replicates each with three blocks of nine plots. There are 27 treatment combinations, so some contrasts from the A.B.C interaction must be confounded with blocks. (For an explanation of *confounding* see Section 4.7.1.) The "which version" question shows that there are four different ways in which we can do this. By choosing one of each version to provide the required four replicates, we ensure that each of the possible contrasts forming the A.B.C interaction is confounded in one of the replicates. As a result the design is balanced, as you can see from the analysis of variance at the end of the example. If, however, we had not selected all four versions the design would have been partially balanced and AGDESIGN would have generated the necessary pseudo-factors (see 4.7.3) for it to be analysable. Like AGHIERARCHICAL (Section 4.9.1) AGDESIGN not only forms the factors, it also sets the block and treatment formulae (using the BLOCKSTRUCTURE and TREATMENTSTRUCTURE directives) to allow the design to be analysed by ANOVA.

Example 4.9.3

```
> AGDESIGN

Which design would you like:
a         Single replicate of a 2x2x2 factorial in blocks of size 4
b         Single replicate of a 2x2x2x2 factorial in blocks of size 8
c         Single replicate of a 2x2x2x2x2x2 factorial in blocks of size 16
d         Single replicate of a 2x2x2x2 factorial in blocks of size 4
e         Single replicate of a 2x2x2x2x2 factorial in blocks of size 8
f         Single replicate of a 2x2x2x2x2x2 factorial in blocks of size 8
g         Single replicate of a 3x3x3 factorial in blocks of size 9
h         Single replicate of a 3x3x3x3 factorial in blocks of size 9
i         Three replicates of a 2x2x3 factorial in blocks of size 6
j         Three replicates of a 2x2x2x3 factorial in blocks of size 6
k         Single replicate of a 2x3x3 factorial in blocks of size 6
l         Single replicate of a 4x4 factorial in blocks of size 4
m         Single replicate of a 4x2x2 factorial in blocks of size 8
n         Three replicates of a 4x2x3 factorial in blocks of size 12
o         Single replicate of a 4x2x2x2 factorial in blocks of size 8
p         Half replicate of a 4x2x2x2x2 factorial in blocks of size 8
Code (a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p) > g

    Which version(s) would you like:
1   different d.f. of A.B.C confounded with blocks: a+2b+2c
2   a+2b+c
3   a+b+2c
4   a+b+c
Numbers > 1,2,3,4

What would you like to call treatment factor 1?
Identifier > A

What would you like to call treatment factor 2?
Identifier > B

What would you like to call treatment factor 3?
Identifier > C

Pseudo-factors in the treatment formula will have suffixed identifiers
(for example pf[1], pf[2] ...). What identifier would you like to use?
Identifier (Default:Pseudofa) > PF

What would you like to call the replicate (version) factor?
Identifier (Default:Replicat) > Rep

What would you like to call the block factor?
```

```
Identifier (Default:Blocks) > Block

What would you like to call unit-within-block factor?
Identifier (Default:Units) > Plot

Seed for randomization (-1 for none)?
Number (Default: -1) > 349865

Do you want to print the design?
n         no
y         yes
Code (n,y; Default n) > y

Treatment combinations on each unit of the design
================================================

       Block   1       2       3
   Rep  Plot
     1     1   3 2 2   3 3 3   3 1 1
           2   2 3 3   1 1 3   1 3 3
           3   3 1 3   1 3 1   1 2 1
           4   1 3 2   2 1 1   2 2 2
           5   2 2 1   2 3 2   3 3 2
           6   3 3 1   3 2 1   3 2 3
           7   1 2 3   2 2 3   1 1 2
           8   1 1 1   1 2 2   2 1 3
           9   2 1 2   3 1 2   2 3 1
     2     1   1 1 2   1 2 3   2 2 1
           2   2 2 3   1 3 2   1 2 2
           3   3 1 3   2 3 1   2 3 3
           4   1 3 3   2 2 2   3 1 1
           5   3 3 1   2 1 3   3 2 3
           6   2 1 1   3 2 1   1 3 1
           7   2 3 2   1 1 1   2 1 2
           8   3 2 2   3 3 3   1 1 3
           9   1 2 1   3 1 2   3 3 2
     3     1   3 3 1   3 2 1   2 1 2
           2   1 2 2   3 3 2   3 1 1
           3   2 1 3   2 1 1   3 2 2
           4   1 1 1   1 3 1   2 3 1
           5   2 3 2   2 2 2   3 3 3
           6   3 2 3   2 3 3   2 2 3
           7   1 3 3   1 1 2   1 3 2
           8   3 1 2   3 1 3   1 1 3
           9   2 2 1   1 2 3   1 2 1
     4     1   3 1 3   2 3 3   3 3 3
           2   1 3 3   1 1 3   3 2 2
           3   3 3 2   2 1 1   2 3 2
           4   2 3 1   3 3 1   2 1 3
           5   2 1 2   1 3 2   1 1 2
           6   3 2 1   2 2 2   1 3 1
           7   1 2 2   1 2 1   3 1 1
           8   1 1 1   3 2 3   1 2 3
           9   2 2 3   3 1 2   2 2 1

Treatment factors are listed in the order: A, B, C.

Do you want to check the design by ANOVA?
n         no
y         yes
Code (n,y; Default n) > y

Analysis of variance
====================

Source of variation     d.f.

Rep stratum                3

Rep.Block stratum
A.B.C                      8

Rep.Block.Plot stratum
```

```
A                          2
B                          2
C                          2
A.B                        4
A.C                        4
B.C                        4
A.B.C                      8
Residual                  70

Total                    107


Information summary
===================

Model term                 e.f.  non-orthogonal terms

Rep.Block stratum
  A.B.C                 0.250

Rep.Block.Plot stratum
  A.B.C                 0.750  Rep.Block

> Do you want to form a unit label variate?
n         no
y         yes
Code (n,y; Default n) > n
```

The repertoire of factorial designs with confounding is accessed automatically by AGDESIGN from a subfile of one of the backing-store files that accompanies the procedure library. The backing-store file is specified by the FILENAME option. Its default file has four subfiles.

FACTORIAL – factorial designs (with confounding), as used in Example 4.9.3 above.

LATTICE – square lattice designs: designs for a single treatment factor with number of levels that is the square of some integer *k*; the design has replicates, each containing *k* blocks of *k* plots, and different treatment contrasts can be confounded with blocks in each replicate. (A wider selection of square lattices are available, however, from procedure AGSQLATTICE: see 4.9.6.)

LATTSQ – lattice squares: these are similar to lattices except that the blocking structure with the replicates has rows crossed with columns; again different treatment contrasts can be confounded with the rows and columns in each replicate. (AGSQLATTICE also provides a wider selection of lattice squares: see 4.9.6.)

LATIN – Latin squares: designs are available for 3 to 14 treatments; several different orthogonal squares are available for most of these so, for example, Graeco Latin squares can be formed by using a different square for each of the two treatment factors. (These are also available, however, from procedure AGLATIN: see 4.9.4.)

If the default FILENAME is being used, the usual abbreviation rules are used to match SUBFILE with the names of the subfiles in the default file, and the FACTORIAL subfile is taken by default.

You can also form your own repertoires of designs using the FDESIGNFILE procedure. This requires a data file, details of whose format can be obtained by setting option PRINT=filestructure when running FDESIGNFILE. Further information is given by Payne (1995), or in the description of procedure FDESIGNFILE in Part 3 of the *Genstat Reference Manual*.

AGDESIGN has two other options. The PRINT option can be set to design to print the plan of the design. By default, if you are running Genstat in batch, the plan is not printed. If you do not set PRINT when running interactively, AGDESIGN will ask whether or not you wish to print the design. The other setting catalogue lists the designs in the subfile. Similarly the ANALYSE option governs whether or not AGDESIGN produces a skeleton analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). Again

AGDESIGN assumes that this is not required if ANALYSE is unset in a batch run, and asks whether it is required if ANALYSE is unset in an interactive run.

The information required to select the design and give identifiers to its factors can be defined using the parameters of AGDESIGN. In an interactive run, as shown in Example 4.9.3, AGDESIGN will ask questions to obtain any necessary information that is not supplied in this way; when running in batch, if any of the required information has not been specified, AGDESIGN will terminate with a warning message.

The DESIGN parameter can supply a variate whose first value selects the "type" of design: for example, in the LATTICE subfile, this would select between a 3×3 lattice, a 4×4 lattice, and so on. Some of these designs are available in several different "versions": for example, in lattice designs there are several ways of defining which treatment contrasts are to be confounded with blocks. If there is more than one version, the second and subsequent values of the DESIGN variate indicate which version, or versions, are required. These need not be distinct so, for example, you can replicate a basic design several times. If the variate has a single value, AGDESIGN will select the first version.

The TREATMENTFACTORS parameter can specify a pointer to supply identifiers for the treatment factors in the design. For example, if there are two factors you could define their identifiers to be A and B by forming the pointer Tf (say) with the statement

```
POINTER [VALUES=A,B] Tf
```

and then setting TREATMENTFACTORS=Tf. Alternatively, and more succinctly, you could put TREATMENTFACTORS=!p(A,B), where !p(A,B) is an unnamed pointer containing the required two identifiers. Similarly the BLOCKFACTORS parameter can specify a pointer to define the identifiers for the block factors in the basic design. If you have requested several versions, or several replicates, of the basic design AGDESIGN will also need a factor to represent the replicates. The identifier of this factor can be supplied using the REPLICATEFACTOR parameter. Partially balanced designs, such as lattices, will require pseudo-factors in the treatment formula to enable the design to be analysed by ANOVA. Identifiers can be supplied for these using the PSEUDOFACTORS parameter.

The UNITLABELS parameter can specify a variate to store a unique number to label each of the plots in the design. In the first replicate (or version) in the generated design, the variate contains the numbers one up to the number of plots per replicate. The second replicate (if any) contains these numbers plus the smallest power of ten greater than the number of plots per replicate, the third replicate contains the numbers plus twice this power of ten, and so on.

The SEED parameter allows you to specify a seed to randomize the design. In a batch run, this has a default of –1, to suppress randomization. If SEED is unset in an interactive run, you will be asked to provide a seed (and again a negative value will leave the design unrandomized).

The STATEMENT parameter is useful when you are using AGDESIGN interactively. It allows you to save a Genstat text structure containing a command specifying the same information that you will have given in answer to the questions asked by AGDESIGN.

4.9.4 Latin and Youden squares

AGLATIN procedure

Generates mutually orthogonal Latin squares (I. Wakeling & R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (design, squares, list); if unset in an interactive run AGLATIN will ask whether the design is to be printed, in a batch run the default is not to print anything
ANALYSE = *string token*	Controls whether or not to analyse the design, and

	produce a skeleton analysis-of-variance table using ANOVA (no, yes); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run

Parameters

NROWS = *scalars*	Specifies the number of rows (and columns) in each square
NSQUARES = *scalars*	Number of squares to form (i.e. number of treatment factors to generate)
SEED = *scalars*	Seed to be used to randomize each design; a negative value implies no randomization
TREATMENTFACTORS = *pointers*	Pointer to identifiers for the treatment factors
ROWS = *factors*	Identifier for the row factor
COLUMNS = *factors*	Identifier for the column factor
MAXNSQUARES = *scalars*	Returns the maximum number of squares available with the specified number of rows and columns
STATEMENT = *texts*	Saves a command to recreate each design (useful if the design information has been specified in response to questions from AGLATIN)

AGLATIN generates a Latin square, or a set of orthogonal Latin squares (for example two orthogonal squares provides a Graeco-Latin square). If you are running Genstat interactively, you need not set any of the options or parameters of AGLATIN. The information required to generate the squares is then obtained by questions. You need set the parameters only if you wish to anticipate some of the questions, or if you wish to use AGLATIN in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The size of the squares (i.e. the number of rows and columns) can be specified by the NROWS parameter, and the number of squares (i.e. the number of treatment factors to be generated) can be specified by the NSQUARES parameter. The MAXNSQUARES parameter can be used to ascertain how many squares are available. If this is set but NSQUARES is not set, the procedure then stops. Otherwise, when AGLATIN is being used interactively, if NSQUARES is unset you will be asked how many squares you want.

The squares are represented as a row factor, a column factor and NSQUARES treatment factors all of length NROWS**2. The ROWS and COLUMNS parameters can supply identifiers for the row and column factors, so that they are accessible outside the procedure. The TREATMENTFACTORS parameter can specify a pointer to supply identifiers for the treatment factors. For example, if there is one factors you could define its identifiers to be Treat by forming the pointer Tf (say) with the statement

```
POINTER [VALUES=Treat] Tf
```

and then setting TREATMENTFACTORS=Tf. Alternatively, and more succinctly, you could put TREATMENTFACTORS=!p(Treat), where !p(Treat) is an unnamed pointer containing the required identifier, see line 4 of Example 4.9.4. Similarly you can have a pointer with two identifiers if there are two treatment factors, as in line 7 of Example 4.9.4.

The SEED parameter allows you to specify a seed to randomize the design. In a batch run, this has a default of –1, to suppress randomization. If SEED is unset in an interactive run, you will be asked to provide a seed (and again a negative value will leave the design unrandomized).

The PRINT option controls whether AGLATIN prints the design. The setting design prints it as a square table of treatment factors tabulated by the row and column factors, squares prints each treatment factor separately (again tabulated by rows and columns), and list prints row,

column and treatment factor values as a list. By default, if you are running Genstat in batch, the nothing is printed. If you do not set PRINT when running interactively, AGLATIN will ask what you want to print. Similarly the ANALYSE option governs whether or not AGLATIN produces a skeleton analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). Again AGLATIN assumes that this is not required if ANALYSE is unset in a batch run, and asks whether it is required if ANALYSE is unset in an interactive run.

AGLATIN generates the squares using Galois fields, obtained from procedure GALOIS. Details are given in the description of AGLATIN in Part 3 of the *Genstat Reference Manual*.

Example 4.9.4a uses AGLATIN to generate a six by six Latin square and then a 12 by 12 Graeco-Latin square (that is a square design with two orthogonal treatment factors, here called Latin and Graeco).

Example 4.9.4a

```
   2  " 6 x 6 Latin square."
   3  AGLATIN  [PRINT=design; ANALYSE=yes] 6; NSQUARES=1;\
   4    TREATMENTFACTOR=!p(Treat); ROWS=Rows; COLUMNS=Columns; SEED=876413

Treatments on each unit of the design
=====================================

 Columns   1   2   3   4   5   6
    Rows
       1   5   6   4   3   1   2
       2   3   1   2   4   5   6
       3   1   2   3   5   6   4
       4   2   3   1   6   4   5
       5   4   5   6   2   3   1
       6   6   4   5   1   2   3

Treatment factor: Treat.


Analysis of variance
====================

Source of variation     d.f.

Rows stratum               5

Columns stratum            5

Rows.Columns stratum
Treat                      5
Residual                  20

Total                     35


   5  " 7 x 7 Graeco-Latin square."
   6  AGLATIN  [PRINT=design; ANALYSE=yes] 7; NSQUARES=2;\
   7    TREATMENTFACTORS=!p(Latin,Graeco); ROWS=Rows; COLUMNS=Columns;\
   8    SEED=712753

Treatment combinations on each unit of the design
=================================================

 Columns   1     2     3     4     5     6     7
    Rows
       1   1 2   6 5   2 4   4 1   3 6   5 3   7 7
       2   2 3   7 6   3 5   5 2   4 7   6 4   1 1
       3   3 4   1 7   4 6   6 3   5 1   7 5   2 2
       4   7 1   5 4   1 3   3 7   2 5   4 2   6 6
       5   6 7   4 3   7 2   2 6   1 4   3 1   5 5
       6   4 5   2 1   5 7   7 4   6 2   1 6   3 3
       7   5 6   3 2   6 1   1 5   7 3   2 7   4 4

Treatment factors are listed in the order: Latin, Graeco.
```

```
Analysis of variance
====================

Source of variation     d.f.

Rows stratum               6

Columns stratum            6

Rows.Columns stratum
Latin                      6
Graeco                     6
Residual                  24

Total                     48
```

AGCROSSOVERLATIN procedure

Generates Latin squares balanced for carry-over effects (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (design); if unset in an interactive run ACROSSOVERGLATIN will ask whether the design is to be printed, in a batch run the default is not to print anything
ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (yes, no); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run

Parameters

LEVELS = *scalars* or *variates*	Number of treatments (scalar) or levels for the treatments
SEED = *scalars*	Seed to be used to randomize the design; a negative value implies no randomization
TREATMENTS = *factors*	Identifier for a factor to represent the direct effects of the treatments
SUBJECTS = *factors*	Identifier for a factor to represent the subjects
PERIODS = *factors*	Identifier for a factor to represent the periods
CARRYOVERFACTOR = *factors*	Identifier for a factor to represent the carry-over (or "residual") effect of the treatments in the period immediately after the period in which they were applied
NOCARRYOVER = *factors*	Identifier for a factor to represent the comparison between none and any carry-over effect of the treatments
STATEMENT = *texts*	Saves a command to recreate each design (useful if the design information has been specified in response to questions from AGCROSSOVERLATIN)

Genstat can also generate the specialized Latin squares that are used for cross-over trials. These are designed to study the effects of various treatments on a set of plots (in a field experiment) or subjects (in a medical trial). The special feature of these experiments is that the same plots or subjects are treated during several successive time periods, and there is interest both in the direct effect of a treatment during the period in which it is applied and its carry-over (or

"residual") effect during later periods. AGCROSSOVERLATIN can generate designs for a single treatment factor for the most usual situation, where the carry-over effect is assumed to last over only one subsequent period. The design balances the direct and carry-over effects by ensuring that each treatment follows each other treatment an equal number of times. For an even number of treatments t the design consists of a single $t \times t$ Latin square, while for an odd number t it is formed from a pair of Latin squares.

The design can be analysed by ANOVA by setting

```
BLOCKSTRUCTURE Subjects * Periods
TREATMENTSTRUCTURE Nocarryover / Carryover + Treatments
```

The factor Carryover represents the carry-over effects of the treatments, and factor Nocarryover assesses whether there were any carry-over effects at all (essentially this is a comparison between the periods 2 onwards where there were carry-over effects from earlier times, and period 1 where there was none). So the treatment formula expands to specify terms

Nocarryover	none versus any carry-over effect
Nocarryover.Carryover	differences in carry-over effect amongst the treatments (assuming that there was an earlier treatment)
Treatments	direct effects of treatments, eliminating any carry-over effect

The direct and carry-over effects are not orthogonal, so it may be of interest also to specify

```
TREATMENTSTRUCTURE Treatments + Nocarryover / Carryover
```

in order to estimate the carry-over effects eliminating the direct effects.

AGCROSSOVERLATIN operates similarly to AGLATIN. If it is used interactively the information required to generate the design can be obtained by questions. You need set the parameters only if you wish to anticipate some of the questions, or if you wish to use AGCROSSOVERLATIN in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The number of treatments can be defined using the LEVELS parameter. The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is –1, to suppress randomization. If you do not set SEED when running interactively AGCROSSOVERLATIN will ask for a seed, and again a negative value suppresses any randomization.

Parameters TREATMENTS, CARRYOVERFACTOR and NOCARRYOVER allow you to specify identifiers for factors to represent the direct effects of the treatments, the carry-over effects in the subsequent period, and the comparison between none and any carry-over effect. Similar the parameters SUBJECTS and PERIODS can specify identifiers for factors to represent the subjects (or plots) and time periods respectively. If these parameters are not specified in a batch run, AGCROSSOVERLATIN will use identifiers that are local within the procedure and thus lost at the end of the procedure. If you are running interactively, AGCROSSOVERLATIN will ask you to provide identifiers, and these will remain available after AGCROSSOVERLATIN has finished running.

The PRINT options can be set to design to print the design. By default, if you are running Genstat in batch, the nothing is printed. If you do not set PRINT when running interactively, AGCROSSOVERLATIN will ask what you want to print. The ANALYSE option similarly controls whether AGCROSSOVERLATIN produces a dummy analysis-of
-variance table, exactly as in AGLATIN.

Example 4.9.4b generates a cross-over design for five treatments. This is based on two Latin squares, and so there are ten subjects (and five periods).

Example 4.9.4b

```
   2  AGCROSSOVERLATIN [PRINT=design; ANALYSE=yes] 5; SEED=33841;\
   3    TREATMENTS=Direct; CARRYOVERFACTOR=Carryover;\
   4    NOCARRYOVER=Nocarryover; SUBJECTS=Subjects; PERIODS=Periods

Treatment combinations on each unit of the design
=================================================

  Periods   1       2       3       4       5
 Subjects
        1   3 1 0   4 2 3   5 2 4   1 2 5   2 2 1
        2   3 1 0   5 2 3   4 2 5   2 2 4   1 2 2
        3   2 1 0   5 2 2   1 2 5   3 2 1   4 2 3
        4   1 1 0   2 2 1   4 2 2   5 2 4   3 2 5
        5   2 1 0   1 2 2   5 2 1   4 2 5   3 2 4
        6   4 1 0   3 2 4   1 2 3   5 2 1   2 2 5
        7   5 1 0   3 2 5   2 2 3   4 2 2   1 2 4
        8   1 1 0   4 2 1   2 2 4   3 2 2   5 2 3
        9   4 1 0   1 2 4   3 2 1   2 2 3   5 2 2
       10   5 1 0   2 2 5   3 2 2   1 2 3   4 2 1

Treatment factors are listed in the order: Direct, Nocarryover, Carryover.

 Labels of Nocarryover:
          1 no carry-over
          2    carry-over


* MESSAGE: non-orthogonality between treatment terms. The effects (printed or
used to calculate means), the efficiency factor and the sum of squares for each
treatment term are for that term eliminating previous terms in the TREATMENT
formula and ignoring subsequent terms.


Analysis of variance
====================

Source of variation     d.f.

Subjects stratum
Nocarryover.Carryover      4
Residual                   5

Periods stratum
Nocarryover                1
Residual                   3

Subjects.Periods stratum
Nocarryover.Carryover      4
Direct                     4
Residual                  28

Total                     49


Information summary
===================

Model term                e.f.  non-orthogonal terms

Subjects stratum
  Nocarryover.Carryover  0.050

Subjects.Periods stratum
  Nocarryover.Carryover  0.950  Subjects
  Direct                 0.947  Nocarryover.Carryover
```

AGQLATIN procedure

Generates complete and quasi-complete Latin squares (R.W. Payne).

Options

PRINT = *string token*	Controls printing of the design (design); if unset in an interactive run AGQLATIN will ask whether the design is to be printed, in a batch run the default is not to print anything
ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (no, yes); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run

Parameters

NROWS = *scalars*	Specifies the number of rows (and columns) in the square
SEED = *scalars*	Seed to be used to randomize each design; a negative value implies no randomization
TREATMENTS = *factors*	Identifier for the treatment factor
ROWS = *factors*	Identifier for the row factor
COLUMNS = *factors*	Identifier for the column factor
STATEMENT = *texts*	Saves a command to recreate each design (useful if the design information has been specified in response to questions from AGQLATIN)

A complete Latin square is a Latin square in which each ordered pair of treatments appears exactly once within the rows of the square, and exactly once within the columns. For example, in the four-by-four square below, the pair (1,2) is in row 1 (and only in row 1) while the pair (2,1) is only in row 4. Likewise (1,2) is only only in column 1 and (2,1) only in column 4.

```
Columns   1   2   3   4
   Rows
      1   1   2   4   3
      2   2   3   1   4
      3   4   1   3   2
      4   3   4   2   1
```

A quasi-complete Latin has similar properties, but here each unordered pair occurs exactly twice within the rows, and exactly twice within the columns. See, for example, the five-by-five Latin square below.

```
Columns   1   2   3   4   5
   Rows
      1   1   2   5   3   4
      2   2   3   1   4   5
      3   5   1   4   2   3
      4   3   4   2   5   1
      5   4   5   3   1   2
```

Complete Latin squares can be constructed for any even number of rows, while quasi-complete squares are available for any odd number of rows. They are constructed using the method of Williams (1949), which is based upon terraced groups (Bailey 1984). Designs based on these squares are useful for example in experiments where there is the possibility of interference between a plot and its neighbours. Complete Latin squares should be used if the interference is likely to be directional, as for example in a field experiment to assess fungicides where spores

may be carried from one plot to another by a prevailing wind. Otherwise the choice of design will depend upon wether an odd or even number of treatments is required.

If you are running Genstat interactively, you need not set any of the options or parameters of AGQLATIN. All the information required to generate the design is then obtained by a series of questions. You need set the parameters only if you wish to anticipate some of the questions, or if you wish to use AGQLATIN in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The size of the square (i.e. the number of rows and columns) can be specified by the NROWS option. The ROWS, COLUMNS and TREATMENTS parameters can supply identifiers for the row, column and treatment factors, so that they are accessible outside the procedure.

The SEED parameter allows you to specify a seed to randomize the design, by making a random permutation of the treatment labels. In a batch run, SEED has a default of –1, to suppress randomization. If SEED is unset in an interactive run, you will be asked to provide a seed (and again a negative value will leave the design unrandomized).

The PRINT option can be set to design to print the design. By default, if you are running Genstat in batch, the nothing is printed. If you do not set PRINT when running interactively, AGQLATIN will ask what you want to print. Similarly the ANALYSE option governs whether or not AGQLATIN produces a skeleton analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). Again AGQLATIN assumes that this is not required if ANALYSE is unset in a batch run, and asks whether it is required if ANALYSE is unset in an interactive run.

AGYOUDENSQUARE procedure

Generates a Youden square (W. van den Berg).

Options

PRINT = *string tokens*	Controls printed output (design, lambda, list); default is to ask what to print if this is unset in an interactive run, in a batch run the default is not to print anything
ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (no, yes); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run

Parameters

NROWS = *scalars*	Specifies the number of rows in the square
NCOLUMNS = *scalars*	Specifies the number of columns (and treatments) in the square
SEED = *scalars*	Seed to be used to randomize each design; a negative value implies no randomization
LAMBDA = *scalars*	Saves the number of times each pair of treatments occurs in the same column
TREATMENTS = *factors*	Identifiers for the treatment factor
ROWS = *factors*	Identifier for the row factor
COLUMNS = *factors*	Identifier for the column factor

STATEMENT = *texts*	Saves a command to recreate each design (useful if the design information has been specified in response to questions from AGYOUDENSQUARE)

The simplest Youden square is formed by taking a Latin square, and deleting one of the rows. Each treatment occurs once in every row, and each pair of treatments occurs the same number of times in the columns. So the columns are like the blocks of a balanced-incomplete-block design (4.9.8), and the design can be analysed by ANOVA. In this way, AGYOUDENSQUARE can form any design where the number of rows is one less than the number of columns. Other squares are provided by tables in Cochran & Cox (1957) pages 522-535. The available combinations are tabulated below.

Number of columns	Number of rows
7	3 or 4
11	5 or 6
13	4 or 9
15	7 or 8
16	6 or 10
19	9 or 10
21	5
25	9
31	6 or 10
37	9
57	8
73	9
91	10

AGYOUDENSQUARE is easiest to use interactively. All the information required to generate the squares is then obtained by (clearly explained) questions. You need set the parameters only if you wish to anticipate some of the questions, or if you wish to use AGYOUDENSQUARE in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command to specify the same information.

The number of rows can be specified by the NROWS parameter, and the number of columns can be specified by the NCOLUMNS parameter. The number of rows must be less than the number of columns. The number of columns defines the number of treatments, and the number of rows defines the number of replicates.

The TREATMENTS, ROWS and COLUMNS and parameters can supply identifiers for the row, column and treatment factors, so that they are accessible outside the procedure.

The SEED parameter can specify a seed to randomize the design. In a batch run, this has a default of –1, to suppress randomization. If SEED is unset in an interactive run, you will be asked to provide a seed (and again a negative value will leave the design unrandomized).

The PRINT option controls the printed output, with settings:

design	to print the design as a square table of treatment factors tabulated by the row and column factors;
lambda	to the number of times each pair of treatments occurs in the same column; and
list	to print the row, column and treatment factor values as a list.

By default, if you are running Genstat in batch, nothing is printed. If you do not set PRINT when running interactively, AGYOUDENSQUARE will ask what you want to print.

Similarly the ANALYSE option governs whether or not AGYOUDENSQUARE produces a skeleton analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). Again AGYOUDENSQUARE assumes that this is not required if ANALYSE is unset in a batch run, and asks whether it is required if ANALYSE is unset in an interactive run.

The LAMBDA parameter can save the number of times each pair of treatments occurs in the same column. This is given by

```
LAMBDA  =  NROWS * (NROWS - 1) / (NCOLUMNS - 1)
```

Example 4.9.4c constructs a Youden square with 5 rows and 6 columns.

Example 4.9.4b

```
  2  AGYOUDENSQUARE [PRINT=design; ANALYSE=yes] NROWS=5; NCOLUMNS=6;\
  3    TREATMENTS=Treatments; ROWS=Rows; COLUMNS=Columns; SEED=-1

Treatments on each unit of the design
=====================================


 Columns   1   2   3   4   5   6
    Rows
       1   1   2   3   4   5   6
       2   2   3   1   5   6   4
       3   3   1   2   6   4   5
       4   4   5   6   1   2   3
       5   5   6   4   2   3   1


Treatment factor: Treatments.


Analysis of variance
====================

Source of variation     d.f.

Rows stratum               4

Columns stratum
Treatments                 5

Rows.Columns stratum
Treatments                 5
Residual                  15

Total                     29


Information summary
===================

Model term                e.f.  non-orthogonal terms

Columns stratum
  Treatments             0.040
```

```
Rows.Columns stratum
  Treatments             0.960  Columns
```

4.9.5 Semi-Latin squares

An $(n \times n)/k$ semi-Latin square is like an $n \times n$ Latin square except that there are k letters in each cell. The combinations of the rows and columns of a semi-Latin square are called blocks. Each of the $n \times k$ letters occurs once in each row and once in each column. The design thus has n rows and columns, k (sub-) units within each row × column combination (or block), and $n \times k$ treatments. The analysis should contain strata for rows, columns, rows.columns and rows.columns.units, as well as treatment effects which may be estimated in either the rows.columns or the rows.columns.units strata. Procedure AGSEMILATIN can construct three types of semi-Latin square.

Trojan squares: a Trojan square consist of a set of k mutually orthogonal $n \times n$ Latin squares, on k disjoint sets of treatments. Each block of the semi-Latin square contains the treatments which occur in the corresponding cell of all the individual squares (Bailey 1988). AGSEMILATIN can construct Trojan squares for any value of n for which a Graeco-Latin square exists. Thus, for example, no Trojan square exists for $n = 6$. In a Trojan square k must be greater than 1 and less than n (Edmondson 1998), and for some values of n, k must be less than that. The maximum values of k for n up to 15 for a Trojan square are

n:	3	4	5	7	8	9	11	12	13	14	15
k:	2	3	4	6	6	8	10	2	12	2	2

In a Trojan square, some treatment effects are estimated in both the rows.columns and the rows.columns.units strata, while others (which need to be represented by a pseudo-factor) are estimated only in the rows.columns.units stratum. Trojan squares are optimal semi-Latin squares (Bailey 1992).

Inflated Latin squares: an $(n \times n)/k$ inflated Latin square consists of an $n \times n$ Latin square with each letter replaced by k new symbols (Bailey 1988). AGSEMILATIN can construct inflated Latin squares for any value of n greater than 2, and any value of k greater than 1. The analysis requires a pseudo-factor to distinguish the treatment contrasts that are estimated in the rows.columns stratum from those estimated in the rows.columns.units stratum.

Interleaving Latin squares: these are formed similarly to the Trojan square, except that there is no longer the requirement for the k Latin squares to be orthogonal (Bailey 1988). If the squares are orthogonal, the design is a Trojan square and can be analysed by ANOVA with the help of a pseudo-factor as described above. For n=2 the design is an inflated Latin square and can be analysed by ANOVA, again with the help of a pseudo-factor. Otherwise, the design is unbalanced. It is possible to generate a balanced analysis by omitting the row.column stratum, but this is not reasonable and Yates (1935) advises against such an analysis. AGSEMILATIN can construct interleaving Latin squares for any value of n or k greater than 1.

AGSEMILATIN procedure

Generates semi-Latin squares (W. van den Berg).

Options

PRINT = *string token*	Controls whether or not to print a plan of the design (design); if unset in an interactive run AGSEMILATIN will ask whether the design is to be printed, in a batch run the default is not to print anything
METHOD = *string token*	Method to use to construct the semi-Latin square (Trojan, interleaving, inflated); if unset in an interactive run AGSEMILATIN will ask what type is required, in a batch run the default is Trojan

ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (no, yes); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run

Parameters

NROWS = *scalars*	Number of rows and columns of the semi-Latin square
NUNITS = *scalars*	Number of units (i.e. treatments) within each block
SEED = *scalars*	Seed for randomization; a negative value implies no randomization
TREATMENTS = *factors*	Identifier for the treatment factor
ROWS = *factors*	Identifier for the row factor
COLUMNS = *factors*	Identifier for the column factor
UNITS = *factors*	Identifier for the unit factor
PSEUDOFACTOR = *factors*	Identifier for the pseudo-factor
STATEMENT = *texts*	Saves a command to recreate the design (useful if the design information has been specified in response to questions from AGSEMILATIN)

AGSEMILATIN generates the factors and pseudo-factor required to define a semi-Latin square. It also sets the block and treatment formulae (using the BLOCKSTRUCTURE and TREATMENTSTRUCTURE directives) to allow the design, if balanced, to be analysed by ANOVA.

The type of semi-Latin square can be chosen using the METHOD option with setting either Trojan, inflated, or interleaving. In a batch run the default is Trojan, while in an interactive run AGSEMILATIN will ask what type you want. AGSEMILATIN has two other options. The PRINT option can be set to design to print the plan of the design. By default, if you are running Genstat in batch, the plan is not printed. If you do not set PRINT when running interactively, AGSEMILATIN will ask whether or not you wish to print the design. Similarly the ANALYSE option governs whether or not AGSEMILATIN produces a skeleton analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). Again AGSEMILATIN assumes that this is not required if ANALYSE is unset in a batch run, and asks whether it is required if ANALYSE is unset in an interactive run.

The information required to select the design and give identifiers to its factors can be defined using the parameters of AGSEMILATIN. The number of rows and columns of the design (n) can be defined using the parameter NROWS. Similarly, the number of units (k) for each row-column combination (that is, the number of treatments per block) can be defined by the parameter NUNITS. Parameters TREATMENTS, ROWS, COLUMNS, UNITS and PSEUDOFACTOR allow you to specify identifiers for the treatment, row, column and unit factors, and for the pseudo-factor. The SEED parameter allows you to specify a seed to randomize the design. In a batch run, this has a default of −1, to suppress randomization. If SEED is unset in an interactive run, you will be asked to provide a seed (and again a negative value will leave the design unrandomized). If one of the other parameters is unset in an interactive run, you will be asked to provide a name.

The STATEMENT parameter allows you to save a Genstat text structure containing a command to recreate the design. This is particularly useful when you are running AGSEMILATIN interactively, and specifying the information in response to questions.

The various types of square are illustrated in Example 4.9.5.

Example 4.9.5

```
 2  AGSEMILATIN [PRINT=design; METHOD=trojan; ANALYSE=yes]\
 3    NROWS=5; NUNITS=4; SEED=135143; TREATMENTS=Treat;\
```

```
   4    COLUMNS=Column; ROWS=Row; UNITS=Plot; PSEUDOFACTOR=Pseudo

Trojan Square: NROWS (n) = 5, NUNITS (k) = 4.

                  [1]
            Column   1   2   3   4   5
      Row     Plot
        1        1   3  11  13   8   5
                 2   9  17   4  20   7
                 3  19  15  12  18  16
                 4   1  10   2  14   6
        2        1   2  20  19  12   4
                 2   6  13  15   9  17
                 3  11   7  14  16   8
                 4  18   1   5  10   3
        3        1   5   6   1   2   9
                 2  10  12   8  19  13
                 3  20  14  16   7  15
                 4   4   3  11  17  18
        4        1  12  16  17  11   1
                 2  15  19   9   3  14
                 3   7  18  20   5   2
                 4   8   4   6  13  10
        5        1  14   2   7   6  20
                 2  16   8  10   1  12
                 3  17   9  18   4  11
                 4  13   5   3  15  19


Analysis of variance
====================

Source of variation     d.f.

Row stratum               4

Column stratum            4

Row.Column stratum
Treat                    16

Row.Column.Plot stratum
Treat                    19
Residual                 56

Total                    99


Information summary
===================

Model term               e.f.  non-orthogonal terms

Row.Column stratum
  Treat                 0.250

Row.Column.Plot stratum
  Treat                 0.750  Row.Column

   5  AGSEMILATIN [PRINT=design; METHOD=inflated; ANALYSE=yes]\
   6    NROWS=5; NUNITS=4; SEED=314612; TREATMENTS=Treat;\
   7    COLUMNS=Column; ROWS=Row; UNITS=Plot; PSEUDOFACTOR=Pseudo

Inflated Latin Square: NROWS (n) = 5, NUNITS (k) = 4.

                  [1]
            Column   1   2   3   4   5
      Row     Plot
        1        1   9   7  14  15  16
                 2   4   8  17  19  20
                 3   3  10   1  11   5
                 4   2  13  12   6  18
        2        1   8  15   9  16  12
```

```
                2  10   6   4   5  17
                3   7  19   3  20   1
                4  13  11   2  18  14
        3       1   5  14  11   2  10
                2  18  17  19   9   7
                3  16   1  15   3  13
                4  20  12   6   4   8
        4       1   1   4   5  13  19
                2  17   9  20   8  15
                3  12   3  16  10  11
                4  14   2  18   7   6
        5       1  15  16   8  17   4
                2  19  20   7   1   3
                3  11   5  10  14   9
                4   6  18  13  12   2


Analysis of variance
====================

Source of variation     d.f.

Row stratum                4

Column stratum             4

Row.Column stratum
Treat                      4
Residual                  12

Row.Column.Plot stratum
Treat                     15
Residual                  60

Total                     99


   8  AGSEMILATIN [PRINT=design; METHOD=interleaving; ANALYSE=yes]\
   9    NROWS=5; NUNITS=6; SEED=235978; TREATMENTS=Treat;\
  10    COLUMNS=Column; ROWS=Row; UNITS=Plot; PSEUDOFACTOR=Pseudo

Interleaving Latin Square: NROWS (n) = 5, NUNITS (k) = 6.

                [1]
           Column   1   2   3   4   5
      Row     Plot
        1        1  27  21  11   5   3
                 2   4  28  20   9  17
                 3  14  18   2  12  30
                 4   8  24  13   7  23
                 5  16  15  29  26  22
                 6   6   1  10  25  19
        2        1  20  23   7  27   6
                 2  28  25   3  29  12
                 3  26   4  24   2  15
                 4   9   5  17   1  14
                 5  22  11  21  19  18
                 6  13   8  16  30  10
        3        1  15  17   1  10   2
                 2  11  13   9  21  16
                 3   7  12   6   8   5
                 4  18   3  14  22  28
                 5  30  20  26   4  29
                 6  19  27  23  24  25
        4        1  23  19   5  17  13
                 2   2   9  15  28   4
                 3  21  30  18  11   8
                 4  29  10  25   3  20
                 5  12  16  27   6   1
                 6  24  26  22  14   7
        5        1   1   7  12  18  26
                 2  10   6  19  16   9
                 3  17  29  28  15  11
```

```
4   3  14   4  23  27
5   5   2   8  13  24
6  25  22  30  20  21

******** Warning from AGSEMILATIN:
Interleaved Latin Squares with 5 rows and columns and 6 columns are
unbalanced, and so cannot be analysed by ANOVA.
```

4.9.6 Square lattice and lattice square designs

A square lattice is a design for a single treatment factor with a number of levels that is the square of some integer *k*; the design has replicates, each containing *k* blocks of *k* units (or plots), and different treatment contrasts are confounded with blocks in each replicate. The block structure of the design is thus

```
Replicates / Blocks / Units
```

The lattice square is similar, but it has a row-by-column structure with *k* rows and *k* columns within each replicate. So the block structure is now

```
Replicates / (Rows * Columns)
```

These designs can be generated by the procedure AGSQLATTICE. They are used, for example, in variety trials where there are many treatments to examine and the variability of the units is such that the block size needs to be kept reasonably small. For some numbers of treatments, it is possible to generate enough different replicates so that every treatment contrast is confounded with blocks in one of the replicates of a square lattice, or with rows and with columns in one of the replicates of a lattice square. The design is then balanced. If insufficient replicates are available, or if you choose to use less than the full set available, the design is unbalanced and needs pseudofactors for its analysis by the ANOVA directive. However, AGSQLATTICE can generate these for you automatically.

AGSQLATTICE procedure

Generates square lattice and lattice square designs (R.W. Payne).

Options

PRINT = *string token*	Controls whether or not to print a plan of the design (design); if unset in an interactive run AGSQLATTICE will ask whether the design is to be printed, in a batch run the default is not to print the design
ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (no, yes); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run
DESIGNTYPE = *string token*	What type of design to form (squarelattice, latticesquare); default squa

Parameters

LEVELS = *scalars*	Number of treatments in each design
NREPLICATES = *scalars*	Number of replicates in each design, taken by default to be the maximum number available in a batch run
SEED = *scalars*	Seed for randomization; a negative value implies no randomization
TREATMENTS = *factors*	Identifier for the treatment factor for each design
PSEUDOFACTORS = *pointers*	Identifier for the pseudofactors required if the design is

	not a balanced lattice
REPLICATES = *factors*	Identifier for the replicate factor for each design
BLOCKS = *factors*	Identifier for the factor to index the blocks within replicates of each design
ROWS = *factors*	Identifier for the factor to index the rows within replicates of a lattice square
COLUMNS = *factors*	Identifier for the factor to index the columns within replicates of a lattice square
UNITS = *factors*	Identifier for the factor to index the units (or plots) within the blocks of each design
STATEMENT = *texts*	Saves a command to recreate the design (useful if the design information has been specified in response to questions from AGSQLATTICE)
EXCLUDEREPLICATES = *scalars* or *variates*	Replicates to exclude during randomization

If you are running Genstat interactively, you need not set any of the options or parameters of AGSQLATICE. The information required to generate the design is then obtained by a series of questions. Its options and parameters allow you to anticipate questions, or to define all the necessary information if you want to use AGSQLATTICE in batch. However, if you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The DESIGNTYPE option controls whether a square lattice or a lattice square is generated. By default, if you are running Genstat in batch, a square lattice is generated. If you do not set DESIGNTYPE when running interactively, AGSQLATTICE will ask what sort of design you want.

The number of treatments can be defined using the LEVELS parameter. Similarly, the NREPLICATES parameter can define the number of replicates; by default, in a batch run, the maximum available number of replicates is formed. The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is −1, to suppress randomization. If you do not set SEED when running interactively AGSQLATTICE will ask for a seed, and again a negative value suppresses any randomization. You can use the EXCLUDEREPLICATES parameter to specify a scalar or variate giving numbers of replicates that you do not wish to randomize. (This can be useful in "demonstration experiments", when the treatments may need to be kept in a systematic order in some parts of the trial, but it is not a good idea in more normal situations.)

The TREATMENTS and REPLICATES parameters allow you to specify identifiers for the treatment and replicate factors, and the PSEUDOFACTORS parameter allows you to specify a pointer to represent the pseudo-factors if these are required. The BLOCKS and UNITS parameters specify identifiers for the block-within-replicate and unit-within-block factors of a square lattice, while the ROWS and COLUMNS parameters specify identifiers for the row- and column-within-replicate factors of a lattice square. If any of these parameters is not specified in a batch run, AGSQLATTICE will use an identifier that is local within the procedure and thus lost at the end of the procedure. If you are running interactively, AGSQLATTICE will ask you to provide identifiers, and these will remain available after it has finished running.

AGSQLATTICE has a PRINT option which can be set to design to print the plan of the design. By default, if you are running Genstat in batch, the plan is not printed. If you do not set PRINT when running interactively, AGSQLATTICE will ask whether or not you wish to print the design. Similarly the ANALYSE option governs whether or not AGSQLATTICE produces a skeleton analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). Again AGSQLATTICE assumes that this is not required if ANALYSE is unset in a batch run, and asks whether it is required if ANALYSE is unset in an interactive run.

Example 4.9.6 generates a 5 by 5 square lattice with three replicates.

Example 4.9.6

```
   2  " 5 x 5 Square lattice with 3 replicates."
   3  AGSQLATTICE [PRINT=design; ANALYSE=yes; DESIGNTYPE=squarelattice] 25;\
   4    NREPLICATES=3; SEED=-1; TREATMENTS=variety; PSEUDOFACTORS=pf;\
   5    REPLICATES=rep; BLOCKS=block; UNITS=plot

Treatments on each unit of the design
=====================================

        plot    1    2    3    4    5
   rep block
     1     1    1    2    3    4    5
           2    6    7    8    9   10
           3   11   12   13   14   15
           4   16   17   18   19   20
           5   21   22   23   24   25
     2     1    1    6   11   16   21
           2    2    7   12   17   22
           3    3    8   13   18   23
           4    4    9   14   19   24
           5    5   10   15   20   25
     3     1    1   10   14   18   22
           2    2    6   15   19   23
           3    3    7   11   20   24
           4    4    8   12   16   25
           5    5    9   13   17   21

Treatment factor: variety.


Analysis of variance
====================

Source of variation     d.f.

rep stratum                2

rep.block stratum
variety                   12

rep.block.plot stratum
variety                   24
Residual                  36

Total                     74


Information summary
===================

Model term                e.f.  non-orthogonal terms

rep.block stratum
  pf[1]                  0.333
  pf[2]                  0.333
  pf[3]                  0.333

rep.block.plot stratum
  pf[1]                  0.667  rep.block
  pf[2]                  0.667  rep.block
  pf[3]                  0.667  rep.block

   6  ASTATUS

Treatment structure:  variety // pf[1],pf[2],pf[3]
Block structure:  rep/block/plot
Covariates: not set
```

4.9.7 Alpha designs

Alpha designs form a very flexible class of resolvable incomplete block designs. A resolvable design is one in which each block contains only a selection of the treatments, but the blocks can be grouped together into subsets in which each treatment is replicated once. The groupings of blocks thus form replicates, and the block structure of the design is

```
Replicates / Blocks / Units
```

Such designs are particularly useful when there are many treatments to examine and the variability of the units is such that the block size needs to be kept small. Alpha designs were thus devised originally for the analysis of plant breeding trials (Patterson & Williams 1976), where many varieties may need to be evaluated in a single trial, and have the advantage that they can provide effective designs for any number of treatments. The designs are unbalanced, and are analysed using REML (Chapter 5).

The formation of an alpha design requires a generating array, and the effectiveness of the design that is produced will be very dependent on the choice of array. Procedure AGALPHA provides arrays for up to 100 treatments.

AGALPHA procedure

Forms alpha designs by standard generators for up to 100 treatments (M.F. Franklin & R.W. Payne).

Option

PRINT = *string token*	Controls whether or not to print a plan or the generator of of the design (design, generator); if unset in an interactive run AGALPHA will ask whether the design and generator are to be printed, in a batch run the default is not to print anything

Parameters

LEVELS = *scalars*	Number of treatments
NREPLICATES = *scalars*	Number of replicates
NBLOCKS = *scalars*	Number of blocks per replicate
SEED = *scalars*	Seed for randomization; a negative value implies no randomization
TREATMENTS = *factors*	Identifier for the treatment factor
REPLICATES = *factors*	Identifier for the replicate factor
BLOCKS = *factors*	Identifier for the factor to index the blocks within replicates
UNITS = *factors*	Identifier for the factor to index the units (or plots) within each block
STATEMENT = *texts*	Saves a command to recreate each design (useful if the design information has been specified in response to questions from AGALPHA)

If you are running Genstat interactively, you need not set any of the options or parameters of AGALPHA. It then asks questions to determine the necessary information to select the generating array: for example, the number of treatments, the number of blocks per replicate and so on. The parameters allow you to anticipate questions, or to define all the necessary information if you want to use AGALPHA in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The number of treatments can be defined using the LEVELS parameter. Similarly, the NREPLICATES and NBLOCKS parameters define the number of replicates and the number of blocks per replicate. If the number of blocks per replicate is greater than or equal to the number of units (or plots) per block, generators are available for either two, three or four replicates; otherwise there can only be two. The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is -1, to suppress randomization. If you do not set SEED when running interactively AGALPHA will ask for a seed, and again a negative value suppresses any randomization. The remaining parameters, TREATMENTS, REPLICATES, BLOCKS and UNITS, allow you to specify identifiers for the treatment, replicate, block-within-replicate and unit-within-block factors. If these are not specified in a batch run, AGALPHA will use identifiers that are local within the procedure and thus lost at the end of the procedure. If you are running interactively, AGALPHA will ask you to provide identifiers, and these will remain available after AGALPHA has finished running.

AGALPHA has a PRINT option which can be set to design to print the plan of the design, and generator to print the generator of the design. By default, if you are running Genstat in batch, neither are printed. If you do not set PRINT when running interactively, AGALPHA will ask whether you wish to print the design or generator.

Example 4.9.7 uses AGALPHA to generate an alpha design for 30 treatments (varieties) with three replicates each with six blocks of five plots.

Example 4.9.7

```
  2  " Alpha design for 30 treatments, with 3 replicates
 -3    and 6 blocks per replicate."
  4  AGALPHA  [PRINT=design] 30; NREPLICATES=3; NBLOCKS=6; SEED=37653;\
  5    TREATMENTS=Variety; REPLICATES=Rep; BLOCKS=Block; UNITS=Plot

Treatments on each unit of the design
=====================================

        Plot    1    2    3    4    5
   Rep Block
     1     1   20   19    6   14    4
           2    2   26   15    9   12
           3   28   16    5   25    3
           4   24   11   22    8   27
           5   10   29   30   21    1
           6   13   17   23    7   18
     2     1   28   19   26   22   29
           2   15    4   30   13    3
           3   20   18   10   27    9
           4   12   17   21    5    8
           5    7   14   25    1   11
           6    6   16    2   23   24
     3     1   22   18    1    6    3
           2   20   21   15   16    7
           3    2   25    8   19   10
           4   13   29   24    5    9
           5   17   14   26   27   30
           6   28   11   23    4   12

Treatment factor: Variety.
```

AGALPHA provides a repertoire of alpha arrays from Patterson, Williams & Hunter (1978) and Williams (1975). If you have your own array, you can generate the design using procedure AFALPHA (which is used by AGALPHA). This has a very similar syntax to AGALPHA, except that the GENERATOR parameter (which specifies the generator) replaces the NREPLICATES and NBLOCKS parameters of AGALPHA (the numbers of replicates and blocks are determined by the dimensions of the alpha array).

AFALPHA procedure

Generates alpha designs (R.W. Payne).

Option

PRINT = *string token*	Whether to print the design (design); default * i.e. no printing

Parameters

GENERATOR = *matrices*	generating array (of size number-of-plots-per-block by number-of-reps)
LEVELS = *scalars* or *variates*	Defines the levels of each treatment factor; if this is omitted, the levels of the TREATMENT factor are used, if available, otherwise LEVELS is determined from the generating array on the assumption that the blocks are to be of equal size
SEED = *scalar*	Seed to be used to randomize the design, if required
TREATMENTS = *factors*	Specifies the treatment factor for each design
REPLICATES = *factors*	Specifies the replicate factor
BLOCKS = *factors*	Specifies the block factor
UNITS = *factors*	Specifies the factor to index the units within each block

4.9.8 Balanced-incomplete-block designs

Incomplete block designs occur when the units in an experiment need to be divided into blocks that are not large enough to contain a unit for every treatment. In a balanced-incomplete-block design the contents of the blocks are arranged so that every pair of treatments occurs in an equal number of blocks. All comparisons between treatments are thus made with equal accuracy, so the design is balanced and, in particular, can be analysed by ANOVA. AGBIB can generate a balanced-incomplete-block design for any number of treatments in blocks of size two. It also has a selection of designs whose blocks contain more than two plots, which are generated from Hadamard matrices as described by Hedayat & Wallis (1978).

AGBIB procedure

Generates balanced incomplete block designs (R.W. Payne).

Options

PRINT = *string token*	Controls whether or not to print a plan of the design and whether to print a catalogue of the designs in the subfile (design, catalogue); if unset in an interactive run AGBIB will ask whether the design is to be printed, in a batch run the default is not to print anything
ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (no, yes); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run

Parameters

LEVELS = *scalars*	Number of treatments
NBLOCKS = *scalars*	Number of blocks

NUNITS = *scalars*	Number of units per block
SEED = *scalars*	Seed for randomization; a negative value implies no randomization
TREATMENTS = *factors*	Identifier for the treatment factor
BLOCKS = *factors*	Identifier for the factor to index the blocks
UNITS = *factors*	Identifier for the factor to index the units within each block
STATEMENT = *texts*	Saves a command to recreate each design (useful if the design information has been specified in response to questions from AGBIB)

If you are running Genstat interactively, you need not set any of the options or parameters of AGBIB. It then asks questions to determine the necessary information to form the design. The options and parameters allow you to anticipate questions, or to define all the necessary information if you want to use AGBIB in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

Example 4.9.8 shows the questions and answers (printed in bold font) to form a balanced-incomplete-block design for seven treatments in seven blocks of four plots. In Genstat *for Windows* the questions would be the same, but would appear in pop-up menus.

Example 4.9.8

```
> AGBIB

Do you want blocks of size 2?
n         no
y         yes
Code (n,y; Default n) > n

Design  Number  of  Number of  Number of plots  No. blocks  containing
number  treatments    blocks      per block     each set of treatments
     1           3          3               2                        1
     2           3          6               2                        2
     3           4          6               2                        1
     4           5         10               2                        1
     5           5         10               3                        3
     6           6         10               3                        2
     7           7          7               3                        1
     8           7          7               4                        2
     9           7         14               3                        2
    10           7         14               4                        4
    11           8         14               4                        3
    12           8         14               4                        1
    13           9         18               4                        3
    14           9         18               5                        5
    15          10         18               5                        4
    -1  exit
     0  more designs...
Number (Default: 0) > 8

What would you like to call the treatment factor?
Identifier (Default:Treatmen) > Treat

What would you like to call the block factor?
Identifier (Default:Blocks) > Block

What would you like to call the unit-within-block factor?
Identifier (Default:Units) > Plot

Seed for randomization (-1 for none)?
Number (Default: -1) > 583109
```

```
Do you want to print the design?
n         no
y         yes
Code (n,y; Default n) > y

Treatments on each unit of the design
=====================================

 Block   1   2   3   4   5   6   7
  Plot
     1   4   7   2   4   6   3   5
     2   1   6   4   2   7   7   3
     3   6   2   7   3   1   5   1
     4   5   5   1   6   3   4   2

Treatment factor: Treat.

Do you want to check the design by ANOVA?
n         no
y         yes
Code (n,y; Default n) > y

Analysis of variance
====================

Source of variation     d.f.

Block stratum
Treat                      6

Block.Plot stratum
Treat                      6
Residual                  15

Total                     27


Information summary
===================

Model term                e.f.  non-orthogonal terms

Block stratum
  Treat                 0.125

Block.Plot stratum
  Treat                 0.875  Block
```

Alternatively, you can set the LEVELS parameter to the required number of treatments, the NBLOCKS parameter to the number of blocks and the NUNITS parameter to the number of units per block; AGBIB then selects the design (if available) automatically.

The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is −1, to suppress randomization. If you do not set SEED when running interactively AGBIB will ask for a seed, and again a negative value suppresses any randomization.

Parameters TREATMENTS, BLOCKS and UNITS, allow you to specify identifiers for the treatment, the block and unit-within-block factors. If these are not specified in a batch run, AGBIB will use identifiers that are local within the procedure and thus lost at the end of the procedure. If you are running interactively, AGBIB will ask you to provide identifiers, and these will remain available after AGBIB has finished running.

The PRINT option controls printed output, with setting design to print a plan of the design, and catalogue to print a list of the available designs. By default, if you are running Genstat in batch, nothing is printed. If you do not set PRINT when running interactively, AGBIB will ask whether or not you wish to print the design, after it has been generated. Similarly the ANALYSE

option governs whether or not AGBIB produces a skeleton analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). Again AGBIB assumes that this is not required if ANALYSE is unset in a batch run, and asks whether it is required if ANALYSE is unset in an interactive run.

4.9.9 Cyclic designs

Cyclic designs provide an effective way of assessing treatments using a block design where the blocks are each too small to hold all the treatments. In its simplest form, the cyclic method of generation starts with an initial block, containing some subset of the treatments. The members of this subset are then represented by ordinal numbers in the range 0...$m-1$ where m is the number of treatment levels. The second and subsequent blocks are then generated by successive addition modulo m of one to the numbers in the subset. Thus, for seven treatments (0...6) and an initial block (0,1,4), the subsequent blocks would contain treatments (1,2,5), (2,3,6), (3,4,0), (4,5,1), (5,6,2) and (6,0,3). As can be seen, if m is a prime number, m blocks are generated with each initial block. However, if m can be expressed as the product of other integers, shorter cycles can occur. For example, for m=8 and initial block (0,1,4,5), four blocks are generated altogether, the others being (1,2,5,6), (2,3,6,7) and (3,4,7,0). Procedure AFCYCLE, which generates cylic designs in Genstat, allows for all of this. It is also possible to have more than one initial block, and the increment need not be one.

The efficiency of the design depends very much on the choice of initial blocks. Procedure AGCYCLIC provides a repertoire of initial blocks mainly from the program DSIGNX (Franklin & Mann 1986), and including designs from Davis & Hall (1969), Hall & Williams (1973) and John, Wolock & David (1972). Cyclic designs are generally unbalanced, and are thus analysed using REML (Chapter 5).

AGCYCLIC procedure

Generates cyclic designs from standard generators (M.F. Franklin & R.W. Payne).

Options

PRINT = *string token*	Controls whether or not to print a plan of the design (design); if unset in an interactive run AGCYCLIC will ask whether the design is to be printed, in a batch run the default is not to print the design
METHOD = *string token*	Type of design – ordinary cyclic, cyclic change-over or cyclic superimposed (cyclic, changeover, superimposed); if unset in an interactive run AGCYCLIC will ask about the type of design, in a batch the default is assumed to be cyclic

Parameters

LEVELS = *scalars*	Number of treatments
NBLOCKS = *scalars*	Number of blocks
NUNITS = *scalars*	Number of units per block, or number of periods in a cyclic change-over design
SEED = *scalars*	Seed for randomization; a negative value implies no randomization
TREATMENTS = *factors*	Identifier for the treatment factor
SUPERIMPOSED = *factors*	Identifier for the second treatment factor in a cyclic superimposed design
BLOCKS = *factors*	Identifier for the factor to index the blocks
UNITS = *factors*	Identifier for the factor to index the units within each

	block, or the periods of a cyclic change-over design
INITIALBLOCKS = *variates* or *pointers*	
	To save one (variate) or more (pointer to variates) initial blocks
STATEMENT = *texts*	Saves a command to recreate the design (useful if the design information has been specified in response to questions from AGCYCLIC)

If you are running Genstat interactively, you need not set any of the options or parameters of AGCYCLIC. It then asks questions to determine the necessary information to form the design. It will also tell you which block sizes are available for your chosen number of treatments. The options and parameters allow you to anticipate questions, or to define all the necessary information if you want to use AGCYCLIC in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The first question, which can be anticipated by setting the METHOD option, determines the type of cyclic design. In addition to the standard cyclic designs, AGCYCLIC can also generate the cyclic change-over designs of Davis & Hall (1969) and the cyclic superimposed designs of Hall & Williams (1973). The change-over designs are used for trials in which subjects are given different treatments in different time periods; these thus have a crossed block structure subjects*periods. (Note that procedure AFCARRYOVER can be used after AGCYCLIC to generate factors to represent the carry-over effects if required.) The extension in the cyclic superimposed design is that there are two treatment factors (each with the same number of levels); the design is intended to estimate their main effects but not their interaction.

The PRINT option controls whether AGCYCLIC prints a plan of the design. By default, if you are running Genstat in batch, the plan is not printed. If you do not set PRINT when running interactively, AGCYCLIC will ask whether or not you wish to print the design, after it has been generated.

The number of treatments can be defined using the LEVELS parameter. Similarly, the NBLOCKS and NUNITS parameters define the number of blocks and the number of units per block (or the number of periods in a cyclic change-over design). The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is −1, to suppress randomization. If you do not set SEED when running interactively AGCYCLIC will ask for a seed, and again a negative value suppresses any randomization.

Parameters TREATMENTS, SUPERIMPOSED, BLOCKS and UNITS, allow you to specify identifiers for the treatment, the superimposed treatment (for a cyclic superimposed design), the block and unit-within-block factors. If these are not specified in a batch run, AGCYCLIC will use identifiers that are local within the procedure and thus lost at the end of the procedure. If you are running interactively, AGCYCLIC will ask you to provide identifiers, and these will remain available after AGCYCLIC has finished running. Finally, the INITIAL parameter allows you to save the initial blocks, in a variate if there is only one, or in a pointer (to a list of variates) if there are several.

Example 4.9.9 uses AGCYCLIC to generate a cyclic design for 20 treatments in blocks of size three.

Example 4.9.9

```
" Cyclic design for 20 treatments in blocks of size 3."
AGCYCLIC [PRINT=design; METHOD=cyclic] 20; NBLOCKS=20; NUNITS=3;\
  SEED=149634; TREATMENTS=Treat; BLOCKS=Block; UNITS=Plot
```

```
Treatments on each unit of the design
=====================================

 Block    1    2    3    4    5    6    7    8    9   10   11   12   13   14
  Plot
     1   11   12    6    1   19   13   17   16   20    5    9    2    4   14
     2    7    8    1   20   14   12   16   11    4   10   14   17    8   18
     3    6    7    2    5   15   17    1   12   19    6   10   18    3   13

 Block   15   16   17   18   19   20
  Plot
     1   16   11   13    7   18    5
     2   20   10    9    2   19    9
     3   15   15    8    3    3    4

Treatment factor: Treat.
```

If you have your own initial blocks, you can generate the design using AFCYCLIC. The INITIAL parameter specifies the initial blocks. If the design is to be generated from a single initial block, INITIAL should be set to a variate containing the levels corresponding to the treatments concerned; if there are several, the appropriate variates should be placed into a pointer. Similarly the INCREMENT parameter, which specifies the increment to be used, should be set to a scalar if the same increment is to be used for all the initial blocks, otherwise to a pointer of scalars. The LEVELS, SEED, TREATMENTS, BLOCKS and UNITS parameters operate as in AGCYCLIC.

AFCYCLIC procedure

Generates block and treatment factors for cyclic designs (R.W. Payne).

Option

PRINT = *string token*	Whether to print the design (design); default * i.e. no printing

Parameters

INITIALBLOCKS = *variates* or *pointers*	Defines one (variate) or more (pointer to variates) initial blocks for a treatment factor
INCREMENT = *scalars* or *pointers*	Defines the size of the successive increments (scalar) or increments (pointer to scalars) for each initial block
LEVELS = *scalars* or *variates*	Defines the levels of each treatment factor; this need not be specified if the factor has already been declared
SEED = *scalar*	Seed to be used to randomize each design, if required
TREATMENTS = *factors*	Specifies treatment factors
BLOCKS = *factors*	Specifies block factors
UNITS = *factors*	Specifies factors to index the units within each block

4.9.10 Neighbour-balanced designs

In experiment designs it is often necessary to allow for the possibility that a treatment may have an effect on neighbouring plots, as well as on its own plot. For example, in variety trials, tall varieties may shade their neighbours. Likewise, in experiments on insecticides and fungicides, there may be cross infection from plots receiving control or ineffective treatments to neighbouring plots. In both of these examples the neighbour effect may depend on direction (for example of prevailing wind or of sunlight), so it is usual to distinguish between left and right neighbours. To avoid bias when comparing the effects of treatments in these situations, it is important to ensure that no treatment is unduly disadvantaged by its neighbours. This is best done by using a neighbour-balanced design. Here the allocation of treatments is such that every

treatment occurs equally often with each other treatment as a right neighbour, and as a left neighbour.

The table below shows a design for five treatments in 5 blocks of size 4. Notice that in addition to the experimental plots, the design also needs a line of treated border plots on each side. These provide the neighbouring treatments for plots 1 and 4, but do not provide yields or other response variables. The border plots are not included in the generated factor values.

Plot	border	1	2	3	4	border
Block						
1	5	2	3	1	5	2
2	3	5	4	1	3	5
3	4	2	5	3	4	2
4	1	4	3	2	1	4
5	4	5	1	2	4	5

Methods of constructing and randomizing neighbour-balanced designs for n treatments in either n blocks of $n-1$ plots or in $n-1$ blocks of n plots are described by Azais, Bailey & Monod (1993) together with generators for $3 \le n \le 16$ (other than for n=4 or 6 with $n-1$ blocks of size n, for which no designs are available). AGNEIGHBOUR uses these methods and generators, together with some further generators for blocks of $n-1$ plots formed using the method of Azais (1987).

AGNEIGHBOUR procedure

Generates neighbour-balanced designs (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (catalogue, design); if unset in an interactive run AGNEIGHBOUR will ask whether the design is to be printed, in a batch run the default is not to print anything
METHOD = *string token*	Type of design, $n-1$ blocks of n plots, or n blocks of $n-1$ plots (N_1BLOCKS, NBLOCKS); if unset in an interactive run AGNEIGHBOUR will ask about the type of design, in a batch the default is assumed to be n blocks of n−1 plots

Parameters

LEVELS = *scalars*	Number of treatments
SEED = *scalars*	Seed for randomization; in batch there is a default of 12345
TREATMENTS = *factors*	Identifier for the treatment factor
BLOCKS = *factors*	Identifier for the factor to index the blocks within replicates
UNITS = *factors*	Identifier for the factor to index the units within each block, or the periods of a cyclic change-over design
LEFTNEIGHBOUR = *factors*	To save the treatment on the left neighbouring unit
RIGHTNEIGHBOUR = *factors*	To save the treatment on the right neighbouring unit
STATEMENT = *texts*	Saves a command to recreate each design (useful if the design information has been specified in response to questions from AGNEIGHBOUR)

If you are running Genstat interactively, you need not set any of the options or parameters of AGNEIGHBOUR. It then asks questions to determine the necessary information to form the design, and indicates the numbers of treatments for which designs are available. The options and parameters allow you to anticipate questions, or to define all the necessary information if you

want to use AGNEIGHBOUR in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The first question, which can be anticipated by setting the METHOD option, determines the type of design: n blocks of $n-1$ plots (METHOD=nblocks) or in $n-1$ blocks of n plots (METHOD=n_1blocks). The default in batch is n_1block. The PRINT option controls printed output, with setting design to print a plan of the design, and catalogue to print a list of the available designs. By default, if you are running Genstat in batch, nothing is printed. If you do not set PRINT when running interactively, AGNEIGHBOUR will ask whether or not you wish to print the design, after it has been generated.

The number of treatments can be defined using the LEVELS parameter. This can be set to zero to avoid constructing a design, as may be required if you merely wish to print the catalogue. The SEED parameter allows you to specify a seed to be used to randomize the design. If you do not set SEED when running interactively AGNEIGHBOUR will ask for a seed. In batch there is a default of 12345. Setting a negative seed suppresses any randomization. Parameters TREATMENTS, BLOCKS and UNITS, allow you to specify identifiers to save the treatment, the block and unit-within-block factors. If these are not specified in a batch run, AGNEIGHBOUR will use identifiers that are local within the procedure and thus lost at the end of the procedure. If you are running interactively, AGNEIGHBOUR will ask you to provide identifiers and these will remain available after AGNEIGHBOUR has finished running. There are also parameters LEFTNEIGHBOUR and RIGHTNEIGHBOUR to allow you to save the treatments on the left and right neighbouring plots.

Some of the designs are such that each ordered pair of treatments occurs the same number of times as the left and right neighbours of some other treatment, the design is then said to be neighbour-balanced at distance 2. These designs have the further advantage that they are balanced if analysed with ANOVA with

```
BLOCKSTRUCTURE     BLOCKS / UNITS
TREATMENTSTRUCTURE TREATMENTS+ LEFTNEIGHBOUR \
  + RIGHTNEIGHBOUR
```

(Other designs can be analysed by REML; Chapter 5.)

Example 4.9.10

```
  2  " Neighbour design for 7 treatments in blocks of size 7."
  3  AGNEIGHBOUR [PRINT=catalogue,design; METHOD=n_1block] 7; \
  4    SEED=2041996; TREATMENTS=Treat; BLOCKS=Block; UNITS=Plot; \
  5    LEFTNEIGHBOUR=Left; RIGHTNEIGHBOUR=Right

Neighbour designs
-----------------

Balanced neighbour designs are available for n treatments in n blocks of n-1
plots for any value of n>2, or in n-1 blocks of n plots for the following
values of n:  3, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 19.


Treatments on each unit of the design
=====================================

  Plot   1   2   3   4   5   6   7
 Block
     1   3   1   4   6   2   5   7
     2   5   4   7   6   3   2   1
     3   7   4   5   1   2   3   6
     4   1   3   7   5   2   6   4
     5   7   2   4   3   5   6   1
     6   7   1   6   5   3   4   2

Treatment factor: Treat.
```

```
The design assumes that the plots in each block are arranged in a continuous
line, and that there is a gap between each pair of blocks. There must also be
border plots: the treatments in the left-hand plots must be duplicated on the
right-hand-side, and those in the right-hand plots must be duplicated on the
left-hand side.
   6  " This design is balanced: produce a dummy analysis."
   7  BLOCKSTRUCTURE Block / Plot
   8  TREATMENTSTRUCTURE Treat + Left + Right
   9  ANOVA

* MESSAGE: non-orthogonality between treatment terms. The effects (printed or
used to calculate means), the efficiency factor and the sum of squares for each
treatment term are for that term eliminating previous terms in the TREATMENT
formula and ignoring subsequent terms.


Analysis of variance
====================

Source of variation     d.f.

Block stratum              5

Block.Plot stratum
Treat                      6
Left                       6
Right                      6
Residual                  18

Total                     41


Information summary
===================

Model term                e.f.  non-orthogonal terms

Block.Plot stratum
  Left                   0.972  Treat
  Right                  0.933  Treat  Left
```

4.9.11 Central composite designs

Central composite designs are used for estimating quadratic response surfaces: that is, the model to be fitted to the results is a quadratic function of the various factors. The design is made up of three sets of points.

a) a factorial design: usually this contains all combinations of the factors at a pair of levels (l_1,l_2), but for five or more factors it is feasible to use a fractional factorial (and still be able to estimate all the parameters of the response surface)
b) star points: this set contains a pair of points for each factor where the other factors take the value $(l_1+l_2)/2$ and the factor has the values s_1 and s_2
c) centre points: here all the factors have the value $(l_1+l_2)/2$

AGCENTRALCOMPOSITE procedure

Generates central composite designs (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (design); if unset in an interactive run AGCENTRALCOMPOSITE will ask whether the design is to be printed, in a batch run the default is not to print anything
NCENTRALPOINTS = *scalar*	Defines the number of central points to include; default 4

NSTARPOINTS = *scalar*	Defines the number of star points to include; default 1
LFACTORIAL = *variate*	Defines the treatment levels in the factorial part of the design; default !(-1,1)
LSTAR = *variate*	Defines the treatment levels for the star points; default is to use the levels defined by LFACTORIAL
FRACTION = *scalar*	Denominator for fractional factorial; default 1 specifies a complete design
SEED = *scalar*	Seed to be used to randomize each design; a negative value implies no randomization
STATEMENT = *text*	Saves a command to recreate the design (useful if the design information has been specified in response to questions from AGCENTRALCOMPOSITE)

Parameter

TREATMENTFACTOR = *factors*	Treatment factors

The treatment factors for AGCENTRALCOMPOSITE are listed using the TREATMENTFACTOR parameter. If this is omitted in an interactive run, you will be asked how many factors you want and their names. The number of central points is specified by the NCENTRALPOINTS option; by default this is taken to be four. The LFACTORIAL option can supply a variate to specify the levels to be used in (a); the defaults are 1 and −1 (so the central point is at zero). Similarly, LSTAR specifies the levels for (b), which are taken, by default, to be the same as in (a). The star levels must, however, be equally spaced around the centre point. Option NSTARPOINTS defines how may replicates to have of each star point. The FRACTION option supplies the denominator of a fractional design, if required for (a); the default of one indicates that a complete factorial design is to be used. The SEED option allows you to specify a seed to be used to randomize the design. In batch the default seed is −1, to suppress randomization. If you do not set SEED when running interactively AGCENTRALCOMPOSITE will ask for a seed, and again a negative value suppresses any randomization. The PRINT option can be set to design to print the plan of the design. By default, if you are running Genstat in batch, the plan is not printed. If you do not set PRINT when running interactively, AGCENTRALCOMPOSITE will ask whether or not you wish to print the design.

The STATEMENT option allows you to save a Genstat text structure containing a command to recreate the design. This is particularly useful if AGCENTRALCOMPOSITE is being used interactively, and the information to define the design has been provided in response to questions from the procedure.

Example 4.9.11

```
" Unrandomized plan of a central composite design for
  2 treatment factors with factorial levels -1 and +1,
  star points at -1.5 and +1.5, and 4 central points at 0."
AGCENTRALCOMPOSITE [PRINT=design; NCENTRAL=4; NSTAR=1;\
  LFACTORIAL=!(-1,1); LSTAR=!(-1.5,1.5); SEED=-1] A,B
```

```
     A          B
  -1.0       -1.0
  -1.0        1.0
   1.0       -1.0
   1.0        1.0
   0.0        0.0
   0.0        0.0
   0.0        0.0
   0.0        0.0
  -1.5        0.0
   1.5        0.0
   0.0       -1.5
```

```
      0.0         1.5
```

4.9.12 Box-Behnken designs

Box-Behnken designs are often used to study response surfaces. The design is usually formed to allow a quadratic response surface to be fitted. The factors are studied at three equally-spaced levels, below denoted by −1, 0 and 1. The construction uses a balanced incomplete block design to select successive sets of factors to be applied at all factorial combinations of −1 and +1, while other factors are held at 0. For example, with three factors `A`, `B` and `C`, the relevant balanced incomplete block design would have three blocks (`A`,`B`), (`A`,`C`) and (`B`,`C`). So the design would first have a section with `A` and `B` varying but `C` constant

```
 A    B    C
-1   -1    0
-1   +1    0
+1   -1    0
+1   +1    0
```

then a section where `B` is held constant but `A` and `C` take all combinations of −1 and +1

```
 A    B    C
-1    0   -1
-1    0   +1
+1    0   -1
+1    0   +1
```

and finally a section with `A` constant

```
 A    B    C
 0   -1   -1
 0   -1   +1
 0   +1   -1
 0   +1   +1
```

In addition, there can be some "central points", where all the factors take the central value

```
 A    B    C
 0    0    0
 0    0    0
 0    0    0
 0    0    0
```

`AGBOXBEHNKEN` procedure

Generates Box Behnken designs (R.W. Payne).

Options

`PRINT` = *string token*	Controls printed output (`design`); if unset in an interactive run `AGBOXBEHNKEN` will ask whether the design is to be printed, in a batch run the default is not to print anything
`NCENTRALPOINTS` = *scalar*	Defines the number of central points to include; default 4
`LEVELS` = *variate*	Defines the outer levels to be used; default `!(-1,1)`
`NCOMBINATIONS` = *scalar*	Number of factors to vary in combination at once; default 2
`SEED` = *scalar*	Seed to be used to randomize each design; a negative value implies no randomization
`STATEMENT` = *text*	Saves a command to recreate the design (useful if the design information has been specified in response to questions from `AGBOXBEHNKEN`)

Parameter

TREATMENTFACTOR = *factors* Treatment factors

The treatment factors for AGBOXBEHNKEN are listed using the TREATMENTFACTOR parameter. If this is omitted in an interactive run, you will be asked how many factors you want and their names. The number of central points is specified by the NCENTRALPOINTS option; by default this is taken to be four. The LEVELS option can supply a variate to specify the outer treatment levels; the defaults are 1 and −1 (so the central point is at zero). The NCOMBINATIONS option defines the number of factors whose combinations of (outer) levels are to be varied at once. For the default of two, the relevant balanced incomplete block design is formed within AGBOXBEHNKEN. Other values can be supplied, but the corresponding balanced incomplete block design must be one of those obtainable from procedure AGBIB. You can find out the possibilities by putting

```
AGBIB [PRINT=catalogue]
```

The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is −1, to suppress randomization. If you do not set SEED when running interactively AGBOXBEHNKEN will ask for a seed, and again a negative value suppresses any randomization. The PRINT option can be set to design to print the plan of the design. By default, if you are running Genstat in batch, the plan is not printed. If you do not set PRINT when running interactively, AGBOXBEHNKEN will ask whether or not you wish to print the design.

The STATEMENT option allows you to save a Genstat text structure containing a command to recreate the design. This is particularly useful if AGBOXBEHNKEN is being used interactively, and the information to define the design has been provided in response to questions from the procedure.

Example 4.9.12

```
" Unrandomized plan of a Box-Behnken design for 4 treatments."
AGBOXBEHNKEN [PRINT=design; NCENTRAL=4; LEVELS=!(-1,1);\
  SEED=-1] A,B,C,D
```

A	B	C	D
-1	-1	0	0
-1	1	0	0
1	-1	0	0
1	1	0	0
-1	0	-1	0
-1	0	1	0
1	0	-1	0
1	0	1	0
-1	0	0	-1
-1	0	0	1
1	0	0	-1
1	0	0	1
0	-1	-1	0
0	-1	1	0
0	1	-1	0
0	1	1	0
0	-1	0	-1
0	-1	0	1
0	1	0	-1
0	1	0	1
0	0	-1	-1
0	0	-1	1
0	0	1	-1
0	0	1	1
0	0	0	0
0	0	0	0
0	0	0	0

0 0 0 0

4.9.13 Plackett Burman (main effect) designs

AGMAINEFFECT procedure

Generates designs to estimate main effects of two-level factors (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (design, catalogue); if unset in an interactive run AGMAINEFFECT will ask whether the design or catalogue are to be printed, in a batch run the default is not to print anything
ANALYSE = *string token*	Controls whether or not to analyse the design, and produce a skeleton analysis-of-variance table using ANOVA (no, yes); default is to ask if this is unset in an interactive run, and not to analyse if it is unset in a batch run
FOLDED = *string token*	Whether to include an extra "folded" replicate with the levels of each factor interchanged (no, yes); default no
SEED = *scalar*	Seed to be used to randomize each design; a negative value implies no randomization
STATEMENT = *texts*	Saves a command to recreate the design (useful if the design information has been specified in response to questions from AGMAINEFFECT)

Parameter

TREATMENTFACTOR = *factors*	Treatment factors

AGMAINEFFECT generates designs for estimating main effects of factors with two levels, using a minimum number of experimental units; see Plackett & Burman (1946). The designs are based on Hadamard matrices, which are generated by procedure FHADAMARDMATRIX. However, the numbers of treatment factors for which designs are available can be printed by setting option PRINT=catalogue. They are all expressible as $4n-1$ for some integer n. The treatment factors are listed using the TREATMENTFACTOR parameter. If this is omitted in an interactive run, you will be asked how many factors you want and their names.

The basic design allows the main effects to be estimated, but has no residual degrees of freedom. This is fine if you merely want to screen the main effects to identify the largest. Otherwise you can generate a design for more factors than are needed, and then use the degrees of freedom of the unnecessary factors to provide the residual. Alternatively, if you set option FOLDED=yes, AGMAINEFFECT will include a "folded" replicate of the design: this is identical to the initial replicate except that the levels of the factors are swapped (level one instead of level two and vice versa). This particular arrangement has the advantage that no main effect is aliased with any first-order interaction.

The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is −1, to suppress randomization. If you do not set SEED when running interactively AGMAINEFFECT will ask for a seed, and again a negative value suppresses any randomization. The PRINT option can be set to design to print the plan of the design. By default, if you are running Genstat in batch, the plan is not printed. If you do not set PRINT when running interactively, AGMAINEFFECT will ask whether or not you wish to print the design. Similarly the ANALYSE option governs whether or not AGMAINEFFECT produces a skeleton

analysis-of-variance table (containing just source of variation, degrees of freedom and efficiency factors). Again AGMAINEFFECT assumes that this is not required if ANALYSE is unset in a batch run, and asks whether it is required if ANALYSE is unset in an interactive run. The ANOVA option ORTHOGONAL is set to assumed for the analysis. (If this is not done, the larger designs can take a very long time to analyse.)

The STATEMENT option allows you to save a Genstat text structure containing a command to recreate the design. This is particularly useful if AGMAINEFFECT is being used interactively, and the information to define the design has been provided in response to questions from the procedure.

Example 4.9.13 shows two Plackett-Burman designs for seven treatment factors. The first has only eight units, and thus no residual degrees of freedom. Data from designs like this can be analysed graphically using procedure A2PLOT. The second design also has a folded replicate, and thus eight residual degrees of freedom.

Example 4.9.13

```
  2  " Design with 8 units for 7 factors."
  3  AGMAINEFFECT [PRINT=design; ANALYSE=yes; FOLDED=no;\
  4    SEED=357417] A,B,C,D,E,F,G

        A     2     1     2     1     1     2     1     2
        B     1     1     2     2     2     2     1     1
        C     1     2     2     1     1     2     2     1
        D     1     2     2     1     2     1     1     2
        E     1     1     2     2     1     1     2     2
        F     2     1     2     1     2     1     2     1
        G     2     2     2     2     1     1     1     1

Analysis of variance
====================

Source of variation     d.f.
A                          1
B                          1
C                          1
D                          1
E                          1
F                          1
G                          1
Total                      7


  5  " Include a folded replicate."
  6  AGMAINEFFECT [PRINT=design; ANALYSE=yes; FOLDED=yes;\
  7    SEED=357417] A,B,C,D,E,F,G

        A     2     1     1     1     2     2     1     2
        B     1     1     1     2     1     2     1     1
        C     1     2     1     1     2     1     2     1
        D     1     2     2     1     1     2     1     2
        E     1     1     2     2     2     1     2     2
        F     2     1     2     1     1     1     2     1
        G     2     2     2     2     2     2     1     1

        A     1     1     1     2     2     2     1     2
        B     2     1     2     1     2     2     2     2
        C     2     1     1     2     2     2     2     1
        D     2     1     2     2     2     1     1     1
        E     2     1     1     1     2     1     1     2
        F     1     1     2     2     2     1     2     2
        G     1     1     1     1     2     1     2     1
```

```
Analysis of variance
====================

Source of variation     d.f.
A                          1
B                          1
C                          1
D                          1
E                          1
F                          1
G                          1
Residual                   8
Total                     15
```

4.9.14 Response surface designs

AFRESPONSESURFACE directive

Uses the BLKL algorithm to construct designs for estimating response surfaces.

Options

PRINT = *string token*	Printed output required (monitoring); default * i.e. no printing
TERMS = *formula*	Model to be fitted when the design is used; no default i.e. this option must be specified
CONSTANT = *string token*	How to treat the constant in the model (estimate, omit); default esti
FACTORIAL = *scalar*	Limit for expansion of terms in the model; default 2
NUNITS = *scalar*	Number of units (or trials) in the design
NDELETION = *scalar*	Number of design points to consider for deletion; default takes NUNITS/4, or 4 is this is larger
NINCLUSION = *scalar*	Number of design points to consider for inclusion; default takes NUNITS/4, or 4 is this is larger
NRUNS = *scalar*	Number of times to run the algorithm; default 100
ADJUSTMENTSTEP = *scalar*	Maximum amount by which to perturb the design points in the adjustment algorithm; default * i.e. no adjustments are tried
NBLOCKS = *scalar*	Number of blocks; default 1 i.e. design not blocked
BLOCKFACTOR = *factor*	Saves the block factor (if any) for the design
BLOCKSIZE = *scalar* or *variate*	Number of units in each block of the design
PREVIOUSBLOCKS = *factor*	Supplies values of the blocking factor for any previous experiments that are to be included in the analysis of the results of the design
SEED = *scalar*	Seed for random numbers used to construct the initial design; default 124195
MIXTURE = *variates*	Lists any variates that are part of a mixture (their values must be greater than zero and sum to one)
DETERMINANT = *scalar*	Saves the determinant of the information matrix for the best design
MEANGRID = *scalar*	Saves the mean value of the standardized variance of predictions obtained from the design over a grid of x-values
MAXGRID = *scalar*	Saves the maximum value of the standardized variance of predictions obtained from the design over a grid of x-values

NGRIDPOINTS = *scalar*	Number of grid points in each x-direction to use for MEANGRID and MAXGRID; default 5

Parameters

X = *variates*	Lists the variates to be investigated in the design; these need not be supplied if none of the other parameters are required
X2 = *variates*	Lists identifiers to be used to represent squares of the x-variates in the model
X3 = *variates*	Lists identifiers to be used to represent cubes of the x-variates in the model
SUPPORTPOINTS = *variates*	Support points for each x-variate in the design; if these are not (all) specified, they are formed automatically
PREVIOUSVALUES = *variates*	Supplies values of the x-variates for any previous experiments that are to be included in the analysis of the results of the design

AFRESPONSESURFACE uses the BLKL algorithm of Atkinson & Donev (1992) to construct a design to estimate parameters of a response-surface model – and Alex Donev's assistance with this Genstat implementation is gratefully acknowledged. The algorithm searches for a D-optimal design: that is, a design that will provide a maximum value for the determinant of the information matrix of the model parameters. The model is specified using the TERMS option, with the CONSTANT option indicating whether or not it is to contain the constant term (or intercept). The FACTORIAL sets a limit on the number of variates in each model term; by default this is 2.

The NUNITS option specifies the number of units in the design. If there is to be a blocking factor in the design, the NBLOCKS option specifies its number of levels, and the BLOCKFACTOR option saves its values. The BLOCKSIZE option specifies the number of units to be contained in each block of the design, in a scalar (if they are all the same) or a variate. If the block sizes are fixed, the specified sizes must sum to the number of units. However, if you specify sizes that sum to a value greater that the required number of units, the algorithm will search for the optimum block sizes.

When the model is to contain squares or cubes of x-variables, you will need to specify identifiers to represent these using the parameters of the directive. (When using regression directives such as FIT to fit the model, you can use the POL function but this is not recognised by AFRESPONSESURFACE.) The x-variates in the model must then all be listed by the X parameter. The corresponding squares are listed by the X2 parameter, and the cubes by the X3 parameter.

The BLKL algorithm starts by forming an initial design by making a random selection of points from the set of support points. The SEED option defines the seed for the random numbers used to make the selection (default 124195). The algorithm then uses an exchanges algorithm to improve the design. At each exchange, the *K* points with the lowest variance of prediction amongst the points of design are considered for replacement by the *L* points with the highest variance of prediction amongst the candidate points for inclusion in the design. The algorithm makes the best one of these exchanges, continuing until there are none that increase the determinant. The values for *K* and *L* are specified by the NDELETION and NINCLUSION options respectively. The best values depend on the design parameters, including the number of model parameters and the number of residual degrees of freedom. If they are unset, AFRESPONSESURFACE sets them to the number of units divided by 4, or 4 if this larger. The NRUNS option can be set to request that the algorithm is run several times, with different starting designs; the default is 100. The design parameters are saved only for the best design found, but

you can set option PRINT=monitoring to print information about each attempt.

The DETERMINANT option allows you to saves the determinant of the information matrix for the best design. An alternative way of evaluating the design is to examine the standardized variance of the predictions that would be obtained from the design at other points, not in the design. The MEANGRID option can save the mean value of the standardized variance of prediction over a grid of x-values, and the MAXGRID option can save the maximum value. Number of grid points in each x-direction is specified by the NGRIDPOINTSMETHOD option (default 5).

Example 4.9.14a forms a design with 17 units for estimating a response surface modelled by an equation involving the terms: constant, x_1, x_2, x_3, x_4, x_1x_2, x_1x_3, x_1x_4, x_2x_3, x_2x_4, x_3x_4, x_1^2, x_2^2, x_3^2 and x_4^2. The squared terms are represented by the variates X1_2, X2_2, X3_2, X4_2, specified by the X2 parameter.

Example 4.9.14a

```
 2  AFRESPONSESURFACE [NDELETION=10; NINCLUSION=40; NRUNS=1000; NUNITS=17;\
 3        TERMS=X1 * X2 * X3 * X4 + X1_2 + X2_2 + X3_2 + X4_2;\
 4        CONSTANT=estimate; DETERMINANT=Det; MAXGRID=Dmax; MEANGRID=Dave]\
 5        X=X1,X2,X3,X4; X2=X1_2,X2_2,X3_2,X4_2
 6  PRINT X1,X2,X3,X4; FIELD=8; DECIMALS=3

    X1      X2      X3      X4
-1.000   0.000  -1.000  -1.000
-1.000  -1.000   0.000  -1.000
 1.000   1.000   0.000  -1.000
-1.000   1.000  -1.000   0.000
 1.000   0.000   1.000  -1.000
-1.000   1.000   0.000   1.000
-1.000  -1.000   1.000   0.000
 0.000  -1.000   1.000  -1.000
 0.000   1.000  -1.000  -1.000
-1.000   0.000   1.000   1.000
 1.000  -1.000   1.000   1.000
 0.000   0.000   0.000   0.000
 1.000   1.000   1.000   1.000
 1.000  -1.000  -1.000  -1.000
 1.000   1.000  -1.000   1.000
-1.000   1.000   1.000  -1.000
-1.000  -1.000  -1.000   1.000

 7  &     Det,Dmax,Dave

       Det        Dmax        Dave
1.5288E+13       38.43       15.04
```

If you specify the X parameter, you can also use the SUPPORTPOINTS parameter to specify the x-values of the points to be considered when constructing the design; if this is not specified, these support points are formed automatically. Note that the variates are all assumed to be scaled to have values between −1 and 1. However, the criterion for D-optimality is unaffected by linear transformations of the X-variables. So you can rescale afterwards in any way you like. The PREVIOUSVALUES parameter can supply values of the x-variates for any previous experiments that are to be included in the analysis of the results of the new experiment, or to specify points that must be included in the design. The PREVIOUSBLOCKS option should then indicate the blocks to which these previous observations belonged. These parameters are both illustrated in Example 4.9.14b.

Example 4.9.14b

```
 8  VARIATE [NVALUES=15] S1,S2,S3
 9  READ    [PRINT=data,errors] S1,S2,S3

10  -0.5774 -0.5774 -0.5774
```

```
11   0.5774 -0.5774 -0.5774
12  -0.5774  0.5774 -0.5774
13   0.5774  0.5774 -0.5774
14  -0.5774 -0.5774  0.5774
15   0.5774 -0.5774  0.5774
16  -0.5774  0.5774  0.5774
17   0.5774  0.5774  0.5774
18   1.0000  0.0000  0.0000
19  -1.0000  0.0000  0.0000
20   0.0000  1.0000  0.0000
21   0.0000 -1.0000  0.0000
22   0.0000  0.0000  1.0000
23   0.0000  0.0000 -1.0000
24   0.0000  0.0000  0.0000 :
25  VARIATE [NVALUES=6] P1,P2,P3
26  READ    [PRINT=data,errors] P1,P2,P3

27   1.0000  0.0000  0.0000
28  -1.0000  0.0000  0.0000
29   0.0000  1.0000  0.0000
30   0.0000 -1.0000  0.0000
31   0.0000  0.0000  1.0000
32   0.0000  0.0000 -1.0000 :
33  AFRESPONSESURFACE [NUNITS=16; NDELETION=3; NINCLUSION=3;\
34        TERMS=X1 * X2 * X3 + X1_2 + X2_2 + X3_2; CONSTANT=estimate;\
35        DETERMINANT=Det; MAXGRID=Dmax; MEANGRID=Dave] \
36        X=X1,X2,X3; X2=X1_2,X2_2,X3_2; SUPPORTPOINTS=S1,S2,S3
37  SORT  [INDEX=X1,X2,X3] X1,X2,X3
38  PRINT X1,X2,X3; FIELD=8; DECIMALS=3

    X1      X2      X3
-1.000   0.000   0.000
-0.577  -0.577  -0.577
-0.577  -0.577   0.577
-0.577   0.577  -0.577
-0.577   0.577   0.577
 0.000  -1.000   0.000
 0.000   0.000  -1.000
 0.000   0.000   0.000
 0.000   0.000   0.000
 0.000   0.000   1.000
 0.000   1.000   0.000
 0.577  -0.577  -0.577
 0.577  -0.577   0.577
 0.577   0.577  -0.577
 0.577   0.577   0.577
 1.000   0.000   0.000

39  &     Det,Dmax,Dave

       Det        Dmax        Dave
      2669       106.5       30.81
```

AFRESPONSESURFACE allows for a set of mixture variates, whose values must all be positive and which must sum to 1. The variates in the mixture are specified using the MIXTURE option. This is illustrated by the variates X1, X2 and X3 in Example 4.9.14c.

Example 4.9.14c

```
40  AFRESPONSESURFACE [NUNITS=12; NDELETION=3; NINCLUSION=3; NRUNS=500;\
41        TERMS=X1 + X2 + X3 + X4 + X1.X2 + X1.X3 + X2.X3 + X4_2;\
42        CONSTANT=omit; MIXTURE=X1,X2,X3;\
43        DETERMINANT=Det; MAXGRID=Dmax; MEANGRID=Dave] \
44        X=X1,X2,X3,X4; X2=*,*,*,X4_2
45  PRINT X1,X2,X3,X4; FIELD=8; DECIMALS=3

    X1      X2      X3      X4
 0.000   0.000   1.000  -1.000
 0.000   0.000   1.000   1.000
```

```
  0.000   1.000   0.000   1.000
  0.500   0.500   0.000  -1.000
  1.000   0.000   0.000   1.000
  0.500   0.000   0.500   0.000
  0.000   0.500   0.500   0.000
  1.000   0.000   0.000   0.000
  0.000   1.000   0.000   0.000
  0.000   0.500   0.500  -1.000
  0.500   0.000   0.500  -1.000
  0.500   0.500   0.000   1.000

46  &     Det,Dmax,Dave

       Det        Dmax        Dave
    0.1875       992.0       145.5
```

There is also be a final adjustment algorithm which can be used except when the design contains mixtures. This examines the design points one at a time to see whether the design can be improved by moving it a small amount along any x-axis. If an increase is possible, the point providing the greatest increase is moved. The process is then repeated until no improvment is possible. This phase is selected by setting the `ADJUSTMENTSTEP` option to the maximum amount (e.g. 0.2) by which the point may be moved on any axis.

Procedure `RQUADRATIC` can be used to analyse response-surface designs. This fits a quadratic surface, and can also estimate its stationary point (minimum or maximum).

4.9.15 Designs for nonlinear and generalized linear models

`AFNONLINEAR` procedure

Forms D-optimal designs to estimate the parameters of a nonlinear or generalized linear model (W. van den Berg).

Options

`PRINT` = *string token*	Controls printed output (`results`, `monitoring`); default `resu`, `moni`
`PLOT` = *string token*	Controls whether to plot the design (`design`); default `desi`
`YARGUMENT` = *identifier*	Data structure that stores the results of the function when it is calculated by expressions supplied by the `FUNCTION` option; must be set
`XARGUMENT` = *identifier*	Date structure representing the x-variate in the expressions supplied by the `FUNCTION` option; must be set
`FUNCTION` = *expression structures*	Specifies the function whose parameters are to be estimated; must be set
`FNDERIVATIVES` = *expression structures*	Specifies expressions to calculate derivative of the function with respect to each parameter; must be set
`ITERATIVEWEIGHTS` = *identifier*	Data structure that stores the iterative weights in the expressions supplied by the `FNITERATIVEWEIGHTS` option
`FNITERATIVEWEIGHTS` = *expression structures*	Specifies expressions to calculate the iterative weights when estimating the parameters of a generalized linear model
`XSUPPORT` = *variate*	Supplies the support points for the initial design, and

	saves those of the final design; if no initial values are supplied, an initial design is formed at random
XWEIGHTS = *variate*	Supplies the weights for the support points for the initial design, and saves those of the final design; if no initial values are supplied, equal weights are used initially
GRID = *variate*	Specifies the grid points where the design will be evaluated
A0 = *scalar*	Initial update weight; default 0.1
SEED = *scalar*	Seed for the random numbers used to select the initial design when not supplied by XSUPPORT and XWEIGHTS
NCYCLE = *scalar*	Number of iterations to make between at each value of A0, before halving it for the next batch of iterations; default 100
MAXCYCLE = *scalar*	Maximum number of iterations; default 2500
TOLERANCES = *variate*	Variate with two values specifying the convergence criterion and the tolerance for zero weights; default !(1.E-6,1.E-5)

Parameters

PARAMETER = *scalars*	Parameters of the nonlinear or generalized linear model (with values giving an indication of their likely estimated values)
DERIVATIVE = *identifiers*	Data structures that store the results of the calculation of the derivative for each parameter, in the expressions specified by the FNDERIVATIVES option

AFNONLINEAR constructs a design for estimating the parameters of a nonlinear or generalized linear model involving a single continuous variable *x*. The aim is to find the best values of *x* (i.e. the best *support points*) at which to observe the model, and a weight for each one. The design should then contain replicate observations at each of the support points, with the numbers of replicates in the same proportions as their weights. Suppose, for example, we have support points 1, 2 and 4, with weights 0.25, 0.25 and 0.5. A suitable design might then consist of observations at x-values 1, 2, 4 and 4 (i.e. 4 should have twice the replication of either 1 or 2). The designs that are produced are known as *continuous* designs, as the weights are not constrained to give an exact integer partitioning of the available points for any specific design size *N*. Instead you need to round *N* multiplied by each weight to the nearest feasible integer.

The model is specified in one, or more, expression structures by the FUNCTION option. The YARGUMENT gives the identifier of the data structure that receives the result of the function in the expressions, and the XARGUMENT gives the identifier of the data structure that provides the *x*-values. For example, we could define the negative exponential model

$$y = e^{(-b \times x)} + c$$

by

```
EXPRESSION Func; VALUE=!e( Y = EXP(-1*B*X) + C)
AFNONLINEAR [FUNCTION=Func; YARGUMENT=Y; XARGUMENT=X; ...
```

Notice that the data structures X and Y do not need to be declared. AFNONLINEAR simply needs to know which they are within the expression, so that it can replace them by the sets of *x*- and *y*-values that it really needs (using the REFORMULATE directive).

The parameters of the model (here B and C) must be specified by the PARAMETER parameter. These must be scalars, with values that give an indication of their likely estimated values. AFNONLINEAR also needs to be able to calculate the derivative of the function with respect to each parameter. You must specify expressions to do this using the FNDERIVATIVES option, and

indicate the data structures that will receive the results of the calculations using the DERIVATIVE parameter. So, for the negative exponential above, we need

```
EXPRESSION  Gfunc[1,2]; VALUE=!e( GradB = -1*X*EXP(-1*B*X) ),\
                              !e( GradC = 1 )
AFNONLINEAR [FUNCTION=Function; YARGUMENT=Y; XARGUMENT=X;\
            FNDERIVATIVE=Gfunc[]; XSUPPORT=X; XWEIGHTS=W;\
            GRID=Grid] PARAMETER=B,C; DERIVATIVE=GradB,GradC
```

The GRID option defines the *x*-values at which the design is evaluated. These should cover the range of feasible *x*-values.

The XSUPPORT option saves the support points of the design, in a variate. If the variate has values already defined on entry to AFNONLINEAR, these are used to provide the support points for the initial design where AFNONLINEAR begins its search. Otherwise AFNONLINEAR chooses an initial design at random by selecting *m* points at random from the grid points, where *m* is twice the number of parameters in the model. The SEED option specifies a seed for the random numbers that are used to make the selection. The default value of zero continues an existing sequence of random numbers if any have already been used in the current Genstat job, or obtains a random seed using system clock if none have been used already.

The XWEIGHTS option saves the weights of the support points, in a variate, and can supply weights for an initial design. Otherwise AFNONLINEAR starts with equal weights.

To form designs for generalized linear models, you also need to supply expressions to calculate the iterative weights at various *x*-values. The FNITERATIVEWEIGHTS option supplies the expressions, and the ITERATIVEWEIGHTS option indicate the data structure that will receive the results of the calculations.

By default AFNONLINEAR produces a plot showing the function and prediction variance at the selected grid points, but you can suppress this by setting option PLOT=*. Figure 4.9.15 shows the plot, produced in Example 4.9.15, for the negative model discussed above.

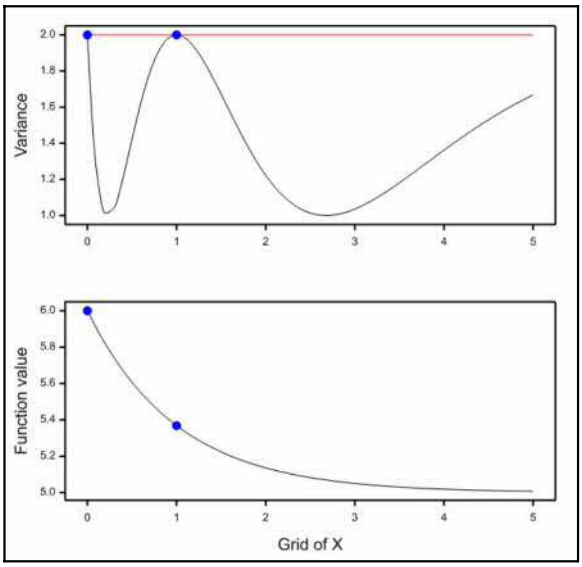


Figure 4.9.15

AFNONLINEAR uses the algorithm of Federov (1972). This involves a sequence of iterations in which a new support point may be added, or the weight of an existing point may be increased. The A0 option specifies the weights to be given to a new point, or to be added to an existing point. (The weights of the other support points are then decreased, proportionally, so that the weights still add up to one.) The NCYCLE option controls how many iterations are made with each value of A0 (default 100); so, at the end of each set of NCYCLE iterations, A0 is divided by two in order for the weights to converge to a stable solution.

The TOLERANCES option can be set to a variate of length two, to specify the convergence criterion and the tolerance for zero weights (defaults 10^{-6} and 10^{-5}, respectively). The algorithm stops when the number of support points equals the number of parameters, and the prediction variance minus the number of parameters is less than the first TOLERANCES value. Weights less than the second TOLERANCES value are set to zero at each iteration (so that the corresponding points leave the design).

Example 4.9.15 uses AFNONLINEAR to form a design for the negative exponential model, discussed above.

Example 4.9.15

```
  2  VARIATE     [VALUES=3,4] X
  3  VARIATE     [VALUES=0.6,0.4] W
  4  VARIATE     [VALUES=0,0.1... 5] Grid
  5  SCALAR      B,C; VALUE=1,5
  6  EXPRESSION  Function; VALUE=!e( Y = EXP(-1*B*X) + C )
  7  EXPRESSION  Gfunction[1,2]; VALUE=!e( GradB = -1*X*EXP(-1*B*X) ),\
  8                                 !e( GradC = 1 )
  9  AFNONLINEAR [PRINT=results; FUNCTION=Function; YARGUMENT=Y; XARGUMENT=X;\
 10              FNDERIVATIVE=Gfunction[]; XSUPPORT=X; XWEIGHTS=W; GRID=Grid]\
 11              PARAMETER=B,C; DERIVATIVE=GradB,GradC

Design for estimating a nonlinear model
=======================================

Function: Y = EXP((-1*B*X)) + C
Number of iterations: 1100
Maximum variance: 2.000
A_0: 0.00004883

Design points     Weights
       1.0000      0.5000
       0.0000      0.5000
```

4.9.16 Reference-level designs

AGREFERENCE procedure

Generates reference-level designs e.g. for microarray experiments (R.W. Payne).

Option

PRINT = *string token*	Controls whether or not to print a plan of the design (design); if unset in an interactive run AGREFERENCE will ask whether the design is to be printed, in a batch run the default is not to print the design

Parameters

LEVELS = *scalars*	Number of treatments
REFLEVEL = *scalars*, *variates* or *pointers*	Reference level(s); if this is unset in an interactive run you will be asked which reference level or levels you want, in a batch run the default is level 1
REFUNIT = *scalars*, *variates* or *pointers*	Unit(s) to which to allocate the reference level(s); if this is unset in an interactive run you will be asked which reference level or levels you want, in a batch run the default is to choose the unit at random within each block
SEED = *scalars*	Seed for randomization; a negative value implies no randomization
TREATMENTS = *factors*	Identifier for the treatment factor
BLOCKS = *factors*	Identifier for the block (plate) factor
UNITS = *factors*	Identifier for the factor for the units within each block (or colours in a microarray experiment)
STATEMENT = *texts*	Saves a command to recreate the design (useful if the design information has been specified in response to questions from AGREFERENCE)

Reference-level designs can be useful in experiments where the main aim is to compare new treatments with a control, or reference, treatment. The design is made up of blocks of size two, each of which compares the control with one of the new treatments. So, if there are four treatment and the reference treatment is treatment 1, the basic design would have three blocks containing the pairs of treatments (1, 2), (1, 3) and (1, 4). The design is particularly relevant to two-colour microarray experiments, where each slide compares a pair of treatments, one of which is stained with a red dye and the other with a green dye.

If you are running Genstat interactively, you do not need to specify any of the options or parameters of AGREFERENCE. It then asks questions to determine the necessary information to form the design: for example, the number of treatments, and which of the treatments is the control. The options and parameters allow you to anticipate questions, or to define all the necessary information if you want to use AGREFERENCE in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The number of treatments (including the reference treatment) can be defined using the LEVELS parameter. Similarly, the REFLEVEL parameter can define the reference treatment or treatments. You can supply a scalar to define a single reference treatment, or a variate, or a pointer containing several scalars, to define several. The REFUNIT similarly indicates which unit is to be used for the reference treatment within each block. (In a microarray experiment, the "unit" would be the colour, red or green, and each block would be a slide.) The numbers specified for the reference unit should be either 1 to use the first unit, or 2 to use the second, or 0 to use a unit selected at random for each block.

You can thus construct several versions of the basic design, each using a different reference level and/or unit. For example

```
VARIATE     [VALUES=1,2] V12
AGREFERENCE 6; REFLEVEL=1; REFUNIT=V12
```

would define a design with two blocks to compare the reference treatment with each of the other five treatments (see Example 4.9.16). In one of the blocks the reference treatment would be on unit one (e.g. colour red on a microarray plate) and in the other it would be on unit two (e.g. colour green). Similarly

```
AGREFERENCE 6; REFLEVEL=V12; REFUNIT=1
```

would generate two versions of the basic design. The first would have treatment one as the reference, and the second would have treatment two as the reference (both allocated to unit one).

```
AGREFERENCE 4; REFLEVEL=V12; REFUNIT=V12
```

would generate two versions of the basic design. The first would have treatment one as the reference (allocated to unit 1), and the second would have treatment two as the reference (allocated to unit 2).

The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is –1, to suppress randomization. If you do not set SEED when running interactively AGREFERENCE will ask for a seed, and again a negative value suppresses any randomization. Note that the randomization takes account of the settings of the REFUNIT parameter.

The remaining parameters, TREATMENTS, BLOCKS and UNITS, allow you to specify identifiers for the factors representing treatments, blocks (or plates in a microarray experiment) and units within blocks (or colours in a microarray experiment). If these are not specified in a batch run, AGREFERENCE will use identifiers that are local within the procedure and thus lost at the end of the procedure. If you are running interactively, AGREFERENCE will ask you to provide identifiers, and these will remain available after AGREFERENCE has finished running.

AGREFERENCE has a PRINT option which can be set to design to print the plan of the design.

By default, if you are running Genstat in batch, neither are printed. If you do not set PRINT when running interactively, AGREFERENCE will ask whether or not you wish to print the design.

Example 4.9.16

```
  2  VARIATE     [VALUES=1,2] V12
  3  AGREFERENCE [PRINT=design] 6; REFLEVEL=1; REFUNIT=V12; SEED=142527;\
  4              TREATMENTS=Treat; BLOCKS=Plate; UNITS=Color

Treatments on each unit of the design
=====================================

 Plate   1   2   3   4   5   6   7   8   9  10
 Color
     1   3   4   1   1   1   1   2   1   6   5
     2   1   1   2   4   3   6   1   5   1   1

Treatment factor: Treat.
```

4.9.17 Loop designs

AGLOOP procedure

Generates loop designs e.g. for time-course microarray experiments (R.W. Payne).

Option

PRINT = *string token*	Controls whether or not to print a plan of the design (design); if unset in an interactive run AGLOOP will ask whether the design is to be printed, in a batch run the default is not to print the design

Parameters

LEVELS = *scalars*	Number of treatments
INCREMENTS = *scalars*, *variates* or *pointers*	Increment or increments to be used to form the loops
SEED = *scalars*	Seed for randomization; a negative value implies no randomization
TREATMENTS = *factors*	Identifier for the treatment factor
BLOCKS = *factors*	Identifier for the block (plate) factor
UNITS = *factors*	Identifier for the factor for the units within each block (or colours in a microarray experiment)
STATEMENT = *texts*	Saves a command to recreate the design (useful if the design information has been specified in response to questions from AGLOOP)

Loop designs are also used in two-colour microarray experiments (see 4.9.16). Suppose that the treatments are t_1, t_2 ... t_n. Then, before randomization in the basic form of the design, the first slide would compare t_1 (using red) with t_2 (using green), the second slide would compare t_2 (red) with t_3 (green), and the nth slide would compare t_n (red) with t_1 (green). The design has the advantage that treatments are balanced with colours. This basic form is also very effective for making comparisons between treatments that are adjacent in the sequence t_1 ... t_n, as might be the main point of interest when the treatments correspond to time.

Comparisons between more widely spaced treatments are less well estimated So an alternative possibility is to choose more than one increment, and construct additional cycles through the treatments using modulo arithmetic. The design is then known as an *interwoven loop design*.

None of the increments, other than 1, must be a divisor of the number of treatments as its cycle would then fail to include all the treatments. For example, with 8 treatments an increment of 3 would be satisfactory (1, 4, 7, 2, 5, 8, 3, 6, 1) but 2 would not (1, 3, 5, 7, 1). Note also, that 5 (which is 8 – 3) would be equivalent to 3 (1, 6, 3, 8, 5, 2, 7, 4, 1); the treatments appear in the reverse order, so the adjacent pairs are the same.

If you are running Genstat interactively, there is no need to specify any of the options or parameters of AGLOOP. It then asks questions to determine the necessary information to form the design: for example, the number of treatments and the increments to use. The option and parameters allow you to anticipate questions, or to define all the necessary information if you want to use AGLOOP in batch. If, however, you wish to recreate the same design later, the STATEMENT parameter allows you to save a Genstat text structure containing a command specifying the same information.

The number of treatments can be defined using the LEVELS parameter. Similarly, the INCREMENTS parameter can supply a scalar defining a single increment, or a variate, or a pointer containing several scalars, to define several. The SEED parameter allows you to specify a seed to be used to randomize the design. In batch the default seed is –1, to suppress randomization. If you do not set SEED when running interactively AGLOOP will ask for a seed, and again a negative value suppresses any randomization. Note that, the randomization is constrained to ensure that the treatments remain balanced with colour.

The remaining parameters, TREATMENTS, BLOCKS and UNITS, allow you to specify identifiers for the factors representing treatments, blocks (or plates in a microarray experiment) and units within blocks (or colours in a microarray experiment). If these are not specified in a batch run, AGLOOP will use identifiers that are local within the procedure and thus lost at the end of the procedure. If you are running interactively, AGLOOP will ask you to provide identifiers, and these will remain available after AGLOOP has finished running.

AGLOOP has a PRINT option which can be set to design to print the plan of the design. By default, if you are running Genstat in batch, neither are printed. If you do not set PRINT when running interactively, AGLOOP will ask whether or not you wish to print the design.

Example 4.9.17 shows an interwoven loop design for 8 treatments with increments of 1 and 3. The design is unrandomized, so that the looping can be seen more clearly.

Example 4.9.17

```
  2  AGLOOP   [PRINT=design] LEVELS=8; INCREMENT=!(1,3); SEED=-1;\
  3           TREATMENTS=time; BLOCKS=plate; UNITS=color

Treatments on each unit of the design
=====================================

 plate   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16
 color
     1   1   2   3   4   5   6   7   8   1   4   7   2   5   8   3   6
     2   2   3   4   5   6   7   8   1   4   7   2   5   8   3   6   1

Treatment factor: time.
```

4.10 Displaying a design

This section describes the procedures for printing designs, displaying field plans and generating data forms.

4.10.1 Printing a design: the PDESIGN procedure

PDESIGN procedure

Prints or stores treatment combinations tabulated by the block factors (R.W. Payne).

Options

PRINT = *string token*	Controls the printing of the design (design); default desi
BLOCKSTRUCTURE = *formula*	Defines the block factors for the design; the default is to take those specified by the BLOCKSTRUCTURE directive
TREATMENTSTRUCTURE = *formula*	Defines the treatment factors for each design; the default is to take those specified by the TREATMENTSTRUCTURE directive
TABLES = *pointer*	Contains tables to store the tabulated factor values for printing outside the procedure in some other format
FREPRESENTATION = *string token*	How to represent the factor values (labels, levels); default leve

No parameters

PDESIGN allows the treatment combinations allocated to each plot in a design to be displayed as tables, classified by the block factors.

The combinations are represented using the levels of the treatment factors. If any factor also has labels these are printed alongside the levels, as a key, after the tables. The levels are printed in formats that are determined automatically in a way that avoids wasted space or unnecessary decimal places. Alternatively, if you set option FREPRESENTATION=labels, the labels are displayed in the table, instead of the levels.

The block factors are obtained from the block structure of the design, which can be specified explicitly using the BLOCKSTRUCTURE option; otherwise PDESIGN will use any structure that has already been defined by a BLOCKSTRUCTURE statement earlier in the job. Similarly, the treatment factors are obtained either from the TREATMENTSTRUCTURE option of the procedure, or from an earlier TREATMENTSTRUCTURE statement.

If the display produced by the procedure is unsuitable, printing can be suppressed by setting option PRINT=* (by default PRINT=design), and the tables of treatment levels can be saved for printing outside the procedure by setting the TABLES option to a pointer. This will be returned with an element for each treatment factor, pointing to a table classified by the block factors and storing the tabulated levels of the treatment.

Example 4.10.1 uses PDESIGN to print the plan of the split-plot design in Example 4.2.1 (continuing from the end of Example 4.6d). We have specified the block structure and treatment structure explicitly, but could have allowed PDESIGN to have taken these from BLOCKSTRUCTURE and TREATMENTSTRUCTURE statements earlier in the example.

Example 4.10.1

```
65  PDESIGN [BLOCKSTRUCTURE=Blocks/Wplots/Subplots;\
66    TREATMENTSTRUCTURE=Variety*Nitrogen]
```

```
Treatment combinations on each unit of the design
=================================================

          Subplots   1     2     3     4
   Blocks   Wplots
        1         1   3 4   3 3   3 2   3 1
                  2   1 1   1 2   1 4   1 3
                  3   2 1   2 2   2 3   2 4
        2         1   3 3   3 1   3 2   3 4
                  2   1 4   1 1   1 2   1 3
                  3   2 2   2 1   2 3   2 4
        3         1   2 2   2 3   2 4   2 1
                  2   3 4   3 2   3 3   3 1
                  3   1 1   1 4   1 2   1 3
        4         1   3 3   3 4   3 1   3 2
                  2   2 1   2 3   2 4   2 2
                  3   1 2   1 3   1 4   1 1
        5         1   2 4   2 1   2 3   2 2
                  2   1 3   1 4   1 1   1 2
                  3   3 3   3 4   3 2   3 1
        6         1   1 3   1 1   1 4   1 2
                  2   2 4   2 3   2 1   2 2
                  3   3 1   3 2   3 3   3 4

Treatment factors are listed in the order: Variety, Nitrogen.

 Labels of Variety:
           1     Victory
           2 Golden rain
           3  Marvellous

 Labels of Nitrogen:
           1       0 cwt
           2     0.2 cwt
           3     0.4 cwt
           4     0.6 cwt
```

4.10.2 Plotting the plan of a design: the DDESIGN procedure

DDESIGN procedure

Plots the plan of an experimental design (K.E. Bicknell & R.W. Payne).

Options

`Y` = *variate*	Specifies the *y* position of the plots in standard coordinates 1 ... number of rows of plots in the experiment (taking 1 as the top row of the window)
`X` = *variate*	Specifies the *x*-coordinate of the plots in standard coordinates 1 ... number of columns of experimental plots
`TITLE` = *text*	Title for the plan
`WINDOW` = *scalar*	Window number for the plan; default 3
`KEYWINDOW` = *scalar*	Window number for the key; default 0
`SCREEN` = *string token*	Whether to clear the screen before plotting (`clear`, `keep`); default `clea`
`KEYDESCRIPTION` = *text*	Overall description for the key; default *
`ENDACTION` = *string token*	Action to be taken after completing the plot (`continue`, `pause`); default * uses the setting from the last `DEVICE` statement
`CHARACTERS` = *scalar*	Sets a limit on the length of each factor label; default * i.e. none
`SIZE` = *scalar*	Provides a multiplier by which to scale the sizes of the

factor labels on the plan

Parameters

FACTOR = *factors*	Factors to be listed on the plan and to define the layout (the procedure determines the style of line to divide each pair of plots in the design from the grid pen of the first factor in the list with which they have different levels); default * forms the list from first the factors specified by a preceding BLOCKSTRUCTURE statement, and then those specified by a preceding TREATMENTSTRUCTURE statement
PEN = *scalars*	Pen to be used to write the levels of each factor on the plan (if PEN=0 the levels of that factor are not included); default 1 if the FACTOR parameter is specified, otherwise 0 for block factors and 1 for treatment factors
PENGRID = *scalars*	Pens to be used to draw the boundaries between the plots in the design (according to the first FACTOR with which they have different levels but ignoring factors with PENGRID=0); default 1,2...
LABELS = *texts*	Labels to be used for each factor if its own levels or labels are inappropriate

DDESIGN uses high-resolution graphics to produce a plan of an experimental design. The plots in the design are assumed to be arranged on a rectangular grid. The rows of the plots are assumed to run from 1 (at the top of the graph) upwards and are specified by a variate supplied by the Y option. The columns (again running from 1 upwards) specified by a variate supplied by the X option. If either Y or X is not specified, DDESIGN will generate values automatically according to the factors in the design.

The TITLE, WINDOW, KEYWINDOW, SCREEN, KEYDESCRIPTION and ENDACTION options operate as usual in high-resolution graphics. The CHARACTERS option allows a limit to be set on the length of each factor label when written on the plan, and the SIZE option allows the size of the plotted factor labels to be scaled (using the SIZE parameter of the PEN directive).

The factors involved in the experiment can be listed using the FACTOR parameter. If this is omitted DDESIGN forms the list firstly from the factors in the previous BLOCKSTRUCTURE statement (or a "units" factor if there was none), and then from the factors (if any) in the previous TREATMENTSTRUCTURE statement.

These factors are then used to draw the plan and to label the plots in the design. The PEN parameter allows the levels or labels of the factors to be drawn using different pens (and thus, for example, in different colours). If the pen for any factor is defined as zero, its levels/labels are not included. However, it can still be used to determine the lines drawn to delimit the plots. For these lines, DDESIGN considers each pair of adjacent plots and checks through the list of factors to find the first one for which they have different levels. It then uses the grid pen (defined by the PENGRID parameter) to draw the dividing line. If the grid pen of any factor is zero, it is ignored.

This makes it very easy to achieve the usual style of plan in which stronger lines are used for example to indicate the boundaries between different blocks than between the plots within blocks. For example, the parameter settings to draw a randomized block design with a single treatment factor Treat in this way would be

```
FACTOR=Block,Plots,Treat; PEN=1; PENGRID=1,2,0
```

if all the factors are to have their levels listed within the plots, or

```
FACTOR=Block,Plots,Treat; PEN=0,0,1; PENGRID=1,2,0
```

if only Treat is to be listed. Note that, as each pair of plots will have different levels of either

`Block` or `Plot` (or both), the `PENGRID` specified here for `Treat` is irrelevant.

If a plot has no neighbour in some direction, `DDESIGN` will check the next but one plot; if this too is not used in the design, the grid pen of the first `FACTOR` is used to mark the boundary.

The final parameter, `LABELS`, allows alternative labels to be specified for each factor if the existing ones are inappropriate.

For example, the plan of the split-plot design in Example 4.2.1 can be plotted by

```
DDESIGN [Y=Row; X=Column] Blocks,Wplots,Subplots,\
  Variety,Nitrogen; PEN=0,0,0,1,1; PENGRID=1,2,3,0,0
```

The resulting graph is in Figure 4.10.2.

Marvellous 0.6 cwt	Marvellous 0.4 cwt	Marvellous 0.4 cwt	Marvellous 0.6 cwt
Marvellous 0.2 cwt	Marvellous 0 cwt	Marvellous 0 cwt	Marvellous 0.2 cwt
Victory 0 cwt	Victory 0.2 cwt	Golden rain 0 cwt	Golden rain 0.4 cwt
Victory 0.6 cwt	Victory 0.4 cwt	Golden rain 0.6 cwt	Golden rain 0.2 cwt
Golden rain 0 cwt	Golden rain 0.2 cwt	Victory 0.2 cwt	Victory 0.4 cwt
Golden rain 0.4 cwt	Golden rain 0.6 cwt	Victory 0.6 cwt	Victory 0 cwt
Marvellous 0.4 cwt	Marvellous 0 cwt	Golden rain 0.6 cwt	Golden rain 0 cwt
Marvellous 0.2 cwt	Marvellous 0.6 cwt	Golden rain 0.4 cwt	Golden rain 0.2 cwt
Victory 0.6 cwt	Victory 0 cwt	Victory 0.4 cwt	Victory 0.6 cwt
Victory 0.2 cwt	Victory 0.4 cwt	Victory 0 cwt	Victory 0.2 cwt
Golden rain 0.2 cwt	Golden rain 0 cwt	Marvellous 0.4 cwt	Marvellous 0.6 cwt
Golden rain 0.4 cwt	Golden rain 0.6 cwt	Marvellous 0.2 cwt	Marvellous 0 cwt
Golden rain 0.2 cwt	Golden rain 0.4 cwt	Victory 0.4 cwt	Victory 0 cwt
Golden rain 0.6 cwt	Golden rain 0 cwt	Victory 0.6 cwt	Victory 0.2 cwt
Marvellous 0.6 cwt	Marvellous 0.2 cwt	Golden rain 0.6 cwt	Golden rain 0.4 cwt
Marvellous 0.4 cwt	Marvellous 0 cwt	Golden rain 0 cwt	Golden rain 0.2 cwt
Victory 0 cwt	Victory 0.6 cwt	Marvellous 0 cwt	Marvellous 0.2 cwt
Victory 0.2 cwt	Victory 0.4 cwt	Marvellous 0.4 cwt	Marvellous 0.6 cwt

Figure 4.10.2

4.10.3 Plans and data forms in spreadsheets: the `ADSPREADSHEET` procedure

`ADSPREADSHEET` procedure

Puts the data and plan of an experimental design into a spreadsheet (R.W. Payne).

Options

`DATA` = *factors* or *variates* — Data variables (e.g. design factors and covariates) to put into the `data` spreadsheet; default takes the factors defined by previous `BLOCKSTRUCTURE` and

	TREATMENTSTRUCTURE directives
NEWDATA = *variates*	New variates (e.g. measurements to be taken during the experiment) to create and put into the data spreadsheet; default * i.e. none
Y = *variate* or *factor*	Specifies the *y* position of the plots for the plan spreadsheet
X = *variate* or *factor*	Specifies the *x*-coordinate of the plots for the plan spreadsheet
CONSTANTFACTORS = *string tokens*	Whether to put factors whose levels are constant in the y- or x-direction in a separate row or column of the Plan spreadsheet (y, x); default * i.e. neither
SEPARATOR = *text*	Separator for factor values in the plan spreadsheet; default '; '
OMITGAPS = *string token*	Whether to omit gaps when the plots in the plan are equally spaced (yes, no); default no
FOREGROUND = *scalar*, *variate* or *text*	Foreground colours to use for the plots in the experiment; default 'Black'
BACKGROUND = *scalar*, *variate* or *text*	Background colours to use for the plots in the experiment; default 'BlanchedAlmond'
CFACTORS = *factors*	Factors to determine the colour to use for each plot; default uses the first block factor or no colouring otherwise
GAPFOREGROUND = *text* or *scalar*	Foreground colour for gaps and surrounding plots; default 'Black'
GAPBACKGROUND = *text* or *scalar*	Background colour for gaps and surrounding plots; default 'LightGreen'
YFOREGROUND = *text* or *scalar*	Foreground colour for factors constant in y-direction; default 'Black'
YBACKGROUND = *text* or *scalar*	Background colour for factors constant in y-direction; default 'PaleTurquoise'
XFOREGROUND = *text* or *scalar*	Foreground colour for factors constant in x-direction; default 'Black'
XBACKGROUND = *text* or *scalar*	Background colour for factors constant in x-direction; default 'LightCyan'
SPREADSHEET = *string tokens*	Which spreadsheets to form (data, plan); default data
OUTFILENAME = *texts*	Name of Genstat workbook file (.gwb) or Excel (.xls or .xlsx) file to create

Parameters

FACTOR = *factors*	Factors to include in the plan spreadsheet; if unset, includes the factors defined by a previous TREATMENTSTRUCTURE directive
LABELS = *texts*	Labels to be used for each factor if its own levels or labels are inappropriate

---

ADSPREADSHEET puts information about an experimental design into a spreadsheet. By default the spreadsheet is opened within Genstat itself, but you can save it to an external file by supplying its name using the OUTFILENAME option. The file can be a Genstat workbook (.gwb)

or an Excel spreadsheet (.xls or .xlsx). If the name is specified without a suffix, '.gwb' is added (so that a Genstat workbook is saved).

The contents of the data spreadsheet are specified by the DATA and NEWDATA options. The DATA option lists existing data variables (i.e. design factors and covariates) to put into the data spreadsheet. If this is unset, the default is to take the factors defined by previous BLOCKSTRUCTURE and TREATMENTSTRUCTURE directives; ADSPREADSHEET gives a failure diagnostic if the DATA option is unset and there has been no previous BLOCKSTRUCTURE or TREATMENTSTRUCTURE. The NEWDATA option allows you to include new spreadsheet columns to provide blank cells for new variates like measurements that are to be taken during the experiment. For security all the existing variables are protected so that they are read-only.

The locations of the plots in the plan spreadsheet are specified by variates or factors supplied by the X and Y parameters; these define the row and column of the plots in the sheet, respectively (with row coordinates increasing from top to bottom, and column coordinates increasing from left to right in the usual way). The plots need not be equally spaced. However, ADSPREADSHEET looks to see whether the coordinates in either direction are taken from a regular grid, possibly with some gaps: for example coordinates (1, 2, 4, 6) are on a grid with spacing 1 and gaps at 3 and 5. If so, ADSPREADSHEET will include rows or columns for all the coordinates, including the gaps (i.e, 1, 2, 3, 4, 5 and 6 for the example), unless you set option OMITGAPS=yes. The x-coordinates are shown in a units column of the spreadsheet, and the y-coordinates are given in a row at the bottom of the plan. If either Y or X is not specified, ADSPREADSHEET will generate values automatically according to the factors in the design – factors from a previous BLOCKSTRUCTURE directive, if available, otherwise from a previous TREATMENTSTRUCTURE directive.

The factors to include in the plan can be specified using the FACTOR parameter. If this is omitted, ADSPREADSHEET takes the factors from a previous TREATMENTSTRUCTURE directive (and fails if there has been none). The values of each factor are represented by its labels, if available, or otherwise its levels. The LABELS parameter allows alternative labels to be specified for each factor, if the existing levels or labels are too unsuitable. The values of the factors in each plot are listed in the equivalent cell of the spreadsheet. By default, they are separated from each other by a semi-colon and a space, but you can supply alternative separating characters using the SEPARATOR option. You can set option CONSTANTFACTORS to x to list the values of factors whose values are constant in the x direction separately, in a column on the left-hand side of the sheet. Similarly, the setting y causes factors whose values are constrant in the y-direction to be listed in a row at the top of the sheet.

The colouring of the cells in a Genstat can be controlled using the FOREGROUND, BACKGROUND, CFACTORS, GAPFOREGROUND, GAPBACKGROUND, YFOREGROUND, YBACKGROUND, XFOREGROUND and XBACKGROUND options. The colours can be specified as numbers defining RGB values, or texts containing names of the standard Genstat colours; see the PEN diective for details. The FOREGROUND and BACKGROUND options control the colours of the text and background, respectively, of the spreadsheet cells that correspond to plots in the experiment. You can give the plots different colours by supplying several values (in texts or variates). ADSPREADSHEET then uses a different colour for each combination of levels of the factor or factors specified by the CFACTORS option. If several colours are defined, but CFACTORS is not set, the first factor in the block factor (in BLOCKSTRUCTURE) is used. If there are no block factors, the first defined colour is used for all the plots. The GAPFOREGROUND and GAPBACKGROUND options define the colour to use for the cells representing gaps in the experiment or surrounding it. The YFOREGROUND and YBACKGROUND options specify the colour for the text and background in the cells containing the names and levels of the factors constant in the y-direction. The XFOREGROUND and XBACKGROUND options similarly specify the colour for the text and background for the factors constant in the x-direction.

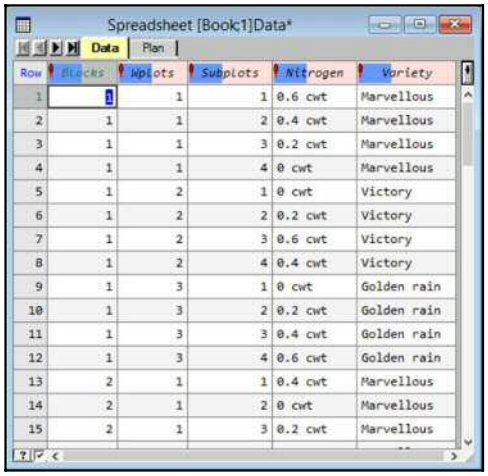
If X or Y or any of the factors in the plan is restricted, only the unrestricted plots will be

included in the plan spreadsheet.

For example, the factor values and plan of the split-plot design in Example 4.2.1 can be displayed in a spreadsheet by

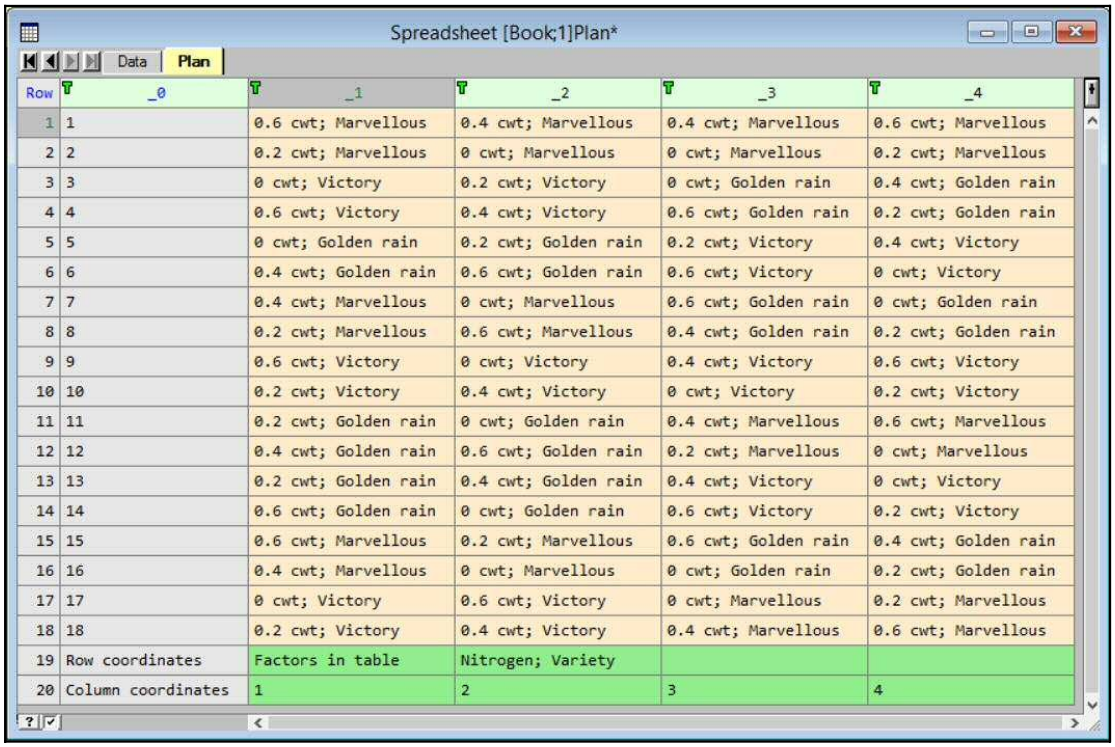
```
ADSPREADSHEET [Y=Row; X=Column; SPREADSHEET=data,plan]
```

The Data tab of the resulting spreadsheet is shown in Figure 4.10.3a, and the Plan tab is shown in Figure 4.10b. The blue marks on the factor columns of the Data tab indicate that these columns are "protected" to prevent their values being changed.



Row	Blocks	Wplots	Subplots	Nitrogen	Variety
1	1	1	1	0.6 cwt	Marvellous
2	1	1	2	0.4 cwt	Marvellous
3	1	1	3	0.2 cwt	Marvellous
4	1	1	4	0 cwt	Marvellous
5	1	2	1	0 cwt	Victory
6	1	2	2	0.2 cwt	Victory
7	1	2	3	0.6 cwt	Victory
8	1	2	4	0.4 cwt	Victory
9	1	3	1	0 cwt	Golden rain
10	1	3	2	0.2 cwt	Golden rain
11	1	3	3	0.4 cwt	Golden rain
12	1	3	4	0.6 cwt	Golden rain
13	2	1	1	0.4 cwt	Marvellous
14	2	1	2	0 cwt	Marvellous
15	2	1	3	0.2 cwt	Marvellous

Figure 4.10.3a



Row	_0	_1	_2	_3	_4
1	1	0.6 cwt; Marvellous	0.4 cwt; Marvellous	0.4 cwt; Marvellous	0.6 cwt; Marvellous
2	2	0.2 cwt; Marvellous	0 cwt; Marvellous	0 cwt; Marvellous	0.2 cwt; Marvellous
3	3	0 cwt; Victory	0.2 cwt; Victory	0 cwt; Golden rain	0.4 cwt; Golden rain
4	4	0.6 cwt; Victory	0.4 cwt; Victory	0.6 cwt; Golden rain	0.2 cwt; Golden rain
5	5	0 cwt; Golden rain	0.2 cwt; Golden rain	0.2 cwt; Victory	0.4 cwt; Victory
6	6	0.4 cwt; Golden rain	0.6 cwt; Golden rain	0.6 cwt; Victory	0 cwt; Victory
7	7	0.4 cwt; Marvellous	0 cwt; Marvellous	0.6 cwt; Golden rain	0 cwt; Golden rain
8	8	0.2 cwt; Marvellous	0.6 cwt; Marvellous	0.4 cwt; Golden rain	0.2 cwt; Golden rain
9	9	0.6 cwt; Victory	0 cwt; Victory	0.4 cwt; Victory	0.6 cwt; Victory
10	10	0.2 cwt; Victory	0.4 cwt; Victory	0 cwt; Victory	0.2 cwt; Victory
11	11	0.2 cwt; Golden rain	0 cwt; Golden rain	0.4 cwt; Marvellous	0.6 cwt; Marvellous
12	12	0.4 cwt; Golden rain	0.6 cwt; Golden rain	0.2 cwt; Marvellous	0 cwt; Marvellous
13	13	0.2 cwt; Golden rain	0.4 cwt; Golden rain	0.4 cwt; Victory	0 cwt; Victory
14	14	0.6 cwt; Golden rain	0 cwt; Golden rain	0.6 cwt; Victory	0.2 cwt; Victory
15	15	0.6 cwt; Marvellous	0.2 cwt; Marvellous	0.6 cwt; Golden rain	0.4 cwt; Golden rain
16	16	0.4 cwt; Marvellous	0 cwt; Marvellous	0 cwt; Golden rain	0.2 cwt; Golden rain
17	17	0 cwt; Victory	0.6 cwt; Victory	0 cwt; Marvellous	0.2 cwt; Marvellous
18	18	0.2 cwt; Victory	0.4 cwt; Victory	0.4 cwt; Marvellous	0.6 cwt; Marvellous
19	Row coordinates	Factors in table	Nitrogen; Variety		
20	Column coordinates	1	2	3	4

Figure 4.10.3b

4.11 Randomization

Randomization can be done using either the ARANDOMIZATION procedure or the RANDOMIZE directive. ARANDOMIZE uses RANDOMIZE internally, but packages the facilities more conveniently for experimental designs. It also allows the ordering of the levels of the treatment factors to be permuted, as may be required for example in incomplete-block designs.

4.11.1 The RANDOMIZE directive

RANDOMIZE directive

Randomizes the units of a designed experiment or the elements of a factor or variate.

Options

BLOCKSTRUCTURE = *formula*	Block model according to which the randomization is to be carried out; default * i.e. as a completely-randomized design
EXCLUDE = *factors*	(Block) factors whose levels are not to be randomized
SEED = *scalar*	Seed for the random-number generator; default 0

Parameters

factors or *variates*	Structures whose units are to be randomized according to the defined block model

In its simplest form, RANDOMIZE merely performs a random permutation of the units of a list of factors or variates. You list these structures with the parameter of RANDOMIZE. Genstat gives them all exactly the same permutation, which is produced by a set of random numbers generated from the SEED option. For example

```
RANDOMIZE [SEED=144556] X,Y
```

puts the values of X and Y into an identical random order. The seed can be any positive integer, but only the last six digits of its integer part are used. Thus the seeds 2144556 and 7144556.3 are both equivalent to the seed 144556. If you put SEED=*, or leave it unset, Genstat picks a seed at random.

If you have restricted any of the structures in the parameter list (1:4.4.1), then all will be treated as though they were restricted; moreover, all the restricted structures must be restricted in exactly the same way.

The main use of RANDOMIZE, however, is to randomize the allocation of treatments to units in a designed experiment. In the analysis of designed experiments, the underlying structure of an experiment is defined by the block formula, as described in 4.2. Provided the only operators in a block formula are the nesting (/) and crossing (*) operators, this also specifies the correct randomization of the experiment.

The nesting operator specifies that one factor is to be randomized within another one. The simplest example is the randomized block design: its block formula is Blocks/Plots; a separate randomization of plots is done for each block. Another example is a split-plot design, the formula for which is Blocks/Wplots/Subplots; this means randomize first the levels of Blocks, then the levels of Wplots within levels of Blocks, and finally the levels of Subplots within the levels of Blocks and Wplots. In other words, there is a separate randomization of Wplots for each Block, and a separate randomization of Subplots for each Wplot. A similar formula and randomization would apply to a resolvable incomplete-block design.

The crossing operator specifies that the factors are to be randomized independently of each other. For example the formula Rows*Cols means randomize the levels of Rows and Cols separately. Thus the same randomization of Cols appears within each Row. This is the block formula associated with a row and column design, for example a Latin square. This is illustrated in Example 4.11.2, which does the randomization using ARANDOMIZE (which then uses RANDOMIZE).

You specify the block formula by the BLOCKSTRUCTURE option, which thus defines the way in which the randomization is to be carried out. Genstat does not randomize the factors in the block structure themselves, unless you put them into the parameter list. This is because the

original order of the block-factor levels often describes actual positions in the experiment; for example, in a field. So you will be interested in keeping these values, rather than the random ordering of them that is used to allocate treatments.

Example 4.11.1a, shows the randomization of a randomized block design.

Example 4.11.1a

```
2  UNITS [NVALUES=16]
3  FACTOR [LEVELS=4; VALUES=4(1...4)] Blocks
4  & [VALUES=(1...4)4] Plots
5  & [LABELS=!T(A,B,C,D)] Dose
6  PRINT Blocks,Plots,Dose

   Blocks       Plots        Dose
        1           1           A
        1           2           B
        1           3           C
        1           4           D
        2           1           A
        2           2           B
        2           3           C
        2           4           D
        3           1           A
        3           2           B
        3           3           C
        3           4           D
        4           1           A
        4           2           B
        4           3           C
        4           4           D

7  RANDOMIZE [BLOCKSTRUCTURE=Blocks/Plots; SEED=556743] Dose
8  PRINT Blocks,Plots,Dose

   Blocks       Plots        Dose
        1           1           C
        1           2           B
        1           3           D
        1           4           A
        2           1           C
        2           2           B
        2           3           A
        2           4           D
        3           1           B
        3           2           C
        3           3           D
        3           4           A
        4           1           A
        4           2           C
        4           3           D
        4           4           B
```

Notice that the values of the `Blocks` and `Plots` factors have not been randomized because they did not appear in the parameter list. Note also that the block formula for this design is `Blocks/Plots` and not just `Blocks`. This is because the formula must define each experimental unit by a unique combination of the block factor levels, for example block 1, plot 3. To put a block formula of just `Blocks` would not give Genstat any information about what to do with the elements of the blocks.

You should use the `EXCLUDE` option if you want to restrict the randomization so that one or more of the factors in the block formula is not randomized. The most common instance where this is required is when one of the treatment factors is time-order, which cannot be randomized. For example, suppose the main plot treatments in a split-plot experiment were lengths of time between two chemicals being mixed together, and that the analysis is of the amount of gas produced. If all the jars of chemicals needed to be mixed up at the beginning of the day, and the

analyses were performed after the appropriate time lapse, the standing times would have to be in the same order in each replicate. A suitable randomization is shown in Example 4.11.1b.

Example 4.11.1b

```
 2  UNITS [NVALUES=18]
 3  FACTOR [LEVELS=3; VALUES=6(1,2,3)] Block
 4  & [LABELS=!T(A,B,C); VALUES=(1...3)6] Method
 5  & [LEVELS=2; LABELS=!T('2 hours','4 hours'); VALUES=3(1,2)3] Time
 6  & [LABELS=*] Mplot
 7  PRINT Block,Time,Method

     Block        Time      Method
         1     2 hours           A
         1     2 hours           B
         1     2 hours           C
         1     4 hours           A
         1     4 hours           B
         1     4 hours           C
         2     2 hours           A
         2     2 hours           B
         2     2 hours           C
         2     4 hours           A
         2     4 hours           B
         2     4 hours           C
         3     2 hours           A
         3     2 hours           B
         3     2 hours           C
         3     4 hours           A
         3     4 hours           B
         3     4 hours           C

 8  RANDOMIZE [BLOCKSTRUCTURE=Block/Mplot/Method; EXCLUDE=Mplot; \
 9     SEED=888667] Time,Method
10  PRINT Block,Time,Method

     Block        Time      Method
         1     2 hours           C
         1     2 hours           A
         1     2 hours           B
         1     4 hours           A
         1     4 hours           B
         1     4 hours           C
         2     2 hours           C
         2     2 hours           B
         2     2 hours           A
         2     4 hours           C
         2     4 hours           B
         2     4 hours           A
         3     2 hours           C
         3     2 hours           B
         3     2 hours           A
         3     4 hours           B
         3     4 hours           A
         3     4 hours           C
```

In this example we have also used a simplification of the terminology for the block structure: we have used a treatment factor, `Method`, to specify what is actually a term in the block formula. The strict specification of the structure should have a block factor that is synonymous with `Method`; but having to specify such duplicate structures can be wasteful, and may not conform to the way in which such experiments are described colloquially. In fact the `RANDOMIZE` statement in line 8 could be modified further to remove the `Mplot` factor:

```
RANDOMIZE [BLOCKSTRUCTURE=Block/Time/Method; EXCLUDE=Time;\
  SEED=888667] Method
```

The `SEED` option determines which randomization Genstat gives. If you use the same seed, you

will get the same random numbers, and hence the same randomization (provided the block formula and the block factors are the same as before). If you omit SEED Genstat picks a seed at random, and prints a message to tell you what it is in case you want to reproduce the randomization later.

4.11.2 The ARANDOMIZE procedure

ARANDOMIZE procedure

Randomizes and prints an experimental design (R.W. Payne).

Options

PRINT = *string token*	Allows the (randomized) design to be printed; (design); default *
BLOCKSTRUCTURE = *formula*	Defines the block factors according to which the randomization is to be carried out; default takes the existing specification as defined by the BLOCKSTRUCTURE directive
EXCLUDE = *factors*	(Block) factors whose levels are not to be randomized
SEED = *scalar*	Seed to generate the random numbers used to define the randomization; default 0
LPERMUTE = *string token*	Whether to randomly permute treatment factor levels (no, yes); default no

Parameters

OLDVECTOR = *factors* or *variates*	Vectors whose values are to be randomized; default is to use the factors occurring in the formula (if any) specified by the most recent TREATMENTSTRUCTURE directive
NEWVECTOR = *factors* or *variates*	Vectors to store the randomized values; by default these overwrite the values in the original vectors

ARANDOMIZE provides a convenient way of randomizing the treatment allocations in an experimental design. It has several advantages over the RANDOMIZE directive (which is used inside the procedure).

First of all, the BLOCKSTRUCTURE option, which (as in RANDOMIZE) specifies the block model formula to indicate how the randomization is to take place, will use any setting that has already been defined by the BLOCKSTRUCTURE directive as its default. Moreover, the formula need not index all the units of the design, as is required by RANDOMIZE; if necessary ARANDOMIZE will set up an extra factor _units_ simular to the factor *units* used by ANOVA.

ARANDOMIZE allows the original (unrandomized) values to be retained. There are two parameters: OLDVECTOR to specify the factors or variates to be randomized, and NEWVECTOR to allow new structures to be supplied to store the randomized values. If no NEWVECTOR is specified, the randomized values replace the original values of the corresponding OLDVECTOR. By default, NEWVECTOR is assumed to contain the list of factors in the model formula (if any) specified by the previous TREATMENTSTRUCTURE directive. RESTRICT can be used, as usual, to restrict the set of units to be randomized.

The levels of the treatment factors can be randomized by setting option LPERMUTE=yes; ARANDOMIZE then randomly permutes the numbering of the levels of each treatment factor on the units of the design. There is also a PRINT option which can be set to design to print the design. The other two options, EXCLUDE and SEED, are as in RANDOMIZE. EXCLUDE lists block factors whose levels are not to be permuted during the randomization; for example the period

factor might need to be excluded in the randomization of a trial to study carry over effects. SEED defines the seed used to generate the random numbers used for the randomization; the default of 0 ensures that a seed will be chosen at random if SEED is not set.

Example 4.11.2 shows the randomization of a Latin square generated as in Example 4.9.4. AGLATIN contains a BLOCKSTRUCTURE statement setting the block formula to Rows*Cols, and a TREATMENTSTRUCTURE statement setting a treatment formula of Treat, so there is no need to set the BLOCKSTRUCTURE and TREATMENTSTRUCTURE options of ARANDOMIZE. In the randomization, Rows and Cols are randomized separately, so the same randomization of Treat appears within each row and column – thus preserving the properties of the Latin square.

Example 4.11.2

```
   2  AGLATIN [PRINT=design; ANALYSE=no] 6; NSQUARES=1;\
   3    TREATMENTFACTOR=!p(Treat); ROWS=Rows; COLUMNS=Columns; SEED=-1

Treatments on each unit of the design
=====================================

 Columns   1   2   3   4   5   6
    Rows
       1   1   2   3   4   5   6
       2   2   3   1   5   6   4
       3   3   1   2   6   4   5
       4   4   5   6   1   2   3
       5   5   6   4   2   3   1
       6   6   4   5   3   1   2

Treatment factor: Treat.

   4  ARANDOMIZE [SEED=876413]
   5  PDESIGN

Treatments on each unit of the design
=====================================

 Columns   1   2   3   4   5   6
    Rows
       1   5   6   4   3   1   2
       2   3   1   2   4   5   6
       3   1   2   3   5   6   4
       4   2   3   1   6   4   5
       5   4   5   6   2   3   1
       6   6   4   5   1   2   3

Treatment factor: Treat.
```

4.12 Sample size and power calculations

Genstat has procedures for determining the sample size (i.e. replication) required for experiments that are to be analysed by t-tests (4.12.1), analysis of variance (4.12.2) or non-parametric tests (4.12.5-8), or by using either product moment correlations (4.12.9) or Lin's concordance correlation coefficient (4.12.10). You can also calculate the power (or probability of detection) for terms in analysis of variance (4.12.3). Power calculations for regression analyses are described in 3.1.8.

4.12.1 Sample size for t-tests

STTEST procedure

Calculates the sample size for t-tests, including equivalence tests (R.W. Payne).

Options

PRINT = *string token*	What to print (replication, power); default repl, powe
NSAMPLES = *scalar*	Number of samples for the t-test (1 or 2); default 2
PROBABILITY = *scalar*	Significance level at which the response is to be tested; default 0.05
POWER = *scalar*	The required power (i.e. probability of detection) of the test; default 0.9
TMETHOD = *string token*	Type of test to be done (onesided, twosided, equivalance, noninferiority); default ones
RATIOREPLICATION = *scalar*	Ratio of replication sample2:sample1 (i.e. the size of sample 2 should be RATIOREPLICATION times the size of sample 1); default 1
REPLICATION = *variate*	Replication values for which to calculate and print or save the power; default * takes 11 replication values centred around the required number of replicates

Parameters

RESPONSE = *scalars*	Response to be detected
VAR1 = *scalars*	Anticipated variance of sample 1
VAR2 = *scalars*	Anticipated variance of sample 2; default * assumes the same variance as sample 1
NREPLICATES = *scalars*	Saves the required number of replicates
VREPLICATION = *variates*	Numbers of replicates for which powers have been calculated
VPOWER = *variates*	Power (i.e. probability of detection) for the various numbers of replicates

STTEST calculates the number of replicates (or *sample size*) required for various types of t-test. The calculations can be done for a one-sample t-test (testing for evidence that the mean of the sample differs from a specific value) or a two-sample test (testing that means of the samples are different). The number of samples is specified by the NSAMPLES option (default 2).

The size of response that should be detectable is supplied by the RESPONSE parameter. (This is difference between the sample mean of a one-sample test and the specific value, or the difference between the means of the two samples in a two-sample test.) The VAR1 parameter supplies the variance of the observations in the sample of a one-sample test or of the first sample of a two-sample test. If the second sample of a two-sample test has a different variance from the first sample, this can be supplied by the VAR2 parameter.

The significance level for the test is specified by the PROBABILITY option (default 0.05 i.e. 5%). The required probability for detection of the response (that is, the *power* of the test) is specified by the POWER option (default 0.9). It is generally assumed that the sizes of the samples in the two-sample test should be equal. However, you can set the RATIOREPLICATION option to a scalar, R say, to indicate that the size of the second sample should be R times the size of the first sample. The NREPLICATES parameter allows you to save the required size of the first sample.

The PRINT option controls printed output, with settings:

replication	to print the required number of replicates in each sample (i.e. the size of each sample);
power	to print a table giving the power (i.e. probability of detection) provided by a range of numbers of replicates.

By default both are printed.

The replications and corresponding powers can also be saved, in variates, using the VREPLICATION and VPOWER parameters. The REPLICATION option can specify the replication values for which to calculate and print or save the power; if this is not set, the default is to take 11 replication values centred around the required number of replicates.

By default, STTEST assumes a one-sided t-test is to be used, but you can set option TMETHOD=twosided to take a two-sided t-test instead. Other settings of TMETHOD enable you to test for equivalence or for non-inferiority. To demonstrate equivalence of the two samples (TMETHOD=equivalence), their means m_1 and m_2 must differ by less than some threshold d; this is specified by RESPONSE and should represent a limit below which the difference can be assumed to have no physical (or clinical) importance. Statistically, equivalence implies comparing a null hypothesis that the samples are not equivalent, i.e.

$$(m_1 - m_2) \leq -d$$

or

$$(m_1 - m_2) \geq d$$

with the alternative hypothesis that they are equivalent, i.e.

$$-d < (m_1 - m_2) < d$$

A one-sample test for equivalence operates similarly, but here d specifies the threshold for the sample mean itself. To demonstrate non-inferiority of sample 1 compared to sample 2, the null hypothesis becomes

$$(m_1 - m_2) \geq -d$$

(which, in fact, represents a simple one-sided t-test). For more details of the method, see Part 3 of the *Genstat Reference Manual*, or the description of ASAMPLESIZE (4.12.2).

The DSTTEST procedure can produce plots showing the probability distributions for the null and alternative hypotheses, to help you to understand the various types of test.

DSTTEST procedure

Plots power and significance for t-tests, including equivalence tests (R.W. Payne).

Options

NSAMPLES = *scalar*	Number of samples for the t-test (1 or 2); default 2
PROBABILITY = *scalar*	Significance level at which the response is to be tested; default 0.05
TMETHOD = *string token*	Type of test to be done (onesided, twosided, equivalence, noninferiority); default ones
RATIOREPLICATION = *scalar*	Ratio of replication sample2:sample1 (i.e. the size of sample 2 should be RATIOREPLICATION times the size of sample 1); default 1

Parameters

RESPONSE = *scalars*	Response to be detected
VAR1 = *scalars*	Anticipated variance of sample 1
VAR2 = *scalars*	Anticipated variance of sample 2; default * assumes the same variance as sample 1
NREPLICATES = *scalars*	Number of replicates

RDF = *scalars*	Number of residual degrees of freedom; default * calculates these automatically, assuming a standard t-test

In the plots the area of the distribution for the null hypothesis, where the null hypothesis would be rejected, is coloured in red. Its size corresponds to the significance level of the t-test, which is set by the PROBABILITY option (default 0.05), as in STTEST. The area of the distribution for the alternative hypothesis, where the null hypothesis would be rejected, is coloured in dark blue, unless it overlaps the red colour of the null hypothesis. The size of the dark blue area (including that overlapped by red) corresponds to the power of the test. The area of the distribution for the alternative hypothesis, where the null hypothesis would still be accepted, is coloured in light blue.

The TMETHOD and RATIOREPLICATION options also operate as in STTEST, as do the RESPONSE, VAR1, VAR2 and NREPLICATES parameters. However, DSTTEST has an additional parameter, RDF, which can be used to specify the number of degrees of freedom for the test. By default DSTTEST calculates these automatically assuming a standard t-test. RDF allows DSTTEST to be used, for example, to show t-tests of differences of means from an analysis of variance.

Example 4.12.1 illustrate s the various types of test. Figures 4.12.1a-f display the corresponding hypotheses.

Example 4.12.1

```
   2  "1) one-sample test, required response 2, anticipated variance 3."
   3  STTEST  [PRINT=replication,power; NSAMPLES=1] 2; VAR1=3;\
   4          NREPLICATES=nrep

Sample size for t-tests
=======================


Power for a one-sample t-test
-----------------------------

Response 2, variance 3, one-sided significance level 0.05.

No. replicates         Power
             3         0.384
             4         0.552
             5         0.684
             6         0.782
             7         0.852
             8         0.901
             9         0.934
            10         0.957
            11         0.972
            12         0.982
            13         0.988


Replication
-----------

To detect a response of 2, with sample variance 3, at a one-sided significance
level of 0.05, with a power of 0.9, using a one-sample t-test, requires a
replication of 8.

   5  DSTTEST [NSAMPLES=1] 2; VAR1=3; NREPLICATES=nrep
   6  "2) two-sample test, required response 2, anticipated variances 5."
   7  STTEST  [PRINT=replication,power] 2; VAR1=5; NREPLICATES=nrep

Sample size for t-tests
=======================
```

```
Power for a two-sample t-test
-----------------------------

Response 2, variance 5, one-sided significance level 0.05.

No. replicates         Power
            18         0.838
            19         0.855
            20         0.871
            21         0.886
            22         0.899
            23         0.910
            24         0.920
            25         0.930
            26         0.938
            27         0.945
            28         0.951


Replication
-----------

To detect a response of 2, with sample variance 5, at a one-sided significance
level of 0.05, with a power of 0.9, using a two-sample t-test, requires a
replication of 23 for each sample.

   8  DSTTEST 2; VAR1=5; NREPLICATES=nrep
   9  "3) two-sample test, required response 2, anticipated variances 5 & 6."
  10  STTEST  [PRINT=replication,power] 2; VAR1=5; VAR2=6; NREPLICATES=nrep

Sample size for t-tests
=======================


Power for a two-sample t-test
-----------------------------

Response 2, sample 1 variance 5, sample 2 variance 6, one-sided significance
level 0.05.

No. replicates         Power
            20         0.842
            21         0.858
            22         0.872
            23         0.885
            24         0.897
            25         0.908
            26         0.917
            27         0.926
            28         0.934
            29         0.941
            30         0.947


Replication
-----------

To detect a response of 2, with sample variances 5 and 6, at a one-sided
significance level of 0.05, with a power of 0.9, using a two-sample t-test,
requires a replication of 25 for each sample.

  11  DSTTEST 2; VAR1=5; VAR2=6; NREPLICATES=nrep
  12  "4) two-sample test, required response 2, anticipated variance 5,
 -13      sample sizes in a ratio 1:2."
  14  STTEST  [PRINT=replication,power; RATIOREPLICATION=2] 2; VAR1=5;\
  15          NREPLICATES=nrep

Sample size for t-tests
=======================


Power for a two-sample t-test
-----------------------------
```

```
Response 2, variance 5, one-sided significance level 0.05.

No. reps. sample 1  No. reps. sample 2         Power
                12                  24         0.798
                13                  26         0.826
                14                  28         0.851
                15                  30         0.873
                16                  32         0.891
                17                  34         0.907
                18                  36         0.921
                19                  38         0.933
                20                  40         0.943
                21                  42         0.952
                22                  44         0.959


Replication
-----------

To detect a response of 2, with sample variance 5, at a one-sided significance
level of 0.05, with a power of 0.9, using a two-sample t-test, requires a
replication of 17 for sample 1 and 34 for sample 2.

  16  DSTTEST [RATIOREPLICATION=2] 2; VAR1=5; NREPLICATES=nrep
  17  "5) demonstrating equivalence with threshold 5, anticipated variance 20,
 -18      significance level 0.05, power 0.95."
  19  STTEST  [PRINT=replication,power; POWER=0.95;\
  20          TMETHOD=equivalence] 5; VAR1=20; NREPLICATES=nrep

Sample size for t-tests
=======================


Power for a test of equivalence
-------------------------------

Threshold for non-equivalence 5, variance 20, significance level 0.05.

No. replicates         Power
            17         0.878
            18         0.899
            19         0.917
            20         0.932
            21         0.945
            22         0.955
            23         0.963
            24         0.970
            25         0.976
            26         0.980
            27         0.984


Replication
-----------

To demonstrate equivalence with a threshold of 5, sample variance 20, a
significance level of 0.05 and a power of 0.95, requires a replication of 22 for
each sample.

  21  DSTTEST [TMETHOD=equivalence] 5; VAR1=20; NREPLICATES=nrep
  22  "6) demonstrating non-inferiority with threshold 4,
 -23      anticipated variance 20, significance level 0.05, power 0.90."
  24  STTEST  [PRINT=replication,power; TMETHOD=noninferiority] 4; VAR1=20;\
  25          NREPLICATES=nrep

Sample size for t-tests
=======================


Power for a test of non-inferiority
-----------------------------------

Threshold for non-equivalence 4, variance 20, significance level 0.05.
```

```
No. replicates          Power
            18          0.838
            19          0.855
            20          0.871
            21          0.886
            22          0.899
            23          0.910
            24          0.920
            25          0.930
            26          0.938
            27          0.945
            28          0.951


Replication
-----------

To demonstrate non-inferiority with a threshold of 4, sample variance 20, a
significance level of 0.05 and a power of 0.9, requires a replication of 23 for
each sample.

  26  DSTTEST [TMETHOD=noninferiority] 4; VAR1=20; NREPLICATES=nrep
```

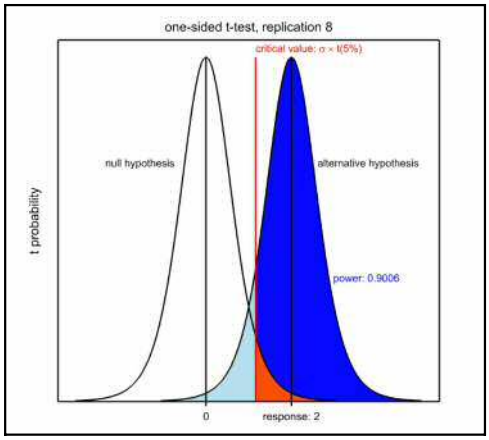


Figure 4.12.1a

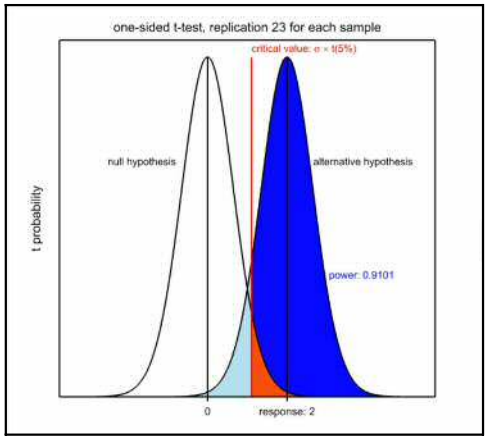


Figure 4.12.1b

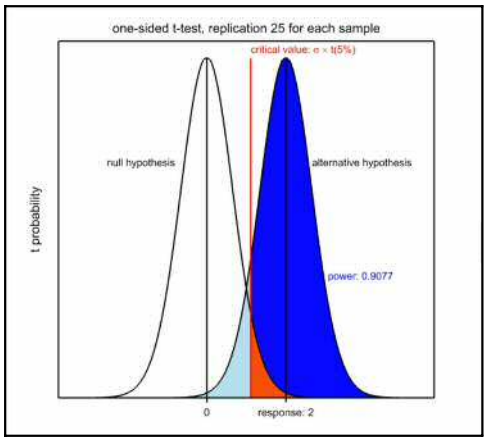


Figure 4.12.1c

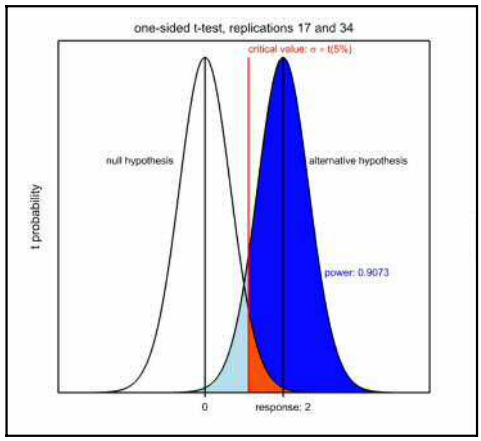


Figure 4.12.1d

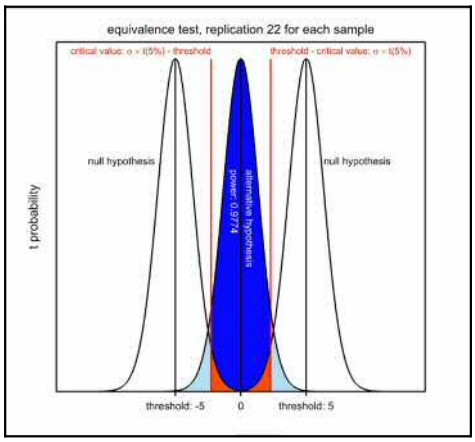


Figure 4.12.1e

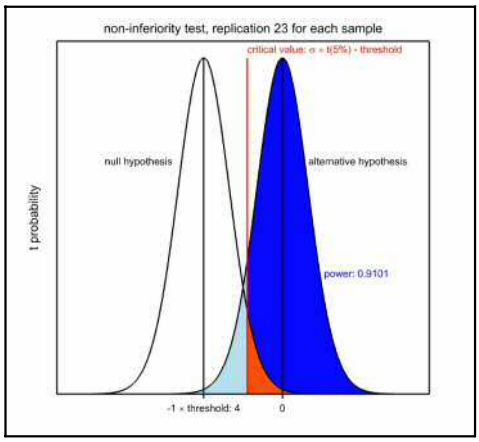


Figure 4.12.1f

4.12.2 Sample size for analysis of variance

`ASAMPLESIZE` procedure

Finds the replication to detect a treatment effect or contrast (R.W. Payne & P. Brain).

Options

`PRINT` = *string tokens*	Prints the replication or produces a printed summary of the power etc for the various amounts of replication (`power`, `replication`); default `powe`, `repl`
`TERM` = *formula*	Treatment term to be assessed in the analysis
`REPLICATES` = *factor*	Factor identifying the replication in the design
`MINREPLICATION` = *scalar*	Minimum number of replicates to try; default 2
`MAXREPLICATION` = *scalar*	Maximum feasible number of replicates; default `*` i.e. no limit
`TREATMENTSTRUCTURE` = *formula*	Treatment structure of the design; determined automatically from an `ANOVA` save structure if `TREATMENTSTRUCTURE` is unset or if `SAVE` is set
`BLOCKSTRUCTURE` = *formula*	Block structure of the design; determined automatically from an `ANOVA` save structure if `BLOCKSTRUCTURE` is unset or if `SAVE` is set
`COMPONENTS` = *variate* or *scalar*	Variate of variance components of all the terms in the block structure or, if `TERM` is estimated in the final stratum of the design, scalar containing only the variance component of the final stratum itself; determined automatically (if possible) from an `ANOVA` save structure if unset
`FACTORIAL` = *scalar*	Limit on the number of factors in treatment terms; default 3
`PROBABILITY` = *scalar*	Significance level at which the response is required to be detected (assuming a one-sided test); default 0.05
`POWER` = *scalar*	The required power (i.e. probability of detection) of the test; default 0.9
`TMETHOD` = *string token*	Type of test to be made (`onesided`, `twosided`, `equivalence`, `noninferiority`, `fratio`); default `ones`

XCONTRASTS = *variate*	X-variate defining a a contrast to be detected
CONTRASTTYPE = *string token*	Type of contrast (regression, comparison) default rege
SAVE = *asave*	ANOVA save structure to provide the information about the design

Parameters

RESPONSE = *scalars*	Size of the difference or contrast between TERM effects that is to be detected
NREPLICATES = *scalars*	Number of replicates required to detect RESPONSE

When designing an experiment, it is often possible to vary the replication of the treatments. For example, in a randomized block design you can adjust the number of blocks, or in a design with no blocking structure you can choose how many units to allocate to each of the treatments.

To decide how many replicates to include, you need to specify the size of difference between treatment effects that you would like the design to be able to detect. The treatment term of interest is specified using the TERM option of ASAMPLESIZE, and the difference that you want to detect between its effects is given by the RESPONSE parameter. As an alternative to detecting a difference between treatment effects, you can ask to detect a contrast, but here the treatment term must be a main effect (that is, TERM must involve just one factor). The XCONTRASTS option then specifies a variate containing the coefficients defining the contrast, and the CONTRASTTYPE option indicates whether this is a regression contrast (as specified by the REG function) or a comparison (as specified by COMPARISON).

The PROBABILITY option specifies the significance level that you will be using in the future analysis to detect the treatment difference (default 0.05, i.e. 5%). The POWER option specifies the probability with which you want the experiment to be able to detect the difference (that is, the *power* of the test); by default this is 0.9 i.e. 90%. In the language of hypothesis testing, PROBABILITY specifies the type I error rate, and POWER specifies one minus the type II error rate. By default, ASAMPLESIZE assumes a one-sided t-test is to be used, but you can set option TMETHOD=twosided to take a two-sided t-test instead. Alternatively, you can save the information explicitly in an ANOVA save structure, using the SAVE parameter of ANOVA, and then use this same save structure as the setting of the SAVE option of ASAMPLESIZE.

Other settings of TMETHOD enable you to test for equivalence or for non-inferiority. With equivalence (TMETHOD=equivalence), RESPONSE provides a threshold below which the treatments can be assumed to be equivalent. If the treatments have effects e_1 and e_2, the null hypothesis that the treatments are not equivalent is that either

$$(e_1 - e_2) \leq -\text{RESPONSE}$$

or

$$(e_1 - e_2) \geq \text{RESPONSE}$$

with the alternative hypothesis that they are equivalent, i.e.

$$-\text{RESPONSE} < (e_1 - e_2) < \text{RESPONSE}$$

With non-inferiority (TMETHOD=noninferiority), RESPONSE again specifies the threshold for the effect of one treatment to be superior to another. So, for example, to demonstrate non-inferiority of treatment 1 compared to treatment 2, the null hypothesis becomes

$$(e_1 - e_2) \geq -\text{RESPONSE}$$

(which, in fact, represents a simple one-sided t-test).

To determine the replication, ASAMPLESIZE needs to know the about the structure of the design, and the likely amount of variability. This is most easily obtained by taking the analysis of a design with similar units and the same block and treatment structures as those that are to be used in the new design. To do this, you should analyse the earlier set of data with the ANOVA directive in the usual way. First define the strata (or error terms) for the design using the

BLOCKSTRUCTURE directive, and the treatment model to be fitted using the TREATMENTSTRUCTURE directive. Then analyse the y-variate using the ANOVA. Provided you do not give any other ANOVA commands in the interim, ASAMPLESIZE will pick up the information automatically from the save information held within Genstat about that analysis. Alternatively, you can save the information explicitly in an ANOVA save structure, using the SAVE parameter of ANOVA, and then use this same save structure as the setting of the SAVE option of ASAMPLESIZE.

If you do not have a suitable earlier set of data, you should set up the design factors to contain the values required to define the units of the design for any convenient number of replicates. (It does not matter how many replicates you choose, as the form of the design should be the same in every replicate.) Then use the TREATMENTSTRUCTURE and BLOCKSTRUCTURE options of ASAMPLESIZE to define the treatment model and the block model, and the COMPONENTS option to specify the variance components of the strata. Note: if TERM is estimated in the bottom (or final) stratum of the design, COMPONENTS can be set to a scalar to specify only the variance component of this stratum – which is then equal to its residual mean square.

There is also the compromise possibility that you can take the information about the design and the block and treatment model from an ANOVA save structure (generated for example by the analysis of an artificial data set), but use the COMPONENTS option to specify different variance components from those in the analysis in the save structure.

The treatment terms to be included are controlled by the FACTORIAL option. This sets a limit (by default 3) on the number of factors in a treatment term. Treatment terms containing more than that number are deleted.

Finally, you must set the REPLICATES option to the factor in the block formula whose number of levels is to be increased or decreased to change the replication of the treatments. You can set the MINREPLICATION option to indicate the minimum number of replicates to try; by default this is 2. You can use the MAXREPLICATION option to define a maximum feasible number of replicates; by default this is no limit. The number of replicates that is required can be saved using the NREPLICATES parameter.

The PRINT option controls the printed output, with settings:

power	prints a table summarising the situation for a range of numbers of replicates (defined by MINREPLICATION and MAXREPLICATION if set, otherwise set automatically to a range covering the required number of replicates) – the table contains the residual degrees of freedom, the residual mean square, the standard error of difference (sed), RESPONSE divided by the sed, the t-value for a difference of RESPONSE, and the detection probability (i.e. power) at the level defined by the PROBABILITY option;
replication	prints the required replication.

By default both are printed.

Example 4.12.2a determines the number of blocks required to detect a treatment difference of 3 in a randomized block design with an anticipated residual mean square of 2.5 in the final stratum Block.Plot (i.e. within blocks); there is a single treatment factor Treat with 3 levels. We first use AGHIERARCHICAL to define the design for one replicate (or block), and then call ASAMPLESIZE to discover how many blocks are actually needed.

Example 4.12.2a

```
2  AGHIERARCHICAL [PRINT=*; ANALYSE=no; SEED=-1] Block,Plot;\
3                 TREATMENTFACTORS=*,Treat; LEVELS=1,3
4  ASAMPLESIZE    [PRINT=power,rep; TERM=Treat; REPLICATES=Block;\
5                 TREATMENTSTRUCTURE=Treat;\
6                 BLOCKSTRUCTURE=Block/Plot; COMPONENT=2.5]\
```

```
   7                1; NREPLICATES=Nrep

Sample size for analysis of variance
====================================

Power
-----

 Number of  Residual   Residual    s.e.d.  RESPONSE   t-value     Power
replicates      d.f.       m.s.           / s.e.d.
        39        76      2.500    0.3581     2.793     1.665     0.869
        40        78      2.500    0.3536     2.828     1.665     0.877
        41        80      2.500    0.3492     2.864     1.664     0.884
        42        82      2.500    0.3450     2.898     1.664     0.891
        43        84      2.500    0.3410     2.933     1.663     0.897
        44        86      2.500    0.3371     2.966     1.663     0.903
        45        88      2.500    0.3333     3.000     1.662     0.909
        46        90      2.500    0.3297     3.033     1.662     0.914
        47        92      2.500    0.3262     3.066     1.662     0.919
        48        94      2.500    0.3227     3.098     1.661     0.924
        49        96      2.500    0.3194     3.130     1.661     0.928


Replication
-----------

To detect a treatment difference of 1.000, at a significance level of 0.050,
with a power of 0.900, using a one-sided test, requires a replication of 44.
```

In Example 4.12.2b, we have a split-plot design, with block structure `Rep/Wplot/Subplot`. The factor `Variety` with 3 levels is applied to whole plots (and is thus estimated in the `Rep.Wplot` stratum) and the factor `Nitrogen` with 4 levels is applied to the sub-plots (and is thus estimated in the `Rep.Wplot.Subplot` stratum). The variance components for `Rep`, `Rep.Wplot` and `Rep.Wplot.Subplot` are anticipated to be 6, 3 and 5 respectively, and we wish to detect varietal differences of 3. Again we first define a split-plot with a single replicate, and then use `ASAMPLESIZE` to find out how many reps we need.

Example 4.12.2b

```
   2  AGHIERARCHICAL [PRINT=*; ANALYSE=no; SEED=-1] Rep,Wplot,Subplot;\
   3                 TREATMENTFACTORS=*,Variety,Nitrogen; LEVELS=1,3,4
   4  ASAMPLESIZE    [PRINT=power,rep; TERM=Variety; REPLICATES=Rep;\
   5                 TREATMENTSTRUCTURE=Variety*Nitrogen;\
   6                 BLOCKSTRUCTURE=Rep/Wplot/Subplot;\
   7                 COMPONENTS=!(6,3,5)] 3; NREPLICATES=Nrep

Sample size for analysis of variance
====================================

Power
-----

 Number of  Residual   Residual    s.e.d.  RESPONSE   t-value     Power
replicates      d.f.       m.s.           / s.e.d.
         4         6     17.00     1.458     2.058     1.943     0.570
         5         8     17.00     1.304     2.301     1.860     0.676
         6        10     17.00     1.190     2.521     1.812     0.758
         7        12     17.00     1.102     2.722     1.782     0.821
         8        14     17.00     1.031     2.910     1.761     0.869
         9        16     17.00     0.972     3.087     1.746     0.905
        10        18     17.00     0.922     3.254     1.734     0.931
        11        20     17.00     0.879     3.413     1.725     0.950
        12        22     17.00     0.842     3.565     1.717     0.965
        13        24     17.00     0.809     3.710     1.711     0.975
        14        26     17.00     0.779     3.850     1.706     0.982
```

```
Replication
-----------

To detect a treatment difference of 3.000, at a significance level of 0.050,
with a power of 0.900, using a one-sided test, requires a replication of 9.
```

ASAMPLESIZE calculates the standard error of difference between two treatment effects using the equation

$$\sqrt{(s^2 \times 2 / (r \times e))}$$

where s^2 is the stratum variance of the stratum where the treatment term is estimated, e is the efficiency factor, and r is the replication of each effect as in 4.1.3. For a regression contrast the standard error is

$$\sqrt{(s^2 \times 2 / (r \times sdiv \times e))}$$

where *sdiv* is the sum of squares of the XCONTRASTS variate, and for a comparison contrast the standard error is

$$\sqrt{(s^2 \times sdiv / (r \times e))}$$

(see 4.5). ASAMPLESIZE assumes that the treatment effects have equal replication, and also that all the effects (or residuals) of each block term have equal replication.

The stratum variance can be calculated as the variance component of the stratum S where the treatment term is estimated multiplied by the replication of its effects (residuals), plus the variance component of each stratum to which the stratum S is marginal, again multiplied by the replication of its effects (residuals). See for example Payne & Tobias (1992).

Comparing the null hypothesis that the treatments are not equivalent, i.e.

$$(m_1 - m_2) \leq -d$$

or

$$(m_1 - m_2) \geq d$$

with the alternative hypothesis that they are equivalent, i.e.

$$-d < (m_1 - m_2) < d$$

defines an *intersection-union* test, in which each component of the null hypothesis must be rejected separately. Here this implies performing two one-sided t-tests (this is known as a *TOST* procedure). If the significance level for the full test is to be α, each t-test must have significance level α (see Berger & Hsu 1996). To obtain a detection probability (or power) of $(1 - \beta)$, each of the t-tests must have detection probabilities of $(1 - \beta/2)$.

To demonstrate non-inferiority of treatment 1 compared to treatment 2, the null hypothesis is

$$(m_1 - m_2) \geq -d$$

This is equivalent to a one-sided t-test.

For the F-test, it is assumed that one effect will be $-0.5 \times$ RESPONSE, another will be $0.5 \times$ RESPONSE, and the others will be zero. This gives the smallest sum of squares for any table of effects with a maximum pair-wise difference of RESPONSE, which represents the most difficult case that needs to be detected.

4.12.3 Power for analysis of variance

APOWER procedure

Calculates the power (probability of detection) for terms in an analysis of variance (R.W. Payne).

Options

PRINT = *string token*	Prints the power (power); default powe
TERM = *formula*	Treatment term to be assessed in the analysis
TREATMENTSTRUCTURE = *formula*	Treatment structure of the design; determined automatically from an ANOVA save structure if

	TREATMENTSTRUCTURE is unset or if SAVE is set
BLOCKSTRUCTURE = *formula*	Block structure of the design; determined automatically from an ANOVA save structure if BLOCKSTRUCTURE is unset or if SAVE is set
FACTORIAL = *scalar*	Limit on the number of factors in treatment terms; default 3
PROBABILITY = *scalar*	Significance level at which the response is required to be detected (assuming a one-sided test); default 0.05
TMETHOD = *string token*	Type of test to be made (onesided, twosided, equivalence, noninferiority, fratio); default ones
XCONTRASTS = *variate*	X-variate defining a contrast to be detected
CONTRASTTYPE = *string token*	Type of contrast (regression, comparison) default rege
SAVE = *asave*	ANOVA save structure to provide the information about the design

Parameters

RESPONSE = *scalars*, *variates* or *tables*	Size of the difference or contrast between the effects of TERM that is to be detected, or (for TMETHOD=fratio) pattern of effects or means to be detected
RMS = *scalars*	Anticipated residual mean square corresponding to TERM; can be omitted if a SAVE structure is available
POWER = *scalars* or *variates*	Power (i.e. probability of detection) for RESPONSE

When assessing an experimental design, it can be useful to know how likely a treatment response of a specified size may be detected. This probability of detection, known as the *power* of the design with respect to the response of interest, helps to determine whether the experiment is sufficiently large or accurate to achieve its purpose.

The treatment term to test is specified using the TERM option of APOWER, and the difference that you want to detect between its effects is given by the RESPONSE parameter. As an alternative to detecting a difference between treatment effects, you can ask to detect a contrast. However, here the treatment term must be a main effect (that is, TERM must involve just one factor). The XCONTRASTS option then species a variate containing the coefficients defining the contrast, and the CONTRASTTYPE option indicates whether this is a regression contrast (as specified by the REG function) or a comparison (as specified by COMPARISON).

The PROBABILITY option specifies the significance level that you will be using in the analysis to detect the treatment difference or contrast; the default is 0.05, i.e. 5%. By default, APOWER assumes that a one-sided t-test is to be used, but you can set option TMETHOD=twosided to take a two-sided t-test instead.

Other settings of TMETHOD enable you to test for equivalence or for non-inferiority. With equivalence (TMETHOD=equivalence), RESPONSE defines a threshold below which the treatments can be assumed to be equivalent. If the treatments have effects e_1 and e_2, the null hypothesis that the treatments are not equivalent is that either

$$(e_1 - e_2) \leq -\text{RESPONSE}$$

or

$$(e_1 - e_2) \geq \text{RESPONSE}$$

with the alternative hypothesis that they are equivalent, i.e.

$$-\text{RESPONSE} < (e_1 - e_2) < \text{RESPONSE}$$

(see 4.1.2 for further details). With non-inferiority (TMETHOD=noninferiority), RESPONSE

again specifies the threshold for the effect of one treatment to be superior to another. So, for example, to demonstrate non-inferiority of treatment 1 compared to treatment 2, the null hypothesis becomes

$(e_1 - e_2) \geq -$RESPONSE

which represents a simple one-sided t-test.

You can also set TMETHOD=fratio, to assess the power of the F test in the analysis of variance table to detect a pattern of effects for TERM. You can specify the pattern by setting RESPONSE to a table containing the anticipated effects or means. Alternatively, you can set it to a y-variate containing, in each unit, the value of the effect or mean for the treatment (or treatment combination) to be applied to that unit of the design.

To determine the power, you need to define the design and specify the anticipated residual mean square for the stratum where the treatment term is estimated. This is most easily obtained by taking the analysis of a design with similar units and the same block and treatment structures as those that are to be used in the new design. To do this, you should analyse the earlier set of data with the ANOVA directive in the usual way. First define the strata (or error terms) for the design using the BLOCKSTRUCTURE directive, and the treatment model to be fitted using the TREATMENTSTRUCTURE directive. Then analyse the y-variate using the ANOVA directive. Provided you do not give any other ANOVA commands in the interim, APOWER will pick up the information automatically from the save information held within Genstat about the most recent ANOVA analysis. Alternatively, you can save the information explicitly in an ANOVA save structure, using the SAVE parameter of ANOVA, and then use this same save structure as the setting of the SAVE option of APOWER.

If you do not have a suitable earlier set of data, you should set up the design factors to contain the values required to define the units of the design. Then use the BLOCKSTRUCTURE and TREATMENTSTRUCTURE options of APOWER to define the strata and the treatment model, and the RMS option to specify the anticipated residual mean square for the stratum where TERM is estimated. There is also the compromise possibility that you can take the information about the design, the strata and treatment model from an ANOVA save structure (generated for example by the analysis of an artificial data set), but use the RMS parameter to specify a different residual mean square from the one in the analysis in the save structure. The treatment terms to be included are controlled by the FACTORIAL option; this sets a limit (by default 3) on the number of factors in a treatment term: terms containing more than that number are deleted.

The POWER parameter can save the power. This is printed by default, but you can set option PRINT=* to stop this.

Example 4.12.3 takes the split-plot design analysed in Example 4.2.1. The statement in line 67 determines the power of the design to detect a difference between two varieties of 20, assuming that the corresponding mean square (for the Blocks.Wplots stratum) will be 600 (and the defaults of a one-sided test with significance level of 0.05 i.e. 5%). Line 70 determines the power of the design to detect a difference between two nitrogen levels of 15, assuming that the corresponding mean square (for the Blocks.Wplots.Subplots stratum) will be 200. Then lines 71 and 72-3 determine the power to detect a linear regression contrast of Nitrogen (defined by the variate Nitlev: see 4.5), of 25.

Example 4.12.3

```
  70  APOWER   [PRINT=power; TERM=Variety] 20; RMS=600

Power of analysis of variance
=============================

For testing a treatment difference at a significance level of 0.050 using a
one-sided test.
```

```
      Response         Power
         20.00         0.838

  71  APOWER   [PRINT=power; TERM=Nitrogen] 15; RMS=200

Power of analysis of variance
=============================

For testing a treatment difference at a significance level of 0.050 using a
one-sided test.

      Response         Power
         15.00         0.932

  72  APOWER   [PRINT=power; TERM=Nitrogen; XCONTRASTS=Nitlev;\
  73           CONTRASTTYPE=regression] 25; RMS=200

Power of analysis of variance
=============================

For testing a regression contrast at a significance level of 0.050 using a
one-sided test.

      Response         Power
         25.00         0.951
```

4.12.4 Sizes of effects and contrasts detectable in an analysis of variance

ADETECTION procedure

Calculates the minimum size of effect or contrast detectable in an analysis of variance (R.W. Payne).

Options

PRINT = *string token*	Prints the minimum size of response that can be detected (detected); default dete
TERM = *formula*	Treatment term to be assessed in the analysis
TREATMENTSTRUCTURE = *formula*	Treatment structure of the design; determined automatically from an ANOVA save structure if TREATMENTSTRUCTURE is unset or if SAVE is set
BLOCKSTRUCTURE = *formula*	Block structure of the design; determined automatically from an ANOVA save structure if BLOCKSTRUCTURE is unset or if SAVE is set
FACTORIAL = *scalar*	Limit on the number of factors in treatment terms; default 3
PROBABILITY = *scalar*	Significance level at which the response is required to be detected (assuming a one-sided test); default 0.05
TMETHOD = *string token*	Type of test to be made (onesided, twosided, equivalence, noninferiority); default ones
XCONTRASTS = *variate*	X-variate defining a contrast to be detected
CONTRASTTYPE = *string token*	Type of contrast (regression, comparison); default rege
TOLERANCE = *scalar*	Tolerance for the iterations to calculate the detectable response
SAVE = *ANOVA save structure*	Save structure to provide the information about the design

Parameters

POWER = *scalars* or *variates*	Specifies the power i.e. probability with which the

	response should be detected
RMS = *scalars*	Anticipated residual mean square corresponding to TERM; can be omitted if a SAVE structure is available
DETECTED = *scalars* or *variates*	Minimum size of difference or contrast between the effects of TERM that is to be detected

ADETECTION finds the minimum size of effect or contrast that is detectable with a specified power (or probability) in an analysis of variance. The treatment term to test is specified using the TERM option of ADETECTION, and the power with which you want to detect it is given by the POWER parameter. You can save the size of response using the DETECTED parameter. This is printed by default, but you can set option PRINT=* to stop this.

As an alternative to detecting a difference between treatment effects, you can ask to detect a contrast. However, here the treatment term must be a main effect (that is, TERM must involve just one factor). The XCONTRASTS option then species a variate containing the coefficients defining the contrast, and the CONTRASTTYPE option indicates whether this is a regression contrast (as specified by the REG function) or a comparison (as specified by COMPARISON).

The PROBABILITY option specifies the significance level that you will be using in the analysis to detect the treatment difference or contrast; the default is 0.05, i.e. 5%. By default, ADETECTION assumes that a one-sided t-test is to be used, but you can set option TMETHOD=twosided to take a two-sided t-test instead.

As with ASAMPLESIZE (4.12.2) and APOWER (4.12.3) other settings of TMETHOD enable you to test for equivalence or for non-inferiority. With equivalence (TMETHOD=equivalence), RESPONSE defines a threshold below which the treatments can be assumed to be equivalent. If the treatments have effects e_1 and e_2, the null hypothesis that the treatments are not equivalent is that either

$$(e_1 - e_2) \leq -\text{RESPONSE}$$

or

$$(e_1 - e_2) \geq \text{RESPONSE}$$

with the alternative hypothesis that they are equivalent, i.e.

$$-\text{RESPONSE} < (e_1 - e_2) < \text{RESPONSE}$$

With non-inferiority (TMETHOD=noninferiority), RESPONSE again specifies the threshold for the effect of one treatment to be superior to another. So, for example, to demonstrate non-inferiority of treatment 1 compared to treatment 2, the null hypothesis becomes

$$(e_1 - e_2) \geq -\text{RESPONSE}$$

which represents a simple one-sided t-test. See 4.12.2 for more details.

ADETECTION needs to know the design, and the size of residual mean square anticipated for the stratum where the treatment term is estimated. This is provided most easily by supplying the analysis of a design with similar units and the same block and treatment structures as those that are to be used in the new design. To do this, you should analyse the earlier set of data with the ANOVA directive in the usual way. First define the strata (or error terms) for the design using the BLOCKSTRUCTURE directive, and the treatment model to be fitted using the TREATMENTSTRUCTURE directive. Then analyse the y-variate using the ANOVA directive. Provided you do not give any other ANOVA commands in the interim, ADETECTION will pick up the information automatically from the save information held within Genstat about the most recent ANOVA analysis. Alternatively, you can save the information explicitly in an ANOVA save structure, using the SAVE parameter of ANOVA, and then use this same save structure as the setting of the SAVE option of ADETECTION.

If you do not have a suitable earlier set of data, you should set up the design factors to contain the values required to define the units of the design. Then use the BLOCKSTRUCTURE and TREATMENTSTRUCTURE options of ADETECTION to define the strata and the treatment model, and the RMS option to specify the anticipated residual mean square for the stratum where TERM

is estimated. There is also the compromise possibility that you can take the information about the design, the strata and treatment model from an ANOVA save structure (generated for example by the analysis of an artificial data set), but use the RMS parameter to specify a different residual mean square from the one in the analysis in the save structure. The treatment terms to be included are controlled by the FACTORIAL option; this sets a limit (by default 3) on the number of factors in a treatment term: terms containing more than that number are deleted.

The procedure involves an iterative search to find the response that gives the specified power. The TOLERANCE option sets the convergence criterion (on the probability scale); the default is 10^{-7}.

Example 4.12.4 takes the same situations as in Example 4.12.3. The statement in line 74 determines the minimum size of variety effect that is detectable with power 0.9, assuming that the corresponding mean square (for the Blocks.Wplots stratum) will be 600 (and taking the default options of a one-sided test with significance level of 0.05 i.e. 5%). Line 75 determines the minimum size of nitrogen effect that is detectable with power 0.9, assuming that the corresponding mean square (for the Blocks.Wplots.Subplots stratum) will be 200. Then lines 75 and 76-7 determine minimum detectable regression contrast of Nitrogen (defined by the variate Nitlev: see 4.5).

Example 4.12.4

```
  74  ADETECTION [TERM=Variety] 0.9; RMS=600

Response detected by analysis of variance
=========================================

For testing a treatment difference at a significance level of 0.050 using a
one-sided test.

         Power      Response
         0.900         22.27

  75  ADETECTION [TERM=Nitrogen] 0.9; RMS=200

Response detected by analysis of variance
=========================================

For testing a treatment difference at a significance level of 0.050 using a
one-sided test.

         Power      Response
         0.900         14.01

  76  ADETECTION [TERM=Nitrogen; XCONTRASTS=Nitlev; CONTRASTTYPE=regression]\
  77              0.9; RMS=200

Response detected by analysis of variance
=========================================

For testing a regression contrast at a significance level of 0.050 using a
one-sided test.

         Power      Response
         0.900         22.15
```

4.12.5 Sample size for binomial tests

SBNTEST procedure

Calculates the sample size for binomial tests (R.W. Payne & D.A. Murray).

Options

PRINT = *string token*	What to print (replication, power); default repl, powe
PRMETHOD = *string token*	Method to be used to calculate the probabilities for the binomial test (angular, normalapproximation, exact); default norm
PROBABILITY = *scalar*	Significance level for the test; default 0.05
POWER = *scalar*	The required power (i.e. probability of detection) of the test; default 0.9
TMETHOD = *string token*	Type of test to be done (onesided, twosided); default ones
NULL = *scalar*	Probability under the null hypothesis for the one-sample test; default 0.5
RATIOREPLICATION = *scalar*	Ratio of replication sample2:sample1 (i.e. the size of sample 2 should be RATIOREPLICATION times the size of sample 1); default 1
REPLICATION = *variate*	Replication values for which to calculate and print or save the power; default * takes 11 replication values centred around the required number of replicates

Parameters

P1 = *scalars*	Probability to detect in sample 1
P2 = *scalars*	Probability to detect in sample 2
NREPLICATES = *scalars*	Saves the required number of replicates
VREPLICATION = *variates*	Numbers of replicates for which powers have been calculated
VPOWER = *variates*	Power (i.e. probability of detection) for the various numbers of replicates

SBNTEST calculates the number of replicates (or sample size) required for a binomial test (2.3.4). A one-sample binomial test assesses the evidence that the probability of success within a sample differs from some specific value. The probability that needs to be detected is specified by the P1 parameter, and the value from which it needs to be distinguished (i.e. the value under the null hypothesis) is specified by the NULL option. If NULL is not set, the default is 0.5. Alternatively, a two-sample test assess the evidence that probabilities within two samples are different. The anticipated probability within the first sample is then specified by the P1 parameter, and the probability within the second sample (from which it must be distinguished) is specified by the P2 parameter.

The PRMETHOD option defines the type of binomial test that is to be done. The normalapproximation setting relates to a test based on the Normal approximation to the binomial distribution (see the BNTEST procedure), while the angular setting is for a test using an angular transformation of the probabilities. The final setting, exact, is available only for the one-sample test and assumes an exact test using the binomial distribution.

The significance level for the test is specified by the PROBABILITY option (default 0.05 i.e. 5%). The required probability for detection of the difference between the probabilities (that is, the *power* of the test) is specified by the POWER option (default 0.9). It is generally assumed that

the sizes of the samples in the two-sample test should be equal. However, you can set the RATIOREPLICATION option to a scalar, R say, to indicate that the size of the second sample should be R times the size of the first sample. By default, SBNTEST assumes a one-sided test is to be used, but you can set option TMETHOD=twosided to take a two-sided test instead. The NREPLICATES parameter allows you to save the required size of the first sample.

The PRINT option controls printed output, with settings:

replication	to print the required number of replicates in each sample (i.e. the size of each sample);
power	to print a table giving the power (i.e. probability of detection) provided by a range of numbers of replicates.

By default both are printed.

The replications and corresponding powers can also be saved, in variates, using the VREPLICATION and VPOWER parameters. The REPLICATION option can specify the replication values for which to calculate and print or save the power; if this is not set, the default is to take 11 replication values centred around the required number of replicates.

Example 4.12.5 first discovers that a sample size of 53 would be needed to detect a probability of 0.7 (compared to the default of 0.5, with a power of 0.9, using a significance level of 0.05). As this is a one-sample test, Genstat can make an exact calculation for the probabilities, using the binomial distribution. The second part of the example uses the Normal approximation to determing that a replication of 106 is needed in each of two samples to detect the difference between probabilities 0.400 and 0.600 (at the default one-sided significance level of 0.050 and a power of 0.900).

Example 4.12.5

```
   2  SBNTEST [PRINT=replication,power; PRMETHOD=exact] 0.7

Sample size for a binomial test
===============================

Power
-----

To detect 0.700 compared to probability 0.500 under null hypothesis,
significance level 0.050 (one-sided).

Sample size         Power
         48         0.836
         49         0.881
         50         0.859
         51         0.898
         52         0.880
         53         0.914
         54         0.897
         55         0.879
         56         0.913
         57         0.897
         58         0.926


(calculated using the binomial distribution)


Replication
-----------

To detect a probability of 0.700 compared to a null hypothesis value of 0.500,
at a one-sided significance level of 0.050 and a power of 0.900, requires a
replication of 53.

   3  SBNTEST [PRINT=replication,power] 0.4; P2=0.6
```

```
Sample size for a binomial test
===============================

Power
-----

First probability 0.400, second probability 0.600, significance level 0.050
(one-sided).

Sample size         Power
        101         0.889
        102         0.892
        103         0.895
        104         0.897
        105         0.900
        106         0.902
        107         0.904
        108         0.907
        109         0.909
        110         0.911
        111         0.913

(calculated using a Normal approximation)


Replication
-----------

To detect the difference between probabilities 0.400 and 0.600, at a one-sided
significance level of 0.050 and a power of 0.900, requires a replication of
106 for each sample.
```

4.12.6 Sample size for Poisson tests

SPNTEST procedure

Calculates the sample size for a Poisson test (R.W. Payne & D.A. Murray).

Options

PRINT = *string token*	What to print (replication, power); default repl, powe
PRMETHOD = *string token*	Method to be used to calculate the probabilities for the test (normalapproximation, exact); default norm
PROBABILITY = *scalar*	Significance level for the test; default 0.05
POWER = *scalar*	The required power (i.e. probability of detection) of the test; default 0.9
TMETHOD = *string token*	Type of test to be done (onesided, twosided); default ones
NULL = *scalar*	Mean under the null hypothesis for the one-sample test; must be set when MU2 is unset
RATIOREPLICATION = *scalar*	Ratio of replication sample2:sample1 (i.e. the size of sample 2 should be RATIOREPLICATION times the size of sample 1); default 1
REPLICATION = *variate*	Replication values for which to calculate and print or save the power; default * takes 11 replication values centred around the required number of replicates

Parameters

MU1 = *scalars*	Mean to detect in sample 1
MU2 = *scalars*	Mean to detect in sample 2
NREPLICATES = *scalars*	Saves the required number of replicates

VREPLICATION = *variates*	Numbers of replicates for which powers have been calculated
VPOWER = *variates*	Power (i.e. probability of detection) for the various numbers of replicates

SPNTEST calculates the number of replicates (or sample size) required for a Poisson test. In the one-sample Poisson test, the data consist of a set of counts that are assumed to have been generated by the same Poisson distribution, and the sample size is the number of counts that have been observed. The mean that needs to be detected is specified by the MU1 parameter, and the value from which it needs to be distinguished (i.e. the value under the null hypothesis) is specified by the NULL option.

Alternatively, a two-sample test assesses the evidence that the there is a difference between the means of the Poisson distributions that have generated two separate samples of counts. The anticipated mean for the first sample is then specified by the MU1 parameter, and the mean for the second sample is specified by the MU2 parameter.

The other options and parameters operate as in SBNTEST (4.12.5).

Example 4.12.6

```
   2  "1) one-sample test, to detect a mean of 3,
  -3      against the null hypothesis that the mean is 2."
   4  SPNTEST [NULL=3; PRMETHOD=exact] 2

Sample size for a Poisson test
==============================


Power
-----

To detect a mean of 2.00 compared to a mean of 3.00 under null hypothesis,
significance level 0.05 (one-sided).

Sample size         Power
         18         0.822
         19         0.854
         20         0.880
         21         0.875
         22         0.898
         23         0.916
         24         0.912
         25         0.928
         26         0.941
         27         0.938
         28         0.949


(calculated using the Poisson distribution)


Replication
-----------

To detect a mean of 2.00 compared to a null hypothesis value of 3.00, at a
one-sided significance level of 0.05 and a power of 0.90, requires a replication
of 23.

   5  "2) two-sample test, to distinguish samples with means of 5 and 10."
   6  SPNTEST [TMETHOD=twosided] MU1=5; MU2=10

Sample size for a Poisson test
==============================
```

```
Power
-----

First mean 5.00, second mean 10.00, significance level 0.05 (two-sided).

Sample size         Power
          2         0.447
          3         0.609
          4         0.733
          5         0.823
          6         0.885
          7         0.927
          8         0.955
          9         0.972
         10         0.983
         11         0.990
         12         0.994


(calculated using a Normal approximation)


Replication
-----------

To detect the difference between means 5.00 and 10.00, at a two-sided
significance level of 0.05 and a power of 0.90, requires a replication of 7 for
each sample.
```

4.12.7 Sample size for sign tests

SSIGNTEST procedure

Calculates the sample size for a sign test (R.W. Payne).

Options

PRINT = *string token*	What to print (replication, power); default repl, powe
PROBABILITY = *scalar*	Significance level at which the response is to be tested; default 0.05
POWER = *scalar*	The required power (i.e. probability of detection) of the test; default 0.9
TMETHOD = *string token*	Whether to a one- or two-sided test is to be made (onesided, twosided); default twos
REPLICATION = *variate*	Replication values for which to calculate and print or save the power; default * takes 11 replication values centred around the required number of replicates

Parameters

RESPONSE = *scalars*	Probability of response (i.e. the probability that an observation in one sample will be greater than the equivalent observation in the other sample) that should be detectable
NREPLICATES = *scalars*	Saves the required number of replicates
VREPLICATION = *variates*	Numbers of replicates for which powers have been calculated
VPOWER = *variates*	Power (i.e. probability of detection) for the various numbers of replicates

SSIGNTEST calculates the number of replicates (or sample size) required for a sign test (2.4.2). The probability of response (i.e. the probability that an observation in one sample will be greater than the equivalent observation in the other sample) that should be detectable is supplied by the RESPONSE parameter. The other options and parameters operate as in SBNTEST (4.12.5).

Example 4.12.7

```
   2  SSIGNTEST [PRINT=replication,power] 0.7

Sample size for a sign test
===========================

Power
-----

To detect a probability of response of 0.700, at a significance level of 0.050
(one-sided).

No. replicates         Power
            48         0.836
            49         0.881
            50         0.859
            51         0.898
            52         0.880
            53         0.914
            54         0.897
            55         0.879
            56         0.913
            57         0.897
            58         0.926


Replication
-----------

To detect a probability of response of 0.700 using a sign test with a
one-sided significance level of 0.050 and a power of 0.900 requires a replication of
53.
```

4.12.8 Sample-size for McNemar's test

SMCNEMAR procedure

Calculates sample sizes for McNemar's test (R.W. Payne).

Options

PRINT = *string token*	What to print (replication, power); default repl, powe
PRMETHOD = *string token*	Method to be used to calculate the power of the McNemar test (normalapproximation, exact); default exac
PROBABILITY = *scalar*	Significance level at which the test is to be made; default 0.05
POWER = *scalar*	The required power (i.e. probability of detection) of the test; default 0.9
TMETHOD = *string token*	Whether a one- or two-sided test is to be made (onesided, twosided); default twos
REPLICATION = *variate*	Sample sizes for which to calculate and print or save the power; default * takes 11 replication values centred around the required number of replicates

Parameters

CHANGEPROBABILITY = *scalars*	Probability of any sort of change
RATIOPROBABILITIES = *scalars*	Ratio of the two probabilities of change
NREPLICATES = *scalars*	Saves the required sample size
VREPLICATION = *variates*	Sample sizes for which powers have been calculated
VPOWER = *variates*	Power (i.e. probability of detection) for the various numbers of replicates

The McNemar test is useful for analysing studies where subjects are assessed before and after a treatment. The response on each occasion is assumed to be categorized by a factor with two levels, with level 1 usually representing a *negative* response, and level 2 a *positive* response. The test is based on a table giving the numbers of subjects giving each combination of responses over the two occasions. Suppose that the table contains the values A, B, C and D as below:

	Second occasion	
First occasion	negative	positive
positive	A	B
negative	C	D

The test statistic assesses the equality of A and D, which represent the changes from positive to negative, and negative to positive, respectively; see 2.9.3.

In its original form, the test leads to a chi-square test. However, this may be inaccurate when there are small numbers of subjects. Consequently procedure MCNEMAR also provides an exact probability (based on the binomial distribution). Similarly SMCNEMAR has an option, PRMETHOD, to select whether you want to calculate the power of the test by approximating the probabilities by a Normal distribution, or using the binomial distribution as in the exact calculation (settings normalapproximation and exact, respectively). The default is exact.

To calculate the sample size, SMCNEMAR needs to know the overall probability of change (i.e. the probability of a subject being amongst those in either A or D), and the ratio of the probabilities of the two types of change (A versus D). These are specified by parameters CHANGEPROBABILITY and RATIOPROBABILITIES, respectively. By default the calculations are done for a one-sided test (testing for evidence that the change is in a specific direction (e.g. negative to positive). However, you can set option TMETHOD=twosided for a two-sided test (testing for either type of change).

As in SBNTEST (4.12.5), the significance level for the test and the power of the test are specified by options PROBABILITY and POWER option. The options PRINT and REPLICATION, and the parameters NREPLICATES, VREPLICATION and VPOWER, also operate as described in 4.12.5.

Example 4.12.8 shows that 115 subjects are needed if the aim is to detect a ratio of probabilities of 2, assuming that there is an overall probability of change of 0.7 (and taking the default of a one-sided significance level of 0.05 and a power of 0.9).

Example 4.12.8

```
  2  SMCNEMAR CHANGEPROBABILITY=0.7; RATIOPROBABILITIES=2

Sample size for McNemar's test
==============================

(Exact calculation using the binomial distribution)

Power
-----
```

```
Ratio of probabilities of change to detect 2.000, assuming a probability of
any change of 0.700, with significance level 0.050 (one-sided).

Sample size         Power
        110         0.888
        111         0.891
        112         0.894
        113         0.896
        114         0.899
        115         0.901
        116         0.904
        117         0.906
        118         0.908
        119         0.910
        120         0.913

Replication
-----------

To detect a ratio of change probabilities of 2.000, assuming an overall
probability of change of 0.700, at a one-sided significance level of 0.050 and
a power of 0.900, requires a replication of 115.
```

4.12.9 Sample size for the Mann-Whitney test

SMANNWHITNEY procedure

Calculates the sample sizes for the Mann-Whitney test (R.W. Payne).

Options

PRINT = *string token*	What to print (replication, power); default repl, powe
PROBABILITY = *scalar*	Significance level at which the test is to be made; default 0.05
POWER = *scalar*	The required power (i.e. probability of detection) of the test; default 0.9
TMETHOD = *string token*	Whether to a one- or two-sided test is to be made (onesided, twosided); default twos
RATIOREPLICATION = *scalar*	Ratio of replication sample2:sample1 (i.e. the size of sample 2 should be RATIOREPLICATION times the size of sample 1); default 1
REPLICATION = *variate*	Sample sizes for which to calculate and print or save the power; default * takes 11 replication values centred around the required number of replicates

Parameters

NULLPROBABILITIES = *variates*	Probabilities under null hypothesis
ODDSRATIO = *scalars*	Odds ratio for test group vs. control
NREPLICATES = *scalars*	Saves the required sample size
VREPLICATION = *variates*	Sample sizes for which powers have been calculated
VPOWER = *variates*	Power (i.e. probability of detection) for the various numbers of replicates

The Mann-Whitney U test is a non-parametric test for differences in location between two samples (2.5.1). This procedure, SMANNWHITNEY, allows you to calculate the sample sizes required for the test, provided you can supply some information about the probability distributions from which the samples are likely to be generated. For simplicity, the data are assumed to be classified into ordered categories. These may be natural categories (such as "very

good", "good", "moderate" and "poor") or they may be formed by splitting a continuous scale intervals (e.g. "under 18", "18-25", "25-40", "40-60" and "over 60"). You then use the NULLPROBABILITIES parameter to specify a variate containing the probability value for each category. This indicates the probability distribution which you feel would generate the data of both samples under the null hypothesis. The accuracy of the subsequent calculations will depend on how many categories you take for a continuous variate. However, Whitehead (1993) suggests that there is little to gain in taking more than five.

To assess the power of the test, you next need to indicate how small a difference between the sample distributions the test should be able to detect. The assumption now is that there will be a control sample, with probability distribution as supplied, and a test sample for which the distribution is shifted by multiplying the odds (i.e. $p/(1-p)$) of the cumulative distribution by a constant amount. (This corresponds to the proportional-odds model of McCullagh 1980.) This constant is supplied by the ODDSRATIO parameter. An example, with odds-ratio 2, is show below.

Null hypothesis			Alternative hypothesis		
probability	cumulative probability	odds	probability	cumulative probability	odds
0.20	0.20	0.25	0.33	0.33	0.50
0.40	0.60	1.50	0.42	0.75	3.00
0.30	0.90	9.00	0.20	0.95	18.00
0.10	1.00	*	0.05	1.00	*

The cumulative probabilities are produced as part of the information generated by setting the PRINT option to power. So you can evaluate possible ratios to check that they generate plausible distributions.

By default the calculations are done for a one-sided test, but you can set option TMETHOD=twosided for a two-sided test instead. It is generally assumed that the sizes of the samples in the two-sample test should be equal. However, you can set the RATIOREPLICATION option to a scalar, R say, to indicate that the size of the second sample should be R times the size of the first sample.

As in SBNTEST (4.12.5), the significance level for the test and the power of the test are specified by options PROBABILITY and POWER option. The options PRINT, RAQTIOREPLICATION and REPLICATION, and the parameters NREPLICATES, VREPLICATION and VPOWER, also operate as described in 4.12.5

Example 4.12.9 considers the second example of Whitehead (1993). Note, however, that the results below differ slightly, as the Genstat implementation omits the approximation of taking $n/(n+1)$ to be equal to one.

Example 4.12.9

```
  2  VARIATE [VALUES=0.2,0.5,0.2,0.1] Controlprob
  3  SMANNWHITNEY [TMETHOD=twosided] Controlprob; ODDSRATIO=EXP(0.887)

Sample size for a two-sided Mann-Whitney test
=============================================

Power
-----

Sample size         Power
         90         0.885
         91         0.889
         92         0.892
```

```
         93         0.895
         94         0.899
         95         0.902
         96         0.905
         97         0.907
         98         0.910
         99         0.913
        100         0.916

Tested at a significance level of 0.050, assuming that the data are ordered
categories with probabilities
  Sample 1  Sample 2
     0.200     0.378
     0.700     0.850
     0.900     0.956
     1.000     1.000
(based on an odds-ratio of 2.428).

Replication
-----------

To detect an odds-ratio of 2.43 between the cumulative probabilities of the
samples, using a Mann-Whitney test with a two-sided significance level of
0.050 and a power of 0.900, requires a replication of 95 for each sample.
```

4.12.10 Sample size for correlations

SCORRELATION procedure

Calculates the sample size to detect specified correlations (R.W. Payne).

Options

PRINT = *string token*	What to print (replication, power); default repl, powe
PROBABILITY = *scalar*	Significance level at which the correlation or difference between correlations is to be tested; default 0.05
POWER = *scalar*	The required power (i.e. probability of detection) of the test; default 0.9
TMETHOD = *string token*	Whether to a one- or two-sided test is to be made (onesided, twosided); default ones
RATIOREPLICATION = *scalar*	Ratio of replication sample2:sample1 (i.e. the size of sample for group 2 should be RATIOREPLICATION times the size of sample for group 1); default 1
REPLICATION = *variate*	Replication values for which to calculate and print or save the power; default * takes 11 replication values centred around the required number of replicates

Parameters

COR1 = *scalars*	Anticipated correlation in group 1
COR2 = *scalars*	Anticipated correlation in group 2
NREPLICATES = *scalars*	Saves the required number of replicates
VREPLICATION = *variates*	Numbers of replicates for which powers have been calculated
VPOWER = *variates*	Power (i.e. probability of detection) for the various numbers of replicates

SCORRELATION can be used to determine sample sizes when you wish to assess the correlation between two variables within a single group of subjects, or when you wish to compare the

correlations between two groups of subjects. The correlation in this case is the product moment correlation coefficient, as calculated by the CORRELATION function (1:4.2.2), the FCORRELATION procedure (2.8.1) or the CORRELATE directive (7.1.1). (So the variables are assumed to have Normal distributions.)

If there is a single group of subjects the correlation is specified (in a scalar) by the COR1 parameter, and the assumption is that we wish to assess whether this is non-zero. With two groups the correlations are specified by the COR1 and COR2 parameters (again in scalars). As in SBNTEST (4.12.5), the significance level for the test and power are specified by options PROBABILITY and POWER option. The options PRINT, RATIOREPLICATION and REPLICATION, and the parameters NREPLICATES, VREPLICATION and VPOWER, also operate as described in 4.12.5.

Example 4.12.10 first shows that a replication of 30 is required to detect a correlation of 0.5 compared to a value of zero, with power 0.9, and one-sided significance level 0.05. It then shows that replication of 75 is required for each of two samples to detected correlations 0.2 and 0.6, with power 0.9, at one-sided significance level 0.05.

Example 4.12.10

```
   3  SCORRELATION [PRINT=replication,power] 0.2; COR2=0.6

Sample size for testing a correlation
=====================================

Power
-----

Correlation 0.500, significance level 0.050 (one-sided).

No. replicates         Power
            25         0.851
            26         0.865
            27         0.877
            28         0.889
            29         0.899
            30         0.909
            31         0.918
            32         0.926
            33         0.933
            34         0.940
            35         0.945


Replication
-----------

To detect the difference between correlation 0.500 and zero, at a
one-sided significance level of 0.050 and a power of 0.900, requires
a replication of 30.

   3  SCORRELATION [PRINT=replication,power] 0.2; COR2=0.6

Sample size for comparing correlations
======================================

Power
-----

First correlation 0.200, second correlation 0.600, significance level 0.050
(one-sided).

No. replicates         Power
            70         0.884
            71         0.888
            72         0.892
            73         0.896
            74         0.899
```

```
          75        0.903
          76        0.906
          77        0.910
          78        0.913
          79        0.916
          80        0.919


Replication
-----------

To detect the difference between correlations 0.200 and 0.600, at a
one-sided significance level of 0.050 and a power of 0.900, requires
a replication of 75 for each sample.
```

4.12.11 Sample size for Lin's concordance correlation coefficient

SLCONCORDANCE procedure

Calculates the sample size for Lin's concordance correlation coefficient (R.W. Payne).

Options

PRINT = *string token*	What to print (replication, power); default repl, powe
PROBABILITY = *scalar*	Significance level at which the non-reproducibility is to be tested; default 0.05
POWER = *scalar*	The required power (i.e. probability of detection) of the test; default 0.9
REPLICATION = *variate*	Replication values for which to calculate and print or save the power; default * takes 11 replication values centred around the required number of replicates

Parameters

CORRELATION = *scalars*	Correlation for two samples with the smallest amount of non-reproducibility required to be detected
CONCORDANCE = *scalars*	Value of Lin's concordance for two samples with the smallest amount of non-reproducibility required to be detected
MEANSHIFT = *scalars*	Value of the shift in means (divided by the harmonic mean of the standard deviations) for two samples with the smallest amount of non-reproducibility required to be detected
SDRATIO = *scalars*	Value of the ratio of the standard deviations for two samples with the smallest amount of non-reproducibility required to be detected
NREPLICATES = *scalars*	Saves the required number of replicates
VREPLICATION = *variates*	Numbers of replicates for which powers have been calculated
VPOWER = *variates*	Power (i.e. probability of detection) for the various numbers of replicates

Procedure SLCONCORDANCE determines sample sizes for assessments involving Lin's concordance correlation coefficient (2.8.7). The coefficient is defined by the equation

$$\rho_c = \rho \times C_b$$

The term ρ is the standard Pearson product-moment correlation coefficient, while C_b is a bias

correction factor which is calculated by

$$C_b = 2 / (v + 1/v + u^2$$
$$v = s_1 / s_2$$
$$u = (m_1 - m_2) / \sqrt{(s_1 \times s_2)}$$

where m_i and s_i ($i = 1,2$) are the mean and standard deviation of the i^{th} set of measurements. The quantity u represents the shift in the mean between the two sets of measurements divided by the harmonic mean of their standard deviations, while v is the ratio of the two standard deviations.

If the coefficient is given a Z-transformation, the result has an approximate Normal distribution, with a standard deviation that depends on ρ_c, ρ and u (see Lin 1989, 2000). So, to calculate the sample size, SLCONCORDANCE needs to know the values of these quantities for two sets of measurements displaying the smallest amount of non-reproducibility that is required to be detected. The correlation coefficient (ρ) is specified by the CORRELATION parameter, the concordance coefficient by the CONCORDANCE parameter, and u by the MEANSHIFT parameter. Alternatively, you can omit either CONCORDANCE or MEANSHIFT provided you specify the ratio of the standard deviations, v, using the SDRATIO parameter. (SLCONCORDANCE can then calculate the omitted quantity using the equations above.) As in SBNTEST (4.12.5), the significance level for the test and power are specified by options PROBABILITY and POWER option. The options PRINT, RATIOREPLICATION and REPLICATION, and the parameters NREPLICATES, VREPLICATION and VPOWER, also operate as described in 4.12.5.

Example 4.12.11 shows that a replication of 33 would be needed to detect samples with a correlation of 0.95 and a concordance of 0.9 with power 0.9 by a one-sided test with significance level 0.05.

Example 4.12.11

```
   2  SLCONCORDANCE CORRELATION=0.95; CONCORDANCE=0.9; MEANSHIFT=0.1

Sample size for Lin's concordance coefficient
=============================================

Power
-----

To detect samples with a correlation of 0.950 and a concordance of 0.900 by
a one-sided test with significance level 0.050.

No. replicates         Power
            28         0.854
            29         0.866
            30         0.876
            31         0.886
            32         0.895
            33         0.904
            34         0.912
            35         0.919
            36         0.926
            37         0.932
            38         0.937


Replication
-----------

To detect samples with a correlation of 0.950 and a concordance of 0.900 by
a one-sided test with significance level 0.050 and a power of 0.900 requires
a replication of 33.
```

4.13 Design tools

This section describes the specialist commands for constructing designs. The GENERATE directive (4.13.1) provides an easy way of generating blocking factors or any other factors whose values occur in a systematic order. It can also form the values of treatment factors, using the design-key method, and define values for the pseudo-factors required to specify some types of partially balanced experimental design. The AKEY procedure (4.13.2) provides an alternative interface to the facilities in GENERATE, customized to be more convenient for experimental designs. In particular, it combines the two main uses of GENERATE, allowing you to generate the block factors in systematic order, and then (in the same statement) to generate the treatment factors using a design key. It can also print the design. The AMERGE procedure can add additional plots to a design (4.13.3), the APRODUCT procedure can combining simple designs into more complicated arrangements (4.13.4), and the AFAUGMENTED procedure can add plots for control treatments to a basic design to form an augmented design (4.13.5). The FKEY directive (4.13.6) forms design keys for new multi-stratum experimental designs (allowing you to control the confounding and aliasing of treatments). You can then use the FPSEUDOFACTORS directive (4.13.7) to determine the patterns of confounding and aliasing from the design key, and extend the treatment model to incorporate the necessary pseudo-factors for the design to be analysed by ANOVA. Alternatively, the FBASICCONTRASTS can be used to split up a model term into all its basic contrasts (4.13.8). These will be the parts of the term that may have been aliased or allocated to different strata.

4.13.1 Generating factor values: the GENERATE directive

GENERATE directive

Generates factor values for designed experiments.

Options

TREATMENTS = *formula*	Model term for which pseudo-factors are to be generated; default *
REPLICATES = *formula*	Factors defining replicates of the design; default *
BLOCKS = *formula*	Block formula (for design-key generation) or term (for generation of pseudo-factors); default *
KEY = *matrix*	Key matrix (number of factors in the parameter list by number of factors in the BLOCKS formula) to generate the factors by the design key method; default *
BASEVECTOR = *variate*	Base vector for design key generation; default *

Parameter

factors	Factors whose values are to be generated

GENERATE is invaluable when you have a set of data that is to be read in a systematic order: for example, you may want to take all the observations within one group, then the same number of observations within the next group, and so on until an equal number of observations has been read for every group. You can then define values of the grouping factor or factors by GENERATE; so the only values that you need to read are the observed data. Designed experiments are the obvious instance where the data are structured in this way: for example, you might have all the data from the first block, then all those from the second block, and so on.

The best way to understand GENERATE is to look at some examples. The values of a set of factors that you have defined by GENERATE are said to be in *standard order*: that is their units are arranged so that the levels of the first factor occur in the same order as in its levels vector

then, within each level of the first factor, the levels of the second factor are arranged similarly, and so on. For example

```
FACTOR [NVALUES=24; LEVELS=2] A
& [LEVELS=!(4,1,2)] B
& [LEVELS=4] C
GENERATE A,B,C
```

gives A, B and C the values

```
A: 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2
B: 4 4 4 4 1 1 1 1 2 2 2 2 4 4 4 4 1 1 1 1 2 2 2 2
C: 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4
```

Placing a number or a scalar in the parameter list has the same effect as if a factor with that number of levels had been listed. Thus to generate values only for A and C, all that you require is

```
GENERATE A,3,C
```

To generate values for just B and C is even simpler since the cycling process is itself recycled until all the units have been covered. Omitting A therefore causes all combinations of a level of B with a level of C to be used twice, in the same pattern as displayed above; so you need specify only

```
GENERATE B,C
```

You get a warning if one of the cycles is incomplete, as would happen for example if B and C had 18 values instead of 24.

This first use of GENERATE, then, is particularly appropriate for generating the blocking factors in an experimental design as can be seen in line 7 of Example 4.2.1a, line 11 of Example 4.3, line 9 of Example 4.7.1a, and line 8 of Example 4.7.3c.

Another use, obtained by setting the BLOCKS, KEY and BASEVECTOR options, is to form values of treatment factors using the design-key method. This method, described by Patterson (1976) and Patterson & Bailey (1978), provides a very flexible way of specifying the allocation of treatments in an experimental design. The method assumes that the units are identified by a set of what are called "plot" factors. In Genstat terms, these will often be the same as the factors that occur in the block formula of the design (4.2), and they are specified by the BLOCKS option of GENERATE. The setting is a formula, but remember this can be just a list of factors if you do not wish to indicate their inter-relationships; if the setting is more than just a list, Genstat forms the set of plot factors by taking the factors from the block formula in the order in which they occur there. Of course, the factors need not be identical to those in the block formula. For example if one these factors has a non-prime number of levels, it may need to be specified instead as the combination of two or more (pseudo) factors: for example, in a block design with blocks of size eight, the plots might need to be indexed by three factors with two levels. The treatment factors to be generated are again specified by the parameter of GENERATE.

The KEY option specifies a matrix known as the *design key*, which indicates how the values of each treatment factor are to be calculated from the plot factors. The matrix has a row for each treatment factor and a column for each plot factor; below k_{ij} represents the element in row i and column j. (This is the transpose of the form used by Patterson 1976, but in Genstat it seems more convenient to specify the treatments by rows.) There is also an option called BASEVECTOR, which can specify a variate with an element b_i for each treatment factor to allow the levels of the factor to be shifted cyclically; if this is unset, Genstat assumes b_i=0.

The calculation assumes that the values of the plot factors are represented by the integers zero upwards (and GENERATE will perform this mapping automatically if necessary). The value $q[i]_u$ in unit u of treatment factor i is then given by

$$q[i]_u = b_i + k_{i1} \times p[1]_u + k_{i2} \times p[2]_u + \ldots + k_{in} \times p[n]_u \quad \text{modulo } t_i$$

where $p[1]_u \ldots p[n]_u$ are the values of the plot factors in unit u, and t_i is the number of levels of

treatment factor *i* (which should be a prime number). The calculated values are integers in the range 0, 1 ... $t_i - 1$, but GENERATE will again map these to the defined levels if necessary. Further details are given in Section 4.13.2, which describes the procedure AKEY. This procedure extends the GENERATE facilities by allowing the block factors to be generated automatically, and providing a convenient way of handling plot or treatment factors with non-prime numbers of levels. It also allows the design to be printed after the factors have been generated. The use of a design key in GENERATE is shown in Example 4.13.6a.

GENERATE can also be used to form the values of pseudo-factors in partially-balanced designs, as shown in line 9 of Example 4.7.3c:

```
GENERATE [TREATMENTS=Variety; REPLICATES=Rep;\
  BLOCKS=Block] A,B
```

The treatment term to which the pseudo-factors are to be linked is specified by the TREATMENTS option; here this is the main effect of variety. The factors that identify the replicates are specified by the REPLICATES option, and those that identify the blocks within each replicate are specified by the BLOCKS option. The settings of these two options are model formulae, but Genstat merely scans them to find which factors they contain; so you may again find it easiest simply to give the factors as a list. Here the replicates and blocks are identified by the single factors Rep and Block respectively. The parameter of GENERATE lists the pseudo-factors. These have as many levels as there are blocks within each replicate. The blocks in the first replicate are used to determine which combinations of the factors in the treatment term correspond to each level of the first pseudo-factor, those in the second replicate are used for the second pseudo-factor, and so on. Here the first pseudo-factor is A, and the five blocks of replicate 1 contain Variety levels 1-5, 6-10, 11-15, 16-20 and 21-25. Thus the plots with varieties 1 to 5 are allocated level 1 of A, and so on. If a treatment combination occurs in more than one block within the same replicate, the level of the corresponding pseudo-factor is not determined uniquely and Genstat will report an error.

4.13.2 Generating factor values using design keys: the AKEY directive

---

AKEY procedure

Generates values for treatment factors using the design key method (R.W. Payne).

Options

PRINT = *string token*	Allows the generated TREATMENTFACTOR values to be printed, tabulated by the BLOCKFACTORS (design); default * i.e. no printing
BLOCKFACTORS = *factors*	Defines the block factors for the design; default is to take those in the formula already specified by the BLOCKSTRUCTURE directive, in the order in which they occur there
KEY = *matrix*	Matrix (number of treatment factors × number of block factors) key for the design
BASEVECTOR = *variate*	Base vector (length = number of treatment factors) for the design; default is a variate of zeros
ROWPRIMES = *variate*	Prime numbers for the rows of the KEY matrix
COLPRIMES = *variate*	Prime numbers for the columns of the KEY matrix
ROWMAPPINGS = *variate*	Mappings from the rows of the KEY to the TREATMENTFACTORS
COLMAPPINGS = *variate*	Mappings from the columns of the KEY to the BLOCKFACTORS

Parameter	
TREATMENTFACTORS = *factors*	Defines the treatment factors for the design; default is to take those in the formula already specified by the TREATMENTSTRUCTURE directive, in the order in which they occur there

AKEY generates the values of the block factors, if necessary, in systematic order and then generates the treatment factors from the block factors using a design key.

The design key method, described by Patterson (1976) and Patterson & Bailey (1978), provides a very flexible way of specifying the allocation of treatments in an experimental design. Patterson & Bailey (1978) provide several examples of keys. These are used in the on-line examples of AKEY which can be accessed using procedure LIBEXAMPLE:

```
LIBEXAMPLE 'AKEY'; EXAMPLE=Keyex
```

Two of them are also used in the examples below.

The method assumes that the units are identified by a set of what are termed "plot" factors. Generally these will be the same factors that are used in the block formula. Thus, in the procedure, they are specified by an option called BLOCKFACTORS which will take the factors from the formula already set by the BLOCKSTRUCTURE directive (outside the procedure) as its default. However, if any of these factors has a non-prime number of levels, it will need to be defined as the combination of two or more (pseudo) factors, as shown in Example 4.13.2b. The method can also be used to generate pseudo-factors for use in the treatment formula; the "plot" factors may then be the treatment factors themselves (see Example 4.13.7). If these "plot" factors do not already have values, they will be generated in "standard order" using the GENERATE directive.

The factors whose values are to be generated are specified by the TREATMENTFACTORS parameter. Again this can be omitted, and AKEY will then take the factors from the existing setting of the TREATMENTSTRUCTURE directive, in the order in which they occur there.

If any of the factors is restricted, only the part of the design not excluded by the restriction will be generated.

The generated values of the factors can be printed by setting option PRINT=design. The other options define how the values are generated. The KEY option specifies a matrix known as the design key, which indicates how the values of each treatment factor are to be calculated from the plot factors. The matrix has a row for each treatment factor and a column for each plot factor; below k_{ij} represents the element in row i and column j. (This is the transpose of the form used by Patterson 1976, but in Genstat it seems more convenient to specify the treatments by rows.) There is also an option called BASEVECTOR, which can specify a variate with an element B_i for each treatment factor to allow the levels of the factor to be shifted cyclically; by default this is a variate of zeros.

The calculation assumes that the values of the plot factors are represented by the integers zero upwards (and AKEY will perform this mapping automatically if necessary). The value $q[i]_u$ in unit u of treatment factor i is then given by

$$q[i]_u = b_i + k_{i1} \times p[1]_u + k_{i2} \times p[2]_u + \ldots + k_{in} \times p[n]_u \quad \text{modulo } t_i$$

where $p[1]_u \ldots p[n]_u$ are the values of the plot factors in unit u, and t_i is the number of levels of treatment factor i. The calculated values are integers in the range $0, 1 \ldots t_i - 1$, but AKEY will again map these to the defined levels if necessary. However, all this takes place behind the scenes, within AKEY. The numbers of levels t_i must be prime numbers. They need not all be equal, but the key will usually be zero in any element where the row and column factors have different numbers of levels: that is, each treatment factor will usually be generated only from "plot" factors with the same number of levels as the treatment factor itself.

To illustrate the process, the treatments to be allocated (before randomization) to the plots of an $N \times N$ Latin Square may be calculated as

Latin-factor-value = Row-factor-value + Column-factor-value modulo N

The values of the extra factor in a Graeco-Latin square can then be formed as

Graeco-factor-value = Row-factor-value + 2 × Column-factor-value modulo N

The design key thus has rows (1,1) and (1,2); Example 4.13.2a uses this to this generate a 5 × 5 Graeco-Latin square.

Example 4.13.2a

```
  2  " 5x5 Graeco-Latin square."
  3  FACTOR [NVALUES=25; LEVELS=5] Row,Column,A,B; DECIMALS=0
  4  GENERATE Row,Column
  5  " Specify key matrix (row and column labelling is unnecessary
 -6    other than to indicate how the matrix is stored)."
  7  MATRIX [ROWS=!t(A,B); COLUMNS=!t(Row,Column); VALUES=1,1, 1,2] GLkey
  8  AKEY [PRINT=design; BLOCKFACTORS=Row,Column; KEY=GLkey] A,B

Treatment combinations on each unit of the design
=================================================

 Column   1     2     3     4     5
    Row
      1   1 1   2 3   3 5   4 2   5 4
      2   2 2   3 4   4 1   5 3   1 5
      3   3 3   4 5   5 2   1 4   2 1
      4   4 4   5 1   1 3   2 5   3 2
      5   5 5   1 2   2 4   3 1   4 3

Treatment factors are listed in the order: A, B.
```

If any of the block or treatment factors has a non-prime number of levels, it must be specified as the combination of two or more (pseudo) factors: for example, in a block design with blocks of size four, the plots will need to be specified by two (pseudo) factors with two levels. Thus the `COLPRIMES` option allows you to supply a variate listing the prime numbers for each column of the key, and the `COLMAPPINGS` option a variate to indicate the "plot" factor corresponding to each column. In Example 4.13.2b, we have four blocks of four plots. The `COLPRIME` option in line 10 specifies that the prime for each column is 2. The `COLMAP` option specifies that the first two columns correspond to the first "plot" factor (`Block` in the example), and columns 3 and 4 correspond to the second "plot" factor (`Plot` in the example). The default for `COLMAP` is a variate containing the integers 1 up to the number of "plot" factors, so it can be omitted if no pseudo-factors are required. `COLPRIME` can also be omitted, provided the "plot" factors have already been declared with their numbers of levels (and provided there are no "plot" pseudo-factors); see Example 4.13.2a. The `ROWPRIME` and `ROWMAP` options similarly allow you to specify pseudo-factors to generate the treatment factors.

Notice that we need not have specified the `BLOCKFACTORS` option of `AKEY` in line 9 of Example 4.13.2b, but could have let `AKEY` construct the setting from the block formula defined by the `BLOCKSTRUCTURE` statement in line 6. Likewise, the parameter setting `A,B,C,D` in line 10 could have been omitted, and deduced instead from the treatment formula defined in line 7.

Example 4.13.2b

```
 2  " Single-replicate design with 4 blocks of 4 plots
-3    and 4 treatment factors each with 2 levels."
 4  FACTOR [NVALUES=16; LEVELS=2] A,B,C,D
 5  & [LEVELS=4] Block,Plot
 6  BLOCKSTRUCTURE Block/Plot
 7  TREATMENTSTRUCTURE A*B*C*D
 8  MATRIX [ROWS=4; COLUMNS=4; VALUES=0,0,1,0, 0,0,0,1, 1,0,1,1, 0,1,1,1] Bkey
 9  AKEY [PRINT=design; BLOCKFACTORS=Block,Plot; KEY=Bkey;\
10    COLPRIME=!(4(2)); COLMAP=!(1,1,2,2)] A,B,C,D
```

```
Treatment combinations on each unit of the design
=================================================

  Plot   1         2         3         4
 Block
     1   1 1 1 1   1 2 2 2   2 1 2 2   2 2 1 1
     2   1 1 1 2   1 2 2 1   2 1 2 1   2 2 1 2
     3   1 1 2 1   1 2 1 2   2 1 1 2   2 2 2 1
     4   1 1 2 2   1 2 1 1   2 1 1 1   2 2 2 2

Treatment factors are listed in the order: A, B, C, D.

11  ANOVA [FACTORIAL=4]

Analysis of variance
====================

Source of variation     d.f.

Block stratum
C.D                        1
A.B.C                      1
A.B.D                      1

Block.Plot stratum
A                          1
B                          1
C                          1
D                          1
A.B                        1
A.C                        1
B.C                        1
A.D                        1
B.D                        1
A.C.D                      1
B.C.D                      1
A.B.C.D                    1

Total                     15
```

The design key thus provides a very convenient way of defining treatment factors. Essentially, the key identifies each factor i with the set of contrasts (in the usual terminology)

$$p[1]^{k_{i1}}\ p[2]^{k_{i2}}\ \ldots\ p[n]^{k_{in}}$$

and the skill when forming a design is in selecting the best set for each factor. The Genstat design system has a repertoire of keys, which are used by procedures DESIGN and AGDESIGN to generate a range of designs, including factorials, fractional factorials, Latin squares and Lattices (4.9.3). You can also construct new design keys using the directive FKEY, described in Section 4.13.6.

4.13.3 Adding extra units to a design: the AMERGE procedure

AMERGE procedure

Merges extra units into an experimental design (R.W. Payne).

Option

SORT = *string token*	Whether to sort the factors afterwards (no, yes); default no

Parameters

FACTOR = *factors*	Factors to which the new units are to be added

NEWUNITS = *factors*, *variates* or *scalars*
Extra units to be added to each factor

AMERGE provides a convenient way of adding extra units into an experimental design. In Example 4.13.3 we use AMERGE to incorporate an extra, control, treatment replicated twice to each block of a randomized block design generated by AGHIERARCHICAL (4.9.1). More complicated uses may join together two completely different designs, for example a randomized block design to a balanced incomplete block design.

The factors of the design which is to be augmented are specified using the first parameter (FACTOR), and the units that are to be added to each one are specified by the NEWUNITS parameter. The same number of units must be added to every FACTOR, and their levels (and labels) will be extended, if necessary, according to those defined on the units that are added. New units of a factor that are to receive different levels should be specified in a factor or a variate. Alternatively, if every new unit is to receive the same level of the FACTOR, NEWUNIT can be set to a scalar. Any restrictions on the vectors are ignored.

The SORT option allows the FACTOR values to be sorted after the new units have been added. Otherwise, they are simply placed at the end of the existing values.

Example 4.13.3

```
   2  AGHIERARCHICAL [PRINT=design; ANALYSE=no; SEED=-1] Blocks,Units;\
   3    TREATMENTFACTORS=*,!p(Type,Amount);\
   4    LEVELS=3,!p(2,3)

Treatment combinations on each unit of the design
=================================================

 Blocks   1     2     3
  Units
      1   1 1   1 1   1 1
      2   1 2   1 2   1 2
      3   1 3   1 3   1 3
      4   2 1   2 1   2 1
      5   2 2   2 2   2 2
      6   2 3   2 3   2 3

Treatment factors are listed in the order: Type, Amount.

   5  AMERGE [SORT=yes] Blocks,Units,Type,Amount;\
   6    NEWUNITS=!(2(1...3)),!((7,8)3),0,0
   7  PDESIGN [BLOCKSTRUCTURE=Blocks/Units; TREATMENTSTRUCTURE=Type*Amount]

Treatment combinations on each unit of the design
=================================================

  Units   1     2     3     4     5     6     7     8
 Blocks
      1   1 1   1 2   1 3   2 1   2 2   2 3   0 0   0 0
      2   1 1   1 2   1 3   2 1   2 2   2 3   0 0   0 0
      3   1 1   1 2   1 3   2 1   2 2   2 3   0 0   0 0

Treatment factors are listed in the order: Type, Amount.

   8  ARANDOMIZE [PRINT=design; BLOCKSTRUCTURE=Blocks/Units;\
   9    SEED=266203] Type,Amount

Treatment combinations on each unit of the design
=================================================

  Units   1     2     3     4     5     6     7     8
 Blocks
      1   2 1   1 3   1 2   2 2   0 0   0 0   2 3   1 1
      2   1 3   0 0   2 2   0 0   2 3   1 2   1 1   2 1
      3   2 3   0 0   2 2   2 1   1 1   0 0   1 3   1 2
```

```
Treatment factors are listed in the order: Type, Amount.

  10  FACTOR [LEVELS=2] Control
  11  CALCULATE Control = NEWLEVELS(Type; !(1,2,2))
  12  BLOCKSTRUCTURE Blocks/Units
  13  TREATMENTS Control/(Type*Amount)
  14  ANOVA

Analysis of variance
====================

Source of variation     d.f.

Blocks stratum             2

Blocks.Units stratum
Control                    1
Control.Type               1
Control.Amount             2
Control.Type.Amount        2
Residual                  15

Total                     23
```

For clarity, we first print the design with the units sorted. We then randomize the design, using the ARANDOMIZE procedure (4.11.2). Finally we define a factor Control to represent the comparison between the new control and the other treatments (see Section 4.3), and produce a dummy analysis of variance for the complete design.

4.13.4 Taking the product of two experimental designs: the APRODUCT procedure

APRODUCT procedure

Forms a new experimental design from the product of two designs (R.W. Payne).

Options

PRINT = *string token*	Controls printing of the design (design); default desi
ANALYSE = *string token*	Whether to analyse the design by ANOVA (yes, no); default no
METHOD = *string token*	How to combine the designs (cross, nest); default nest
BF1 = *formula*	Block formula for design 1
TF1 = *formula*	Treatment formula for design 1
BF2 = *formula*	Block formula for design 2
TF2 = *formula*	Treatment formula for design 2

No parameters

APRODUCT forms an experimental design by taking the product of two other designs. The METHOD option controls whether the product is formed by nesting the second design within the first, or by crossing the two designs together. Example 4.13.4 extends the Latin square formed in Example 4.11.2 to include an extra stratum of subplots nested within the plots of the square, with a two-level factor Subtreat applied to the (two) subplots within each plot. This is achieved by nesting an extra design, with single block factor Subplot and treatment factor Subtreat below the original design. The block structure for the new design is

```
(Row * Column) / Subplot
```

and the treatment structure is

```
Treat * Subtreat
```

Nesting is thus useful when you want to subdivide the units of a design and apply further treatments (in this case those defined by the factor `Subtreat`) to the resulting subplots.

Alternatively, suppose that the extra design has a single factor `Extra` in the block structure and a single treatment factor `Newtreat`. If we cross the two designs, the new design will have a block structure of `(Rows*Columns)*Extra`, that is `Rows*Columns*Extra`, in which we have duplicated the Latin square for every level of Extra. Crossing is useful if you need to introduce a new blocking structure into an existing design. For example, the factor `Extra` might represent different time periods or different locations in which a Latin square design is to be used, and the factor `Newtreat` the different systematic conditions that might apply on each occasion.

With both nesting and crossing, the new design will contain a unit for every combination of the block factors in the two original designs, and so every combination of the treatment factors in the first design will occur with every combination of the treatment factors in the second design. The treatment structure is thus defined for the new design by crossing the treatment structures of the two original designs, to estimate all the original treatment terms and their interactions.

`APRODUCT` redefines the values of the factors as required for the new design. None of the factors must be restricted, and any existing restrictions are cancelled. `APRODUCT` also executes `BLOCKSTRUCTURE` and `TREATMENTSTRUCTURE` directives with the new block and treatment formulae. These are thus available for subsequent commands, such as the `ARANDOMIZE` command used to randomize the allocation of `Subtreat` in line 9 of Example 4.13.4, and the `ANOVA` command used in line 10 to produce a dummy analysis-of-variance table. The new formulae can also be accessed, outside the procedure, using the `ASTATUS` procedure (4.9.1).

The `PRINT` option of `APRODUCT` can be set to `design` to print the new design, and the `ANALYSE` option can be set to `yes` to produce a skeleton analysis of variance from `ANOVA`. Options `BF1`, `TF1`, `BF2`, and `TF2` define the block structure and treatment structure of the first and then the second design.

Example 4.13.4

```
  6  FACTOR [LEVELS=2; VALUES=1,2] Subplot,Subtreat
  7  APRODUCT [PRINT=*; METHOD=nest; ANALYSE=no; \
  8    BF1=Rows*Columns; TF1=Treat; BF2=Subplot; TF2=Subtreat]
  9  ARANDOMIZE [PRINT=design; SEED=641732]

Treatment combinations on each unit of the design
=================================================

         Subplot   1     2
    Rows Columns
       1       1   3 2   3 1
               2   5 1   5 2
               3   1 2   1 1
               4   4 1   4 2
               5   2 2   2 1
               6   6 1   6 2
       2       1   6 1   6 2
               2   2 1   2 2
               3   4 1   4 2
               4   1 2   1 1
               5   5 2   5 1
               6   3 1   3 2
       3       1   5 2   5 1
               2   1 2   1 1
               3   6 2   6 1
               4   3 2   3 1
               5   4 2   4 1
               6   2 1   2 2
       4       1   1 1   1 2
```

```
              2   6 2   6 1
              3   2 1   2 2
              4   5 1   5 2
              5   3 2   3 1
              6   4 2   4 1
      5       1   4 2   4 1
              2   3 2   3 1
              3   5 1   5 2
              4   2 2   2 1
              5   6 1   6 2
              6   1 2   1 1
      6       1   2 2   2 1
              2   4 1   4 2
              3   3 2   3 1
              4   6 1   6 2
              5   1 2   1 1
              6   5 2   5 1

Treatment factors are listed in the order: Treat. Subtreat.

  10  ANOVA

Analysis of variance
====================

Source of variation     d.f.

Rows stratum               5

Columns stratum            5

Rows.Columns stratum
Treat                      5
Residual                  20

Rows.Columns.Subplot stratum
Subtreat                   1
Treat.Subtreat             5
Residual                  30

Total                     71
```

4.13.5 Augmented designs: the AFAUGMENTED procedure

AFAUGMENTED procedure

Forms an augmented design (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (design); default * i.e. none
TREATMENTSTRUCTURE = *formula*	Treatment terms, other than GENOTYPES, to be included in the analysis
BLOCKSTRUCTURE = *formula*	Defines the block structure of the basic design
COVARIATE = *variates*	Specifies any covariates to be included in the analysis
LEVTEST = *variate*	Test genotypes to add to the design
LEVCONTROL = *scalar* or *variate*	Specifies the control genotype(s) if these are not already in the GENOTYPES factor
GENOTYPES = *factor*	Genotype factor
CONTROLS = *factor*	Factor identifying the controls
TESTVSCONTROL = *factor*	Factor representing the comparison between test and control genotypes
SUBPLOTS = *factor*	Factor to represent the subplots to be created for the test genotypes in the basic design

NSUBPLOTS = *scalar*	Number of subplots to create within each plot of the basic design
SUBCONTROLS = *scalar* or *variate*	Subplots to be used for control genotypes, if not already pre-allocated in the GENOTYPES and SUBPLOTS factors; default selects subplots for the controls at random within each whole plot
NREPTEST = *scalar* or *variate*	Number of times to replicate the test genotypes; default 1
SEED = *scalar*	Seed for the random numbers used to randomize the allocation of the genotypes (a negative value implies no randomization); default 0

No parameters

An augmented design is a design for assessing large numbers of treatments, usually test genotypes in a variety trial. The trial also contains controls; these are replicated while the tests are usually unreplicated.

The design is constructed from a basic design, which can be any standard design, for example, a randomized complete block design or a Latin square. In the simplest situation, a control genotype is allocated to each plot of the basic design. The design is then expanded, or *augmented*, so that each plot of the basic design is split into subplots. (So the plots of the basic design become the whole plots of the augmented design.) The control genotype is allocated to one of the subplots in each plot, and test genotypes are allocated to the other subplots.

So you first need to generate the basic design, using a procedure like AGHIERARCHICAL or AGLATIN. You can then use AFAUGMENTED to augment it.

In the simplest situation, the basic design has blocking factors identifying its plots, and a treatment factor defined to indicate the control genotype allocated to each plot. For example, Lin & Poushinsky (1983) used a 4 × 4 Latin square as their basic design, with 4 different control genotypes. In Genstat this can be constructed using AGLATIN (4.9.4), as shown in lines 4-5 of Example 4.13.5a. They then split each plot into 9 subplots, allocating the control to subplot 5 in each plot, and randomly allocated 128 test genotypes (numbered 5-132) to the other subplots across the design (lines 6-8). The BLOCKSTRUCTURE option specifies the blocking structure of the basic design (here rows crossed with columns), and thus the blocking factors that need to be expanded. The GENOTYPES option specifies the genotypes factor which, on input, indicates the control genotype on each plot. The NSUBPLOTS option specifies the number of subplots to define within each plot, and the SUBCONTROL option specifies the subplot to contain the control. The LEVTEST option specifies which levels of the augmented GENOTYPES factor are to represent the test genotypes. Setting option PRINT=design prints the design, using procedure PDESIGN; by default it is not printed.

Example 4.13.5a

```
 2  " Augmented design based on a 4x4 Latin square,
-3    as in Lin & Poushinsky (1983, Biometrics)."
 4  AGLATIN     [PRINT=*; ANALYSE=no] NROWS=4; NSQUARES=1; SEED=584578;\
 5              TREATMENTFACTORS=!p(Genotype); ROWS=Row; COLUMNS=Column
 6  AFAUGMENTED [PRINT=design; BLOCKSTRUCTURE=Row*Column;\
 7              LEVTEST=!(5...132); LEVCONTROL=5; GENOTYPES=Genotype;\
 8              NSUBPLOTS=9; SUBCONTROL=5; TESTVSCONTROL=TvsC; CONTROLS=Control
```

Treatments on each unit of the design
=====================================

Row	Subplots / Column	1	2	3	4	5	6	7	8	9
1	1	21	35	16	47	3	53	29	91	93
	2	114	32	120	6	1	48	107	92	106
	3	52	61	7	118	2	130	115	109	46
	4	86	12	78	38	4	113	101	22	97
2	1	80	125	73	18	4	75	83	105	68
	2	51	77	102	132	2	14	104	89	59
	3	119	40	17	31	1	39	41	62	9
	4	5	11	60	10	3	85	15	13	43
3	1	57	23	90	37	1	100	42	67	117
	2	108	82	34	27	3	24	65	124	26
	3	71	66	56	49	4	84	74	131	36
	4	44	111	63	128	2	70	122	58	99
4	1	96	98	25	45	2	50	126	110	28
	2	76	127	81	94	4	121	64	79	55
	3	72	112	30	116	3	103	33	95	87
	4	123	129	20	54	1	8	88	19	69

Treatment factors are listed in the order: Genotype.

```
  9  PRINT       TvsC,Genotype,Control,Row,Column
```

TvsC	Genotype	Control	Row	Column
test	21	5	1	1
test	35	5	1	1
test	16	5	1	1
test	47	5	1	1
control	3	3	1	1
test	53	5	1	1
test	29	5	1	1
test	91	5	1	1
test	93	5	1	1
test	114	5	1	2
test	32	5	1	2
test	120	5	1	2
test	6	5	1	2
control	1	1	1	2
test	48	5	1	2
test	107	5	1	2
test	92	5	1	2
test	106	5	1	2
test	52	5	1	3
test	61	5	1	3
test	7	5	1	3
test	118	5	1	3
control	2	2	1	3
test	130	5	1	3
test	115	5	1	3
test	109	5	1	3
test	46	5	1	3
test	86	5	1	4
test	12	5	1	4
test	78	5	1	4
test	38	5	1	4
control	4	4	1	4
test	113	5	1	4
test	101	5	1	4
test	22	5	1	4
test	97	5	1	4
test	80	5	2	1
test	125	5	2	1
test	73	5	2	1
test	18	5	2	1
control	4	4	2	1
test	75	5	2	1
test	83	5	2	1
test	105	5	2	1

test	68	5	2	1
test	51	5	2	2
test	77	5	2	2
test	102	5	2	2
test	132	5	2	2
control	2	2	2	2
test	14	5	2	2
test	104	5	2	2
test	89	5	2	2
test	59	5	2	2
test	119	5	2	3
test	40	5	2	3
test	17	5	2	3
test	31	5	2	3
control	1	1	2	3
test	39	5	2	3
test	41	5	2	3
test	62	5	2	3
test	9	5	2	3
test	5	5	2	4
test	11	5	2	4
test	60	5	2	4
test	10	5	2	4
control	3	3	2	4
test	85	5	2	4
test	15	5	2	4
test	13	5	2	4
test	43	5	2	4
test	57	5	3	1
test	23	5	3	1
test	90	5	3	1
test	37	5	3	1
control	1	1	3	1
test	100	5	3	1
test	42	5	3	1
test	67	5	3	1
test	117	5	3	1
test	108	5	3	2
test	82	5	3	2
test	34	5	3	2
test	27	5	3	2
control	3	3	3	2
test	24	5	3	2
test	65	5	3	2
test	124	5	3	2
test	26	5	3	2
test	71	5	3	3
test	66	5	3	3
test	56	5	3	3
test	49	5	3	3
control	4	4	3	3
test	84	5	3	3
test	74	5	3	3
test	131	5	3	3
test	36	5	3	3
test	44	5	3	4
test	111	5	3	4
test	63	5	3	4
test	128	5	3	4
control	2	2	3	4
test	70	5	3	4
test	122	5	3	4
test	58	5	3	4
test	99	5	3	4
test	96	5	4	1
test	98	5	4	1
test	25	5	4	1
test	45	5	4	1
control	2	2	4	1
test	50	5	4	1
test	126	5	4	1
test	110	5	4	1
test	28	5	4	1

```
    test          76          5          4          2
    test         127          5          4          2
    test          81          5          4          2
    test          94          5          4          2
 control           4          4          4          2
    test         121          5          4          2
    test          64          5          4          2
    test          79          5          4          2
    test          55          5          4          2
    test          72          5          4          3
    test         112          5          4          3
    test          30          5          4          3
    test         116          5          4          3
 control           3          3          4          3
    test         103          5          4          3
    test          33          5          4          3
    test          95          5          4          3
    test          87          5          4          3
    test         123          5          4          4
    test         129          5          4          4
    test          20          5          4          4
    test          54          5          4          4
 control           1          1          4          4
    test           8          5          4          4
    test          88          5          4          4
    test          19          5          4          4
    test          69          5          4          4
```

Note that, if there are insufficient test genotypes, some plots may contain NSUBPLOTS minus one subplots. An error is given if there are too few genotypes for any of the plots to contain NSUBPLOTS subplots.

The SEED option specifies a seed for the random numbers that are used to make the allocations. The default value of zero continues an existing sequence of random numbers if any have already been used in the current Genstat job, or obtains a random seed using the system clock if none have been used already. You can also set SEED=-1 if you want to suppress any randomization.

If the design has other treatments (as well as GENOTYPES), these can be specified using the TREATMENTSTRUCTURE option. This takes a model formula as its setting (so you would define the treatment terms that are to be included in the analysis). However, but it is sufficient just to list the factors if you prefer. These will then be expanded similarly to the blocking factors. Likewise, if you have covariates whose values are defined on the plots of the basic design, these can be specified using the COVARIATE option.

You can use the CONTROLS option to save a factor with a level for each control, and another level for all the test genotypes. You can also use the TESTVSCONTROL option to save a factor with one level for the control genotypes, and another level for the test genotypes. (These will be identical if there is only one control genotype.)

If you want to specify several controls in each whole plot of the augmented design, you can define the basic design to have subplots already, namely those with the controls. Example 4.13.5b has a balanced-incomplete-block design for three treatments as the basic design. The first block has controls 1 and 3, the second has 2 and 3, and the third has 1 and 2. So we start with two subplots. The AFAUGMENTED command in lines 14-15 expands the design to have eight subplots, adding 18 test genotypes. . The SUBCONTROLS option is now set to a variate to put the controls onto subplots 3 and 6, randomizing the allocation within each plot.

Example 4.13.5b

```
 10  " Augmented design based on a balanced-incomplete-block design to
-11    show how to form a design with more than one control per whole-plot."
 12  FACTOR      [LEVELS=3; VALUES=1,1,2,2,3,3] Blocks
 13  FACTOR      [LEVELS=3; VALUES=1,3,2,3,1,2] Genotypes
```

```
 14  AFAUGMENTED [PRINT=design; BLOCKSTRUCTURE=Blocks; LEVTEST=!(101...118);\
 15              GENOTYPES=Genotypes; NSUBPLOTS=8; SUBCONTROL=!(3,6)

Treatments on each unit of the design
=====================================


 Subplots     1     2     3     4     5     6     7     8
   Blocks
        1   102   108     1   115   107     3   104   106
        2   109   101     2   110   103     3   105   117
        3   113   116     2   114   118     1   112   111


Treatment factor: Genotypes.
```

You can predefine the SUBPLOTS factor if you want to allocate the controls to the subplots explicitly, yourself. For example,

```
FACTOR      [LEVELS=8; VALUES=3,6,3,6,3,6] Blocks
AFAUGMENTED [PRINT=design; BLOCKSTRUCTURE=Blocks;\
             LEVTEST=Tests; GENOTYPES=Genotypes;\
             NSUBPLOTS=8; SUBCONTROL=Csubs
```

puts control 1 in block 1 explicitly onto subplot 3, and control 2 in block 1 explicitly onto subplot 6, etc. The NSUBPLOTS option of AFAUGMENTED then need not be set, but will default to the number of levels defined for SUBPLOTS. Of course, if you do predefine the SUBPLOTS factor, you no longer need to have the same number of controls in each plot.

You can even define a null basic design. The "augmented" design will then simply consist of some control and test genotypes allocated to the (sub)plots within the field (with the SUBPLOTS and SUBCONTROL options determining the allocation of the controls as before). This provides a way of defining the controls in a systematically repeating way, as shown in Example 4.13.4c.

Example 4.13.5c

```
 16  " Augmented design with a null basic design, to show how
-17    to form a design with systematic repeating controls."
 18  " design with systematic repeating controls "
 19  FACTOR      [LEVELS=32; VALUES=2,6...30] plots
 20  FACTOR      [LEVELS=2; VALUES=(1,2)4] genotypes
 21  AFAUGMENTED [SUBPLOTS=plots; LEVTEST=!(3...26);\
 22              GENOTYPES=genotypes; CONTROLS=controls
 23  PRINT       plots,genotypes,controls

      plots   genotypes    controls
          1           4           3
          2           1           1
          3          13           3
          4          11           3
          5          19           3
          6           2           2
          7           5           3
          8          23           3
          9          15           3
         10           1           1
         11           6           3
         12          25           3
         13           3           3
         14           2           2
         15          24           3
         16          17           3
         17          26           3
         18           1           1
         19          21           3
         20           9           3
         21          20           3
         22           2           2
```

```
23          12          3
24          22          3
25           7          3
26           1          1
27          18          3
28          14          3
29           8          3
30           2          2
31          10          3
32          16          3
```

By default, the test genotypes are unreplicated. You can set the NREPTEST option to a scalar to replicate every test genotype the same number of times, or to a variate to have different numbers of replicates (as, for example. in a partially-replicated design).

4.13.6 Construction of design keys

Design keys provide the basis of the representation used to store the repertoire of designs obtainable from procedure AGDESIGN (4.9.3). This covers a range of standard situations, but cannot allow for every eventuality. The FKEY directive allows you to form keys for other circumstances and, if these are likely to occur frequently, you can extend or replace the standard repertoire using procedure FDESIGNFILE (see Part 3 of the *Genstat Reference Manual*).

FKEY directive

Forms design keys for multi-stratum experimental designs, allowing for confounded and aliased treatments.

Options

BASICFACTORS = *factors*	Factors indexing the units of the design
ADDEDFACTORS = *factors*	Factors to be allocated to the units of the design
KEY = *matrix*	Stores the design key (ADDEDFACTORS × BASICFACTORS)
INKEY = *matrix*	Can be used to input existing allocations for some of the added factors
HIERARCHIES = *matrix*	Can be used to specify that some of the factors must be constant within each combination of levels of other factors; the matrix has a row for each added factor and columns first for the basic factors and then for the added factors, ones in the entries where the row factor must be constant within the combinations of the column factors, zero elsewhere
SEED = *scalar*	Can provide a seed to generate a random permutation of the sets of basic effects that may be allocated to each added factor, thus producing design randomly selected from all those that might be possible; default * i.e. no permutation
ROWPRIMES = *variate*	Prime numbers for the rows of the KEY matrix
COLPRIMES = *variate*	Prime numbers for the columns of the KEY matrix
ROWMAPPINGS = *variate*	Mappings from the rows of the KEY to the TREATMENTFACTORS
COLMAPPINGS = *variate*	Mappings from the columns of the KEY to the BLOCKFACTORS
SAVE = *identifier*	Structure to save all the information about the formation of the design; this can then be input later to give a

different design (if possible) with the same properties

Parameters

REQUIRED = *formula structures*	Formulae each defining a list of terms that are to be estimated in the analysis
NONNEGLIGIBLE = *formula structures*	Formulae each specifying terms that cannot be ignored in the context of the corresponding REQUIRED formula

The assumption in FKEY is that the units of the design are indexed by a set of factors known as the *basic* factors. The key allows the values of another set of factors, known here as the *added* factors, to be calculated from the basic factors. These factors are listed using the BASICFACTORS and ADDEDFACTORS options. They must all have been declared previously as factors, and their numbers of levels must have been defined. Usually the basic factors are the factors that will be used to define the block formula of the design (for example, blocks, plots, rows, columns, subplots and so on) and the added factors are the treatment factors, but in partial replicates, for example, the basic factors may be the treatment factors and the added factors the block factors.

If the basic and added factors all have prime numbers of levels the key is saved, by the KEY option, as a matrix with a row for each added factor and a column for each basic factor. However, if the levels are not all prime, factors that do not have prime numbers of levels need to be broken up into "pseudo-factors". Thus, a factor with six levels will be represented by the combinations of levels of two pseudo-factors, one with two levels and one with three levels. In simple cases it is straightforward to do this by hand, as shown in Example 4.13.6a. Alternatively, FPSEUDOFACTORS can do the pseudo-factoring automatically, and this is illustrated in Example 4.13.6b.

The main properties of the design are derived from the REQUIRED and NONNEGLIGIBLE parameters. Example 4.13.6a considers the simple case of a block design containing three blocks of nine plots. The experiment is to have three treatment factors, A, B and C, and these will be the added factors. The design has a block structure of plots nested within blocks

```
Blocks/Plots
```

but as there are nine plots within each block we use two plot factors Plot1 and Plot2, each with three levels, to identify the plots and the block structure becomes

```
Blocks/(Plot1.Plot2)
```

So we have three basic factors, Block, Plot1 and Plot2. In the analysis we wish to be able to estimate all main effects and interactions of the factors A, B and C, except the three-factor interaction A.B.C; these terms are specified by the formula structure supplied using the REQUIRED parameter. The NONNEGLIGIBLE parameter specifies model terms that cannot be ignored in the analysis: that is, the model terms with which these required terms cannot be confounded. Here we have the main effect Blocks and all main effects and interactions of the factors A, B and C.

The key is saved in matrix K, and then used at line 7 to generate the values of A, B and C from the values of Block, Plot1 and Plot2, generated in line 6. The dummy analysis of variance table (from line 10) shows that the main effects and two-factor interactions can all be estimated within blocks as required.

Example 4.13.6a

```
 2  FACTOR [NVALUES=27; LEVELS=3] Block,Plot1,Plot2,A,B,C
 3  FKEY [BASIC=Block,Plot1,Plot2; ADDED=A,B,C; KEY=K] \
 4    REQUIRED=!f(A*B*C-A.B.C); NONNEGLIGIBLE=!f(Block+A*B*C)
 5  PRINT K; DECIMALS=0
```

```
                    K
                    1           2           3

          1         0           1           0
          2         0           0           1
          3         1           1           1
```

```
   6  GENERATE Block,Plot1,Plot2
   7  & [BLOCKS=Block,Plot1,Plot2; KEY=K] A,B,C
   8  BLOCKSTRUCTURE Block/(Plot1.Plot2)
   9  TREATMENT A*B*C
  10  ANOVA [FACTORIAL=2]

Analysis of variance
====================

Source of variation     d.f.

Block stratum              2

Block.Plot1.Plot2 stratum
A                          2
B                          2
C                          2
A.B                        4
A.C                        4
B.C                        4
Residual                   6

Total                     26
```

When pseudo-factors are required for the added factors, the ROWPRIMES option can be used to save a variate storing the (prime) number of levels corresponding to each row of the key, and the ROWMAPPINGS option can save a variate with an element for each row containing the number of the corresponding added factor. So, if we had two added factors, one with five and one with six levels, the ROWPRIMES variate might contain the values 5, 2, and 3, and the ROWMAPPINGS variate the values 1, 2, and 2. The second added factor (with six levels) would then be represented by two pseudo-factors, corresponding to the second and third rows of the key. The COLPRIMES and COLMAPPINGS options can similarly save details of the pseudo-factors required for basic factors with non-prime numbers of levels.

In Example 4.13.6b, we repeat the construction of the design in Example 4.13.6a but now with a nine-level factor Plot. This is broken up automatically by FKEY into two pseudo-factors. The variate Cprime stores the primes for the columns of the key (all 3), and variate Cmap indicates that the first column corresponds to Block (the first basic factor), and the second and third columns correspond to Plot (the second basic factor). The variates saved by ROWPRIMES, COLPRIMES, ROWMAPPINGS, and COLMAPINGS can be used in procedure AKEY (4.13.2), together with the key, to generate the factors automatically without the need to worry about the pseudo-factoring; see lines 16 and 17.

Example 4.13.6b

```
  11  FACTOR [LEVELS=9] Plot
  12  FKEY [BASIC=Block,Plot; ADDED=A,B,C; KEY=K; \
  13    COLPRIMES=Cprime; COLMAPPINGS=Cmap] \
  14    REQUIRED=!f(A*B*C-A.B.C); NONNEGLIGIBLE=!f(Block+A*B*C)
  15  PRINT K,Cprime,Cmap; DECIMALS=0

                    K
                    1           2           3

          1         0           1           0
          2         0           0           1
          3         1           1           1
```

```
       Cprime           3           3           3

         Cmap           1           2           2

  16  AKEY [PRINT=design; BLOCKFACTORS=Block,Plot; KEY=K; \
  17    COLPRIMES=Cprime; COLMAPPINGS=Cmap] A,B,C

Treatment combinations on each unit of the design
=================================================

  Plot  1      2      3      4      5      6      7      8      9
 Block
    1  1 1 1  1 2 2  1 3 3  2 1 2  2 2 3  2 3 1  3 1 3  3 2 1  3 3 2
    2  1 1 2  1 2 3  1 3 1  2 1 3  2 2 1  2 3 2  3 1 1  3 2 2  3 3 3
    3  1 1 3  1 2 1  1 3 2  2 1 1  2 2 2  2 3 3  3 1 2  3 2 3  3 3 1

 Treatment factors are listed in the order: A B C

  18  BLOCKSTRUCTURE Block/Plot
  19  TREATMENT A*B*C
  20  ANOVA [FACTORIAL=2]

Analysis of variance
====================

Source of variation     d.f.

Block stratum              2

Block.Plot stratum
A                          2
B                          2
C                          2
A.B                        4
A.C                        4
B.C                        4
Residual                   6

Total                     26
```

The algorithm that FKEY uses to construct the key is based on the method developed by Franklin & Bailey (1977), Franklin (1985) and Kobilinsky (1995). Essentially this considers the possible orthogonal sets of contrasts amongst the main effects and interactions of the basic factors, and tries in turn to find a feasible set against which to confound each added factor. Often there are several feasible ways in which this can be done. To avoid FKEY selecting the same key every time, you can set the SEED option to an integer that will be used to generate a random permutation of the order in which the sets of basic contrasts are considered, thus producing design randomly selected from all those that might be possible; by default no permutation takes place. Alternatively, you can use the SAVE option to save all the information about the formation of the design; this can then be input later to provide the next possible key (if available) with the requested properties.

In Example 4.13.6c, we first use the SEED option to select a key at random from those that are feasible, and then the SAVE option to select three different keys.

Example 4.13.6c

```
  21  " Use the SEED option to select a feasible design at random."
  22  FKEY [BASIC=Block,Plot; ADDED=A,B,C; KEY=K; SEED=284762]\
  23    REQUIRED=!f(A*B*C-A.B.C); NONNEGLIGIBLE=!f(Block+A*B*C)
  24  PRINT K; DECIMALS=0
```

```
                        K
                        1           2           3

            1           0           0           1
            2           0           1           2
            3           1           1           1
```

```
  25  AKEY [BLOCKFACTORS=Block,Plot; KEY=K; \
  26    COLPRIMES=Cprime; COLMAPPINGS=Cmap] A,B,C
  27  ANOVA [FACTORIAL=2]

Analysis of variance
====================

Source of variation     d.f.

Block stratum              2

Block.Plot stratum
A                          2
B                          2
C                          2
A.B                        4
A.C                        4
B.C                        4
Residual                   6

Total                     26
```

```
  28  " form three keys."
  29  FOR [NTIMES=3]
  30    FKEY [BASIC=Block,Plot; ADDED=A,B,C; KEY=K; SAVE=Ksave] \
  31      REQUIRED=!f(A*B*C-A.B.C); NONNEGLIGIBLE=!f(Block+A*B*C)
  32    PRINT K; DECIMALS=0
  33  ENDFOR

                        K
                        1           2           3

            1           0           1           0
            2           0           0           1
            3           1           1           1


                        K
                        1           2           3

            1           0           1           0
            2           1           0           1
            3           0           1           1


                        K
                        1           2           3

            1           0           1           0
            2           0           1           1
            3           1           0           1
```

If the design has more than two strata suitable for the estimation of treatment effects, the REQUIRED and NONNEGLIGIBLE parameters can specify lists of formulae, in parallel, one pair of formulae for each stratum. Each REQUIRED formula specifies the terms that must be estimated in one of the strata (or in a stratum below it), and the corresponding NONNEGLIGIBLE formula specifies the terms that cannot be ignored there. In Example 4.13.6d we have a block formula

```
Block / Wplot / Subplot
```

which produces three strata

```
Block + Block.Wplot + Block.Wplot.Subplot
```

The Subplot factor has nine levels, so FKEY again breaks this down, as in Example 4.13.6b.

The first formula in the REQUIRED list !f((A+B+C)*(A+B+C)), in parallel with the formula !f(Block+Block.Wplot) in the NONNEGLIGIBLE list, indicates that we do not want the main effects or two-factor interaction of factors A, B and C to be confounded with each other nor with Block or Block.Wplot; this ensures that they will be estimated in the Block.Wplot.Subplot stratum. The second pair of formulae, !f((A+B+C+D+E)*(A+B+C+D+E)) and !f(Block), indicate that we want to estimate the main effects and two-factor interactions of all the five treatment factors A, B, C, D and E in the Block.Wplot stratum or below; in effect this means we are willing to have D and E and any of their interactions estimated in the Block.Wplot stratum. As a result, D and part of the A.E interaction are estimated in the Block.Wplot stratum. Section 4.13.7 shows how to set up a pseudo-factor for this part of the A.E interaction and thus ensure the correct analysis.

Example 4.13.6d

```
  2  FACTOR [NVALUES=81; LEVELS=3] Block,Wplot,A,B,C,D,E
  3  & [LEVELS=9] Subplot
  4  FKEY [BASIC=Block,Wplot,Subplot; ADDED=A,B,C,D,E; KEY=K; \
  5    COLPRIMES=Clevel; COLMAPPINGS=Cmapping] \
  6    REQUIRED=!f((A+B+C)*(A+B+C)),!f((A+B+C+D+E)*(A+B+C+D+E)); \
  7    NONNEGLIGIBLE=!f(Block+Block.Wplot),!f(Block)
  8  PRINT K,Clevel,Cmapping; FIELD=6; DECIMALS=0

                K
                1     2     3     4

          1     0     0     1     0
          2     0     0     0     1
          3     1     0     1     1
          4     0     1     0     0
          5     1     1     2     0

      Clevel    3     3     3     3

    Cmapping    1     2     3     3

  9  AKEY [BLOCKFACTORS=Block,Wplot,Subplot; KEY=K; \
 10    COLPRIMES=Clevel; COLMAPPINGS=Cmapping] A,B,C,D,E
 11  BLOCKSTRUCTURE Block/Wplot/Subplot
 12  TREATMENTSTRUCTURE A*B*C*D*E
 13  ANOVA [FACTORIAL=2]

******** Warning, code AN 17, statement 1 on line 13

Command: ANOVA [FACTORIAL=2]
Partial confounding.
A.E is partially confounded with Block.Wplot


Analysis of variance
====================

Source of variation     d.f.

Block stratum              2

Block.Wplot stratum
D                          2
A.E                        4

Block.Wplot.Subplot stratum
A                          2
B                          2
C                          2
E                          2
A.B                        4
A.C                        4
```

```
B.C                    4
A.D                    4
B.D                    4
C.D                    4
A.E                    4
B.E                    4
C.E                    4
D.E                    4
Residual              24

Total                 80
```

In a multi-stratum design, you may wish to insist that some factors are applied to complete units of one of the strata; for example, in the split-plot design in Section 9.1 varieties are applied to complete whole-plots within each of the blocks. This can be done using the `HIERARCHIES` option, which allows you to indicate that some of the added factors must be constant within each combination of levels of other factors. For example, in Example 4.2.1, the levels of the factor `Variety` must remain constant within each combination of `Wplots` and `Blocks`. These constraints are specified, if required, by supplying a matrix with a row for each added factor and columns first for the basic factors and then for the added factors. The matrix contains ones in the entries where the row factor must be constant within the combinations of the column factors, and zeros elsewhere. So, in Example 4.2.1, we would specify the matrix

```
                Blocks  Wplots  Subplots  Variety  Nitrogen
Variety              1       1         0        0         0
Nitrogen             0       0         0        0         0
```

Notice that the combinations of factors within which the added factor must remain constant can include other added factors.

In Example 4.13.6e we use the `HIERARCHIES` option to ensure that the factor `E` is applied to complete whole-plots within each block. So the fifth row of `Hmat` (which corresponds to `E`) has a one in the first column (`Block`) and the second column (`Wplot`), and zero elsewhere.

Example 4.13.6e

```
 14  MATRIX [ROWS=5; COLUMNS=8; VALUES=32(0), 2(1),6(0)] Hmat
 15  PRINT Hmat; FIELD=6; DECIMALS=0

             Hmat
                1     2     3     4     5     6     7     8

          1     0     0     0     0     0     0     0     0
          2     0     0     0     0     0     0     0     0
          3     0     0     0     0     0     0     0     0
          4     0     0     0     0     0     0     0     0
          5     1     1     0     0     0     0     0     0

 16  FKEY [BASIC=Block,Wplot,Subplot; ADDED=A,B,C,D,E; \
 17    KEY=K; HIERARCHIES=Hmat] \
 18    REQUIRED=!f((A+B+C)*(A+B+C)), !f((A+B+C+D+E)*(A+B+C+D+E)); \
 19    NONNEGLIGIBLE=!f(Block+Block.Wplot),!f(Block)
 20  PRINT K; FIELD=6; DECIMALS=0

                K
                1     2     3     4

          1     0     0     1     0
          2     0     0     0     1
          3     1     0     1     1
          4     0     1     1     0
          5     1     1     0     0

 21  AKEY [BLOCKFACTORS=Block,Wplot,Subplot; KEY=K; \
 22    COLPRIMES=Clevel; COLMAPPINGS=Cmapping] A,B,C,D,E
 23  BLOCKSTRUCTURE Block/Wplot/Subplot
```

```
  24  TREATMENTSTRUCTURE A*B*C*D*E
  25  ANOVA [FACTORIAL=2]

******** Warning, code AN 17, statement 1 on line 25

Command: ANOVA [FACTORIAL=2]
Partial confounding.
A.D is partially confounded with Block.Wplot


Analysis of variance
====================

Source of variation     d.f.

Block stratum              2

Block.Wplot stratum
E                          2
A.D                        4

Block.Wplot.Subplot stratum
A                          2
B                          2
C                          2
D                          2
A.B                        4
A.C                        4
B.C                        4
A.D                        4
B.D                        4
C.D                        4
A.E                        4
B.E                        4
C.E                        4
D.E                        4
Residual                  24

Total                     80
```

FKEY can also be used to extend an existing design, by allocating further factors to the units. The existing key should then be input using the INKEY option, with zeros in the rows for the new added factors.

In Example 4.13.6f we start with a key that generates a design for three 2-level factors A, B and C in two blocks of four plots. Originally, we thus have a basic factor Block (with two levels) for the blocks, and two basic factors Plot1 and Plot2 (also with two levels) to represent the four plots.

We then extend the design by replicating it twice (to give four blocks altogether) and by splitting the plots each into two subplots. So we now have factors Block1 and Block2 for the blocks, Plot1 and Plot2 for the plots, and Subplot for the subplots. The key KeyABC indicates how the factors A, B and C are derived from the extended set of blocking (or basic) factors, and has two rows of zeros for two extra factors D and E (both at two levels) that the design is to contain. These two rows are then filled in by FKEY to give the full key ExtKey.

Example 4.13.6f

```
  2  FACTOR [NVALUES=8; LEVELS=2] Block,Plot1,Plot2,A,B,C
  3  MATRIX [ROWS=3; COLUMNS=3; VALUES=1,1,1, 0,1,0, 0,0,1] Key
  4  GENERATE Block,Plot1,Plot2
  5  & [BLOCKS=Block,Plot1,Plot2; KEY=Key] A,B,C
  6  BLOCKSTRUCTURE Block / (Plot1.Plot2)
  7  TREATMENTSTRUCTURE A * B * C
  8  ANOVA [FACTORIAL=2]
```

```
Analysis of variance
====================

Source of variation     d.f.

Block stratum              1

Block.Plot1.Plot2 stratum
A                          1
B                          1
C                          1
A.B                        1
A.C                        1
B.C                        1

Total                      7

   9  FACTOR [NVALUES=32; LEVELS=2] Block1,Block2,Plot1,Plot2,Subplot,\
  10    A,B,C,D,E
  11  MATRIX [COLUMNS=!t(Block1,Block2,Plot1,Plot2,Subplot); \
  12    ROWS=!t(A,B,C,D,E); VALUES=0,1,1,1,0, 0,0,1,0,0, 0,0,0,1,0, \
  13    0,0,0,0,0, 0,0,0,0,0] KeyABC
  14  PRINT KeyABC; FIELD=9; DECIMALS=0

               KeyABC
               Block1   Block2    Plot1    Plot2  Subplot

            A        0        1        1        1        0
            B        0        0        1        0        0
            C        0        0        0        1        0
            D        0        0        0        0        0
            E        0        0        0        0        0

  15  FKEY [BASIC=Block1,Block2,Plot1,Plot2,Subplot; ADDED=A,B,C,D,E; \
  16    KEY=ExtKey; INKEY=KeyABC] REQUIRED=!f((A+B+C+D+E)*(A+B+C+D+E)); \
  17    NONNEGLIGIBLE=!f(Block1.Block2+(A+B+C+D+E)*(A+B+C+D+E))
  18  PRINT ExtKey; FIELD=9; DECIMALS=0

               ExtKey
                    1        2        3        4        5

            1        0        1        1        1        0
            2        0        0        1        0        0
            3        0        0        0        1        0
            4        0        0        0        0        1
            5        1        0        1        0        1

  19  GENERATE Block1,Block2,Plot1,Plot2,Subplot
  20  & [BLOCKS=Block1,Block2,Plot1,Plot2,Subplot; KEY=ExtKey] A,B,C,D,E
  21  BLOCKSTRUCTURE (Block1.Block2) / (Plot1.Plot2) / Subplot
  22  TREATMENTS A * B * C * D * E
  23  ANOVA [FACTORIAL=2]

Analysis of variance
====================

Source of variation     d.f.

Block1.Block2 stratum      3

Block1.Block2.Plot1.Plot2 stratum
A                          1
B                          1
C                          1
A.B                        1
A.C                        1
B.C                        1
D.E                        1
Residual                   5
```

```
Block1.Block2.Plot1.Plot2.Subplot stratum
D                              1
E                              1
A.D                            1
B.D                            1
C.D                            1
A.E                            1
B.E                            1
C.E                            1
Residual                       8

Total                         31
```

FKEY can form keys for small designs fairly quickly, but for complicated arrangements you may find that it takes some time to check the various possibilities.

4.13.7 Forming pseudo-factors from a design key

FPSEUDOFACTORS directive

Determines patterns of confounding and aliasing from design keys, and extends the treatment model to incorporate the necessary pseudo-factors.

Options

TREATMENTSTRUCTURE = *formula*	Treatment model for the design
BLOCKSTRUCTURE = *formula*	Block model for the design
FACTORIAL = *scalar*	Limit on the number of factors in each treatment term
LROWS = *factors* or *scalars*	Numbers of levels of factors, or factors, corresponding to the rows of the key matrices
LCOLUMNS = *factors* or *scalars*	Numbers of levels of factors, or factors, corresponding to the columns of the key matrices
NEWTREATMENTSTRUCTURE = *identifier*	Store the extended treatment model
PSEUDOFACTORS = *pointer*	Pseudo-factors required for the keys
NPSEUDOFACTORS = *scalar*	Number of pseudo-factors required for the keys
KEYPSEUDOFACTORS = *matrix*	Key to generate the pseudo-factors from the treatment factors
KEYCONTRASTS = *matrix*	Key partitioning the treatment terms into orthogonal sets of contrasts

Parameters

KEY = *matrices*	Design keys
KEYINVERSE = *matrices*	Store the inverses of the design keys
ALIASSETS = *variates*	Stores aliasing information about the orthogonal sets of treatment contrasts
RESOLUTION = *scalars*	Saves the resolution number of the design constructed by each key

The FPSEUDOFACTORS directive operates on a list of design keys, specified using the KEY parameter. It assumes that a design is to be formed by generating a replicate using each design key, and forms pseudo-factors to allow the ANOVA directive to cope with partial confounding or aliasing in the design. The factors corresponding to the rows of the keys are specified by the LROWS option, and those for the columns are specified by the LCOLUMNS option. If LROWS is not specified, FPSEUDOFACTORS will take the factors from the formula specified by the

TREATMENTSTRUCTURE parameter, in the order that they occur there. Similarly, the BLOCKSTRUCTURE option can provide a default for LCOLUMNS.

The KEYINVERSE parameter allows the inverse keys to be saved (provided the keys are invertible). These are keys that would allow the factors corresponding to the columns of the original key to be generated from those corresponding to the rows (instead of row factors from column factors, as with the original key). If you merely wish to save the inverses, you can specify scalars defining the numbers of levels of the factors instead of the factors themselves.

The BLOCKSTRUCTURE option defines the block structure within each replicate, so the full block structure would be Rep/(#BLOCKSTRUCTURE) where Rep is factor to identify the replicates. The TREATMENTSTRUCTURE option specifies the treatment terms to be estimated using the design, and the FACTORIAL option allows a limit to be set on the number of factors in the terms that are generated as, for example, in the ANOVA directive. FPSEUDOFACTORS examines the keys to see whether any treatment terms are partially aliased or partially confounded. Provided the factors of each such term all have the same (prime) number of levels it can then extend the treatment formula, inserting pseudo-factors for these terms, so that the ANOVA directive can produce a correct analysis. The extended formula can be saved using the NEWTREATMENTSTRUCTURE option, and the NPSEUDOFACTORS option saves the number of pseudo-factors that are needed. The pseudo-factors themselves are represented by the elements of a pointer specified by the PSEUDOFACTORS option, and the KEYPSEUDOFACTORS option can save the key matrix required to generate their values from the values of the treatment factors.

This is illustrated in Example 4.13.7a which continues Example 4.13.6e. First of all, in lines 26-28, we form factors Subplot1 and Subplot2 to represent the nine subplots. Unlike the FKEY directive, described in Section 4.13.6, FPSEUDOFACTORS requires all the factors to have prime numbers of levels.

The block structure is now Block/Wplot/(Subplot1.Subplot2) and the stratum Block.Wplot.Subplot1.Subplot2 corresponds to the stratum Block.Wplot.Subplot in Example 4.13.7a.

The key K defines the relationship between the treatment factors A, B, C, D and E, and the factors in the block structure Block, Wplot, Subplot1 and Subplot2. Notice that, as LROWS and LCOLUMNS are not specified, the factors for the rows and columns of the key are taken from the treatment and block formulae.

In the new treatment structure Ntreat, pseudo-factor Pf[1] is attached to the term A.D to represent the part of this term that is estimated in the Block.Wplot.Subplot1.Subplot2 stratum (the remainder of the term is estimated in the Block.Wplot stratum). The pseudo-factor key PfK indicates that, in fact, Pf[1] represents the contrasts D^1E^1. This key is used to generate the pseudo-factors at line 36, the new treatment structure is specified for ANOVA in line 36, and you can see that the resulting analysis (from line 37) now has the correct degrees of freedom for A.D.

Example 4.13.7a

```
26  FACTOR [NVALUES=81; LEVELS=3] Subplot1,Subplot2
27  CALCULATE Subplot1, Subplot2 = NEWLEVELS(Subplot; \
28    !(1,1,1,2,2,2,3,3,3), !(1,2,3,1,2,3,1,2,3) )
29  FPSEUDOFACTORS [TREATMENTSTRUCTURE=A*B*C*D*E; FACTORIAL=2; \
30    BLOCKSTRUCTURE=Block/(Wplot)/(Subplot1.Subplot2); \
31    NEWTREATMENTSTRUCTURE=Ntreat; PSEUDOFACTORS=Pf; \
32    KEYPSEUDOFACTORS=PfK] K
33  PRINT PfK,Ntreat; DECIMALS=0

            PfK
              1   2   3   4   5

          1   1   0   0   1   0
```

```
Ntreat
A + B + C + D + E + A.B + A.C + B.C + A.D // Pf[1] + B.D + C.D + A.E
+ B.E + C.E + D.E

  34  FACTOR [NVALUES=81; LEVELS=3] Pf[]
  35  GENERATE [BLOCKS=A,B,C,D,E; KEY=PfK] Pf[]
  36  TREATMENTSTRUCTURE #Ntreat
  37  ANOVA [FACTORIAL=2]

Analysis of variance
====================

Source of variation     d.f.

Block stratum              2

Block.Wplot stratum
E                          2
A.D                        2
Residual                   2

Block.Wplot.Subplot stratum
A                          2
B                          2
C                          2
D                          2
A.B                        4
A.C                        4
B.C                        4
A.D                        2
B.D                        4
C.D                        4
A.E                        4
B.E                        4
C.E                        4
D.E                        4
Residual                  26

Total                     80
```

FPSEUDOFACTORS can also determine the aliasing relationships of treatment terms in fractional factorial designs. The KEYCONTRASTS option can save a design key that partitions the treatment terms into orthogonal sets of contrasts. (The matrix thus has a row for each set of contrasts, and a column for each treatment factor.) The ALIASSETS parameter saves a variate, for each design key, with length equal to the number of rows in the KEYCONTRASTS matrix. The variate stores integers indicating the alias group of each set of contrasts so, if two elements of the variate are equal, this indicates that the corresponding sets of contrasts are aliased in the replicate generated by the design key concerned. The RESOLUTION parameter saves the resolution number for the replicate generated by each design key. This is the minimum number of factors involved in any pair of aliased terms.

This is illustrated in Example 4.13.7b which generates a design containing four 3-level factors in three blocks of 9 plots (it is thus a 1/3rd fraction of a 3^4 design in blocks of size nine). Generating a factional factorial is easy with FKEY. We simply specify more treatment factors than blocking factors. The REQUIRED formula indicates that we want to estimate the main effects of all the treatment factors, and the NONNEGLIGIBLE formula indicates that we do not want them to be confounded with blocks.

Example 4.13.7b

```
  2  " Generate a fractional factorial design: a 1/3 fraction of a 3**4."
  3  FACTOR [NVALUES=27; LEVELS=3] A,B,C,D,Block,Pl1,Pl2
  4  MATRIX [ROWS=!t(A,B,C,D); COLUMNS=!t(Block,Plot)] Key3to4th
  5  FKEY [BASIC=Block,Pl1,Pl2; ADDED=A,B,C,D; KEY=Key3to4th] \
  6    REQUIRED=!f(A+B+C+D); NONNEGLIGIBLE=!f(A+B+C+D+Block)
```

```
  7  PRINT Key3to4th; FIELD=4; DECIMALS=0

           Key3to4th
           Block Plot

          A   0   1   0
          B   1   1   0
          C   1   2   0
          D   0   0   1

  8  AKEY [BLOCKFACTORS=Block,Pl1,Pl2; KEY=Key3to4th] A,B,C,D
  9  BLOCKSTRUCTURE Block/(Pl1.Pl2)
 10  TREATMENTSTRUCTURE A+B+C+D
 11  ANOVA

Analysis of variance
====================

Source of variation     d.f.

Block stratum              2

Block.Pl1.Pl2 stratum
A                          2
B                          2
C                          2
D                          2
Residual                  16

Total                     26


 12  " Determine how the interactions are aliased."
 13  FPSEUDOFACTORS [TREATMENTSTRUCTURE=A*B*C*D; FACTORIAL=2; \
 14    BLOCKSTRUCTURE=Block/(Pl1,Pl2); KEYCONTRAST=Kcon] \
 15    KEY=Key3to4th; ALIASSET=Alias; RESOLUTION=Resolution
 16  PRINT Kcon,Alias; DECIMALS=0; FIELD=4

           Kcon             Alias
              1   2   3   4

          1   1   0   0   0   1
          2   0   1   0   0   2
          3   0   0   1   0   3
          4   0   0   0   1   4
          5   1   1   0   0   3
          6   2   1   0   0   5
          7   1   0   1   0   5
          8   2   0   1   0   2
          9   0   2   1   0   1
         10   0   2   2   0   5
         11   1   0   0   1   6
         12   1   0   0   2   7
         13   0   1   0   1   8
         14   0   1   0   2   9
         15   0   0   1   1  10
         16   0   0   1   2  11

 17  &     Resolution; DECIMALS=0

 Resolution
          3
```

FPSEUDOFACTORS is then used to determine how the interactions are aliased. For example, the first and ninth elements of alias both contain one, indicating that a is aliased with b^2c.

4.13.8 Forming the basic contrasts of a model term

FBASICCONTRASTS procedure

Breaks a model term down into its basic contrasts (R.W. Payne).

Options

TERM = *formula*	Model term to split into basic contrasts
PSEUDOFACTORS = *pointer*	Pseudo-factors representing the basic contrasts
NEWTERMS = *formula structure*	Model formula containing the term followed by the pseudofactors

No parameters

If you do not have the design keys that were used to generate a partially-confounded design, an alternative is to use the FBASICCONTRASTS procedure to break up each partially-confounded interaction into its sets of basic contrasts.

The interaction is specified using the TERM option. The PSEUDOFACTORS option saves a pointer containing the factors generated to represent the basic contrasts. Finally, the NEWTERMS option can save a new model formula containing the interaction followed by the pseudo-factor operator // and then the list of pseudo-factors. For example, for the interaction of two 3-level factors A and B, the NEWTERMS formula would be

```
A.B // (Pf[1,2])
```

where Pf[] is the pointer of pseudo-factors.

4.13.9 Minimum aberration designs

AFMINABERRATION directive

Forms minimum aberration factorial or fractional-factorial designs.

Options

PRINT = *string tokens*	Controls printed output (summary, keyblocks, keydefining, monitoring); default *
NTIMES = *scalar*	Number of designs to try in a random search; default 0 does the full search
SEED = *scalar*	Seed for the random number generator used to search the designs randomly; default 0

Parameters

LEVELS = *scalars*	Number of levels of the treatment factors, must be a power of a prime number
NTREATMENTFACTORS = *scalars*	Number of treatment factors
NUNITS = *scalars*	Number of units in each block of a block design or in the principal block of a fractional factorial
NSUBUNITS = *scalars*	Number of units in each (sub-)block
KEYBLOCKS = *matrices*	Design key for the blocks and sub-blocks
KEYDEFINING = *matrices*	Design key specifying the defining contrasts
RESOLUTION = *scalars*	Saves the resolution of the design
ABERRATION = *scalars*	Saves the aberration of the design
SUBRESOLUTION = *scalars*	Saves the resolution of the sub-design
SUBABERRATION = *scalars*	Saves the aberration of the sub-design

NDESIGN = *scalars*	Saves or defines the design number
NSUBDESIGN = *scalars*	Saves or defines the sub-design number

The concept of *minimum aberration* provides an effective way of selecting either a full factorial design where treatment contrasts are confounded with blocks, or a fractional factorial. (Essentially, these are equivalent – the fractional factorial design is formed by taking only one block of the full factorial.) The *resolution* of the design is defined as the largest integer *r* such that no interaction term with *r* factors is confounded with blocks (or aliased). The *aberration* of the design is the number of interaction terms with *r*+1 factors that are confounded (or aliased). A *minimum aberration* design is a design with the smallest aberration out of the designs with the highest available resolution. It is thus a design that is closest to the next level of resolution.

AFMINABERRATION searches for minimum aberration designs using the algorithm of Laycock & Rowley (1995), and we gratefully acknowledge Patrick Laycock's assistance with the implementation into Genstat. The number of treatment factors is specified by the NFACTORS parameter. Their number of levels is specified by the LEVELS parameter. This must be an integer power of a prime number. The number of units in each block (or the number of plots in the equivalent fractional factorial) is specified by the NUNITS parameter, and must be a power of LEVELS.

AFMINABERRATION can also form a sub-blocking factor that can be used to define blocks if the design is to be used to form a fractional factorial. The number of units in each sub-block is defined by the NSUBBLOCKS parameter (and again must be a power of LEVELS).

If there are very many designs to search, you may prefer to examine only a random selection. The NTIMES option sets the number of designs to try; its default of zero requests the standard (full) search. The SEED option sets the seed for the random numbers that are used to select the designs randomly; the default of zero continues the existing sequence or (if none) initializes the seed automatically. (Note that this version of the random number generator is shared with other design construction algorithms, such as FKEY.)

Printed output is controlled by the PRINT option, with settings:

summary	summarizes the design properties;
keyblocks	prints a design key to generate the block and sub-block factors from the treatment factor (or pseudo-factors to generate them if they have more than *p* levels);
keydefining	prints a design key specifying the defining contrasts i.e. all the treatment contrasts confounded with blocks or sub-blocks;
monitoring	prints monitoring information about the design construction.

You can save the design keys using the KEYBLOCKS and KEYDEFINING parameters. In addition, the NDESIGN parameter can save a unique "design number" for the design, and the NSUBDESIGN parameter can save a unique number for the sub-design of the design. You can input these with NDESIGN and NSUBDESIGN later, along with the same settings for NTREATMENTFACTORS, LEVELS, NUNITS and NSUBUNITS, to obtain the design keys without repeating the design search. The RESOLUTION and ABERRATION parameters can save the resolution and aberration of the (main) design, and the SUBRESOLUTION and SUBABERRATION parameters can save the resolution and aberration of a sub-design.

You can use the design keys to form the design using the GENERATE directive or the AFKEY procedure. Alternatively, you may prefer to use the AGFACTORIAL procedure (4.9.2), which combines a call to AFMINABERRATION with a program to form the factors and generate the design automatically.

5 REML analysis of mixed models

This chapter describes the facilities for analysis of linear mixed models, estimation of variance components and modelling of covariance structures using the method of *residual maximum likelihood* (REML), sometimes also known as *restricted* maximum likelihood.

The REML algorithm estimates the treatment effects and variance components in a linear mixed model: that is, a linear model with both fixed and random effects. Like regression, REML can be used to analyse unbalanced data sets; but, unlike regression, it can account for more than one source of variation in the data, providing an estimate of the variance components associated with the random terms in the model. You can also model the covariance structures of the random terms.

The REML method has many applications. It can be used to obtain information on sources and sizes of variability in data sets. This can be of interest where the relative size of different sources of variability must be assessed, for example to identify the least reliable stages in an industrial process, or to design more effective experiments. REML provides efficient estimates of treatment effects in unbalanced designs with more than one source of error. For example, it can be used to provide estimates of treatment effects that combine information from all the strata of an unbalanced design. It can also be used to combine information over similar experiments conducted at different times or in different places. So you can obtain estimates that make use of the information from all the experiments, as well as the separate estimates from each individual experiment. Finally its ability to model correlated error structures can be useful in a wide range of situations, including repeated measurements, spatial data and random coefficient regression.

The model for a REML analysis can be defined using the commands:

VCOMPONENTS	defines the model for REML (5.2.1)
VCYCLE	controls advanced aspects of the algorithm (5.3.10)
VSTRUCTURE	defines a variance structure for random effects in a REML model (5.4.1)
VPEDIGREE	generates an inverse relationship matrix for use in VSTRUCTURE when fitting animal or plant breeding models by REML (5.6.1)
VFPEDIGREE	checks and prepares pedigree information from several factors, for use by VPEDIGREE (5.6.2)
VRESIDUAL	defines the residual term for a REML model (5.8.2)
VRMETAMODEL	forms the random model for a REML meta analysis (5.8.1)
VSTATUS	prints the current model settings for REML (5.4.2)

The REML directive carries out the analysis, and a range of other directives and procedures are then available to save information in Genstat data structures, to produce further output or for other REML-based analyses:

REML	fits a variance-component model by residual (or restricted) maximum likelihood (5.3.1)
VDISPLAY	displays further output from a REML analysis (5.3.2)
VGRAPH	plots tables of means from a REML analysis (5.3.4)
VDEFFECTS	plots one- or two-way tables of effects estimated in a REML analysis (5.3.4)
VPLOT	plots residuals from a REML analysis (5.3.5)
VDFIELDRESIDUALS	display residuals from a REML analysis in field layout (5.3.5)
VBOOTSTRAP	performs a parametric bootstrap of the fixed effects in a REML analysis (5.3.6)

VCRITICAL	uses a parametric bootstrap to estimate critical values for a fixed term in a REML analysis (5.3.6)
VSCREEN	performs screening tests for fixed terms in a REML analysis (5.3.6)
VCHECK	checks standardized residuals from a REML analysis (5.3.7)
VRCHECK	checks effects of a random term in a REML analysis (5.3.7)
VSOM	analyses a simple REML variance components model for outliers using a variance shift outlier model (5.3.7)
VAIC	calculates the Akaike and Schwarz (Bayesian) information coefficients (5.3.8)
VRACCUMULATE	forms a summary accumulating the results of a sequence of REML random models (5.3.8)
VPREDICT	forms predictions from a REML model (5.5.1)
VTCOMPARISONS	calculates comparison contrasts within a multi-way table of predicted means from a REML analysis (5.5.2)
VMCOMPARISON	performs pairwise comparisons between REML means
VKEEP	copies information from a REML analysis into Genstat data structures (5.9.1)
VFRESIDUALS	obtains residuals, fitted values and their standard errors from a REML analysis (5.9.2)
VSPREADSHEET	saves results from a REML analysis in a spreadsheet (5.9.3)
VFIXEDTESTS	saves fixed tests from a REML analysis (5.9.4)
VALLSUBSETS	fits all subsets of the fixed terms in a REML analysis
VAYPARALLEL	does the same REML analysis for several y-variates, and collates the output
VFLC	performs an F-test of random effects in a linear mixed model based on linear combinations of the responses, i.e. an FLC test
VFUNCTION	calculates functions of variance components from a REML analysis
VHERITABILITY	calculates generalized heritability for a random term in a REML analysis
VPERMTEST	does random permutation tests for the fixed effects in a REML analysis (5.3.6)
VPOWER	uses a parametric bootstrap to estimate the power (probability of detection) for terms in a REML analysis
VRFIT	fits terms from a REML fixed model in a Genstat regression
VRADD	adds terms from a REML fixed model into a Genstat regression
VRDISPLAY	displays output for a REML fixed model fitted in a Genstat regression
VRDROP	drops terms in a REML fixed model from a Genstat regression
VRKEEP	saves output for a REML fixed model fitted in a Genstat regression
VRSETUP	sets up Genstat regression to assess terms from a REML fixed model
VRSWITCH	adds or drops terms from a REML fixed model in a Genstat regression
VRTRY	tries the effect of adding and dropping individual terms from a REML fixed model in a Genstat regression

VRPERMTEST	performs permutation tests for random terms in REML analysis
VSAMPLESIZE	estimates the replication to detect a fixed term or contrast in a REML analysis, using parametric bootstrap
VSURFACE	fits a 2-dimensional spline surface using REML, and estimates its extreme point
VUVCOVARIANCE	forms the unit-by-unit variance-covariance matrix for specified variance components in a REML model
FCONTRASTS	modifies a model formula to contain contrasts of factors
FDIALLEL	forms the components of a diallel model for REML or regression
F2DRESIDUALVARIOGRAM	calculates and plots a 2-dimensional variogram from a 2-dimensional array of residuals
TOBIT	linear mixed model analysis of data with fixed-threshold censoring

Procedures are being developed to to provide automatic selection of REML random models for single trials, series of trials and meta analysis.

VABLOCKDESIGN	analyses an incomplete-block design by REML, allowing automatic selection of random and spatial covariance models
VAROWCOLUMNDESIGN	analyses a row-and-column design by REML, with automatic selection of the best random and spatial covariance model
VALINEBYTESTER	provides combinabilities and deviances for a line-by-tester trial analysed by VABLOCKDESIGN or VAROWCOLUMNDESIGN
VLINEBYTESTER	analyses a line-by-tester trial by REML
VASERIES	analyses a series of trials with incomplete-block or row-and-column designs by REML, automatically selecting the best random models
VASDISPLAY	displays further output from an analysis by VASERIES
VASKEEP	copies information from an analysis by VASERIES into Genstat data structures
VASMEANS	saves experiment × treatment means from analysis of a series of trials by VASERIES
VAMETA	performs a REML meta analysis of a series of trials
VFMODEL	forms a model-definition structure for a REML analysis
VFSTRUCTURE	adds a covariance-structure definition to a REML model-definition structure
VMODEL	specifies the model for a REML analysis using a model-definition structure defined by VFMODEL
VAOPTIONS	defines options for the fitting of models by VARANDOM and associated procedures
VARANDOM	finds the best REML random model from a set of models defined by VFMODEL
VARECOVER	recovers when REML, is unable to fit a model, by simplifying the random model

There is also a suite of procedures that use REML to estimate QTLs from single environment, multi-environment and multi-trait trials:

DQMAP	displays a genetic map

DQMKSCORES	plots a grid of marker scores for genotypes and indicates missing data
DQMQTLSCAN	plots the results of a genome-wide scan for QTL effects in multi-environment trials
DQRECOMBINATIONS	plots a matrix of recombination frequencies between markers
DQSQTLSCAN	plots the results of a genome-wide scan for QTL effects in single-environment trials
GPREDICTION	produces genomic predictions (breeding values) using phenotypic and molecular marker information
QBESTGENOTYPES	sorts individuals of a segregating population by their genetic similarity with a defined target genotype, using the identity by descent (IBD) information at QTL positions for one or more traits
QCANDIDATES	selects QTLs on the basis of a test statistic profile along the genome
QDESCRIBE	prints summary statistics of genotypes
QEIGENANALYSIS	uses principal components analysis and the Tracy-Widom statistic to find the number of significant principal components to represent a set of variables
QEXPORT	exports genotypic data for QTL analysis
QFLAPJACK	creates a Flapjack project file from genotypic and phenotypic data
QGSELECT	obtains a representative selection of genotypes by means of genetic distance sampling or genetic distance optimization
QIBDPROBABILITIES	reads molecular marker data and calculates IBD probabilities
QIMPORT	imports genotypic and phenotypic data for QTL analysis
QKINSHIPMATRIX	forms a kinship matrix from molecular markers
QLDDECAY	estimates linkage disequilibrium (LD) decay along a chromosome
QLINKAGEGROUPS	forms linkage groups using marker data from experimental populations
QMAP	constructs genetic linkage maps using marker data from experimental populations
QMASSOCIATION	performs multi-environment marker trait association analysis in a genetically diverse population using bi-allelic and multi-allelic markers
QMATCH	matches different data structures to be used in QTL estimation
QMBACKSELECT	performs a QTL backward selection for loci in multi-environment trials or multiple populations
QMESTIMATE	calculates QTL effects in multi-environment trials or multiple populations
QMKDIAGNOSTICS	generates descriptive statistics and diagnostic plots of molecular marker data
QMKRECODE	recodes marker scores into separate alleles
QMKSELECT	obtains a representative selection of markers by means of genetic distance sampling or genetic distance optimization
QMQTLSCAN	performs a genome-wide scan for QTL effects (Simple and

	Composite Interval Mapping) in multi-environment trials or multiple populations
QMTBACKSELECT	performs a QTL backward selection for loci in multi-trait trials
QMTESTIMATE	calculates QTL effects in multi-trait trials
QMTQTLSCAN	performs a genome-wide scan for QTL effects (Simple and Composite Interval Mapping) in multi-trait trials
QMVAF	calculates percentage variance accounted for by QTL effects in a multi-environment analysis
QMVESTIMATE	replaces missing molecular marker scores using conditional genotypic probabilities
QMVREPLACE	replaces missing marker scores with the mode scores of the most similar genotypes
QRECOMBINATIONS	calculates the expected numbers of recombinations and the recombination frequencies between markers
QREPORT	creates an HTML report from QTL linkage or association analysis results
QSELECTIONINDEX	calculates (molecular) selection indexes by using phenotypic information and/or molecular scores of multiple traits
QSASSOCIATION	performs multi-environment marker trait association analysis in a genetically diverse population using bi-allelic and multi-allelic markers
QSBACKSELECT	performs a QTL backward selection for loci in single-environment trials
QSESTIMATE	calculates QTL effects in single-environment trials
QSIMULATE	simulates marker data and QTL effects for single and multiple environment trials
QSQTLSCAN	performs a genome-wide scan for QTL effects (Simple and Composite Interval Mapping) in single-environment trials
QTHRESHOLD	calculates a threshold to identify a significant QTL
VGESELECT	selects the best variance-covariance model for a set of environments

Section 5.1 introduces the linear mixed models fitted by REML, and describes the underlying methodology. Section 5.2 explains how these models are defined in Genstat using the VCOMPONENTS directive. Section 5.3 describes the REML directive and presents examples to show how to interpret the output of a mixed-models analysis. Section 5.4 describes the VSTRUCTURE directive, which allows you to model the variance structure of the data to cater for correlated random effects. Section 5.5.1 explains how to form predictions using the VPREDICT directive. Section 5.6 is relevant to the analysis of an animal or plant breeding experiment, describing the VPEDIGREE directive which generates a sparse inverse relationship matrix from a given pedigree for use in VSTRUCTURE. Section 5.7 describes how to generate cubic spline terms to be fitted as part of the random model. The smoothing parameter is estimated by REML and the fitted spline is interpreted as a BLUP (best linear unbiased predictor). Spline terms can be particularly useful for investigating non-linear profiles in repeated measurements data. Section 5.8 describes the use of the VRMETAMODEL procedure and VRESIDUAL directive for specifying meta-analyses combining data from several experiments. Finally, Section 5.9 explains how to use the VKEEP directive to copy results from an analysis into Genstat data structures, and the VSPREADSHEET procedure to save them in a spreadsheet.

This chapter corresponds to the Mixed Models (REML) menus in Genstat *for Windows*, and can provide guidance about the model specification for these menus as well as explanations of

the output. The Linear Mixed Models menu is the most general. The Y-Variate field of the menu corresponds to the Y parameter of REML (5.3.1), and the Fixed Model and Random Model fields correspond to the FIXED option and RANDOM parameter of VCOMPONENTS (5.2.1). The Spline Model field allows cubic smoothing splines to be specified, and the subsidiary Linear Mixed Models - Correlated Errors menu uses VSTRUCTURE to specify correlation models. Genstat *for Windows* also has several specialized menus for repeated measurements and spatial data.

5.1 Models for REML estimation

This section describes the linear mixed models that can be fitted using the REML algorithm in Genstat. The fixed and random parts of the model are discussed 5.1.1, before a formal description of the model is given in 5.1.2. Section 5.1.3 then explains the theory behind the residual maximum likelihood method.

5.1.1 Fixed and random effects

Fixed effects are used to describe treatments imposed in an experiment where it is the effect of those specific choices of treatment that are of interest. Random effects are generally used to describe the effects of factors where the values present in the experiment represent a random selection of the values in some larger homogeneous population. It is then possible to make some inference about this population, for example to estimate its variance and to assess the contribution from a factor to the total variation in the data. Predictions of random effects may also be of interest.

For example, consider the split-plot experiment of Section 4.2, used to assess the effects on yield of three oat varieties with four levels of nitrogen application. In this experiment, specific levels of nitrogen application have been used and the aim is to estimate the effects of these levels; so they would be considered as fixed effects in the model, as would the three oat varieties. However, the effects of the actual blocks and plots in the experiment are not of interest in themselves, but they do provide a means of estimating the variability of the more general population of blocks and plots in order to get an estimate of background variation against which to compare the fixed effects. Blocks and plots would therefore be defined as random effects. In this case, the fixed effects correspond to the effects used as treatments in ANOVA and the random effects would correspond to the blocking factors in ANOVA. The REML analysis of this example is shown in 5.3.1.

Another example (from Dempster *et al*. 1984) involves an experiment to assess the effect of an experimental compound on maternal performance (see 5.3.3). Twenty-seven female rats (dams) were treated with either a control substance or a high or low dose of an experimental compound in order to examine the effects on their litters. The experimental data were then the weights of each individual pup. The different treatments are specified as fixed effects. Since litter size and the sex of the pup influence weight, these factors must also be included, and as the effects of the specific values of these factors in the experiment are of interest, we define them as fixed effects. Further variation is introduced into the data from the effects of different dams. Since the dams could be considered as a random selection from a wider homogeneous population they are introduced to the model as a random effect. The effect of pups is clearly also a random effect. In fact, since the pups are the units of the experiment, the variation between pups is the error variance component (*units*).

The choice of fixed and random terms is not always determined by the structure of the experiment, but may depend on the information required. For example, variety trials are often carried out over different sites and in several years. If a general assessment of varieties over time is required, then the years present in the trial are considered as a random selection of years, and year would be defined as a random term in the model. On the other hand, if the effect of the specific years present in the trial was to be assessed, year would be defined as a fixed term.

Further discussion of the choice of fixed and random effects can be found in Snedecor & Cochran (1989) and Searle (1971).

In general, both the fixed and random parts of the model are constructed from several factors or variates. The structure of both parts is specified using model formulae, in the same way that models are specified for regression (3.3.1) or analysis of variance (4.1.1). The model for both the fixed and random parts can contain factors and variates and can use the usual crossing and nesting operators (5.2).

In the split-plot example, the fixed part of the model must include the main effects of oat variety and nitrogen application plus their interaction, and is specified as

```
Nitrogen*Variety
```

where `Nitrogen` is a factor indicating nitrogen application on each unit, and `Variety` is a factor indicating the variety grown on each unit. The random part of the model describes the nested blocking structure of subplots within whole-plots within blocks and is specified as

```
Block/Wplot/Subplot
```

where `Block` is a factor indicating which block contains each unit, `Wplot` is a factor indicating which whole-plot contains the unit within its block, and `Subplot` is a factor indicating which subplot contains the unit within its whole-plot (see 5.3.1).

Similarly, the fixed model in the rat reproductive study described above might be written as `Dose*Sex+Littersize` with random model `Dam/Pup` (5.3.3).

5.1.2 The linear mixed model with independent random effects

Returning to the split-plot example, the model for the yield y_{ijk} from block i, whole-plot j, subplot k is

$$y_{ijk} = m + v_r + a_s + va_{rs} + b_i + w_{ij} + \varepsilon_{ijk}$$

where the fixed part of the model consists of: m the overall constant; v_r the main effect of variety r (where r indicates the variety assigned to unit ijk); a_s the main effect of nitrogen application at level s (where s indicates the nitrogen application on unit ijk); and va_{rs} their interaction. The random model terms are b_i the effect of block i, w_{ij} the effect of whole-plot j within block i, and ε_{ijk} the random error for unit ijk (which here is the same as the subplot effect, since the subplots are the smallest units of the experiment).

This model can be re-written as a general linear mixed model by grouping the fixed and random terms and using matrix and vector notation:

$$y = X\alpha + Z\beta + \varepsilon$$

where

y is a vector of data (length n)
α is a vector of fixed effects (length p) with nxp design matrix X
β is a vector of random effects (length q) with nxq design matrix Z
ε is a vector of random error (length n).

In the split-plot example above, there are 72 units. The vector α contains the fixed effects m, v_1,v_2,v_3, a_1...a_4 and va_{11}...va_{34}. The rows of matrix X correspond to the units of the experiment and the columns correspond to the fixed effects. The values in each row of X are 1 or 0 to indicate presence or absence of each effect for that unit. Similarly, the vector β contains the random effects b_i (i=1...6) and w_{ij} (i=1...6; j=1...3) and matrix Z indicates which units occur within each block and whole-plot.

More generally, the random model $Z\beta$ is constructed from c model terms (in this example, it consists of the two random model terms `Block` and `Block.Wplot`). Z and β can then be partitioned as $Z = \{ Z_1 \mid Z_2 \mid ... \mid Z_c \}$ and $\beta = (\beta_1 \beta_2 ... \beta_c)'$ where β_i is a vector of length q_i. The model can then be written in terms of the separated random model terms as

$$y = X\alpha + \sum_{i=1}^{c} Z_i\beta_i + \varepsilon$$

It is assumed that the random effects β_i and ε are mutually independent Normally distributed random variables with zero mean, such that $Cov(\varepsilon)=\sigma^2 I_n$, where I_n is the identity matrix of size n, $Cov(\beta_i)=\sigma_i^2 I_{qi}$ where I_{qi} is an identity matrix of size q_i, and $Cov(\beta_i,\beta_j)=0$ for $i\neq j$. Therefore, effects that occur in different random model terms are independent. This means that the variance-covariance matrix for the whole set of random effects takes a particularly simple form, since $Cov(\beta)=diag\{\sigma_1^2 I_{q1},...\sigma_c^2 I_{qc}\}$ is diagonal. The variance parameters σ_i^2 associated with the random model terms are called the variance components of the model. The variance parameter σ^2 associated with the random error ε is called the residual variance (or the variance of the factor `*units*`). The REML algorithm estimates the variance components using residual maximum likelihood, and then uses the variance parameter estimates to form the generalized least squares estimates of the treatment effects and the best linear unbiased predictors (BLUPs) of the random effects.

The general linear model defined above has the properties

$$\begin{aligned}
\mathrm{E}(y) &= X\alpha \\
\mathrm{Cov}(y) &= V \\
&= Z\,\mathrm{Cov}(\beta)\,Z' + \sigma^2 I_n \\
&= \sum_i \sigma_i^2 Z_i Z_i' + \sigma^2 I_n \\
&= \sigma^2 \left(\sum_i \gamma_i Z_i Z_i' + I_n \right) \qquad \text{where } \gamma_i = \frac{\sigma_i^2}{\sigma^2} \\
&= \sigma^2 H.
\end{aligned}$$

where $H = Z\Gamma Z' + I_n$ and $\Gamma = \mathrm{diag}\{ \gamma_1 I_{q1} \ldots \gamma_c I_{qc} \}$.

The expected value of the data is a function of the fixed terms alone, and its variance-covariance matrix can be expressed either as a function of the variance components $\{ \sigma_i^2 ; i=1...c \}$ or as a function of σ^2 and the set $\{ \gamma_i ; i=1...c \}$ which are ratios of the variance components to σ^2, the residual variance, and are called the "gammas". When the model is defined solely in terms of its expectation and variance-covariance matrix, the components can be interpreted as constituent parts of the variance-covariance matrix. Therefore, so long as the variance-covariance matrix of the data remains positive definite overall, there is no constraint on the individual variance components to remain positive.

Although the random effects are assumed to be independent here, this model leads directly to a correlated variance structure V for the data. Units of the data vector y will be correlated if they share the same effect of a random term and, assuming all variance components are positive, this correlation will increase as the number of common random effects increases. For example, in the split-plot experiment above, the variance of the data is

$$\mathrm{Var}(y_{ijk}) = \sigma_b^2 + \sigma_w^2 + \sigma^2$$

where σ_b^2 and σ_w^2 are the variance components for blocks and whole-plots respectively. Covariances within blocks and whole-plots are then

$$\mathrm{Cov}(y_{ijk}, y_{iml}) = \sigma_b^2$$
$$\mathrm{Cov}(y_{ijk}, y_{ijl}) = \sigma_b^2 + \sigma_w^2$$

So correlation is higher for two plots within the same whole-plot than for two plots in the same block (but different whole-plots). This is known as a *uniform* or *compound symmetry* variance

structure. Other variance models can be imposed by using the assumption

$$\mathrm{Var}(\beta_i) = G_i$$

for some symmetric matrix G_i. Sections 5.4 onwards show to fit these models.

5.1.3 REML estimation

The method of residual maximum likelihood (REML) was introduced by Patterson & Thompson (1971). It was developed in order to avoid the biased variance component estimates that are produced by ordinary maximum likelihood estimation: because maximum likelihood estimates of variance components take no account of the degrees of freedom used in estimating treatment effects, they have a downwards bias which increases with the number of fixed effects in the model. This in turn leads to under-estimates of standard errors for fixed effects, which may lead to incorrect inferences being drawn from the data. Estimates of variance parameters which take account of the degrees of freedom used in estimating fixed effects, like those generated by `ANOVA` in balanced data sets, are more desirable.

The REML method splits the data into two parts: treatment contrasts which contain information only on the fixed effects; and error contrasts (that is, all contrasts with zero expectation) which contain information on the variance components. The error contrasts alone are then used to estimate the variance parameters, since they contain all of the information available on the variance parameters. This is done by projecting the data into the residual space: the vector space of error contrasts, where all the data contrasts have zero expectation. The projected data has log-likelihood RL where

$$-2RL(y) = (n-p^*)\log 2\pi - \log|X'X| + \log|V| + \log|X'V^{-1}X| + (y-X\hat{\alpha})'V^{-1}(y-X\hat{\alpha})$$

with n as the number of data values and p^* as the number of degrees of freedom used in estimating fixed effects; that is, rank(X). Variance components are then estimated by maximizing the log-likelihood function RL of the projected data.

The log-likelihood of the original data is L, where

$$-2L(y) = n\log 2\pi + \log|V| + (y-X\alpha)'V^{-1}(y-X\alpha).$$

Compared with the usual log-likelihood L, the log-likelihood of the residual data, RL, contains several extra terms. The only extra term involving the variance components (which is therefore the only extra term used in estimating the variance components) is $\log|X'V^{-1}X|$ which effectively removes the degrees of freedom used in estimating the fixed effects.

To take the simplest example, the maximum likelihood estimate of the variance of a set of n observations y_i from the same population would be $\Sigma(y_i-\bar{y})^2/n$ which has expectation $(n-1)\sigma^2/n$, whereas the more usual unbiased (REML) estimate is $\Sigma(y_i-\bar{y})^2/(n-1)$.

Similarly, in an orthogonal design, the REML estimates of the variance components are identical to the unbiased estimates that can be produced from residual mean squares in the analysis of variance. However, REML can also be used with unbalanced data to produce estimates of variance components that do not suffer the downward bias associated with maximum-likelihood estimation.

Once the variance components have been estimated, they are used to construct an estimate of the variance-covariance matrix, $\hat{V}$. The fixed effects are then estimated by generalized least squares

$$\hat{\alpha} = (X'\hat{V}^{-1}X)^{-1}X'\hat{V}^{-1}y \quad \text{with} \quad \mathrm{Var}(\hat{\alpha}) = (X'\hat{V}^{-1}X)^{-1}.$$

Predictions of the random effects are given by the best linear unbiased predictors (BLUPs)

$$\hat{\beta} = (Z'Z+\Gamma^{-1})^{-1}Z'(y-X\hat{\alpha}).$$

5.2 Specifying linear mixed models

The VCOMPONENTS directive sets up the linear mixed model to be analysed by REML similarly to the way in which the TREATMENTSTRUCTURE and BLOCKSTRUCTURE directives set up the model for ANOVA (Chapter 4). This section first summarises the syntax of VCOMPONENTS (5.2.1). It then describes the parameterization of the fixed model (5.2.2), shows how the random model is defined (5.2.3) and explains how to define initial values or set constraints on the variance components (5.2.4).

5.2.1 The VCOMPONENTS directive

VCOMPONENTS directive

Defines the variance-components model for REML.

Options

FIXED = *formula*	Fixed model terms; default *
ABSORB = *factor*	Defines the absorbing factor (appropriate only when REML option METHOD=Fisher); default * i.e. none
CONSTANT = *string token*	How to treat the constant term (estimate, omit); default esti
FACTORIAL = *scalar*	Limit on the number of factors or covariates in each fixed term; default 3
CADJUST = *string token*	What adjustment to make to covariates before analysis (mean, none); default mean
RELATIONSHIP = *matrix*	Defines relationships constraining the values of the components; default *
SPLINE = *formula*	Defines random cubic spline terms to be generated: each term must contain only one variate, if there is more than one factor in a term, separate splines are calculated for each combination of levels of the factors
EXPERIMENTS = *factor*	Factor defining the different experiments in a multi-experiment (meta-) analysis

Parameters

RANDOM = *formula*	Random model terms
INITIAL = *scalars*	Initial values for each component and the residual variance
CONSTRAINTS = *string tokens*	How to constrain each variance component and the residual variance (none, positive, fixrelative, fixabsolute); default none

The VCOMPONENTS directive specifies the linear mixed model to be fitted by subsequent REML statements. The fixed terms in the model are defined by a model formula supplied using the FIXED option, and the random model terms are defined by a model formula supplied by the RANDOM parameter. Thus, for example, the model for the split-plot experiment described in 5.1.1 would be specified by

```
VCOMPONENTS [FIXED=Nitrogen*Variety] \
  RANDOM=Block/Wplot/Subplot
```

where Nitrogen and Variety are factors indicating the treatments applied to each unit, and Block, Wplot and Subplot are factors indicating the block, whole-plot (within block) and subplot (within whole-plot) to which each unit belongs; see Example 5.3.1.

The model for the rat reproduction experiment would be

```
VCOMPONENTS [FIXED=Dose*Sex+Littersize] RANDOM=Dam/Pup
```

In this case, each pup is a separate unit. The analysis of this experiment is shown in 5.3.3.

If you do not specify the fixed model, the default fixed model consists of just the constant term, which then becomes the grand mean. If the random model is unset, only a single source of variation (the residual component) is used. In this case, REML will produce the same analysis as the regression facilities which, since they take full advantage of the simple variance structure of the model, would be computationally more efficient. Note that any model term found in both the fixed and the random model will be deleted from the random model and retained in the fixed model only. A complete definition of the operators available in model formulae is given in 4.1.1.

By default, it is assumed that within each random model term the variance-covariance matrix takes the form $\sigma_i^2 I$, that is, equal variation and no correlation between different levels of the factor. You can use the VSTRUCTURE directive, described in 5.4.1, to define other variance structures.

As well as defining the basic model using the FIXED option and RANDOM parameter, the VCOMPONENTS directive can also be used to modify the model or to add extra information.

A constant term is automatically included in the fixed part of the model but this can be omitted by setting option CONSTANT=omit, provided you have also specified a fixed model.

You can supply initial values for the gamma ratios of the variance components using the INITIAL parameter, and you can impose constraints on the gamma parameters using either the RELATIONSHIP option or the CONSTRAINTS parameter. The CONSTRAINTS parameter allows you to request that any gamma parameter should be held positive or fixed at its initial value. The default setting, none, allows the variance components to become negative, provided the overall estimated variance-covariance matrix for the data remains positive definite. The RELATIONSHIP option can be used to define linear relationships between the variance components, for example that component A should be constrained to be twice component B. Full details are given in 5.2.4.

The ABSORB option allows you to specify a factor from either the fixed or the random model to act as an absorbing factor for the model, when the METHOD option of REML is set to Fisher. The absorbing factor is used to divide the model terms into two groups; this partition is then used in calculations during the fitting process to reduce the size of the matrices that have to be inverted and stored. Use of an absorbing factor can therefore save computing time and data space. However, although exactly the same model is fitted when an absorbing factor is used, some of the standard errors are unavailable (see 5.3.3, 5.3.9). A good choice of absorbing factor might be a factor with a large number of levels, or any factor whose effects and standard errors are not of interest. The choice of an absorbing factor is considered in detail in 5.3.9. Absorbing factors are irrelevant with the AI method of REML, as this uses sparse-matrix methods which are very economical with data space (Gilmour *et al*. 1995).

The SPLINE option, described in more detail in Section 5.7, allows cubic smoothing splines to be defined for inclusion in the random model.

If an EXPERIMENTS factor is specified, a different residual variance will be estimated for each factor level. The VRESIDUAL directive (Section 5.8.2) can be used to specify more complex variance models at each site.

5.2.2 The fixed model

You define the fixed terms to be included in the linear mixed model using the FIXED option of VCOMPONENTS. The model formula that you specify can include both factors and variates.

Factors are used, as in regression and analysis of variance, to represent qualitative effects. Consider a simple example where a factor Dose might be used to describe the effect of different doses (none, low or high). This would be specified using the FIXED option of VCOMPONENTS

```
VCOMPONENTS [FIXED=Dose]
```

and would lead to the model

$$y_{ij} = a + b_i + \varepsilon_{ij}$$

where y_{ij} is the data value for unit j which received dose i, a is the overall constant, and the parameters b_i describe the effects of the different doses. As in regression models, unless the constant term is omitted, some form of constraint is needed to avoid over-parameterization. For model terms containing only factors, the parameters corresponding to the first levels of the factors are constrained to be zero, as in Genstat regression (4.3.2). The parameters for other levels of the factor are then comparisons with the first level. Here, for example, b_1 (`Dose=none`) would be set to zero, and parameter b_2 (`Dose=low`) would estimate the difference between the low dose and no dose. Similarly, b_3 would estimate the difference between high dose and no dose. Note that the parameter a is not the grand mean, since it contains both the grand mean and the effect of the first level of factor `Dose`; it is an estimate of the expected value of y when the first level of the factor is applied. The parameters b_i are then the adjustments to be added to a to estimate the expected value for the other levels of `Dose`.

Also, whenever a parameter is found to be completely aliased with parameters fitted earlier in the model, the aliased parameter is set to zero.

Variates can be included in the model to represent a linear relationship between the y-variate and a covariate. By default covariates are centred so, for example, if a variate `X` is added to the `FIXED` model above

```
VCOMPONENTS [FIXED=Dose+X]
```

the model to be fitted becomes

$$y_{ij} = a + b_i + c \times (x_{ij} - \bar{x}) + \varepsilon_{ij}$$

where x_{ij} is the value of the covariate for unit ij, $\bar{x}$ is the mean of the covariate (weighted when appropriate) and c estimates the slope (or regression coefficient) of the linear relationship between the expected value of y and the covariate `X`. The parameter a is a constant term representing the grand mean plus the effect of dose level 1 (none) plus an adjustment for the covariate mean i.e. a is the intercept for dose level 1 plus $c\bar{x}$; b_2 (or b_3) represent the difference in intercept between dose levels 2 (or 3) and level 1.

When interactions of factors and variates are included in the model, terms are added to fit a different regression coefficient for each level of the factor. If the interaction between `X` and `Dose` is added to our example

```
VCOMPONENTS [FIXED=Dose*X]
```

the model becomes

$$y_{ij} = a + b_i + c \times (x_{ij} - \bar{x}) + d_i \times (x_{ij} - \bar{x}) + e_{ij}$$

Again, constraints are required to avoid over-parameterization: parameter d_1 is constrained to be zero, so that d_2 (or d_3) represents the difference in slope between dose levels 2 (or 3) and level 1.

You can request that uncentred covariates are used by setting option `CADJUST=none`. The fitted model becomes

$$y_{ij} = a + b_i + c\, x_{ij} + d_i x_{ij} + e_{ij}$$

and the constant term a now represents the grand mean plus the intercept for dose level 1. The covariates must either all be centred or all remain unadjusted. One way of centring some covariates but not others is to centre the desired covariates before analysis using `CALCULATE` and then proceed with `CADJUST=none`. With the default setting, `CADJUST=mean`, tables of predicted means will be produced at the mean covariate values. When `CADJUST=none`, predictions will be produced at zero values of each covariate. The amount by which each covariate is adjusted (by `REML`) can be obtained using the `CADJUSTMENT` parameter of `VKEEP` (see 5.9.1).

When the `VCOMPONENTS` option setting `CONSTANT=omit` is used, the same parameterization convention is used, with the exception that the first fixed model term containing only factors (if such a term is present) will have no constraint imposed.

You can set a limit on the number of factors and variates allowed within each term of the fixed model using the FACTORIAL option of the REML directive (see 5.3.1).

5.2.3 The random model

The model formula for the random part of the model is specified using the RANDOM parameter of the VCOMPONENTS directive and can also include both factors and variates. Each random model term defines a set of random effects and an associated variance component. For example, the nested block structure of the split-plot experiment is specified by

```
VCOMPONENTS [FIXED=Nitrogen*Variety] \
  RANDOM=Block/Wplot/Subplot
```

so three variance components are included in the model representing the variation due to the blocks, the whole-plots and the subplots.

The random term which corresponds to the residual variation between units, or error variance σ^2, is called the residual component and is considered separately from the rest of the random terms within the algorithm. If the residual component is not specified, it is automatically added onto the end of the random model. Here the Block.Wplot.Subplot term is the residual component since it represents the variation between units at the lowest level of the experiment; that is, the subplots. The same model could have been specified as

```
VCOMPONENTS [FIXED=Nitrogen*Variety] RANDOM=Block/Wplot
```

and the residual component would have been added automatically. Genstat would then refer to it as *units*.

If the model is viewed in terms of random effects, then for a random model term specified by factors, an effect is included for each combination of the levels of the factors. For example, the random model in the split-plot example is

$$y_{ijk} = \{\text{fixed model terms}\} + b_i + w_{ij} + \varepsilon_{ijk}$$

with one parameter b_i (i=1...6) for each of the blocks and one parameter w_{ij} (i=1...6, j=1...3) for each whole-plot within each block. To avoid over-parameterization, the random effects are constrained so that their sum is zero within each model term. When variance components are estimated as negative values, standard errors are not available for the effects of the corresponding random terms.

You can also include variates in the RANDOM formula to specify random covariates. This may be useful, for example, in specifying models where a linear response to an explanatory variable varies randomly between groups or individuals. Note that covariance between the intercept and a random slope is usually required to give a sensible model, as in random coefficient regression (see Section 5.4.5). Some rescaling of each covariate may be required – if the covariate values are either very small or very large – in order to bring the estimated variance component to a reasonable value. Currently the minimum value allowed is $0.001 \times \sigma^2$.

In general, care must be taken not to specify the residual component more than once, unless some form of constraint is imposed, or unless one of these is the subject of a variance model (specified by VSTRUCTURE) – see Example 5.4.4c. Otherwise no estimation will be possible.

In a very few cases, you may wish to add the residual component onto the end of the random model even though it has already been specified. For example, in some algorithms for fitting generalized linear mixed models, it is necessary to estimate the residual component on the linear predictor scale whilst fixing the variance parameter on the natural scale. You can tell Genstat to add an `extra' residual component to the model by using the string '*units*' at the end of the random model. For example

```
VCOMPONENTS Block/Plot+'*units*'; CONSTRAIN=none,none,fix;\
  INITIAL=1,1,2
REML [WEIGHTS=W] Y
```

will produce the variance structure

$$\sigma_b^2 Z_b I_b Z_b' + \sigma_p^2 I_n + 2\ \text{diag}\{ w_i ; i=1...n \}.$$

This facility should rarely be needed and should be used with care.

5.2.4 Setting initial values and constraints on variance components

Computing time can be saved by specifying good initial values for some, or all, of the variance parameters. This is especially helpful if the data set is large or many model terms are to be fitted. The initial values are specified using the `INITIAL` parameter of the `VCOMPONENTS` directive. The values run in parallel with the expanded form of the `RANDOM` model, which follows the rules given in 4.1.1. The initial values for all except the residual component should be specified in terms of the gamma ratios defined in 5.2.1, that is, as the ratios of the variance components to the error variance. For the residual component, an initial estimate of the residual variance component itself should be supplied.

For example, the split-plot model

```
Block/Wplot/Subplot
```

expands to

```
Block + Block.Wplot + Block.Wplot.Subplot
```

and would require three initial values: e.g.

```
VCOMPONENTS RANDOM=Block/Wplot/Subplot; INITIAL=7,3,1
```

The random model

```
Row*Column
```

would expand to

```
Row + Column + Row.Column
```

and would also require three initial values: e.g.

```
VCOMPONENTS RANDOM=Row*Column; INITIAL=5,8,20
```

As usual, the list of initial values is recycled if it is shorter than the list of terms in the `RANDOM` model formula. You must remember that the residual component will be added onto the end of the random model (unless it is specified explicitly) and so you must give an initial value for the error variance at the end of the list. For example, if the random model for the split-plot experiment is specified as `Block/Wplot`, then the residual component will be added onto the end of the random model to give three terms in total, so three initial values must be specified

```
VCOMPONENTS RANDOM=Block/Wplot; INITIAL=7,3,1
```

Any gamma for which no initial value is available should be given an initial value of 1, which is the default when no initial values are specified.

By default, the estimates of variance components are allowed to take any non-zero value (positive or negative) such that the variance-covariance matrix of the data (V) remains positive definite. However, you may sometimes wish to constrain the components to remain positive. This can be done by setting parameter `CONSTRAINTS=positive`. You can give a list of strings to specify different constraints for each term in the random model. These again run in parallel with the expanded form of the random model (plus residual component if necessary), and will be recycled if the list is too short. For example,

```
VCOMPONENTS RANDOM=Block/Wplot; CONSTRAINTS=positive
```

would constrain all estimates of components to be positive in the split-plot example. If the `Block` component alone was to be held positive, the command would be

```
VCOMPONENTS RANDOM=Block/Wplot; \
  CONSTRAINTS=positive,none,positive
```

The constraints can be relaxed again to allow negative components by setting `CONSTRAINTS=none`, which is the default.

If the value of a gamma or a variance component is sufficiently well known for there to be no need for further estimation, it can be fixed at its initial value. You can fix the gamma (that is, the

ratio of the variance component to the error variance) for a model term by setting CONSTRAINTS=fixrelative (=fix for short). For example, the command

```
VCOMPONENTS Row*Column; INITIAL=5,8,20; \
  CONSTRAINTS=none,fix,none
```

means that the gamma for the second component will be fixed at 8, that is, the Column component will be estimated by $8\sigma^2$.

When the METHOD option of REML is set to Fisher, you can also fix the absolute value of the variance component at its initial value by setting CONSTRAINTS=fixabsolute. You must then specify the value of the component (not the gamma ratio) in the list of initial values. Thus the command

```
VCOMPONENTS Row*Column; INITIAL=5,8,20; \
  CONSTRAINTS=none,fixabs,none
```

means that the final estimated value of the Column component will be 8.

Note that components that are constrained to be fixed (relative or absolute) at their initial values do not appear in the output as estimated variance components although they are included in the model. Components fixed at zero will be reset to $10^{-3}\sigma^2$ with a warning.

Constraints on the residual component are treated slightly differently to those on the other components. Clearly, the error variance cannot be allowed to become negative, so the default constraint is that the error variance remains positive. The error variance can be fixed at its initial value, using either the fixabsolute or fixrelative setting. Note that a single parameter setting CONSTRAINTS=fix will be recycled, to fix all the gamma ratios and the residual component at their initial values.

None of these variance parameters are allowed to become zero. No estimation can take place if the error variance is zero, which may happen because the fixed model contains as many parameters as data values. When the METHOD option of REML is set to Fisher, any random term that is found to be completely aliased with other model terms (so that it cannot be estimated) will be deleted automatically from the random model, and the analysis will be rerun. If any of the gammas becomes very close to zero for any other reason, it will be reset to a small positive value ($10^{-3}\sigma^2$); REML generates a warning diagnostic if this has to be done repeatedly.

If components that have been constrained to be positive are estimated to be negative, they will also be reset to a small positive value ($10^{-3}\sigma^2$). If a component remains negative when the algorithm converges, a warning will be given since the constrained component is being held at an artificial value and may bias other estimates. In this case, it may be wise to estimate the value of the component without constraint to investigate whether the component is effectively zero (see Section 5.3.3) or whether it takes a relatively large negative value, which may indicate some unexpected structure in the variability of the data. Omission of an important term from the fixed model can lead to unexpected negative components, so the structure of the data should also be checked in order to detect any missing terms in the fixed model. Constraining components to be positive does save some data space which may be useful for very large problems.

You can also apply linear equality constraints between the variance components. These are defined by a matrix which is supplied using the RELATIONSHIP option of VCOMPONENTS. The matrix must be square, with one row and one column for each component (including the residual component, even if this is not specified explicitly in the random model). The entries in each row of the matrix define the constraints on the component corresponding to that row, in terms of multiples of the other components.

For example, consider the random model R*C, and suppose we wish to constrain the component for R to be twice the component for C, that is $\sigma_R^2=2\sigma_C^2$. This random model has 3 terms, therefore we need a 3×3 matrix. The rows and columns of the matrix correspond to the terms of the expanded model R+C+R.C in order. The first row is used to define constraints on the component for the first random model term, which is the R component. Since this is to be constrained to be twice the C component, the values for this row are 2 for the column

corresponding to the C component (the second model term and therefore the second column) and zero elsewhere. The second row is used to define constraints on the second component, C. Since this component is unconstrained, the row has value 1 for the C component (second column), and zeros elsewhere, that is, the C component is constrained to be itself. Similarly R.C, the residual component, is unconstrained and the final row has zeros except for value 1 in the R.C (the third) column. The statements to define this model would then be

```
MATRIX [ROWS=3; COLUMNS=3; VALUES=0,2,0, 0,1,0, 0,0,1] M
VCOMPONENTS [FIXED=F; RELATIONSHIP=M] R*C
```

giving

component	R	C	R.C	Constraint
R	0	2	0	$\sigma_R^2=2\sigma_C^2$
C	0	1	0	none
R.C	0	0	1	none

In this case, since the residual component is specified in the model as R.C, there is no need to add an extra row and column to the matrix. However, if the same model had been specified as R+C, a third row and column would still have been needed in the matrix to correspond to the residual component.

If a component is defined to be a multiple of the residual component, it will be treated as if it had been constrained fixed using parameter CONSTRAINTS=fix and will not appear in the list of estimated variance components.

5.3 Analysing linear mixed models

This section explains how to fit and display output from the analysis of a linear mixed model. The REML directive is described in 5.3.1, the VDISPLAY directive in 5.3.2, and the VPLOT procedure in 5.3.5. The split-plot design, already analysed in Section 4.2, is used in the first example (5.3.1 and 5.3.2). This illustrates that REML produces the same results as ANOVA for balanced designs, although the results are presented slightly differently. As with ANOVA, tables of means and effects are available for fixed model terms. REML also provides these tables for random model terms (see 5.3.3).

In general, REML analyses have two purposes: to study fixed effects when there are several sources of variability, and to estimate the variance components and assess the relative importance of the sources of variability. To some extent, of course, most analyses will involve both purposes, but for clarity we look at the two situations separately. Section 5.3.6 illustrates the output from a REML analysis to study the fixed effects in the rat reproduction experiment. This is an unbalanced data set where there is more than one source of variation in the data. REML estimation in this situation is more appropriate than a linear regression analysis since it makes use of all the available information on the fixed effects. Section 5.3.8 illustrates the output available for assessing the structure of the variability in the data from a factory production process. In this situation REML provides estimates of the variance components and a formal assessment of the random model.

5.3.1 The REML directive

Once you have defined a variance components model using VCOMPONENTS, you can then fit the model to the data (the y-variates) using the REML directive.

REML directive

Fits a variance-components model by residual (or restricted) maximum likelihood.

Options

PRINT = *string tokens*	What output to present (model, components, effects, means, stratumvariances, monitoring, vcovariance, deviance, Waldtests, missingvalues, covariancemodels); default mode, comp, Wald, cova
PTERMS = *formula*	Terms (fixed or random) for which effects or means are to be printed; default * implies all the fixed terms
PSE = *string token*	Standard errors to be printed with tables of effects and means (differences, estimates, alldifferences, allestimates, none); default diff
WEIGHTS = *variate*	Weights for the analysis; default * implies all weights 1
MVINCLUDE = *string tokens*	Whether to include units with missing values in the explanatory factors and variates and/or the y-variates (explanatory, yvariate); default * i.e. omit units with missing values in either explanatory factors or variates or y-variates
SUBMODEL = *formula*	Defines a submodel of the fixed model to be assessed against the full model (for METHOD=Fisher only)
RECYCLE = *string token*	Whether to reuse the results from the estimation when printing or assessing a submodel (yes, no); default no
RMETHOD = *string token*	Which random terms to use when calculating RESIDUALS (final, all, notspline); default fina
METHOD = *string token*	Indicates whether to use the standard Fisher-scoring algorithm or the new AI algorithm with sparse matrix methods (Fisher, AI); default AI
MAXCYCLE = *scalar*	Limit on the number of iterations; default 30
TOLERANCES = *variate*	Tolerances for matrix inversion; default * i.e. appropriate default values
PARAMETERIZATION = *string token*	Parameterization to use for the variance component estimation (gammas, sigmas) default * i.e. use whichever is most appropriate for model
CFORMAT = *string token*	Whether printed output for covariance models gives the variance matrices or the parameters (variancematrices, parameters); default vari
FMETHOD = *string token*	Controls whether and how to calculate F-statistics for fixed terms (automatic, none, algebraic, numerical); default auto
WORKSPACE = *scalar*	Number of blocks of internal memory to be allocated for use by the estimation algorithm when METHOD=AI

Parameters

Y = *variates*	Variates to be analysed

RESIDUALS = *variates*	Residuals from each analysis
FITTEDVALUES = *variates*	Fitted values from each analysis
EXIT = *scalar*	Exit status of the fit (0 if successful)
SAVE = *REML save structures*	Saves the details of each analysis for use in subsequent VDISPLAY and VKEEP directives

The REML directive performs the analysis, allowing control over the estimation process and the output that is produced. Some advanced aspects of the estimation process can be controlled using the VCYCLE directive (5.3.10), but these very rarely need to be changed.

The first parameter, Y, lists the variates that are to be modelled. You can restrict any of the y-variates or any of the factors or variates in the fixed and random models to indicate that only a subset of the units are to be used in the analysis (see 1:4.4.1). If more than one of these vectors is restricted, they must all be restricted to the same set of units.

The parameters FITTEDVALUES and RESIDUALS allow you to store the fitted values and residuals from the fitted model. The EXIT parameter saves the "exit status" of each analysis. This is set to zero if it was completed successfully; for details of the other codes, see Section 5.9.1. Parameter SAVE can be used to name the REML save structure for use with later VKEEP and VDISPLAY directives.

The three options PRINT, PTERMS and PSE all control the printed output. The PRINT option selects the output to be displayed. The different settings are explained in detail in different sections of this chapter, as indicated below:

model	description of model fitted (5.3.1)
components	estimates of variance components (5.3.5)
effects	tables of effects; that is, estimates of parameters α and β (5.3.6)
means	tables of means; that is, predicted means for factor combinations (5.3.2)
stratumvariances	approximate stratum variances from a decomposition of the information matrix for the variance components (available only for METHOD=Fisher; see 5.3.2)
monitoring	monitoring information at each iteration (5.4)
vcovariance	variance-covariance matrix of the estimated components (5.4.4)
deviance	deviance of the fitted model ($-2 \times$ log-likelihood *RL*) plus deviance of submodel if specified (5.4.3 and 5.4.4)
waldtests	Wald tests for all fixed terms in model (5.3.6)
missingvalue	estimates of missing values (relevant when option MVINCLUDE=yvariate)
covariancemodels	estimated covariance models (in the format requested by the CFORMAT option; see Section 5.4.4)

The default setting consists of model, components, waldtests and covariancemodels. Options PTERMS and PSE control the tables of means and effects that are printed, and their accompanying standard errors (see 5.3.2). The FMETHOD option controls whether to accompany the Wald tests for fixed effects with approximate F statistics and corresponding numbers of residual degrees of freedom (see 5.3.6).

The FACTORIAL option is used to set a limit on the number of factors and variates allowed in each fixed term; any term containing more than that number is deleted from the model.

The MVINCLUDE option allows the inclusion of units with missing values. By default, units where there is a missing value in the y-variate or in any of the factors or variates in the model terms are excluded. The setting explanatory allows units with missing values in factors or variates in the model to be included. For missing covariate values, this is equivalent to

substituting the mean value. The setting `yvariate` includes units with missing values in the y-variate. This can be useful to retain the balanced structure of the data for use with direct product covariance matrices (see `VSTRUCTURE`, Section 5.4.1), or to produce predictions of data values for given values of explanatory factors and/or variates.

The `WEIGHTS` option can be used to specify a weight for each unit in the analysis. This is useful when it is suspected that the size of the random error varies between units. For example, if the random error for unit i is known to have variance $v_i\sigma^2$, a weight variate should be used containing values $w_i=1/v_i$.

Option `SUBMODEL` is used to specify a sub-model of the fixed model (but only applies when `METHOD=Fisher`). This model will be fitted as well as the full fixed model, using a slightly modified version of the algorithm, and the difference in deviances between the full and sub-model can be used as a likelihood-based test to assess the importance of the fixed terms dropped from the full model. This is explained in detail in 5.3.6. Once the full model has been fitted, the `RECYCLE` option can be used to test a series of sub-models of the fixed model. If option `RECYCLE=yes` is set, then only the estimation for the sub-model is performed. Information for the full fixed model is picked up from the corresponding save structure. When the `RECYCLE` option is set, only the `deviance` and `model` settings of `PRINT` can be used. Note that the change in deviance will not be printed unless the setting `PRINT=deviance` is used.

The `RMETHOD` option controls the way in which residuals and fitted values are formed. For the default setting `RMETHOD=final`, the fitted values $\hat{y}$ are calculated from all the fixed and random effects: $\hat{y} = X\hat{\alpha} + Z\hat{\beta}$. The residuals are the difference between the data and the fitted values and, in this case, are estimates of the values of ε, the `*units*` random error. These residuals can be used to check the Normality and variance homogeneity assumptions for the random error. To get fitted values constructed from the fixed terms alone, omitting all random terms, the setting `RMETHOD=all` must be used. The fitted values are then $\hat{y} = X\hat{\alpha}$, and the residuals are predictors of $Z\beta+\varepsilon$. The setting `RMETHOD=notspline` means that the residuals will be formed from all the random effects, excluding spline terms (see 5.7). Procedure `VPLOT` (5.3.5) can also be used to produce various diagnostic plots.

The `METHOD` option specifies whether to use the `AI`(Average Information) algorithm (Gilmour *et al*. 1995) with sparse matrix methods to maximize the residual likelihood, or Fisher scoring with full matrix manipulation. By default the sparse Average Information algorithm is used, and it will also be used (regardless of the setting of `METHOD`) if covariance models are specified by `VSTRUCTURE` or if the `EXPERIMENTS` option of `VCOMPONENTS` is set to indicate a multi-experiment analysis. The AI algorithm generally runs faster per iteration than Fisher scoring and uses much less workspace, but it may require slightly more iterations to reach convergence. When sparse matrix methods are used, standard errors of differences will not be available for random effects, although standard errors are available. Note that when `METHOD=AI`, the `SUBMODEL` and `RECYCLE` options do not apply.

The `TOLERANCES` option controls the tolerances for matrix inversion. Three values can be specified in a variate. The first two values are matrix inversion tolerances for the information matrix and the mixed model equations respectively and take the value 10^{-5} by default. The third value is used to detect zero frequency counts for factor combinations in the mixed model equations: 10^{-6} is used by default.

Option `MAXCYCLE` can be used to change the maximum number of iterations performed by the algorithm from the default of 30.

The `PARAMETERIZATION` option allows you to control whether the variance model is parameterized in terms of the gamma ratios defined in 5.1.2 (that is as ratios of variance components to the error variance), or whether the variance components themselves are used. By default, `REML` attempts to select the parameterization automatically to suit the model to be fitted.

The `WORKSPACE` option specifies the number of blocks of internal memory to be allocated for use by the estimation algorithm when `METHOD=AI`. If this is not set, `REML` sets the number

automatically according to the complexity of the model to be fitted.

Example 5.3.1 shows how to analyse the split-plot design from Section 4.2.1 using REML.

Example 5.3.1

```
  2  " Split-plot design (Yates 1937, p.74; also John 1971, p.99)."
  3  UNITS [NVALUES=72]
  4  FACTOR [LEVELS=6] Blocks
  5  & [LEVELS=3] Wplots
  6  & [LEVELS=4] Subplots
  7  GENERATE Blocks,Wplots,Subplots
  8  FACTOR [LABELS=!T(Victory,'Golden rain',Marvellous)] Variety
  9  & [LABELS=!T('0 cwt','0.2 cwt','0.4 cwt','0.6 cwt')] Nitrogen
 10  VARIATE Yield; EXTRA=' of oats'
 11  READ [SERIAL=yes] Nitrogen,Variety,Yield

   Identifier   Minimum      Mean   Maximum    Values   Missing
        Yield     53.00     104.0     174.0        72         0

   Identifier    Values   Missing    Levels
     Nitrogen        72         0         4
      Variety        72         0         3

 24  VCOMPONENTS [Nitrogen*Variety] Blocks/Wplots/Subplots
 25  REML [METHOD=Fisher] Yield

REML variance components analysis
=================================

Response variate:  Yield of oats
Fixed model:       Constant + Nitrogen + Variety + Nitrogen.Variety
Random model:      Blocks + Blocks.Wplots + Blocks.Wplots.Subplots
Number of units:   72

Blocks.Wplots.Subplots used as residual term

Non-sparse algorithm with Fisher scoring


Estimated variance components
-----------------------------

Random term               component        s.e.
Blocks                        214.5       168.8
Blocks.Wplots                 106.1        67.9


Residual variance model
-----------------------

Term                         Model(order)  Parameter      Estimate      s.e.
Blocks.Wplots.Subplots       Identity      Sigma2            177.1      37.3


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term             Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Nitrogen                       113.06       3         37.69    45.0  <0.001
Variety                          2.97       2          1.49    10.0   0.272
Nitrogen.Variety                 1.82       6          0.30    45.0   0.932

Dropping individual terms from full fixed model

Fixed term             Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Nitrogen.Variety                 1.82       6          0.30    45.0   0.932
```

```
* MESSAGE: denominator degrees of freedom for approximate F-tests are
calculated using algebraic derivatives ignoring fixed/boundary/singular
variance parameters.
```

This example shows the default output from REML. First, a summary of the model is given by the setting model; this includes details of the response variate, the fixed and random model terms, the number of units analysed and whether options such as the absorbing factor, weights or mvinclude are set. The number of units analysed takes account of units excluded because of restrictions, zero weights or missing values in either the response variate or the factors and variates in the model. After the model description, the estimates of the variance components are printed with their standard errors. Finally, Wald tests are printed for the fixed model terms. Provided the design and models are not too large or complicated, the default setting of the FMETHOD option also produces F statistics with their numerator (n.d.f.) and denominator (d.d.f.) numbers of degrees of freedom. With an orthogonal design, like that in Example 5.3.1, the F statistics are identical to those produced by ANOVA (see Example 4.2.1), and can be used in exactly the same way. In other situations, they have approximate F distributions and so the F probabilities (F pr) should be used with care especially if the value is close to a critical value. The Wald and F statistics, and the FMETHOD option, are explained in more detail in 5.3.6.

5.3.2 Further output: the VDISPLAY directive

VDISPLAY directive

Displays further output from a REML analysis.

Options

PRINT = *string tokens*	What output to present (model, components, effects, means, stratumvariances, monitoring, vcovariance, deviance, Waldtests, missingvalues, covariancemodels); default mode, comp, Wald, cova
CHANNEL = *identifier*	Channel number of file, or identifier of a text to store output; default current output file
PTERMS = *formula*	Terms (fixed or random) for which effects or means are to be printed; default * implies all the fixed terms
PSE = *string token*	Standard errors to be printed with tables of effects and means (differences, estimates, alldifferences, allestimates, none); default diff
CFORMAT = *string token*	Whether printed output for covariance models gives the variance matrices or the parameters (variancematrices, parameters); default vari
FMETHOD = *string token*	Controls whether and how to calculate F-statistics for fixed terms (automatic, none, algebraic, numerical); default auto

Parameter

REML save structures	Save structure containing the details of each analysis; default is to take the save structure from the latest REML analysis

You can store the information from a REML analysis using the parameter SAVE in the REML statement, and then specify the same structure with the SAVE parameter of VDISPLAY. Several SAVE structures can be specified, corresponding to the analyses of several different variates. These need not have been analysed using the same REML statement, or even from the same model (as defined by VCOMPONENTS). Alternatively, if you just want to display output from the last y-variate that was analysed, there is no need to use the SAVE parameter in either REML or VDISPLAY: the save structure for the last y-variate analysed is saved automatically, and provides the default for VDISPLAY.

The options of VDISPLAY are the same as those that control output from REML: PRINT, PTERMS, PSE, CFORMAT and FMETHOD, plus the CHANNEL option which allows output to be directed to another output channel or into a text structure. The available settings of PRINT are identical to those in REML.

Example 5.3.2 continues Example 5.3.1, and uses VDISPLAY to print approximate stratum variances and tables of predicted means. The approximate stratum variances, which are available only when the Fisher method is used (see line 25 of Example 5.3.1), are derived from a decomposition of the information matrix for the variance components and are accompanied by the matrix of coefficients used to construct the stratum variances from the components.

In this orthogonal design, the approximate stratum variances are exactly the same as the residual mean squares from the strata in Example 4.2.1. Note that this will be the case only when the design is orthogonal: that is when the efficiency factors for the treatments are either 1 or 0 in each stratum. Also, under these circumstances, the estimates of variance components are the same as those that can be obtained from the analysis of variance by equating the residual mean squares to their expectations:

EMS(Blocks) $= 3175.1 = 12\sigma_b^2 + 4\sigma_{b.w}^2 + \sigma^2$

EMS(Wplots) $= 601.3 = 4\sigma_{b.w}^2 + \sigma^2$

EMS(Subplots) $= 177.1 = \sigma^2$

then $\sigma_b^2 = 214.5$, $\sigma_{b.w}^2 = 106.1$, $\sigma^2 = 177.1$ as above.

The second part of the output shows the predicted means for all factor combinations from the fixed model. For this design the means are the same as the standard means produced by ANOVA. For non-orthogonal (but balanced) designs, like the lattice in Example 4.7.3, the REML means are the same as the *combined* means produced by ANOVA. That is, in balanced designs where treatment terms can be estimated in several strata, the REML means combine all the available information.

Example 5.3.2

```
  26  VDISPLAY [PRINT=means,stratumvariances]

Approximate stratum variances
-----------------------------

Stratum                            variance    effective d.f.
Blocks                               3175.1         5.00
Blocks.Wplots                         601.3        10.00
Blocks.Wplots.Subplots                177.1        45.00


Matrix of coefficients of components for each stratum:

                  Blocks       12.00        4.00        1.00
            Blocks.Wplots       0.00        4.00        1.00
  Blocks.Wplots.Subplots        0.00        0.00        1.00


Table of predicted means for Constant
-------------------------------------

            104.0    Standard error:    6.64
```

```
Table of predicted means for Nitrogen
-------------------------------------

Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
             79.4     98.9    114.2    123.4

Standard error of differences: 4.436


Table of predicted means for Variety
------------------------------------

Variety      Victory  Golden rain   Marvellous
                97.6        104.5        109.8

Standard error of differences: 7.079


Table of predicted means for Nitrogen.Variety
---------------------------------------------

      Variety      Victory  Golden rain   Marvellous
     Nitrogen
        0 cwt         71.5         80.0         86.7
      0.2 cwt         89.7         98.5        108.5
      0.4 cwt        110.8        114.7        117.2
      0.6 cwt        118.5        124.8        126.8

Standard errors of differences

Average:           9.161
Maximum:           9.715
Minimum:           7.683

Average variance of differences: 84.74

Standard error of differences for same level of factor:

               Nitrogen     Variety
Average:          9.715       7.683
Maximum:          9.715       7.683
Minimum:          9.715       7.683
```

5.3.3 Tables of means and effects for fixed and random terms

This section gives more detail about the tables of effects for fixed and random terms provided by a REML analysis. It then describes how tables of predicted means are constructed by VDISPLAY. Tables of predictions can also be produced by the VPREDICT directive (5.5.1), which provides far more control over the types of predictions that are produced and the way in which they are calculated.

The estimates of parameters α and β in the general linear model are called the effects. Tables of effects generally differ from those obtained from ANOVA since REML uses a different parameterization of the linear model, described in 5.2.2 and 5.2.3.

The estimates of α and β satisfy the"mixed model equations":

$$\begin{pmatrix} X'X & X'Z \\ Z'X & Z'Z+\Gamma^{-1} \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} X'y \\ Z'y \end{pmatrix}$$

The fixed effects are estimated by the usual generalized least squares estimators

$$\hat{\alpha} = (X'\hat{V}^{-1}X)^{-1}X'\hat{V}^{-1}y$$

and the random effects are predicted by best linear unbiased prediction (BLUP)

$$\beta = (Z'Z+\hat{\Gamma}^{-1})^{-1}Z'(y-X\hat{\alpha}).$$

The variance-covariance matrix for the whole set of parameters ($\alpha' \; \beta' - \hat{\beta}'$) is

$$\text{Var}\begin{pmatrix} \alpha \\ \beta - \hat{\beta} \end{pmatrix} = \sigma^2 \begin{pmatrix} X'X & X'Z \\ Z'X & Z'Z + \Gamma^{-1} \end{pmatrix}^{-1}$$

and the variance matrix for the estimated parameters is obtained by using the estimated values of the variance parameters in $\hat{\Gamma}$. The estimated variance-covariance matrix for the fixed effect parameters can then be shown to be $\text{Var}(\hat{\alpha}) = (X'\hat{V}^{-1}X)^{-1}$.

The difference between estimates of fixed and random parameters can be seen from the form of the estimates. If the matrix $\hat{\Gamma}^{-1}$ is zero, the random effects are estimated as though they were fixed effects. For positive $\hat{\Gamma}$, the BLUP estimates $\hat{\beta}$ for random effects are smaller than if the effects had been estimated as fixed effects. For this reason, the BLUP random effects estimates are often called"shrunken" parameter estimates. The amount of shrinkage depends both on the values $\{\gamma_i\}$ and on the information available for each element of $\hat{\beta}$. Consider the simple case of a model

$$y_{ij} = \beta_i + \varepsilon_{ij}$$

where y_{ij} measures the jth replicate for the ith group (i=1...p; j=1...n_i), and there are two variance components σ_1^2 and σ^2. The BLUP estimator for the random effects is

$$\hat{\beta}_i = \frac{n_i}{n_i + \gamma^{-1}} \bar{y}_{i.} \qquad \text{where} \quad \bar{y}_{i.} = \frac{1}{n_i} \sum_{j=1}^{n_i} y_{ij}$$

The amount of shrinkage increases as $\gamma = \sigma_1^2/\sigma^2$ decreases; that is, shrinkage increases as the variability σ_1^2 of the random effect β decreases relative to the residual variance σ^2. The shrinkage discounts the likely contribution from the random error to the apparent random effect, using a factor that depends on their relative variability. This is intuitively satisfactory since high/low values in β may be due partly to high/low values of ε. Clearly this effect would be expected to decrease as the replication for each element of β increases. In fact, for fixed γ, the shrinkage decreases as the amount of information (here the replication n_i) on each random effect increases. So the random effects for which most information is available, where the estimates are most reliable, are shrunk least.

The BLUP estimates can be interpreted as predictions of the random effects given the data, formed by regressing β on residuals calculated by adjusting the data for the fixed effects only.

Tables of effects are obtained by setting option `PRINT=effects`, as shown in Example 5.3.6a. The constraints imposed upon the parameters α and β are explained in Sections 5.2.2 and 5.2.3 respectively.

The setting `PRINT=means` produces tables of predicted means based on the estimates of parameters α and β. In a generally balanced design, the tables of means produced by `REML` for fixed model terms are the same as the combined means produced by setting option `PRINT=cbmeans` in `ANOVA`, which are the same as the ordinary means when the design is orthogonal (see Examples 5.3.2 and 4.2.1). There is no such correspondence for unbalanced data. With `REML`, the means are calculated from a linear transformation of the estimated parameter values, taking no account of the frequency counts for different factor combinations. Therefore, these predicted means will correspond to the averages over the factor combinations only with orthogonal data. In other cases, tables of means can be thought of as mean effects of factor levels adjusted for the mean values of any covariates and for any lack of balance in the other factors: that is, as the means you would have expected if the data had been orthogonal. If there are no random terms in the model, the means from `REML` are those that would be calculated from fitting a regression model to the fixed terms and then using `PREDICT` with option settings

COMBINATIONS=full and ADJUST=equal (see 3.3.4).

Predicted means are calculated using all the parameter estimates and taking means over the model terms not present in the table. For fixed model terms, means need be taken only over the estimates for fixed model terms, since means over random terms will always be zero. For example, in the split-plot design of Example 5.3.1 above, if c, $v_1...v_3$, $a_1...a_4$ and va_{11},$va_{12}...va_{34}$ are the estimated parameters for the constant, Variety, Nitrogen and the Variety.Nitrogen interaction respectively, the means for Variety are calculated by

$$\text{mean}\{\text{ Variety } i \} = c + v_i + \text{mean}\{ a_j \} + \text{mean}\{ va_{ij} \}$$

and those for Variety.Nitrogen by:

$$\text{mean}\{\text{ Variety } i, \text{ Nitrogen } j \} = c + v_i + a_j + va_{ij}\,.$$

For random terms, means must be taken over the parameter estimates for all the terms in the model. Since the means are based on the shrunken parameter estimates described above, predicted means for random terms will also be shrunk.

When various parameter combinations do not occur and the calculation of a mean effect involves taking means over any of the missing combinations, then that mean will also be a missing value.

Option PTERMS controls the model terms for which tables of means or effects are produced. By default, if means or effects are requested but option PTERMS is not set, tables are printed for all the fixed model terms and none of the random terms. For covariates in the model, the linear regression parameter associated with the covariate can be printed as an effect, but predicted means are not available. Predicted means for other model terms are adjusted to the mean value of the covariate. If you want tables for terms from the random model, or for only a subset of terms in the fixed model, you can use PTERMS to list exactly which tables you require. The setting of PTERMS can contain the string 'Constant' (in capital or lower-case letters, or any mixture), to obtain details of the constant term.

By default, each table is accompanied by a summary – minimum, mean and maximum – of standard errors of differences (seds) for the entries in the table. This can be changed by option PSE: putting PSE=* suppresses the production of standard errors, the setting estimates gives a summary of the standard errors of individual table entries, while the settings alldifferences and allestimates give the full matrix of standard errors of differences and the table of standard errors respectively, as well as the summary. Only one setting of PSE is allowed at a time.

When METHOD=AI, the sparse matrix methods that are used do not return the whole covariance matrix for the random effects. So only standard errors, and not standard errors of differences, are available for these terms.

When an absorbing factor is used, the variance-covariance matrix is not available for the estimated parameters in the absorbing factor model. Therefore standard errors cannot be provided for tables of effects for terms in the absorbing factor model. For tables of means the situation is as follows: for fixed model terms, no errors are available for any term which is in the absorbing factor model or has a fixed interaction in the absorbing factor model; for random model terms, no errors are available for any term which is in the absorbing factor model or has an interaction in the absorbing factor model. No standard errors are available for tables of means if there are fixed effects in the absorbing factor model, although standard errors of differences may be available, subject to the conditions above.

In linear mixed models with more than one source of error variation the ratio of an effect to its standard error is not, in general, distributed as Student's t. This happens because the variance of an effect is some linear combination of "stratum variances": that is, a weighted sum of variables proportional to χ^2 distributions, rather than a simple multiple of a single χ^2 variable. For the same situation the F ratios for fixed terms, shown in Example 5.3.2, have only approximate F distributions.

Provided the design and models are not too large or complicated, REML is able to estimate

denominator numbers of degrees of freedom for the F ratios. The estimation uses the methods devised by Kenward & Roger (1997), which are essentially based on the Satterthwaite method used by ANOVA (4.2.1, 4.7.1). These degrees of freedom can also be used as degrees of freedom for (approximate) t-statistics calculated for contrasts within tables of predicted means of the corresponding fixed terms. Note, though, that the degrees of freedom are relevant for assessing the fixed term as a whole, and may vary over the contrasts amongst the means of the term. So they should be used with caution. (If you are interested in a specific comparison, you should set up a 2-level factor to fit this explicitly in the analysis.)

The degrees of freedom can also be used in the VLSD procedure to calculate (approximate) least significant differences for predicted means of fixed terms.

VLSD procedure

Prints approximate least significant differences for REML means (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (means, sed, lsd, df); default lsd
FACTORIAL = *scalar*	Limit on the number of factors in each term; default 3
LSDLEVEL = *scalar*	Significance level (%) to use in the calculation of least significant differences; default 5
DFMETHOD = *string token*	Specifies which degrees of freedom to use for the t-statistics (fddf, given, tryfddf); default fddf
DFGIVEN = *scalar*	Specifies the number of degrees of freedom to use for the t-statistics when DFMETHOD=given, or if d.d.f. are unavailable when DFMETHOD=tryfddf
FMETHOD = *string token*	Controls how to calculate denominator degrees of freedom for the F-statistics, if these are not already available in the REML save structure (automatic, algebraic, numerical); default auto
SAVE = *REML save structure*	Save structure to provide the table of means; default uses the save structure from the most recent REML

Parameters

TERMS = *formula*	Treatment terms whose means are to be compared; default * takes the REML fixed model
MEANS = *pointer* or *table*	Saves the means for each term
SED = *pointer* or *symmetric matrix*	Saves standard errors of differences between means
LSD = *pointer* or *symmetric matrix*	Saves approximate least significant differences matrix for the means
DF = *pointer* or *scalar*	Saves the degrees of freedom used to calculate the t critical values for the LSDs
DDF = *pointer* or *scalar*	Saves the denominator degrees of freedom in the F test for the term
DFRANGE = *pointer* or *scalar*	Saves the range of denominator degrees of freedom in the F tests for the term and any terms that are marginal to the term (available only when denominator degrees of freedom of F-statistics are being used)

The TERMS parameter specifies a model formula to define the fixed terms whose predicted means are to be compared. The means are usually taken from the most recent analysis performed by REML, but you can set the SAVE option to a save structure from another REML if you want to

examine means from an earlier analysis. As in VCOMPONENTS (5.2.1), the FACTORIAL option sets a limit on the number of factors in each term (default 3).

The DFMETHOD option specifies how to obtain the degrees of freedom for the t-statistics. The default is to use the numbers of denominator degrees of freedom printed by REML in the d.d.f. column in the table of tests for fixed tests (produced by setting option PRINT=wald). The degrees of freedom are relevant for assessing the fixed term as a whole, and may vary over the contrasts amongst the means of the term. So the LSDs should be used with caution. (If you are interested in a specific comparison, you should set up a 2-level factor to fit this explicitly in the analysis.) The FMETHOD option controls how the denominator degrees of freedom should be calculated, if they are not already available in the REML save structure (e.g. because they were printed in the original analysis). The settings are the same as in the REML directive (5.3.1), except that there is no none setting. (You would set this option only if you really do want to calculate them.)

In some of the more complicated analyses, REML may be unable to calculate the denominator degrees of freedom. You might then want to supply the number of degrees of freedom yourself, using the DFGIVEN option, rather than having no least significant differences at all. For example, you could use the number of denominator degrees of freedom from the analysis of an earlier similar design. However, the results will only be as good as the degrees of freedom that you have supplied, and thus should be used with caution! You can set option DFMETHOD=tryfddf to use the denominator degrees of freedom, if these can be calculated, or those specified by DFGIVEN otherwise. The setting DFMETHOD=given always uses the degrees of freedom specified by DFGIVEN.

Printed output is controlled by the PRINT option, with settings:

means	prints the means;
sed	prints standard errors for differences between the means;
lsd	prints least significant differences for the means;
df	prints the degrees of freedom used to calculate the t critical value required for the LSD, together with the denominator degrees of freedom in the F test for the term if these are not the same.

The significance level to use in the calculation of the least significant differences can be changed from the default of 5% using the LSDLEVEL option.

The MEANS parameter can save the means. If the TERMS parameter specifies a single term, MEANS must be undeclared or set to a table. If TERMS specifies several terms, you must supply a pointer which will then be set up to contain as many tables as there are terms. Similarly the SED parameter can save the standard errors of differences, the LSD parameter can save the approximate least significant differences, the DF parameter can save the degrees of freedom, and the DDF parameter can save the denominator degrees of freedom in the F tests.

When a term involves several factors, its means may be be formed from the effects of several terms. For example, the means for the term A.B will involve the effects for the terms A and B (if these are in the model), as well as those for the term A.B. Different contrasts between the means will then have different denominator degrees of freedom. For caution, if VLSD is using the number of denominator degrees of freedom, it uses the smallest number over the terms that are involved in calculating each table of the means. (This corresponds to the largest t-statistic.) If the difference in the t-statistics calculated from smallest and largest numbers of degrees of freedom differ by more than 1%, VLSD prints a warning message. If the denominator degrees of freedom are being used, their range for each term can be saved by the DFRANGE parameter.

Example 5.3.3 calculates least significant differences for the Nitrogen means in Example 5.3.2. In this case, the least significant differences are not approximate as the Nitrogen contrasts all have the same variance, as shown in Example 4.2.1a, and so the values match those produced by ANOVA in Example 4.2.1a too.

Example 5.3.3

```
  27  VLSD Nitrogen

Approximate least significant differences (5% level) of REML means
==================================================================

Nitrogen
--------

Nitrogen   0 cwt   1           *
Nitrogen 0.2 cwt   2       8.934           *
Nitrogen 0.4 cwt   3       8.934       8.934           *
Nitrogen 0.6 cwt   4       8.934       8.934       8.934           *
                               1           2           3           4
```

The same methods are used in procedure VMCOMPARISON to perform (approximate) Fisher's LSD tests; details are in Part 3 of the *Genstat Reference Manual*.

5.3.4 Plots of means and effects

VGRAPH procedure

Plots tables of means from REML (R.W. Payne).

Options

GRAPHICS = *string token*	Type of graph (highresolution, lineprinter); default high
METHOD = *string token*	What to plot (points, means, linesandpoints, onlylines, data, barchart, splines); default poin when XFACTOR is a factor, and only when it is a variate
XFREPRESENTATION = *string token*	How to label the x-axis (levels, labels); default labe uses the XFACTOR labels, if available
PSE = *string token*	What to plot to represent variation when points are plotted at the means (differences, lsd, means, allmeans); default diff
LSDLEVEL = *scalar*	Significance level (%) to use for approximate least significant differences; default 5
DFSPLINE = *scalar*	Number of degrees of freedom to use when METHOD=splines
YTRANSFORM = *string tokens*	Transformed scale for additional axis marks and labels to be plotted on the right-hand side of the y-axis (identity, log, log10, logit, probit, cloglog, square, exp, exp10, ilogit, iprobit, icloglog, root); default iden i.e. none
PENYTRANSFORM = *scalar*	Pen to use to plot the transformed axis marks and labels; default * selects a pen, and defines its properties, automatically
KEYMETHOD = *string token*	What to use for the key descriptions when GROUPS specifies more than one factor (labels, namesandlabels); default name
PLOTTITLEMETHOD = *string token*	What to use for the titles of the plots when TRELLISGROUPS specifies more than one factor

	(labels, namesandlabels); default name
PAGETITLEMETHOD = *string token*	What to use for the titles of the pages when PAGEGROUPS specifies more than one factor (labels, namesandlabels); default name
USEAXES = *string token*	Which aspects of the current axis definitions of window 1 to use (none, limits, marks, mpositions, nsubticks,); default none
SAVE = *REML save structure*	Save structure to provide the table of means if the MEANS parameter is unset; default uses the save structure from the most recent REML

Parameters

XFACTOR = *factors* or *variates*	Provides the x-values for each plot; by default this is chosen automatically
GROUPS = *factors* or *pointers*	Factor or factors identifying groups in each plot; by default chosen automatically
TRELLISGROUPS = *factors* or *pointers*	Factor or factors specifying the different plots of a trellis plot of a multi-way table
PAGEGROUPS = *factors* or *pointers*	Factor or factors specifying plots to be displayed on different pages
NEWXLEVELS = *variates*	Values to be used for XFACTOR; default uses the existing levels if XFACTOR is a factor, and the minimum and maximum values if it is a variate
TITLE = *texts*	Title for the graph; default is to define a title automatically if GROUPS is set, or to have none if it is unset
YTITLE = *texts*	Title for the y-axis; default is to use the identifier of the y-variate, or to have no title if this is unnamed
XTITLE = *texts*	Title for the x-axis; default is to use the identifier of the XFACTOR
PENS = *variates*	Defines the pen to use to plot the points and/or line for each group defined by the GROUPS factors

VGRAPH plots tables of predicted means from REML. In its simplest form, the behaviour of VGRAPH depends on the model. If the fixed model contains only main effects, it plots the means for the first factor in the fixed model. Otherwise it looks for the first fixed term involving two factors; it then plots the means with one of these factors as the x-axis, and the second as a grouping factor with levels identified by different plotting colours and symbols.

By default, the means are from the most recent REML. However, you can plot means from an earlier analysis, by using the SAVE option of VGRAPH to specify its save structure (saved using the SAVE parameter of the REML command that performed the analysis). VGRAPH uses the VPREDICT directive (5.5.1) with default option settings to obtain the means. This should give the same means as those printed by REML or VDISPLAY. If you want to use VPREDICT with other option settings, you can plot these using the DTABLE procedure (1:4.11.7).

The GRAPHICS option controls whether a high-resolution or a line-printer graph is plotted; by default GRAPHICS=high.

The METHOD option controls how the predicted means are plotted in high-resolution graphics, with settings:

points	to plot a point at each mean;
means	synonym of points;

linesandpoints	to plot points and join them by lines;
onlylines	to draw lines between the means;
data	to draw lines between the means, and then also plot the original data values;
barchart	to plot the means as a barchart;
splines	to plot points at the means together with a smooth spline to show the trend over each group of means; the DFSPLINE specifies the degrees of freedom for the splines; if this is not set, 2 d.f. are used when there are up to 10 points, 3 if there are 11 to 20, and 4 for 21 or more.

The default is to plot points when XFACTOR is a factor, and onlylines when it is a variate. Only points are available in line-printer graphics.

The PSE option specifies the type of error bar to be plotted, when points are plotted for the means, with settings:

differences	average standard error of difference;
lsd	average approximate least significant difference (calculated using the VLSD procedure);
means	average effective standard error for the means;
allmeans	plots plus and minus the effective standard error around every mean.

The LSDLEVEL option sets the significance level (%) to use for the approximate least significant differences (default 5). The allmeans setting is often unsuitable for plots other than barcharts when there are GROUPS, as the plus/minus e.s.e. bars may overlap each other.

You can define the table of means to plot explicitly, by specifying its classifying factors using the XFACTOR, GROUPS, TRELLISGROUPS and PAGEGROUPS parameters. The XFACTOR parameter can define a factor against whose levels the means are plotted. It can also specify a variate, and VPREDICT then sets up a factor automatically, to classify the table, with levels at the values specified by the NEWXLEVELS parameter. With a multi-way table, there will be a plot of means against the XFACTOR levels for every combination of levels of the factors specified by the GROUPS, TRELLISGROUPS and PAGEGROUPS parameters. The GROUPS parameter specifies factors whose levels are to be included in a single window of the graph. So, for example, if you specify

```
VGRAPH [METHOD=line] XFACTOR=A; GROUPS=B
```

VGRAPH will produce plot the means in a single window with factor A on the x-axis, and a line for each level of the factor B. You can set GROUPS to a pointer to specify several factors to define groups. For example

```
POINTER [VALUES=B,C] Groupfactors
VGRAPH [METHOD=line] XFACTOR=A; GROUPS=Groupfactors
```

to plot a line for every combination of the levels of factors B and C. Similarly, the TRELLISGROUPS option can specify one or more factors to define a trellis plot. For example,

```
VGRAPH [METHOD=line] XFACTOR=A; GROUPS=B; TRELLISGROUPS=C
```

will produce a plot for each level of C, in a trellis arrangement; each plot will again have factor A on the x-axis, and a line for each level of the factor B. Likewise, the PAGEGROUPS parameter can specify factors whose combinations of levels are to be plotted on different pages. So

```
VGRAPH [METHOD=line] XFACTOR=A; GROUPS=B; PAGEGROUPS=C
```

will produce a plot for each level of C, but now on separate pages. Multi-way tables can plotted even if the corresponding model term was not in the ANOVA analysis. For example you can plot a two-way table even if the analysis contained only the main effects of the two factors; however,

the lines will then all be parallel and no standard errors or LSDs can be included.

The NEWXLEVELS parameter enables different levels to be supplied for an XFACTOR factor, if its existing levels are unsuitable. If the factor has labels, these are used to label the x-axis unless you set option XFREPRESENTATION=levels. When XFACTOR is a variate, NEWXLEVELS can specify the values where the predictions are to be made. By default, they are made at its minimum and maximum values.

Note that the values predicted by VPREDICT, for an XFACTOR variate, will not include any spline effects, nor can it take account of any relationships between different variates in the model. (For example, the model may include a variate and its square.) To take account of relationships like these, you should use VPREDICT directly, specifying the linked variables with the PARALLEL parameter (5.5.1). Save the table of predictions, and then plot it using DTABLE (1:4.11.7).

The TITLE, YTITLE and XTITLE parameters can supply titles for the graph, the y-axis and the x-axis, respectively. The symbols, colours and line styles that are used in a high-resolution plot are usually set up by VGRAPH automatically. If you want to control these yourself, you should use the PEN directive to define a pen with your preferred symbol, colour and line style, for each of the groups defined by combinations of the GROUPS factors. The pen numbers should then be supplied to VGRAPH, in a variate with a value for each group, using the PENS parameter. The YTRANSFORM option allows you to include additional axis markings, transformed onto another scale, on the right-hand side of the y-axis. Suppose, for example, suppose you have analysed a variate of percentages that have been transformed to logits. You might then set YTRANSFORM=ilogit (the inverse-logit transformation) to include markings in percentages alongside the logits. The settings are the same as those of the TRANSFORM parameter of AXIS, which is used to add the markings (1:6.9.7). You can control the colours of the transformed marks and labels, by defining a pen with the required properties, and specifying it with the PENYTRANSFORM option. Otherwise, the default is to plot them in blue.

When there is more than one GROUPS factor, the KEYMETHOD controls whether to use the factor names with their labels (or levels for factors with no labels) or just the labels (or levels) in the key descriptions. The default is to use the names and the labels (or levels). Similarly, the PLOTTITLEMETHOD specifies what to use for the titles of the plots when there is more then one TRELLISGROUPS factor, and the PAGETITLEMETHOD specifies what to use for the titles of the plots when there is more then one PAGEGROUPS factor. You can setKEYMETHOD=* to have no key at all.

The USEAXES option allows you to control various aspects of the axes. First you need to use the XAXIS and YAXIS directives to define them for window 1. Then specify which of the aspects of the axes in window 1 are to be used by DTABLE, by specifying USEAXES with the following settings:

limits	y- and x-axis limits (LOWER and UPPER parameters);
marks	location and labelling of the tick marks (MARKS, LABELS, LDIRECTION, LROTATION, DECIMALS, DREPRESENTATION, and VREPRESENTATION parameters);
mpositions	positions of the tick marks (MPOSITION parameter); and
nsubticks	number of subticks per interval (NSUBTICKS parameter).

By default none are used.

Figure 5.3.4a shows the default means plot for the analysis in Examples 5.3.1 and 5.3.2, produced by the statement

```
VGRAPH
```

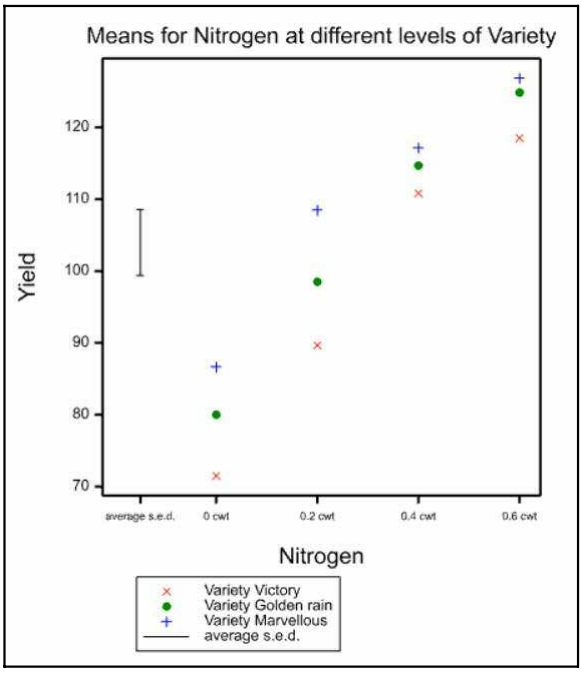


Figure 5.3.4a

VDEFFECTS procedure

Plots one- or two-way tables of effects estimated in a REML analysis (R.W. Payne).

Options

GRAPHICS = *string token*	Type of graph (highresolution, lineprinter); default high
METHOD = *string token*	What to plot (effects, lines); default effe
XFREPRESENTATION = *string token*	How to label the x-axis (levels, labels); default labels uses the XFACTOR labels, if available
PSE = *string*	What s.e. to plot to represent variation (differences, effects, alleffects); default diff
SAVE = *REML save structure*	Save structure of the analysis to display; the default is to take the most recent REML analysis

Parameters

XFACTOR = *factors*	Factor providing the x-values for each plot
GROUPS = *factors*	Factor identifying the different sets of points from a two-way table of effects
COVARIATES = *variates*	X-variates for regression coefficients or pointer
NEWXLEVELS = *variates*	Values to be used for XFACTOR instead of its existing levels
TITLE = *texts*	Title for the graph; default defines a title automatically
YTITLE = *texts*	Title for the y-axis; default ''
XTITLE = *texts*	Title for the x-axis; default is to use the identifier of the XFACTOR

VDEFFECTS plots tables of effects estimated in a REML analysis. By default the effects are from the most recent analysis, but you use the SAVE option to specify the save structure from some other analysis.

The XFACTOR parameter indicates the factor against whose levels the effects are plotted. You can also specify a second factor, using the GROUPS parameter, to plot a two-way table of effects. A separate set of points is then plotted for every level of GROUPS.

By default, the effects will be for the model term XFACTOR (if GROUPS is not set) or XFACTOR.GROUPS (if GROUPS is set). You can also specify one, or more, variates for the term, using the COVARIATES parameter. If COVARIATES is set to a single variate, xvar say, the term will be XFACTOR.xvar or XFACTOR.GROUPS.xvar (representing regression coefficients for xvar). Alternatively, it can be set to a pointer containing several variates, for example x1var and x2var. The term will be then be XFACTOR.x1var.x2var or XFACTOR.GROUPS.x1var.x2var (representing regression coefficients for the product of the variates x1var and x2var).

The NEWXLEVELS parameter enables different levels to be supplied for XFACTOR if the existing levels are unsuitable. If XFACTOR has labels, these are used to label the x-axis unless you set option XFREPRESENTATION=levels.

Usually, each estimate is represented by a point (using pens 1, 2, and so on for each level in turn of the GROUPS factor). However, with high-resolution plots, the METHOD option can be set to lines to draw lines between the points. The GRAPHICS option controls whether a high-resolution or a line-printer graph is plotted; by default GRAPHICS=high.

The PSE option specifies how to represent the variability of the effects, as follows:

differences	plots an error bar showing the average standard error for differences between pairs of effects;
effects	plots an error bar showing the average standard error of the effects;
alleffects	plots a bar around each estimate showing plus and minus its standard error.

The TITLE, YTITLE and XTITLE parameters allow you to supply titles for the graph, the y-axis and the x-axis respectively.

Example 5.3.4b prints and plots the nitrogen effects and e.s.e.'s from the analysis in Examples 5.3.1 and 5.3.2.

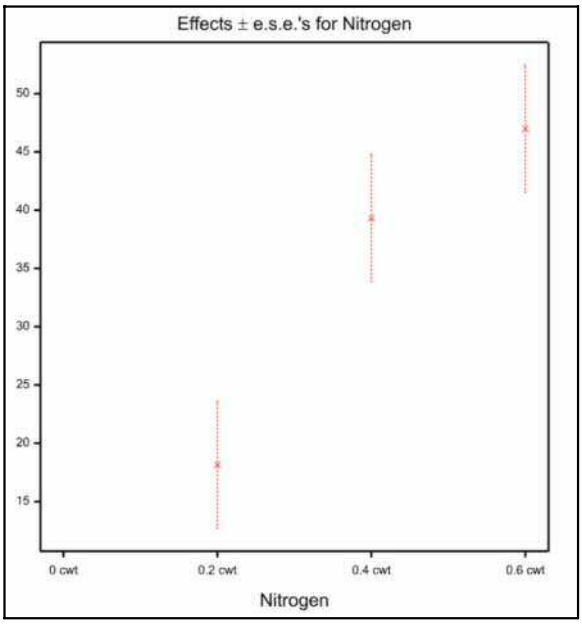


Figure 5.3.4b

Example 5.3.4b

```
  29  VDISPLAY  [PRINT=effects; PTERMS=Nitrogen]

Table of effects for Nitrogen
-----------------------------

Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
             0.00    18.17    39.33    47.00

Standard error of differences: 7.683

  30  VDEFFECTS [PSE=alleffects] Nitrogen
```

5.3.5 Residual plots

This section describes the procedures for plotting residuals. Other procedures for checking residuals, for example to identify potential outliers, are described in Section 5.3.7. (Section 5.3.7 uses the example in Section 5.3.6. This does have some large residuals, unlike the Example 5.3.1 which is used in this section.)

VPLOT procedure

Plots residuals from a REML analysis (S.J. Welham).

Options

RMETHOD = *string token*	Which random terms to use when calculating the residuals (final, all, notspline, stfinal, stall); default uses the setting from the REML statement
INDEX = *variate*	X-variate for an index plot; default !(1,2...)
GRAPHICS = *string token*	What type of graphics to use (lineprinter, highresolution); default high
TITLE = *text*	Overall title for the plots; if unset, the identifier of the y-variate is used
SAVE = *REML save structure*	Specifies the (REML) save structure from which the residuals and fitted values are to be taken; default * uses the SAVE structure from the most recent REML analysis

Parameters

METHOD = *string tokens*	Type of residual plot (fittedvalues, normal, halfnormal, histogram, absresidual, index); default fitt, norm, half, hist
PEN = *scalars*, *variates* or *factors*	Pen(s) to use for each plot

Procedure VPLOT provides up to four types of residual plots from a REML analysis. These are selected using the METHOD parameter, with settings: fitted for residuals versus fitted values, normal for a Normal plot, halfnormal for a half-Normal plot, and histogram for a histogram of residuals, absresidual for a plot of the absolute values of the residuals versus the fitted values, and index for a plot against an "index" variable (specified by the INDEX option). The default is to produce the first four types of plot. The PEN parameter can specify the graphics pen or pens to use for each plot. The TITLE option can supply an overall title. If this is not set, the

identifier of the y-variate is used.

For a Normal plot, the Normal quantiles are calculated as follows:

q_i = NED((*i*−0.375) / (*n*+0.25)) *i*=1...*n*

while for a half−Normal they are given by

q_i = NED(0.5 + 0.5×(*i*−0.375) / (*n*+0.25)) *i*=1...*n*

The residuals and fitted values are accessed automatically from the analysis specified by the SAVE option. If the SAVE option has not been set, they are taken from the last SAVE structure from the most recent REML analysis.

The RMETHOD option controls which random terms are used to calculate the residuals:

all	all the random effects,
final	only the final random term,
notspline	all except any random spline terms,
stall	standardized residuals using all the random effects, and
stfinal	standardized residuals using only the final random term.

The default takes the setting from the REML directive that produced the analysis. Note that residuals based on the final random term will not be calculated when any of the variance components are negative, as the associated negative correlations can generate very misleading patterns. VPLOT will then generate a warning that all the residuals are missing, and you should use RMETHOD=all instead.

By default, high-resolution graphics are used. Line-printer graphics can be obtained by setting option GRAPHICS=lineprinter.

Figure 5.3.5a shows the default set of residual plots for the analysis in Examples 5.3.1 and 5.3.2, produced by the statement

```
VPLOT
```

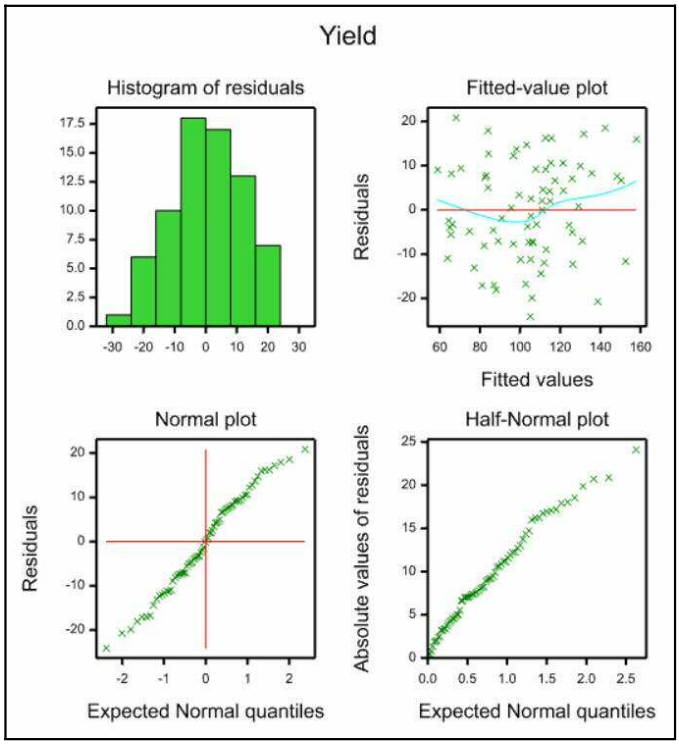


Figure 5.3.5a

VDFIELDRESIDUALS procedure

Display residuals from a REML analysis in field layout (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (table); default * i.e. none
PLOT = *string tokens*	Controls the graphs that are displayed (contour, shade); default cont
RMETHOD = *string token*	Which random terms to use to calculate the residuals (final, all, notspline, stfinal, stall); default all
GRAPHICS = *string token*	Type of graph (highresolution, lineprinter); default high
MARGIN = *string token*	Whether to include margins in printed tables (yes, no); default no
YORIENTATION = *string token*	Y-axis orientation of the plot (reverse, normal); default norm
PENCONTOUR = *scalar*	Pen number to be used for the contours; default 1
PENFILL = *scalar* or *variate*	Pen number(s) defining how to fill the areas between contours; default 3
PENSHADE = *scalar* or *variate*	Pen(s) to use for the shade plot; default 3

Parameters

Y = *variates* or *factors*	Specifies the y-coordinates of the plots
X = *variates* or *factors*	Specifies the x-coordinates of the plots
SAVE = *REML save structures*	Save structure of the REML analysis from which to take the residuals; default is to take the most recent REML analysis
FIELDWIDTH = *scalars*	Field width for printing the residuals; default 12
DECIMALS = *scalars*	Number of decimal places to use when printing the residuals
TITLE = *texts*	Titles for the plots

VDFIRLDRESIDUALS allows you to display residuals from a REML analysis in a two dimensional layout as, for example, from a field experiment. This can be useful to study the spatial pattern of the residuals, for example to see if there are any systematic trends in fertility.

The locations of the plots are defined by the Y and X parameters, specifying variates or factors containing their y- and x-coordinates respectively. By default the residuals are taken from the most recent REML analysis. However, you can take the residuals from some other analysis, by specifying its save structure using the SAVE parameter.

The RMETHOD option controls which random terms are used to calculate the residuals:

all	all the random effects (default),
final	only the final random term,
notspline	all except any random spline terms,
stall	standardized residuals using all the random effects, and
stfinal	standardized residuals using only the final random term.

Usually, the plots in the experiment will all have different coordinates. However, if there are several plots with the same coordinates, mean residuals are calculated for each location. Thus for example, if you wanted only to look at the block and whole-plot residuals in a split-plot design, you could form the residuals from all the random terms, and then set identical coordinates for the (sub-) plots within each whole plot.

VDFIELDRESIDUALS provides two types of graph, selected by the settings of the PLOT option as follows:

contour	generates a contour plot if the plots are on a regular grid, or a line graph if they are arranged in a single line, and
shade	produces a shade plot for plots that are on a regular grid.

By default PLOT=contour. You can also set option PRINT=table to print the residuals in a table, whose structure corresponds to the field layout,

The GRAPHICS option determines the type of graphics that is used, with settings highresolution (the default) and lineprinter. No graphs can be produced if the plots are in an irregular 2-dimensional arrangement. High-resolution contour plots require more than three rows and columns, and line-printer contour plots require more than four rows and columns.

The way in which the lines are drawn in high-resolution contour plots is defined by the properties of the pen specified by the PENCONTOUR option, while the pen specified by the PENFILL parameter defines how to shade the areas between the contours. Their defaults are 1 and 3 respectively. Similarly, the pen or pens specified by the PENSHADE option control the colouring of the shade plot; the default is to use pen 3. For more information see the DCONTOUR and DSHADE directives.

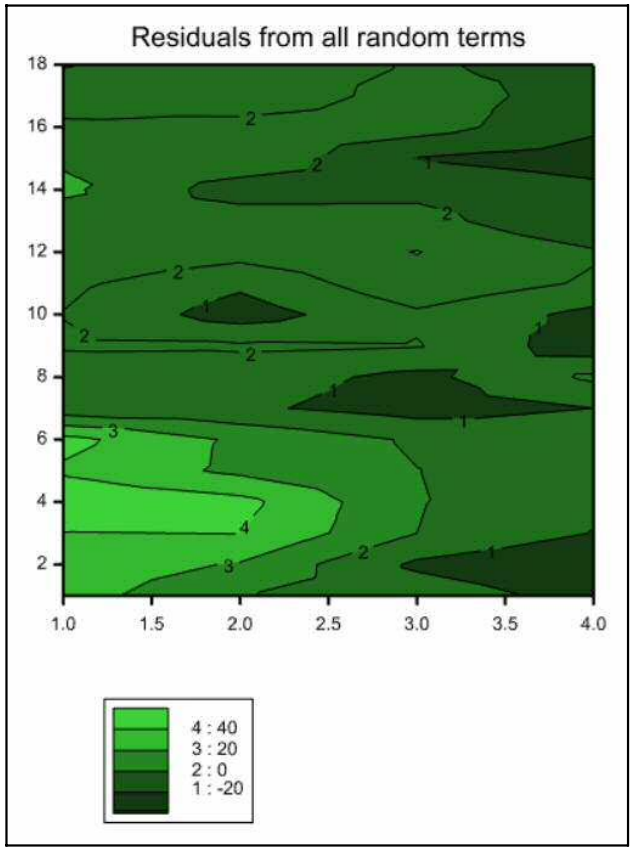


Figure 5.3.5b

The MARGIN option, with settings no (default) and yes, determines whether or not marginal means are included with the printed tables. The FIELDWIDTH and DECIMALS parameters can be used to specify the formats of the printed tables (as in the PRINT directive). The TITLE parameter can supply a title. If this is not set, a default title is formed.

The YORIENTATION option controls the orientation of the y-coordinates in the plots and tables. By default this is normal, so that they run upwards from the bottom of the page (as in a map).

The program below defines coordinates for the plots of the split-plot design in Example 5.3.1, and then displays the residuals in the contour plot shown in Figure 5.3.5b.

```
VARIATE [VALUES=2(1...18)2] Row
& [VALUES=(1,2)18,(3,4)18] Column
VDFIELDRESIDUALS Y=Row; X=Column
```

5.3.6 Assessing and plotting fixed effects

We now consider in more detail the rat reproduction example described in 5.1.1 (Dempster *et al*. 1984). This is an unbalanced design with fixed effects and more than one variance component. In this case, it is the fixed effects, here different doses of the experimental compound and its interactions, that are the primary interest. We describe below how to produce tests for the significance of fixed effects. These tests have only asymptotic distributions and not the exact

distributional properties associated with tests from ANOVA and linear regression. Care is therefore needed when making inferences from small samples.

The experiment was designed to compare three doses of an experimental compound for improving maternal performance (control, low and high), so the thirty female rats (dams) were randomly split into 3 groups of 10, and the three groups were randomly assigned to the three different treatments. All the pups in each litter were then weighed. The fixed model is `Dose * Sex + Littersize`, since the sex of the pup and the size of the litter both affect pup weight, and including the `Dose.Sex` interaction meant that any differential effect of the compound on male or female pups could be estimated. Three of the litters had to be dropped from the study, which meant that one treatment group had only seven litters. Also, litters contain different numbers of male and female pups, as well as being of different total size. This means that the experiment is not balanced, and so cannot be analysed using ANOVA. If it had only one component of variance, the experiment could be analysed by linear regression. However, further variation is introduced into the data by the effects of different dams. Since the dams could be considered as a random selection from the wider population we use the dams as a random effect. The effect of pups is also a random effect. Since the pups are the units of the experiment, the variation between pups is in fact the error variance component. There are therefore two components of variance, due to dams and to pups within dams. Example 5.3.6a shows the analysis of this experiment.

Since the different doses are applied to different dams, most of the information on the compounds is contained in the differences between the dams. Including dams as a random effect means that REML can make use of the between-dam information when estimating the effects of compounds. The variance component due to dam is also estimated, and used to construct appropriate standard errors for the effects.

Example 5.3.6a

```
  2  UNITS [NVALUES=322]
  3  FACTOR [LEVELS=27] Dam
  4  &      [LEVELS=18] Pup
  5  FACTOR [LEVELS=2; LABELS=!T('M','F')] Sex
  6  FACTOR [LEVELS=3; LABELS=!T('C','Low','High')] Dose
  7  VARIATE Littersize,Weight
  8  OPEN 'RATS.DAT'; CHANNEL=2; FILETYPE=input
  9  READ [CHANNEL=2] Dose,Sex,Littersize,Dam,Pup,Weight; \
 10    FREPRESENTATION=2(labels),4(levels)

   Identifier   Minimum      Mean   Maximum    Values   Missing
   Littersize     2.000     13.33     18.00       322         0
       Weight     3.680     6.084     8.330       322         0

   Identifier    Values   Missing    Levels
         Dose       322         0         3
          Sex       322         0         2
          Dam       322         0        27
          Pup       322         0        18


 11  CLOSE 2
 12  VCOMPONENTS [FIXED=Littersize+Dose*Sex] RANDOM=Dam/Pup
 13  REML [PRINT=model,components,effects] Weight

REML variance components analysis
=================================

Response variate:  Weight
Fixed model:       Constant + Littersize + Dose + Sex + Dose.Sex
Random model:      Dam + Dam.Pup
Number of units:   322

Dam.Pup used as residual term

Sparse algorithm with AI optimisation
```

```
All covariates centred


Estimated variance components
-----------------------------

Random term               component        s.e.
Dam                          0.0970      0.0332


Residual variance model
-----------------------

Term                         Model(order)  Parameter      Estimate      s.e.
Dam.Pup                      Identity      Sigma2            0.165    0.0137


Table of effects for Constant
-----------------------------

            6.612    Standard error: 0.1099


Table of effects for Littersize
-------------------------------

          -0.1279    Standard error: 0.01881


Table of effects for Dose
-------------------------

Dose            C          Low         High
           0.0000      -0.4528      -0.9046


Standard errors of differences

Average:          0.1815
Maximum:          0.1936
Minimum:          0.1587

Average variance of differences: 0.03319


Table of effects for Sex
------------------------

Sex            M            F
          0.0000      -0.4116

Standard error of differences: 0.07356


Table of effects for Dose.Sex
-----------------------------

          Sex            M            F
         Dose
            C      0.00000      0.00000
          Low      0.00000      0.07008
         High      0.00000      0.10719


Standard errors of differences

Average:          0.1210
Maximum:          0.1342
Minimum:          0.1063

Average variance of differences: 0.01482
```

Tables of effects contain the values of the estimated parameters $\hat{\alpha}$ and $\hat{\beta}$. By default REML prints estimated effects $\hat{\alpha}$ only for the fixed model terms, as shown in Example 5.3.6a. The effects are subject to constraints (as described in 5.2.2) so that parameters corresponding to the first level of a factor are set to zero. The constant term is then not the grand mean, but the mean for a unit with the first level of all the factors: that is, Dose=Control and Sex=male (and mean value for the covariate Littersize). The other parameters represent differences from the first levels of the factors.

As discussed earlier (5.3.3), individual parameter estimates are not in general distributed as Student's t. However, the importance of individual terms in the model can be assessed formally using either Wald and approximate F statistics or a likelihood-based test.

The Wald statistic to test the null hypothesis $\alpha_1=0$ for a fixed model term is defined as $\hat{\alpha}_1'[\mathrm{Var}(\hat{\alpha}_1)]^{-1}\hat{\alpha}_1$. In an orthogonal design (see 4.7), this corresponds to the treatment sum of squares divided by the stratum mean square. So, under the usual assumption that the residuals come from Normal distributions, the Wald statistic divided by its degrees of freedom will have an F distribution, $F_{m,n}$, where m is the number of degrees of freedom of the fixed term, and n is the number of residual degrees of freedom for the fixed term. Unless the design is large or complicated, Genstat estimates n by default, and prints it in the column headed "d.d.f." (i.e. denominator degrees of freedom); m is in the column headed "n.d.f." (i.e. numerator degrees of freedom). For orthogonal designs, the F statistics and probabilities are identical to those produced by the Analysis of Variance menus, and can be used in exactly the same way. In other situations, the printed F statistics have approximate F distributions. So you need to be careful if the value is close to a critical value.

The degree-of-freedom estimation uses the methods devised by Kenward & Roger (1997). The computations can be time consuming with large or complicated models. So REML and VDISPLAY have an FMETHOD option to control whether and how they are done. With the default setting, automatic, Genstat assesses the model itself and decides automatically whether to do the computations and which method to use. The other settings allow you to decide this for yourself:

none	no F statistics are produced;
algebraic	the calculations use algebraic derivatives (which may involve large matrix calculations);
numerical	the calculations use numerical derivatives (which require an extra evaluation of the mixed model equations for every variance parameter).

The Wald statistics themselves would have exact χ^2 distributions if the variance parameters were known but, as they must be estimated, they are only asymptotically distributed as χ^2. In practical terms, the χ^2 values will be reliable if the residual degrees of freedom for a fixed term is large compared to its own degrees of freedom. Otherwise they tend to give significant results rather too frequently. The F statistics, if available, are more reliable than the Wald statistics. If they are not available, Genstat produces probabilities for the Wald statistics instead, which should again be used with care especially when the value is close to a critical value.

The first part of the table presents Wald and F statistics for a sequential fit of the fixed terms. Each line represents the effect of adding a term to a model containing the terms in all the preceeding lines. When there is only one fixed term, or when the fixed terms are orthogonal, the order is unimportant. However, with non-orthogonal fixed effects, the statistics will depend on the order in which the terms were specified in the fixed model. You may therefore need to specify the model in several different ways to obtain all the required tests. Marginality should be taken into account: that is, main effects must always be listed before their interactions (see 3.3.3). Problems of interpretation associated with non-orthogonal model terms are discussed further in 4.7.4.

As an example, for the fixed model

```
A * B = A + B + A.B
```

there will be three Wald and F statistics in this part of the table: the first, due to A, can be used to compare model H_0: $E(y_{ij})=\mu$ with model H_1: $E(y_{ij})=\mu+a_i$; the second, due to fixed model term B, compares model H_1 with model H_2: $E(y_{ij})=\mu+a_i+b_j$; and the third statistic, due to model term A.B, compares model H_2 with model H_3:

$$E(y_{ij}) = \mu + a_i + b_j + ab_{ij}.$$

The second part of the table looks at the effect of removing terms from the complete fixed model: so the lines here allow you to assess the effects of a term after eliminating all the other fixed terms. This is particularly useful for seeing how the model might be simplified. For the fixed model A*B the only relevant term here would be the A.B interaction. We cannot remove a main effect (such as A or B) from a model that contains an interaction involving that factor.

The Wald and F statistics are obtained by setting the REML option PRINT to waldtests. For some very large models, the statistics cannot be calculated when METHOD=Fisher is used.

Example 5.3.6b

```
  14  VDISPLAY [PRINT=waldtests]

Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Littersize                       27.99       1         27.99    31.5  <0.001
Dose                             24.29       2         12.15    23.9  <0.001
Sex                              57.96       1         57.96   299.8  <0.001
Dose.Sex                          0.80       2          0.40   302.1   0.672

Dropping individual terms from full fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Littersize                       46.25       1         46.25    31.5  <0.001
Dose.Sex                          0.80       2          0.40   302.1   0.672

* MESSAGE: denominator degrees of freedom for approximate F-tests are
calculated using algebraic derivatives ignoring fixed/boundary/singular
variance parameters.
```

The approximate F statistic for the Dose.Sex interaction is 0.4 on 2 and 302.1 degrees of freedom, and is not significant under the corresponding F distribution. To preserve marginality, we would always fit the interaction after the main effects, so there is no need to recalculate the F statistic for the interaction using a different fixed model order. The Dose.Sex interaction can therefore be dropped from the model. To judge which of the main effects should be retained, it is then necessary to fit the model terms in several different orders, as shown in Example 5.3.6c.

Example 5.3.6c

```
  15  VCOMPONENTS [FIXED=Dose+Sex+Littersize] RANDOM=Dam/Pup
  16  REML [PRINT=waldtests] Weight

Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Dose                              9.83       2          4.91    24.0   0.016
Sex                              53.96       1         53.96   301.7  <0.001
Littersize                       46.43       1         46.43    31.4  <0.001
```

```
Dropping individual terms from full fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Dose                             22.84       2         11.42    24.0  <0.001
Sex                              58.27       1         58.27   301.7  <0.001
Littersize                       46.43       1         46.43    31.4  <0.001

* MESSAGE: denominator degrees of freedom for approximate F-tests are
calculated using algebraic derivatives ignoring fixed/boundary/singular
variance parameters.

  17  VCOMPONENTS [FIXED=Sex+Dose+Littersize] RANDOM=Dam/Pup
  18  REML [PRINT=waldtests] Weight

Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Sex                              55.81       1         55.81   301.7  <0.001
Dose                              7.98       2          3.99    24.0   0.032
Littersize                       46.43       1         46.43    31.4  <0.001

Dropping individual terms from full fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Sex                              58.27       1         58.27   301.7  <0.001
Dose                             22.84       2         11.42    24.0  <0.001
Littersize                       46.43       1         46.43    31.4  <0.001

* MESSAGE: denominator degrees of freedom for approximate F-tests are
calculated using algebraic derivatives ignoring fixed/boundary/singular
variance parameters.
```

From Example 5.3.6c, it is clear that for any model order, all three remaining fixed model terms are important in explaining the pattern of the data.

For this data set, where the estimated numbers of residual degrees of freedom are quite large, the F probabilities can be expected to be reasonably reliable. For smaller data sets, or when the F statistics cannot be calculated, the use of a likelihood-based test statistic may be preferable.

A likelihood ratio test statistic for fixed model terms using REML has been proposed by Welham & Thompson (1997) and can be calculated using the `REML` directive when `METHOD=Fisher`. Unlike linear regression, the difference in log-likelihoods between two nested fixed models does not give a sensible test statistic. This is because it is the residual likelihood *RL*, the likelihood of the data after projection into the residual space, that is maximized rather than the likelihood of the original data. For the residual likelihood, two different fixed models correspond to two different projections and, hence, effectively to two different data sets on which the same random terms are estimated. The statistic proposed by Welham and Thompson can be used to test a fixed model against a nested sub-model. The method calculates the likelihood for the full fixed model as usual. The same projection is then used for the sub-model and fixed effects to be dropped in the sub-model are constrained to be zero. This gives log-likelihoods calculated from the same projected data-set, using the same random model, but with some fixed effects constrained to zero for the sub-model. The difference in log-likelihoods therefore gives a likelihood ratio test in the usual way, where $-2(RL-RL_0)$ is the test statistic which has an asymptotic χ^2 distribution with degrees of freedom equal to the degrees of freedom of the fixed model terms constrained to be zero in the sub-model.

Simulations have indicated that for small samples this statistic tends to be slightly conservative, that is, it gives a significant test statistic slightly less often than would be expected when the null hypothesis is true.

You can obtain likelihood ratio test statistics by using the `SUBMODEL` option of `REML` to define the nested sub-model that is to be fitted and compared to the full fixed model. In other words,

the sub-model is the full model with the terms of interest dropped out. These tests are available only with the Fisher estimation method, and so they cannot be calculated when variance models are being fitted (see Section 5.4). For our example above, we would first try dropping the `Dose.Sex` interaction. Some constant terms are omitted from the calculation of the deviances by `REML`, and so the absolute values of the deviances are not usable; in fact, as shown in the example, the printed deviance may even be negative. However, it is only the difference between the deviance, printed in the Change line, that is of interest (and here the omitted constants will have cancelled out).

Example 5.3.6d

```
 19  VCOMPONENTS [FIXED=Littersize+Dose*Sex] RANDOM=Dam/Pup
 20  REML [PRINT=deviance; METHOD=Fisher;\
 21    SUBMODEL=Littersize+Dose+Sex] Weight

Deviance: -2*Log-Likelihood
---------------------------

Submodel:          Constant + Littersize + Dose + Sex
Full fixed model:  Constant + Littersize + Dose + Sex + Dose.Sex

Source             deviance  d.f.
Submodel          -173.6844   315
Full model        -174.4796   313
Change               0.7952     2
```

The inference from the change in deviances is the same as that from the F statistic, again suggesting that the `Dose.Sex` interaction is not important in explaining the pattern of the data. The `Dose.Sex` interaction can thus be removed from the model, and the other fixed model terms can then be dropped in turn to assess their importance.

The option `RECYCLE` is very useful for saving computing time when testing a series of sub-models like this. Ordinarily, each time the `REML` directive is used with the `SUBMODEL` option set, two runs of the algorithm are made: one to estimate the full model and one to estimate the sub-model. Clearly, for subsequent sub-models, the only new information required is from the sub-model run. The `RECYCLE` option is used to specify that only the sub-model run is to be made and the remainder of the information is to be picked up from the save structure. If no save structure is specified, the save structure from the most recent `REML` analysis is used automatically. Note that if you have analysed several y-variates using a single `REML` statement, then unless you specify a save structure for each y-variate (using the `SAVE` parameter), only the information from the last y-variate specified will be available. So if the pointer `Y` held 4 variates to analyse, you would need to use statements of the form

```
REML [PRINT=deviance; SUBMODEL=Sub1] Y[]; SAVE=S[1...4]
& [RECYCLE=yes; SUBMODEL=Sub2] Y[]; SAVE=S[]
```

to get the test statistics for the two submodels for each of the variates.

In Example 5.3.3e, only one variate is analysed, so there is no need to specify the save structure.

Example 5.3.6e

```
 22  VCOMPONENTS [FIXED=Littersize+Dose+Sex] RANDOM=Dam/Pup
 23  REML [PRINT=deviance; METHOD=Fisher; SUBMODEL=Dose+Sex] Weight

Deviance: -2*Log-Likelihood
---------------------------

Submodel:          Constant + Dose + Sex
Full fixed model:  Constant + Dose + Sex + Littersize
```

```
Source              deviance  d.f.
Submodel             -154.85   316
Full model           -182.37   315
Change                 27.52     1

  24  & [PRINT=deviance; SUBMODEL=Littersize+Dose; RECYCLE=yes] Weight

Deviance: -2*Log-Likelihood
---------------------------

Submodel:          Constant + Littersize + Dose
Full fixed model:  Constant + Dose + Sex + Littersize

Source              deviance  d.f.
Submodel             -129.91   316
Full model           -182.37   315
Change                 52.46     1

  25  & [PRINT=deviance; SUBMODEL=Littersize+Sex; RECYCLE=yes] Weight

Deviance: -2*Log-Likelihood
---------------------------

Submodel:          Constant + Littersize + Sex
Full fixed model:  Constant + Dose + Sex + Littersize

Source              deviance  d.f.
Submodel             -166.54   317
Full model           -182.37   315
Change                 15.84     2
```

Again, the results agree with the F statistics, and it seems that all the remaining terms in the fixed model are important in explaining the data.

You can specify a sub-model consisting of the constant term alone by using the string `'Constant'`: that is by putting `SUBMODEL='Constant'`. The string is case-insensitive: any combination of upper and lower case within the string is accepted.

The use of `CONSTRAINTS=positive` in a `VCOMPONENTS` statement may lead to biased results when testing sub-models, since the omission of an important fixed model term often leads to negative estimates of variance components. A warning is given if the constraints have to be enforced when fitting the sub-model, and it is then recommended that the analysis be rerun with parameter setting `CONSTRAINTS=none`.

Other proposals have been made for the testing of fixed effects using REML estimation procedures. Several of these are based on estimating the full fixed model, fixing the values of the gammas, and then estimating the nested sub-model. The change in residual sum of squares under this procedure is equivalent to the Wald statistic. The change in log-likelihood under this procedure may also give a useful test statistic. These statistics can be constructed by fitting several models and fixing the gammas using the `INITIAL` and `CONSTRAINTS` parameters of the `VCOMPONENTS` directive (5.2.4) then saving the required values from the `REML` analysis using the `VKEEP` directive (5.9.1).

Another, more recent, strategy is to use bootstrapping techniques.

VBOOTSTRAP procedure

Performs a parametric bootstrap of the fixed effects in a `REML` analysis (C.J. Brien & R.W. Payne).

Options

`PRINT` = *string tokens* — Controls printed output (`observedteststatistics`, `pvalues`, `vdiagnostics`, `nnotconverged`, `monitoring`, `all`, `ownstatistics`); default `obse`,

	`pval`
`VPRINT` = *string tokens*	Controls the output from the `REML` analysis of each sample (`model`, `components`, `effects`, `means`, `stratumvariances`, `monitoring`, `vcovariance`, `deviance`, `Waldtests`, `missingvalues`, `covariancemodels`); default `*` i.e. none
`PLOT` = *string*	What to plot (`histogram`); default `*`
`NBOOT` = *scalar*	Number of bootstrap samples to take; default 99
`NRETRIES` = *scalar*	Maximum number of extra samples to take when some `REML` analyses fail to converge; default `NBOOT`
`SEED` = *scalar*	Seed for random number generation; default 0 continues an existing sequence or, if none, selects a seed automatically
`METHOD` = *string token*	Indicates whether to use the standard Fisher-scoring algorithm or the new AI algorithm with sparse matrix methods (`Fisher`, `AI`); default `AI`
`MAXCYCLE` = *scalar*	Sets a limit on the number of iterations in the `REML` analyses; default 30
`FMETHOD` = *string token*	Controls whether and how to calculate F statistics for fixed terms (`automatic`, `none`, `algebraic`, `numerical`); default `none`
`WMETHOD` = *string token*	Controls which Wald statistics are saved (`add`, `drop`); default `add`
`WORKSPACE` = *scalar*	Number of blocks of internal memory to be set up for use by the `REML` algorithm
`OWNMETHOD` = *string token*	Type of test required for own statistics (`twosided`, `greaterthan`, `lessthan`); default `twos`
`CIPROBABILITY` = *scalar*	Probability level for the confidence interval for own statistics; default 0.95

Parameters

`SAVE` = *REML save structures*	Specifies the (`REML`) save structure of the original analysis; default `*` uses the `SAVE` structure from the most recent `REML` analysis
`UMEANS` = *variates*	Specifies the expected values for the units under the null hypothesis of no effects from the `FIXEDTERMS`
`UVCOVARIANCE` = *symmetric matrices*	Specifies the variances and covariances of the units under the null hypothesis of no effects from the `FIXEDTERMS`
`FIXEDTERMS` = *formula*	Specifies the fixed terms to test; default `*` tests all the fixed terms in the original analysis
`FSTATISTICS` = *pointers*	Saves a pointer with a variate for each of the `FIXEDTERMS`, containing the F statistics from the bootstrap samples
`PVALUES` = *pointers*	Saves a pointer with a scalar for each of the `FIXEDTERMS`, containing the test probability obtained from the position of its F statistic within those from the bootstrap samples
`NNOTCONVERGED` = *scalars*	Saves the number of bootstrap samples whose `REML` analysis failed to converge

OWNDATA = *pointers*	Data required to calculate own statistics
OWNOBSERVEDVALUES = *variates*	Saves observed values of the own statistics
OWNPROBABILITIES = *variates*	Saves bootstrap probabilities for the own statistics
OWNESTIMATES = *variates*	Saves boostrap estimates for the own statistics
OWNSES = *variates*	Saves boostrap standard errors for the own statistics
OWNLOWERCIS = *variates*	Saves boostrap lower values of the confidence intervals for the own statistics
OWNUPPERCIS = *variates*	Saves boostrap upper values of the confidence intervals for the own statistics
OWNSTATISTICS = *pointers*	Saves the own statistics obtained from the bootstrap samples, in a pointer with a variate for each statistic

VBOOTSTRAP performs a parametric bootstrap for fixed effects in a REML analysis. The model to be fitted must be defined using the VCOMPONENTS and VSTRUCTURE directives, in the usual way. The SAVE parameter supplies the save structure from the original analysis; if this is not set, the most recent REML analysis is used.

The bootstrap samples are generated from a multivariate Normal distribution with dimension equal to the number of units in the analysis. The UMEANS parameter supplies the expected values for the distribution, Usually, this contains the fitted values under the null model for the terms being tested. If UMEANS is not set, a variate containing the grand mean of the response is used. The UVCOVARIANCE parameter supplies the variances and covariances of the units. If this is not set, the unit-by-unit variance-covariance matrix from the original analysis is used (see the UVCOVARIANCE option of VKEEP). Note: you can use the VUVCOVARIANCE procedure to form the variance-covariance matrix, if you know the variance components for a REML model that contains no covariance models.

By default all the fixed terms in the original analysis are tested simultaneously. However, you can set the FIXEDTERMS parameter to test a smaller model, and you should then also set UMEANS to specify the expected values under the null model.

The NBOOT option specifies the number of bootstrap samples to take (default 99). The NRETRIES option specifies the maximum number of extra samples to take when some REML analyses fail to converge; the default is to use the same number as specified by NBOOT. The SEED option supplies the seed for the random number generator used to make the permutations; default 0 continues from the previous generation or (if none) initializes the seed automatically. The NNOTCONVERGED parameter can save the number of samples whose analyses did not converge, in a scalar.

The bootstrap p-values are calculated by taking the proportion of F statistics in the bootstrap samples that are larger than the observed F statistic of each fixed term. The WMETHOD option controls whether these statistics are obtained from the table where terms are added sequentially (the default), or from the table where suitable terms are dropped from the full fixed model. Note that, if you use the table where terms are dropped, the only terms that can be tested are those that are not marginal to any other term in the fixed model: for example, the main effect A cannot be tested if the model contains an interaction, such as A.B.

The bootstrap F statistics can be saved, in a pointer with a variate for each of the FIXEDTERMS, using the FSTATISTICS parameter. The p-values can be saved, in a pointer with a scalar for each of the FIXEDTERMS, using the PVALUES parameter. You can obtain a plot of a histogram showing the position of the observed F statistic, compared to those from the bootstrap samples, by setting option PLOT=histogram.

You can define your own statistics to be assessed by the bootstrap. They are calculated by a procedure _VBOOTownstatistics, which is called by VBOOTSTRAP following the REML analysis of each bootstrap sample. The information required by _VBOOTownstatistics to do the calculations is supplied, in a pointer, by the OWNDATA parameter. The OWNMETHOD option

specifies the type of test to be made. The default, `twosided` tests whether the statistics differ from zero. The `greaterthan` setting tests whether they are greater than zero, and the `lessthan` setting tests whether they are less than zero. Bootstrap estimates, standard errors and confidence intervals are also calculated, The `CIPROBABILITY` option specifies the probability for the confidence intervals (default 0.95). The `OWNOBSERVEDVALUES` parameter can save a variate containing the values of the own statistics from the original data set. The `OWNPROBABILITIES` can save a variate containing the probabilities from the tests. The `OWNESTIMATES` can save a variate containing the bootstrap estimates of the statistics (calculated as the mean of the values obtained from the bootstrap samples) The `OWNSES` can save a variate containing standard errors of bootstrap estimates. The `OWNLOWERCIS` and `OWNUPPERCIS` parameters can save variates containing the lower and upper values, respectively, of the confidence intervals. Finally, the `OWNSTATISTICS` can save the values of the own statistics obtained from the bootstrap samples, in a pointer with a variate for each statistic. The assessment of own statistics is shown in Example 5.3.6h with the `VPERMTEST` procedure, which has a similar subsidiary procedure `_VPERMownstatistics`. They are also assessed in the `VBOOTSTRAP` example, which can be modified to calculate your own statistics instead.

Printed output is controlled buy the `PRINT` option, with settings:

`observedteststatistics`	to print the values of the observed Wald or F statistics for the fixed terms in the original `REML` analysis,
`pvalues`	to print the bootstrap p-values of the observed Wald or F statistics for the fixed terms,
`vdiagnostics`	to print the diagnostics from the `REML` analyses performed on the bootstrap samples,
`nnotconverged`	to print the number of samples whose analyses did not converge,
`ownstatistics`	to print the estimates, standard errors and confidence intervals for the own statistics,
`monitoring`	to print the progress of the bootstrapping, and
`all`	to print all the information.

By default, the observed statistics and the p-values are printed.

The `VPRINT` option controls the output from the `REML` analyses of the bootstrap samples, with the same settings as the `PRINT` option of `REML`. By default, nothing is printed.

The `MAXCYCLE` option sets a limit on the number of iterations in the `REML` analyses (default 30). The `METHOD` option controls whether `REML` uses the standard Fisher-scoring algorithm, or the new AI algorithm with sparse matrix methods (the default). The `FMETHOD` option controls whether and how to calculate F statistics for fixed terms; the default is not to calculate the statistics. (This is relevant if tests for fixed effects are being printed in the `REML` analyses of the bootstrap samples.) The `WORKSPACE` option specifies the number of blocks of internal memory to be set up for use by the `REML` algorithm; the default is to use the same value as in the original `REML` analysis.

Example 5.3.3f uses bootstrapping to test the Dose.Sex interaction is needed in model.

Example 5.3.6f

```
" To perform a bootstrap test for the interaction Dose.Sex:
  1) fit full model to get variances & covariances of the units;"
VCOMPONENTS [FIXED=Littersize+Dose*Sex] RANDOM=Dam/Pup
REML        [PRINT=*] Weight; SAVE=fullfixed
VKEEP       [UVCOVARIANCE=V]
" 2) fit a model with no interaction, and get the fitted values;"
VCOMPONENTS [FIXED=Littersize+Dose+Sex] RANDOM=Dam/Pup
REML        [PRINT=*] Weight; FITTEDVALUES=fit
" 3) parameteric bootstrap to test the interaction."
```

```
 35  VCOMPONENTS [FIXED=Littersize+Dose*Sex] RANDOM=Dam/Pup
 36  VBOOTSTRAP  [NBOOT=999; SEED=265600] SAVE=fullfixed;\
 37              UMEANS=fit; UVCOVARIANCE=V; FIXEDTERMS=!f(Dose.Sex)

Observed test statistics
------------------------


              Numerator df  Observed F Observed Wald Wald p-value
      Source
    Dose.Sex             2      0.3984        0.7968       0.6714


Parametric bootstrap p-values from 999 samples
----------------------------------------------


                 p-values
      Source
    Dose.Sex       0.6660
```

The results confirm the earlier conclusion, that there is no evidence to suggest that there is an interaction.

Bootstrapping can also be used to estimate the true critical values to be used for the Wald and F tests. These take account of the biases in the statistics, discussed earlier in this section.

VCRITICAL procedure

Uses a parametric bootstrap to estimate critical values for a fixed term in a REML analysis (R.W. Payne & C.J. Brien).

Options

PRINT = *string tokens*	Prints the critical values (critical, fcritical, tcritical, wcritical); default crit, fcri, tcri, wcri
VPRINT = *string tokens*	Controls the output from the REML analyses (model, components, effects, means, stratumvariances, monitoring, vcovariance, deviance, Waldtests, missingvalues, covariancemodels); default * i.e. none
TERM = *formula*	Fixed term to be tested
UMEANS = *variate*	Specifies the expected values for the units under the null hypothesis of no effects from the TERM; default is to use the constant from the SAVE structure
UVCOVARIANCE = *symmetric matrix*	Specifies the variances and covariances of the units under the null hypothesis of no effects from the TERM; default is to take this from the SAVE structure
WCRITICAL = *variate*	Saves the critical values of the Wald statistic
FCRITICAL = *variate*	Saves the critical values of the F statistic
NBOOT = *scalar*	Number of bootstrap samples to take; default 99
NRETRIES = *scalar*	Maximum number of extra samples to take when some REML analyses fail to converge; default NBOOT
SEED = *scalar*	Seed for random number generation; default 0 continues an existing sequence or, if none, selects a seed automatically

PROBABILITIES = *scalar* or *variate*	Significance levels for which critical values are required; default 0.05
METHOD = *string token*	Indicates whether to use the Fisher-scoring algorithm or the AI algorithm with sparse matrix methods (Fisher, AI); default AI
MAXCYCLE = *scalar*	Sets a limit on the number of iterations in the REML analyses; default 30
FMETHOD = *string token*	Controls how to calculate estimated denominator degrees of freedom when these are to be saved (automatic, none, algebraic, numerical); default auto
WMETHOD = *string token*	Controls which Wald statistics are saved (add, drop); default add
TMETHOD = *string token*	Type of test to be made for the contrasts (twosided, greaterthan, lessthan, equivalence, noninferiority); default twos
WALD = *variate*	Saves the Wald statistics from the samples
FSTATISTIC = *variate*	Saves the F statistics from the samples
NDF = *scalar*	Saves the numerator degrees of freedom for the Wald and F statistics
DDF = *variate*	Saves the estimated denominator degrees of freedom for the F statistics
NNOTCONVERGED = *scalar*	Saves the number of bootstrap samples whose REML analysis failed to converge
WORKSPACE = *scalar*	Number of blocks of internal memory to be set up for use by the REML algorithm
SAVE = *vsave*	REML save structure to provide the information about the analysis

Parameters

XCONTRASTS = *variates* or *tables*	X-variate defining a contrast to be detected
CONTRASTTYPE = *string tokens*	Type of contrast (regression, comparison) default rege
ESTIMATE = *variates*	Saves the estimated values of the contrasts from the samples
SE = *variates*	Saves the standard errors for the estimates of the contrasts from the samples
CRITICAL = *variates*	Saves the critical values for the contrasts
TCRITICAL = *variates*	Saves the critical values for the t-statistics of the contrasts

As mentioned, earlier in this secton, the conventional way to assess fixed terms in a REML analysis is to use either the Wald or the F tests, in the table of tests for fixed effects that is produced by setting option PRINT=wald in either REML or VDISPLAY. The Wald have the disadvantage of being biased, i.e. they tend to generate significant results too frequently. The F tests are more reliable. However, their denominator degrees of freedom need to be estimated, using the method of Kenward & Roger (1997), and this may not be feasible for some data sets. These denominator degrees of freedom can also be used in t-tests to assess contrasts amongst the effects of a term; see procedure VTCOMPARISONS. However, those tests must be used with caution, as the degrees of freedom are relevant for assessing the fixed term as a whole, and may

differ over the various contrasts.

VCRITICAL provides an alternative method of assessment, that may be useful if the decision from the conventional tests is not clear-cut, or if contrasts are to be assessed. It uses a parametric bootstrap, in the same way as the VBOOTSTRAP procedure. However, it differs from VBOOTSTRAP, in that it generates critical values, rather than assessing the significance of terms in a specific data set. These critical values can be used test hypotheses with a specific data set, and the critical values for the F, Wald and t-statistics may be useful with similar data sets. The critical values for the t-statistics also allow you to determine the size of the contrast that may be detectable in these investigations.

As in VBOOTSTRAP, the model to be fitted must be defined using the VCOMPONENTS and VSTRUCTURE directives. The bootstrap samples are generated from a multivariate Normal distribution with dimension equal to the number of units in the analysis. The UMEANS option supplies the expected values for the distribution. This should contain the fitted values under the null model for the term being tested. The UVCOVARIANCE option supplies the variances and covariances of the units. If either UMEANS or UVCOVARIANCE is not specified, defaults are taken from the REML analysis supplied by the SAVE option, or from the most recent REML if SAVE is not set. For UMEANS the default is a variate containing the constant estimated in that analysis. For UVCOVARIANCE it is the unit-by-unit variance-covariance matrix from the analysis (see the UVCOVARIANCE option of VKEEP). Note: you can use the VUVCOVARIANCE procedure to form the variance-covariance matrix, if you know the variance components for a REML model that contains no covariance models.

The NBOOT option specifies the number of bootstrap samples to take (default 99). The NRETRIES option specifies the maximum number of extra samples to take when some REML analyses fail to converge; the default is to use the same number as specified by NBOOT. The SEED option supplies the seed for the random number generator used to form the samples; default 0 continues from the previous generation or (if none) initializes the seed automatically. The NNOTCONVERGED option can save the number of samples whose analyses did not converge, in a scalar.

The fixed term to be assessed is specified by the TERM option. If the term is a main effect (i.e. if TERM contains just one factor) you can use the XCONTRASTS parameter to specify variates or tables containing the coefficients defining the contrasts amongst the effects of the term. The CONTRASTTYPE option indicates whether each of these is a regression contrast (as specified in analysis of variance by the REG function) or a comparison (as specified by the COMPARISON function).

The TMETHOD option specifies the type of test that is to be used to assess the contrasts, with the following settings.

twosided	assumes a two-sided test to assess whether the contrast differs from zero (default).
lessthan	assumes a one-sided test to assess whether the contrast is less than zero.
greaterthan	assumes a one-sided test to assess whether the contrast is greater than zero.
noninferiority	assumes a test to check that the contrast is not significantly less then zero. (See Method for more details.)
equivalence	assumes a one-sided test to check that the contrast does not differ significantly from zero; see Method for more details.

The PROBABILITIES option specifies the significance levels for which you want to obtain critical values; the default is 0.05, i.e. 5%.

Printed output is controlled buy the PRINT option, with the following settings.

critical	prints critical values for the contrasts,
fcritical	prints critical values for the F statistics,

`tcritical`	prints critical values for the t-statistics of the contrasts,
`wcritical`	prints critical values for the Wald statistics,
`nnotconverged`	prints the number of bootstrap samples whose analysis failed to converge, and
`monitoring`	prints monitoring information, showing the progress of the bootstrap sampling.

By default, all the critical values printed.

The `VPRINT` option controls the output from the `REML` analyses of the bootstrap samples, with the same settings as the `PRINT` option of `REML`. By default, nothing is printed.

The critical values for the contrasts and their t-statistics can be saved, in variates, by the `CRITICAL` and `TCRITICAL` parameters, respectively. The critical values for the F and Wald statistics can be saved, again in variates by the `FCRITICAL` and `WCRITICAL` options.

You can also save the values estimated for the various statistics, in the analyses of the bootstrap samples, in variates (with a unit for each sample). Those for the contrasts and their standard errors can be saved the `ESTIMATES` and `SE` parameters, respectively. The F and Wald statistics can be saved by the `FSTATISTIC` and `WALD` options. The degrees of freedom for the Wald statistics and numerator degrees for the F statistics can be saved, in a scalar, using the `NDF` option. The estimated denominator degrees of freedom for the F tests can be saved, in a variate, using the `DDF` option.

The `MAXCYCLE`, `METHOD`, `WMETHOD`, `FMETHOD` and `WORKSPACE` option control various aspects of the `REML` analyses, as in `VBOOTSTRAP`.

Example 5.3.3g continues from Example 5.3.6f, and uses bootstrapping to estimate critical values for the Dose.Sex interaction. The bias in the Wald statistic is demonstrated by the difference between the value of 6.598, estimated by `VCRITICAL` for the 5% critical value of the Wald statistic, and the value 5.991 calculated from a standard Chi-square distribution.

Example 5.3.6g

```
 38  " Parametric bootstrap to get critical values for Dose.Sex."
 39  VCRITICAL   [PRINT=critical,fcritical,tcritical,wcritical; NBOOT=999;\
 40              SEED=36820; UMEANS=fit; UVCOVARIANCE=V; TERM=Dose.Sex]

Critical values
===============

Term: Dose.Sex
Probability: 0.05
F-statistic: 3.299
Wald statistic: 6.598

 41  CALCULATE   Wcrit = EDCHISQUARE(0.95; 2)
 42  PRINT       Wcrit; DECIMALS=3

      Wcrit
      5.991
```

Another way to assess the fixed model, similar but less complicated than bootstrapping, is to do a random permutation test.

VPERMTEST procedure

Does random permutation tests for the fixed effects in a `REML` analysis (R.W. Payne).

Options

`PRINT` = *string tokens*	Controls printed output (`prwald`, `criticalwald`, `ownstatistics`, `monitoring`); default `prwa`, `crit`

NTIMES = *scalar*	Number of permutation samples to make; default 99
NRETRIES = *scalar*	Maximum number of extra samples to take when some REML analyses fail to converge; default NTIMES
BLOCKSTRUCTURE = *formula*	Model formula defining any blocking to consider during the randomization; default none
EXCLUDE = *factors*	Factors in the block formula whose levels are not to be randomized
SEED = *scalar*	Seed for random number generation; default 0 continues an existing sequence or, if none, selects a seed automatically
WMETHOD = *string token*	Controls which Wald statistics are used (add, drop); default add
OWNMETHOD = *string token*	Type of test required for own statistics (twosided, greaterthan, lessthan); default twos
CIPROBABILITY = *scalar*	Probability level for the confidence interval for own statistics; default 0.95

Parameters

SAVE = *REML save structures*	Specifies the (REML) save structure of the original analysis; default * uses the SAVE structure from the most recent REML analysis
WALD = *pointers*	Wald statistics saved in a pointer with a variate for each term
PRWALD = *pointers*	Critical values for Wald statistics saved in a pointer with a scalar for each term
CRITICALWALD = *pointers*	Saves a pointer with variates for the 5%, 1% and 0.1% significance levels containing the corresponding critical values for the fixed terms, obtained from the quantiles of the Wald statistics from the permuted data sets
NNOTCONVERGED = *scalars*	Saves the number of permutations whose REML analysis failed to converge
OWNDATA = *pointers*	Data required to calculate own statistics
OWNOBSERVEDVALUES = *variates*	Saves observed values of the own statistics
OWNPROBABILITIES = variates	Saves probabilities for the own statistics
OWNESTIMATES = *variates*	Saves estimates for the own statistics
OWNSES = *variates*	Saves standard errors for the own statistics
OWNLOWERCIS = *variates*	Saves lower values of the confidence intervals for the own statistics
OWNUPPERCIS = *variates*	Saves upper values of the confidence intervals for the own statistics
OWNSTATISTICS = *pointers*	Saves the own statistics obtained from the permutation samples, in a pointer with a variate for each statistic

VPERMTEST performs a random permutation test for fixed effects in a REML analysis. The SAVE parameter can supply the save structure from the original analysis; if this is not set, the tests are done for the most recent REML analysis.

The test probabilities are calculated by taking the proportion of Wald statistics in the permutation samples that are larger than the observed Wald statistic of each fixed term. (As a result these should not suffer from the bias that is found in the probabilities for the Wald statistics themselves, which tend to be too low.) The WMETHOD option controls whether the Wald statistics are obtained from the table where terms are added sequentially (the default), or from

the table where suitable terms are dropped from the full fixed model. Note that, if you use the table where terms are dropped, the only terms that can be tested are those that are not marginal to any other term in the fixed model: for example, the main effect A cannot be tested if the model contains an interaction, such as A.B.

The NTIMES option defines how many random permutations to perform; by default there are 99 (as well as the "null" permutation where the data keep their original order). The NRETRIES option specifies the maximum number of extra samples to take when some REML analyses fail to converge; the default is to use the same number as specified by NTIMES. The SEED option allows you to specify the seed to use for the random-number generator that is used to construct the permutation samples. The default, SEED=0, continues the sequence of random numbers from a previous generation or, if this is the first use of the generator in this run of Genstat, it initializes the seed automatically. If NTIMES exceed the maximum possible number of permutations for the data, an "exact" test is performed in which every permutation is used once. This is feasible only for small datasets. There are *n*! (n factorial) permutations of *n* units: 3!=6, 4!=24, 5!=120, 6!=720, 7!=5040, 8!=40320, and so on. The NNOTCONVERGED parameter can save the number of samples whose analyses did not converge, in a scalar.

If the data are from a designed experiment, you may need to use the BLOCKSTRUCTURE option to specify a block model to define how to do the randomization. The EXCLUDE option can then restrict the randomization so that one or more of the factors in the block model is not randomized. See the RANDOMIZE directive for further details.

Output is controlled by the PRINT option, with settings:

prwald	to print the probabilities calculated from the distribution of the Wald statistics;
criticalwald	to print a table giving estimated critical values for each of the Wald statistics, formed from the permutation samples;
ownstatistics	to print the estimates, standard errors and confidence intervals for the own statistics, and
monitoring	to monitor the progress of the test.

The Wald statistics from the permutation tests can be saved, in a pointer with a variate for each of the FIXEDTERMS, using the WALD parameter. The probabilities calculated from the tests can be saved, in a pointer with a scalar for each of the FIXEDTERMS, using the PRWALD parameter.

As in VBOOTSTRAP, you can define your own statistics to be assessed by the test. For VPERMTEST they are calculated by a procedure _VPERMownstatistics, which is called by VPERMTEST following the REML analysis of each permutation sample. Its use is shown in the VPERMTEST example, which can be modified to calculate your own statistics instead. The information required by _VPERMownstatistics to do the calculations is supplied, in a pointer, by the OWNDATA parameter. The OWNMETHOD option specifies the type of test to be made. The default, twosided tests whether the statistics differ from zero. The greaterthan setting tests whether they are greater than zero, and the lessthan setting tests whether they are less than zero. Standard errors and confidence intervals are also calculated, The CIPROBABILITY option specifies the probability for the confidence intervals (default 0.95). The OWNOBSERVEDVALUES parameter can save a variate containing the values of the own statistics from the original data set. The OWNPROBABILITIES can save a variate containing the probabilities from the tests. The OWNESTIMATES can save a variate containing the bootstrap estimates of the statistics (calculated as the mean of the values obtained from the bootstrap samples) The OWNSES can save a variate containing standard errors of bootstrap estimates. The OWNLOWERCIS and OWNUPPERCIS parameters can save variates containing the lower and upper values, respectively, of the confidence intervals. Finally, the OWNSTATISTICS can save the values of the own statistics obtained from the bootstrap samples, in a pointer with a variate for each statistic.

The maximum number of iterations (MAXCYCLE) and number of blocks of internal memory to

be (WORKSPACE) to be used in the REML analyses can be set by a call to the VAOPTIONS procedure before you use VPERMTEST.

Example 5.3.6h continues from Example 5.3.6g, and uses VPERMTEST to do a permutation test to assess the fixed terms. It also defines procedure _VPERMownstatistics to assess the contrasts between the low dose of Drug and control and between the high dose of Drug and control. The setting of the OWNDATA parameter of VPERMTEST is the pointer owninfo. This provides the setting of the DATA parameter of _VPERMownstatistics, and provides the information needed to calculate the own statistics. The estimated contrasts for each permutation sample are made available for VPERMTEST, as a variate, through the STATISTICS parameter of _VPERMownstatistics. The first element of owninfo. is the factor Dose, whose levels are to be compared in the contrasts. The second element provides labels for the STATISTICS variate. The third and fourth elements define the contrasts. The results confirm the strong evidence for differences between the sexes and doses. The own statistics indicate that the low and high doses do each differ from the control.

Example 5.3.6h

```
 43  VCOMPONENTS [FIXED=Littersize+Dose*Sex] RANDOM=Dam/Pup
 44  REML        [PRINT=*] Weight
 45  " Random permutation test with own statistics
-46    to test differences of drugs from control "
 47  PROCEDURE    [WORDLENGTH=long] '_VPERMownstatistics'
 48  PARAMETER      NAME=\
 49    'DATA',      "(I: pointer) information to calculate the statistics"\
 50    'STATISTICS';"(O: variate) estimated statistics"\
 51    MODE=p; TYPE='pointer','variate'
 52    " insert commands to calculate the statistics "
 53    VKEEP      DATA[1]; MEANS=means
 54    VARIATE    vmeans; VALUES=means
 55    VARIATE    [NVALUES=DATA[2]] STATISTICS
 56    CALCULATE  STATISTICS = vmeans$[DATA[3]] - vmeans$[DATA[4]]
 57  ENDPROCEDURE "_VPERMownstatistics"
 58  TEXT         [VALUES='Low - C','High - C'] Contrast
 59  POINTER      [VALUES=Dose,Contrast,!(1,1),!(2,3)] Owninfo
 60  VPERMTEST    [PRINT=prwald,ownstatistics,ownstatistics; SEED=357192]\
 61               OWNDATA=Owninfo

Probabilities for Wald statistics
---------------------------------

      Source
  Littersize       0.010
        Dose       0.010
         Sex       0.010
    Dose.Sex       0.670

(determined from 99 random permutations)


Own statistics
--------------

                Observed
               statistic     p-value
    Contrast
     Low - C      0.4178       0.010
    High - C      0.8510       0.010

              Permutation
                 estimate        s.e.   Lower 95%   Upper 95%

     Low - C   -0.002338     0.07673     -0.1466      0.1729
    High - C   -0.001882     0.11290     -0.2133      0.2271
```

When a fixed model contains many terms, it can be very time-consuming to determine which ones are genuinely required. The VSCREEN procedure may then be useful.

VSCREEN procedure

Performs screening tests for fixed terms in a REML analysis (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (ftests, waldtests); default ftes, wald
EXCLUDEHIGHER = *string token*	Whether to exclude higher-order interactions in the conditional models (yes, no); default no
FORCED = *formula*	Terms that must always be included in the model (no tests on these terms); default *
FSAVE = *pointer*	Saves the F tests
WSAVE = *pointer*	Saves the Wald tests
SAVE = *REML save structure*	Specifies the analysis whose fixed terms are to be tested; by default this will be the most recent REML

No parameters

VSCREEN calculates marginal and conditional tests for fixed terms in a REML analysis. By default, these are from the recent REML analysis. However, you can take an earlier analysis, by using the SAVE option of VSCREEN to specify its save structure (saved using the SAVE parameter of the earlier REML command).

In the marginal test, the term is added to the simplest possible model. For example, the main effect of A would be added to the null model, and the interaction A.B would be added to a model containing only the main effects A and B.

In the conditional test, the term is added to the most complex possible model that contains no terms involving the term to be tested. For example, interaction A.B would be added to the model containing all terms except those involving A.B (such as the interaction A.B.C). By default, the most complex model includes terms with more factors or variates than the term being tested. For example, the interaction C.D.E would be included when testing A.B. You can exclude these higher-order terms by setting option EXCLUDEHIGHER=yes (and VSCREEN will print a message to remind you that this has been done).

You can specify terms that should always be included in the model by using the FORCED option. These terms are fitted first, and are not tested.

The PRINT option controls printed output, with the following settings.

ftests	presents F statistics for the terms. If denominator degrees of freedom (ddf) are available from the earlier REML analysis, probabilities are also given. Note, however, that these ddf are correct only for models that correspond to those in the sequential Wald table in the REML analysis. They should be acceptable for the other models, but you should be cautious when probabilities are close to critical values.
waldtests	presents Wald statistics for the terms. These suffer from the usual biases of Wald tests in REML analyses, and so should again be used with caution.

You can save the results of the F tests and the Wald tests, in pointers, using the FSAVE and WSAVE options, respectively. The elements of the pointers are labelled by the headers of the columns used in the printed output.

Example 5.3.6i calculates screening tests for the fixed model originally fitted in Example 5.3.6a. This shows that Littersize, Dose and Sex are all required in the fixed model.

Example 5.3.6i

```
  62  VSCREEN

Screening tests for fixed effects
---------------------------------


Fixed term    d.f.   Marginal     pr.   Conditional     pr.
                   Wald test            Wald test
Littersize       1      27.99  <0.001         46.25  <0.001
      Dose       2       9.91   0.007         23.01  <0.001
       Sex       1      55.50  <0.001         57.96  <0.001
  Dose.Sex       2       1.24   0.537          0.80   0.671


Fixed term  n.d.f.  d.d.f.   Marginal     pr.   Conditional     pr.
                             F test               F test
Littersize       1      31      27.99  <0.001         46.25  <0.001
      Dose       2      24       4.96   0.016         11.51  <0.001
       Sex       1     300      55.50  <0.001         57.96  <0.001
  Dose.Sex       2     302       0.62   0.538          0.40   0.672
```

An advantage of using VSCREEN to assess the fixed model, rather than running a succession of REML analyses with different fixed models, is that the fixed terms are assessed against identical estimates of the random variation (as in an analysis of variance). When terms are dropped from (or added to) the fixed model in a REML analysis, the random variation will change. For example, it will increase if a term with a Wald statistics greater than its number of degrees of freedom is dropped. It may therefore be difficult to reach consistent decisions about which fixed terms are genuinely required. Similar methods are used by the VALLSUBSETS procedure, which fits all subsets of the fixed terms in a REML analysis. Likewise they are used by VRFIT and its associated procedures. These allow you investigate the fixed model by fitting and modifying subsets of the terms, in a similar way to FIT and its associated directives (3.1, 3.2).

Once you have used VSCREEN to decide which terms to keep in the fixed model, you can use only those terms for prediction, by specifying them in the MODEL option of VPREDICT (5.5.1).

5.3.7 Assessing random effects

VCHECK procedure

Checks standardized residuals from a REML analysis (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (largeresiduals, similarunits, stability); default larg
RMETHOD = *string token*	Which random terms to use when calculating the standardized residuals (final, all); default fina
RLIMIT = *scalar*	Limit for detection of large standardized residuals; if this is not set, the limit is set automatically according to the number of residual degrees of freedom
COMMONFACTORS = *factors*	Factors to define similar units; if this is not set, the factors in the fixed model are used
REPORTFACTORS = *factors*	Additional factors to include in the table of similar units

PROBABILITY = *scalar*	Critical value for the test probabilities to decide whether to generate warning messages from the Levine test for variance stability; default=0.025
NLARGERESIDUALS = *scalar*	Saves the number of large standardized residuals that have been detected
LARGERESIDUALUNITS = *variate*	Saves the unit numbers of the large standardized residuals
SIMILARINFORMATION = *pointer*	Saves details of large standardized residuals and residuals in similar units
STABILITYTEST = *pointer*	Saves the results of the Levene test for stability of the variance of the standardized residuals
SAVE = *REML save structure*	Specifies the analysis to be checked; by default this will be the most recent REML

No parameters

Procedure VCHECK performs some checks on the standardized residuals from a REML analysis. By default, these are taken from the recent REML analysis. However, you can check an earlier analysis, by using the SAVE option of VCHECK to specify its save structure (saved using the SAVE parameter of the earlier REML command).

The RMETHOD option controls which random terms are used to calculate the standardized residuals, with settings:

all	uses all of the random effects, and
final	uses only the final random term (default).

Output is controlled by the PRINT option, with the following settings.

largeresiduals	reports any large standardized residuals, with their unit numbers.
similarunits	reports large standardized residuals, together with the residuals from similar units.
stability	performs two Levene tests to check whether the residual variance differs according to the size of the response. The data are divided into three groups (small, intermediate and large) according to the sizes of their fitted values. The tests compare the variance of the standardized residuals in the first (small) group with those in the third (large) group, and the variance of the second (intermediate) group with the variance of other two groups combined..

By default PRINT=largeresiduals.

The RLIMIT option specifies the limit that must be exceeded by the absolute value of a standardized residual for it to be identified as large. If this is not set, the default is taken as 2.0 if the number of degrees of freedom d of the random terms in the REML analysis is less than 20, and 4.0 if d is greater than 15773. For other values of d, the default is the critical value of the Normal distribution for a two-sided test with significance probability $1/d$. These calculations are the same as those used in regression and analysis of variance, and are intended to ensure that a report should appear for any extreme outlier, but that reports should not appear too often just as a result of random variation.

The NLARGERESIDUALS option saves the number of large standardized residuals that have been found, and the LARGERESIDUALUNITS option can save a variate containing their unit numbers.

The COMMONFACTORS option lists the factors whose levels should be shared by the units that are listed in the report as similar to those with the large residuals. If this is not set, the default

is to take the factors in the fixed model. The REPORTFACTORS option lists any other factors that are to be included in the report. The SIMILARINFORMATION option can save a pointer containing details of the table that has been printed. The first element of the pointer, labelled 'Column labels', contains labels to use as column headings for the other elements, The second element, labelled 'Unit number', contains unit numbers. The third element, labelled 'Unit type', is a factor indicating whether each unit contains a large standardized residual, or the standardized residual from a similar unit. The remaining columns contain the values of the factors displayed in the report.

The results of the Levene test for stability of the variance of the standardized residuals can be saved, in a pointer, by the STABILITYTEST option.

If nothing is to be saved and no printed output is requested, VCHECK provides a safety check. It prints a warning message if any large standardized residuals are detected, or if either of the Levene tests generates a test probability less than or equal to the value specified by the PROBABILITY option. The default value is 0.025 (i.e. 2.5%), which is the same as the value used for the similar messages that may occur with the summary of analysis in regression of from procedure ACHECK following an analysis of variance. It is important to realise that the estimated residuals will be correlated. The Levene tests assume that the residuals are independent Normally-distributed observations. Their test probabilities may therefore be too low – and generate too many significant results. So the use of a smaller critical probability value provides some protection against spurious messages.

Example 5.3.7a examines the residuals from the analysis in Example 5.3.6e.

Example 5.3.7a

```
  63  VCHECK      [PRINT=largeresiduals,stability]

Large residuals
===============


        Unit    Residual
          56      -4.091
          58       3.020
          60       3.202
          66      -7.941
         227      -3.241


Levene tests for stability of variance
======================================


                                    Test t-statistic        d.f.         pr.
               Small vs. large responses       3.878      34.685      <0.001
Intermediate v.s. small & large responses      1.366     295.901       0.173
```

VCHECK has identified five units with large residuals. More worryingly, the Levene test shows strong evidence that the variance differs according to whether the observed weights are small or large. We will discuss this further at the end of this section. Next, though, we show how you can perform similar checks on other random terms.

VRCHECK procedure

Checks effects of a random term in a REML analysis (R.W. Payne).

Options

PRINT = *string tokens* Controls printed output (largeblups, stability);

	default `larg`
`TERM` = *formula*	Random term whose BLUPs are to be assessed; must be set
`RMETHOD` = *string token*	Which random terms to use to form the residuals that are subtracted from the y-variate to provide the fitted values (`all`, `term`); default `all`
`RLIMIT` = *scalar*	Limit for detection of large standardized BLUPs; if this is not set, the limit is set automatically according to the number of BLUPs
`NLARGEBLUPS` = *scalar*	Saves the number of large standardized BLUPs that have been detected
`LARGEBLUPUNITS` = *pointer*	Saves the factor levels of the large standardized BLUPs
`STABILITYTEST` = *pointer*	Saves the results of the Levene test for stability of the variance of the standardized BLUPs
`SAVE` = *REML save structure*	Specifies the analysis from which the BLUPs are to be taken; by default this will be the most recent `REML`

No parameters

Procedure `VRCHECK` checks effects (i.e. BLUPs) of a random term from a `REML` analysis. The `TERM` option must be set to specify the random term to check. By default, its BLUPs are taken from the recent `REML` analysis. However, you can use an earlier analysis, by using the `SAVE` option of `VRCHECK` to specify its save structure (saved using the `SAVE` parameter of the earlier `REML` command).

Output is controlled by the `PRINT` option, with the following settings.

`largeblups`	reports any large standardized BLUPs.
`stability`	performs two Levene tests to check whether the variance of the random term differs according to the size of the response. The BLUPs are divided into three groups (small, intermediate and large) according to the sizes of the corresponding fitted values. The tests compare the variance of the standardized BLUPs in the first (small) group with those in the third (large) group, and the variance of the second (intermediate) group with the variance of other two groups combined.

By default `PRINT=largeblups`.

The `RMETHOD` option specifies how to form the residuals that are subtracted from the y-variate to provide the fitted values. The available settings are:

`all`	uses all of the random effects (default), and
`term`	uses only the random term specified by the `TERM` option.

It is important to realise that the estimated BLUPs will be correlated. The Levene tests assume that they are independent Normally-distributed observations. Their test probabilities may therefore be too low – and generate too many significant results. They should thus be interpreted with care.

The `RLIMIT`, `NLARGERESIDUALS`, `LARGEBLUPUNITS` and `STABILITYTEST` options are the same as in `VCHECK`.

Example 5.3.7b examines the BLUPs for the random term `Dam` in Example 5.3.6e. Again several seem to be large.

Example 5.3.7b

```
  64  VRCHECK     [TERM=Dam]

Large BLUPs
===========


         Dam        BLUP        s.e.   BLUP/s.e.
           5      0.3460      0.1416       2.443
           7      0.3885      0.1557       2.496
           9     -0.6102      0.1485      -4.109
          13     -0.3728      0.1457      -2.558
          17     -0.4039      0.1410      -2.865
          18      0.4311      0.1427       3.021
          20      0.3180      0.1457       2.183
          22     -0.4921      0.1630      -3.020
```

The natural next step, if you think that you have some large residuals, may be to make a more rigorous assessment. The `VSOM` procedure uses a mixed-model analysis with a variance shift outlier model (VSOM) to search for potential outliers amongst the residuals or amongst the effects (BLUPs) of another random term. The model defines an extra component of variation for each unit (an individual or a group), in turn, and estimates the extra variance associated with it.

VSOM procedure

Analyses a simple `REML` variance components model for outliers using a variance shift outlier model (S.J. Welham, F.N. Gumedze & D.B. Baird).

Options

`PRINT` = *string tokens*	Specifies the output to be produced (`fdr`, `outliers`); default `fdr`, `outl`
`VPRINT` = *string tokens*	Controls the output from the `REML` analysis of the baseline model (`model`, `components`, `effects`, `means`, `stratumvariances`, `monitoring`, `vcovariance`, `deviance`, `Waldtests`, `missingvalues`, `covariancemodels`); default `mode`, `comp`, `Wald`, `cova`
`PLOT` = *string tokens*	Controls which plots are produced (`indexplots`, `residual`); default `inde`, `resi`
`INDEXPLOT` = *string tokens*	Selects the index plots to produce (`omega`, `sigma2`, `tsquared`, `lrt`, `method`, `all`); default `meth`
`RTERM` = *formula*	Random term to scan for outliers; default is the residual term
`METHOD` = *string token*	Method for calculating the statistics used to indicate an outlier (`full`, `partial`, `t`); default `t`
`THRMETHOD` = *string token*	Method for obtaining the threshold statistics (`approximate`, `bootstrap`); default `appr` for `METHOD=full` and `boot` otherwise
`NBOOT` = *scalar*	Number of bootstrap samples to take to form the threshold statistics; default `99` for `METHOD=full` and `499` otherwise
`FIXED` = *formula*	Fixed model terms
`RANDOM` = *formula*	Random model terms
`CONSTANT` = *string token*	How to treat the constant term (`estimate`, `omit`);

	default `esti`
`FACTORIAL` = *scalar*	Limit on the number of factors or covariates in each fixed term; default `3`
`VCONSTRAINTS` = *string token*	How to constrain the variance components and the residual variance (`none`, `positive`, `fixrelative`, `fixabsolute`); default `posi`
`INITIAL` = *variate*	Initial values for the variance components; default `1`
`SEED` = *scalar*	Seed for random number generation; default 0 continues an existing sequence or, if none, selects a seed automatically
`SAVEITEMS` = *string tokens*	Selects the items to save (`residuals`, `omega`, `sigma2`, `gamma`, `tsquared`, `lrt`, `fdr`, `approxthresholds`, `thresholdstats`, `outliers`, `method`, `all`); default `resi`, `omeg`, `sigm`, `meth`, `fdr`, `outl`

Parameters

`Y` = *variates*	Response variates
`TITLE` = *texts*	Specifies the title or titles to use for the plots
`SAVE` = *pointer*s	Saves information from the analysis of each y-variate

By default, the `VSOM` assesses the residuals. However, you can set the `RTERM` option to a random term in the analysis, to assess its effects: i.e. to see whether any of the groups of observations defined by the random term seem to be aberrant. The `METHOD` option specifies how the extra variance in the VSOM is estimated, with the following settings.

`full`	refits the full model with the added variance term for each unit; this can be very time-consuming.
`partial`	approximates the change in likelihood by a partial likelihood, where the baseline model parameters are held fixed, and only the extra variance component for each unit is estimated; this is much faster than re-estimating the full model.
`t`	uses the squared *t*-statistics (i.e. squared standardized residuals) to approximate the change in likelihood (default); this is the fastest approach.

To assess whether a unit is outside its expected distribution, thresholds are calculated at various levels of significance. The `THRMETHOD` option specifies the method to use:

`approximate`	uses the asymptotic distribution to calculate the thresholds; and
`bootstrap`	uses parametric bootstrap samples, with the variance components in the baseline model, to calculate the thresholds from the percentiles of the order statistics.

Each bootstrap sample is formed by taking the sum of the fitted fixed effects from the baseline model, together with simulated effects for the random terms in the model. Each random effect is simulated by Normal random numbers, with a mean of zero and the variance that was estimated for that term in the baseline model. The `NBOOT` option defines how many random samples to perform; the default is `99` for `METHOD=full`, and `499` otherwise. The `SEED` option specifies the seed for the random number generator, used by the `GRNORMAL` function to make the bootstrap samples. The default of zero continues the sequence of random numbers from a previous generation or, if this is the first use of the generator in this run of Genstat, it initializes the seed automatically from the computer clock. If you repeat the analysis with the same (non-zero) seed, you will get the same random numbers, and hence the same results.

The FIXED and RANDOM options specify the fixed and random terms to be fitted in the analysis, and the FACTORIAL option sets a limit on the number of factors and variates allowed in each fixed term. If neither FIXED nor RANDOM is specified, their settings are taken from the most recent VCOMPONENTS command. Its FACTORIAL setting is also taken if VCOMPONENTS is providing the fixed model. A fault is given if neither a fixed nor a random model is supplied. Note that the analysis cannot handle covariance models (which would be specified by the VSTRUCTURE directive). The VCONSTRAINTS option specifies constraints on the variance components, using the same settings as the CONSTRAINTS parameter of VCOMPONENTS. The CONSTANT option allows you to omit the constant.

Printed output is controlled by the PRINT option, with the following settings:

outliers	prints a summary of the potential outliers, as measured against the threshold statistics, at various levels of significance; and
fdr	prints the estimated false discovery rates for the potential outliers.

The false discovery rates (FDR) are estimated from the distribution of p-values calculated with the *t*-statistics from the asymptotic model. This uses the FDRMIXTURE procedure, or else the FDRBONFERRONI procedure if that fails. The FDR estimates the probability that the outlier is generated by noise. If this is small, it is likely that the outlier is genuine. However, if it is larger than 0.5, there is more chance that it was generated by noise. The FDR probabilities do not allow for correlations between the estimates. So, if there are only 2-3 replicates of the fixed terms, these may be too small, and should be interpreted with caution.

The VPRINT option controls the output from the REML analysis of the baseline model (as specified by the FIXED and RANDOM options). This has the same settings and default as the PRINT option of REML.

Graphical output is controlled by the PLOT option, with the following settings.

residual	when RTERM is set, the DRESIDUALS procedure is used to plot histograms and Normal plots of the specified random effects; when RTERM is not set, DRESIDUALS is used to plot histograms and Normal plots of the residuals together with a plot of the residuals against the fitted values.
indexplots	plots the statistics, selected by the INDEXPLOT option, against their index (i.e. their position in the y-variate).

For residual and indexplots, points are plotted in red if they are greater than their 5% bootstrap threshold, and in purple or green if greater than the 1% or 5% asymptotic thresholds respectively. The index plot also displays reference lines for the order statistics (OS 1, OS 2...) when METHOD=bootstrap, or the 5%, 1% and 0.1% and 0.01% asymptotic thresholds when METHOD=approximate.

The plots that are produced as components of the index plot can be controlled by the INDEXPLOT option, with the following settings:

omega	variance shift as a ratio to the residual variance,
sigma2	estimated residual variance under VSOM,
tsquared	squared *t*-statistic,
lrt	likelihood ratio test,
method	the statistic associated with the setting of the METHOD option, i.e. lrt for full or partial, and tsquared for t (default), and
all	all the statistics.

The Y parameter specifies the response variate. The TITLE parameter can supply a text, with either one or three values, to label the graphs. If the text has a single value, this is used to prefix the standard descriptions for the three graphs. If it has three values, these give (in full) the titles

for the comparison, indexplots, residual plots, respectively.

The SAVE parameter can save a pointer containing variates, storing the statistics calculated for each group or individual. The labels of the pointer, and the corresponding statistics, are as follows:

'residuals'	the standardized residuals,
'omega'	the variance shift as a ratio to the residual variance,
'sigma2'	the estimated residual variance under VSOM,
'gamma'	the estimated variance component for RTERM under VSOM,
'tsquared'	the squared *t*-statistic,
'LRT'	the partial likelihood ratio test if THRMETHOD=partial or the full likelihood ratio test otherwise,
'method'	the statistic associated with the setting of the METHOD option (lrt for full or partial, and tsquared for t),
'FDR'	the false discovery rate base on the *t*-statistics,
'approxthresholds'	the approximate thresholds used to indicate significant departures,
'thresholdstats'	the 95 percentiles of the order statistics from the bootstrap samples in decreasing order, and
'outliers'	the unit numbers of outliers above the thresholds.

The SAVEITEMS option controls which of the above items are saved.

Example 5.3.7a fits variance shift outlier models to assess the residuals from the analysis in Example 5.3.6e.

Example 5.3.7c

```
 65  VSOM        [VPRINT=*; FIXED=Littersize+Dose+Sex; RANDOM=Dam/Pup] Weight

Variance shift outlier model
============================


Analysis for residual term
==========================

Outlier detection based on test statistic t^2

Thresholds based on bootstrap with 499 simulated data sets
* MESSAGE: Default seed for random number generator used with value 523270


Units above test-wise threshold p <= 0.0001
-------------------------------------------

    Unit   Omega   Residual variance      Test statistic
      66   86.28              0.1322               63.06
      56   17.94              0.1563               16.73

Units above test-wise threshold 0.001 < p <= 0.01
-------------------------------------------------

    Unit   Omega   Residual variance      Test statistic
     227  10.473              0.1596              10.503
      60  10.599              0.1597              10.250
      58   9.269              0.1603               9.119
      48   7.953              0.1608               8.196

Units above test-wise threshold 0.01 < p <= 0.05
------------------------------------------------
```

Unit	Omega	Residual variance	Test statistic
45	5.792	0.1618	6.274
61	5.845	0.1618	6.169
109	5.058	0.1621	5.693
6	4.782	0.1623	5.345
51	4.460	0.1624	5.077
301	3.852	0.1628	4.386
293	3.510	0.1629	4.200
283	4.112	0.1629	4.083

Units above experiment-wise threshold (p=0.05) on order statistics

Unit	Omega	Residual variance	Test statistic	Threshold
66	86.28	0.1322	63.06	14.630
56	17.94	0.1563	16.73	10.316
227	10.47	0.1596	10.50	8.952
60	10.60	0.1597	10.25	7.835
58	9.27	0.1603	9.12	7.132
48	7.95	0.1608	8.20	6.701

False discovery rate analysis

Unit	t^2 statistic	Probability	FDR
66	63.06	< 0.0001	< 0.0001
56	16.73	< 0.0001	0.0062
227	10.50	0.0011	0.1369
60	10.25	0.0013	0.1532
58	9.12	0.0024	0.2461
48	8.20	0.0040	0.3465
45	6.27	0.0120	0.5951
61	6.17	0.0128	0.6086
109	5.69	0.0168	0.6676
6	5.35	0.0205	0.7080
51	5.08	0.0240	0.7373
301	4.39	0.0360	0.8040
293	4.20	0.0402	0.8197
283	4.08	0.0431	0.8292

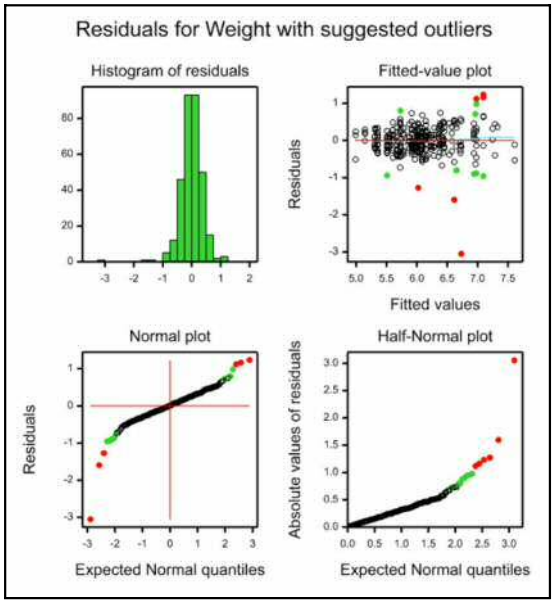


Figure 5.3.7a

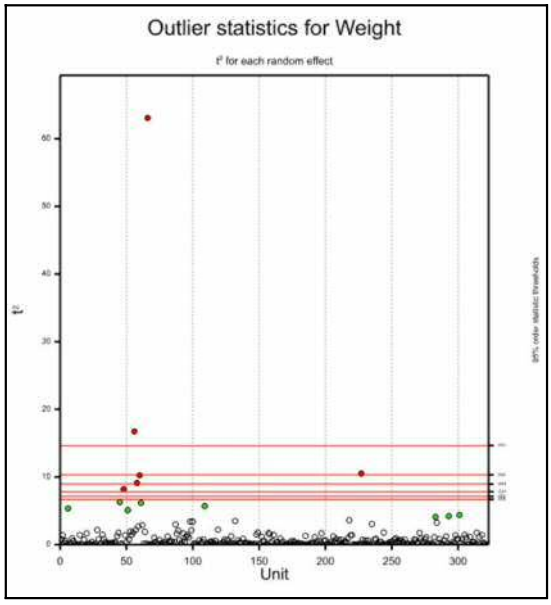


Figure 5.3.7b

The output confirms the earlier conclusion, from VCHECK, that there are some aberrant residuals in that analysis. More interestingly Figure 5.3.7a, which contains residual plots with the potential outliers plotted in red and green, reinforces the concerns about the stability of the variance. It is important to ensure that the variance is stable before checking for outliers. However, Example 5.3.7d shows that some of the units still have large residuals even if the weights are transformed to logarithms.

Example 5.3.7d

```
 66  CALCULATE   LogWeight = LOG10(Weight)
 67  VSOM        [VPRINT=*; FIXED=Littersize+Dose+Sex; RANDOM=Dam/Pup] LogWeight

Variance shift outlier model
============================


Analysis for residual term
==========================

Outlier detection based on test statistic t^2

Thresholds based on bootstrap with 499 simulated data sets

Units above test-wise threshold p <= 0.0001
-------------------------------------------

    Unit   Omega   Residual variance      Test statistic
      66  117.90           0.0006567               80.12
      56   17.47           0.0008335               16.34

Units above test-wise threshold 0.0001 < p <= 0.001
---------------------------------------------------

    Unit   Omega   Residual variance      Test statistic
     227   12.79           0.0008441               12.53

Units above test-wise threshold 0.001 < p <= 0.01
-------------------------------------------------

    Unit   Omega   Residual variance      Test statistic
     109   7.235           0.0008576               7.671
      60   7.526           0.0008577               7.624
      58   6.712           0.0008596               6.921

Units above test-wise threshold 0.01 < p <= 0.05
------------------------------------------------

    Unit   Omega   Residual variance      Test statistic
      48   4.862           0.0008637               5.438
       6   3.852           0.0008663               4.509
```

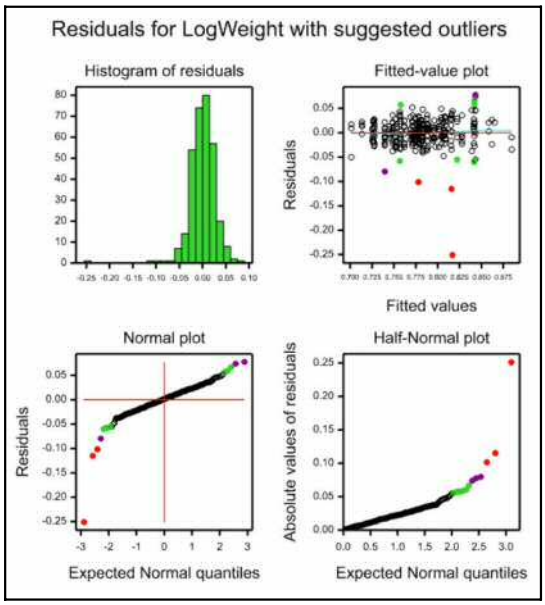


Figure 5.3.7c

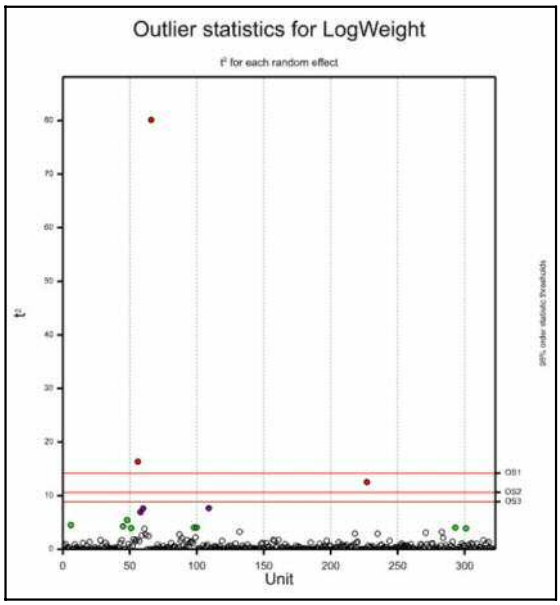


Figure 5.3.7d

5.3.8 Examining sources of variability

Example 5.3.6 showed how REML can be used to estimate variance components in order to form sensible estimates of fixed effects and their standard errors. Sometimes, however, you may be more interested in studying the random effects, in order to gain knowledge about the sources of variability in a data set. The results from REML analyses can help you do this: estimates of the variance parameters are available with their variance covariance matrix; likelihood tests can be used to compare competing random models; and a decomposition of the information matrix for the variance parameters can indicate any underlying structure in the data. Also, procedure VAIC can calculate the Akaike and Schwarz (Bayesian) information coefficients, and VRACCUMULATE can accumulate this information over a sequence of random models to help you assess which one is the most appropriate. Some of these facilities are illustrated in Example 5.3.8.

The data in the example were obtained to investigate sources and sizes of variability in an industrial process, the production of car voltage regulators (Example S from Cox & Snell 1981, Snell & Simpson 1991). Within the factory, each regulator was passed from the production line to a setting station where it was adjusted to operate within the correct range of voltages. It would then be passed to a testing station where it would be tested and sent back if outside the acceptable range. An experiment was designed to examine the sources of variability in the voltages produced by the regulators. This experiment used four testing stations, ten setting stations and between four and eight regulators from each setting station. In this situation, small components of variance can be tested for exclusion from the model and the approximate stratum variances can be used to give insight into the structure of the data.

Using factors Teststat and Setstat to indicate the testing and setting stations used for each unit, and factor Regulator which numbers regulators within each setting station, the random model containing all possible sources of variation is

```
Teststat*(Setstat/Regulator).
```

The three-way interaction Teststat.Setstat.Regulator is the residual error component in this model, and there are no fixed effects except the overall mean.

Example 5.3.8a

```
  2  " Voltage Regulator Performance
 -3    Investigation into sources of variability encountered
```

```
  -4    during the production of voltage regulators for cars.
  -5    (Example S from Applied Statistics Principles and Examples,
  -6    D.R.Cox & E.J.Snell, 1981)."
   7  UNITS  [NVALUES=256]
   8  FACTOR [LEVELS=4; VALUES=(1...4)64] Teststat
   9  FACTOR [LEVELS=10; LABELS=!T(A,B,C,D,E,F,G,H,J,K); \
  10    VALUES=32(1),16(2),28(3),28(4),16(5),28(6), \
  11    32(7),24(8),24(9),28(10)] Setstat
  12  FACTOR [LEVELS=8; VALUES=4( 1...8, 1,2...4, 1,2...7, 1,2...7, \
  13    1,2...4, 1,2...7, 1,2...8, 1,2...6, 1,2...6, 1,2...7)] Regulator
  14  OPEN 'Voltage.dat'; CHANNEL=2; FILETYPE=input
  15  READ [CHANNEL=2] Voltage

    Identifier   Minimum      Mean   Maximum    Values   Missing
       Voltage     15.30     16.12     17.80       256         0

  16  CLOSE 2; FILETYPE=input
  17  VCOMPONENTS [ABSORB=Setstat] Teststat*(Setstat/Regulator)
  18  REML [PRINT=model,components,stratumvariances,deviance;\
  19    METHOD=Fisher] Voltage

REML variance components analysis
=================================

Response variate:  Voltage
Fixed model:       Constant
Random model:      Teststat + Setstat + Teststat.Setstat + Setstat.Regulator
                   + Teststat.Setstat.Regulator
Number of units:   256
Absorbing factor:  Setstat

Teststat.Setstat.Regulator used as residual term

Non-sparse algorithm with Fisher scoring


Estimated variance components
-----------------------------

Random term               component        s.e.
Teststat                    0.00350     0.00320
Setstat                     0.01297     0.00902
Teststat.Setstat           -0.00413     0.00139
Setstat.Regulator           0.02980     0.00851


Residual variance model
-----------------------

Term                          Model(order)   Parameter    Estimate      s.e.
Teststat.Setstat.Regulator    Identity       Sigma2         0.0551   0.00606


Approximate stratum variances
-----------------------------

Stratum                             variance    effective d.f.
Teststat                             0.24453              3.00
Setstat                              0.47527              8.93
Teststat.Setstat                     0.02627             24.03
Setstat.Regulator                    0.17425             54.07
Teststat.Setstat.Regulator           0.05506            164.97


Matrix of coefficients of components for each stratum:

                    Teststat     62.40      0.00      7.04      0.00      1.00
                     Setstat      0.00     25.22      6.30      4.00      1.00
           Teststat.Setstat      0.00      0.00      6.98      0.00      1.00
           Setstat.Regulator     0.00      0.00      0.00      4.00      1.00
  Teststat.Setstat.Regulator     0.00      0.00      0.00      0.00      1.00
```

```
Deviance: -2*Log-Likelihood
---------------------------

                   Deviance   d.f.
                    -410.60   250

Note: deviance omits constants which depend on fixed model fitted.
```

Because a large number of effects are to be fitted in this model (135 parameters), `Setstat` is used as an absorbing factor to reduce the amount of space required. More discussion of the choice of absorbing factor is given in Section 5.3.9.

The `Teststat.Setstat` component is estimated as a small negative value. This would mean that the variability due to the testing station and setting station together is less than the variability expected from simply adding the variability of testing stations and setting stations. Rather than assume this to be the case, and since the negative value is small relative to the other components, it might seem more plausible that in reality the `Teststat.Setstat` component is zero.

The list of estimated variance components indicates that two of the components, `Teststat` and `Teststat.Setstat`, are much smaller than the others. They are small compared to their standard errors, but these estimates are based on only four testing stations. (The variance-covariance matrix and standard errors for the components are obtained from the inverse of their information matrix.) In order to decide whether the smaller components are effectively zero, or whether they are really necessary to explain the variation in the data, you can use a likelihood ratio test. You can obtain this by running `REML` again with the same fixed model but omitting the component from the random model. The test statistic is given by the difference between the deviances of the two models.

Example 5.3.8b

```
  20  VCOMPONENTS [ABSORB=Setstat] Teststat+(Setstat/Regulator)
  21  REML [PRINT=components,deviance] Voltage

Estimated variance components
-----------------------------

Random term               component        s.e.
Teststat                    0.00329     0.00334
Setstat                     0.01194     0.00881
Setstat.Regulator           0.03078     0.00845


Residual variance model
-----------------------

Term                       Model(order)  Parameter      Estimate      s.e.
Residual                   Identity      Sigma2           0.0511   0.00526


Deviance: -2*Log-Likelihood
---------------------------

                   Deviance   d.f.
                    -406.48   251

Note: deviance omits constants which depend on fixed model fitted.
```

The change in log-likelihood of 4.08 is large compared to a χ^2 variable on one d.f. which indicates that the `Teststat.Setstat` component should be retained in the model.

The Akaike's and Schwarz (or Bayesian) information criteria provide alternative ways of assessing the appropriateness of random models in `REML`. (The model with the smallest value of AIC or SIC is considered best.) These can be obtained using the `VAIC` procedure.

VAIC procedure

Calculates the Akaike and Schwarz (Bayesian) information coefficients for REML (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (deviance, aic, bic, sic, dffixed, dfrandom, changes); default aic
INCLUDE = *string tokens*	Which constants to include that depend only on the fixed model (determinant, pi); default pi
DMETHOD = *string token*	Method to use to calculate log(determinant(X′X)) (choleski, lrv); default chol
REPEAT = *string token*	Whether to repeat output from the previous VAIC (yes, no); default no

Parameters

DEVIANCE = *scalars*	Saves the deviance
AIC = *scalars*	Saves the Akaike information coefficient
SIC = *scalars*	Saves the Schwarz (Bayesian) information coefficient
DFFIXED = *scalars*	Saves the number of parameters fitted in the fixed model
DFRANDOM = *scalars*	saves the number of parameters fitted in the random model (and any covariance models)
CHANGES = *variates*	Saves changes since the previous VAIC; the units of the variates are labelled by the names of the coefficients (deviance, aic, sic, dffixed and dfrandom)
SAVE = *REML save structures*	Save structure for which to calculate the coefficients; default uses the save structure from the most recent REML

The coefficients are calculated from the deviance:

$$\text{aic} = \text{deviance} + 2 \times r$$

$$\text{sic} = \text{deviance} + \log(n - p) \times r$$

where n is the total number of usable units in the analysis, r is the number of parameters fitted in the random model (and any covariance models), and p is the number of parameters fitted in the fixed model. They are usually calculated for the most recent REML analysis. However, you can use the SAVE parameter to specify the SAVE structure from an earlier analysis.

The deviance provided by REML omits some constants that depend on the fixed model. In fact the full deviance is given by

$$\text{full-deviance} = \text{REML-deviance} + (n-p)*\log(2\pi) - \log(\det(X'X))$$

where X is the design matrix of the fixed model. Other software systems tend to include the first term, involving π, but omit the log-determinant term which is more time-consuming to calculate. The inclusion of these terms in the calculation is controlled by the INCLUDE option, with settings

determinant	$-\log(\det(X'X))$
pi	$+(n-p)*\log(2\pi)$

The DMETHOD option controls how $-\log(\det(X'X))$ is calculated when this is included. However, the default is INCLUDE=pi.

Printed output is controlled by the PRINT option, with settings:

deviance	prints the deviance (adding the extra terms specified by INCLUDE);
aic	prints the Akaike information coefficient;
sic or sic (synonyms)	prints the Schwarz (Bayesian) information coefficient;

`dffixed`	prints the number of parameters fitted in the fixed model;
`dfrandom`	prints the number of parameters fitted in the random model (and any covariance models).
`changes`	prints changes in the values of the coefficients since the previous use of `VAIC`, provided the fixed model of the `REML` analysis has not also changed.

These can all be saved using the `DEVIANCE`, `AIC`, `SIC`, `DFFIXED`, `DFRANDOM` and `CHANGES` parameters. By default `VAIC` prints just the Akaike information coefficient.

By default, each time that you use `VAIC`, its record of the current and previous `REML` analyses is updated. However, you can set option `REPEAT=yes` to repeat output from the previous `VAIC`. The analysis record is then not updated, so the information required to calculate changes remains available.

The `VRACCUMULATE` procedure can be useful if you have a sequence of random models that you want to evaluate.

`VRACCUMULATE` procedure

Forms a summary accumulating the results of a sequence of `REML` random models (R.W. Payne).

Options

`PRINT` = *string tokens*	Controls printed output (`deviance`, `aic`, `bic`, `sic`, `dffixed`, `dfrandom`, `change`, `exit`); default `devi`, `aic`, `sic`, `dfra`
`METHOD` = *string token*	How to accumulate the current analysis (`add`, `printonly`, `restart`); default `add`
`INCLUDE` = *string tokens*	Which constants to include that depend only on the fixed model (`determinant`, `pi`); default `pi`
`DMETHOD` = *string token*	Method to use to calculate log(determinant(X′X)) (`choleski`, `lrv`); default `chol`
`ACCUMULATED` = *pointer*	Saves the summary

Parameters

`DESCRIPTION` = *text*	Single-line text to describe the analysis; default lists the random terms added or deleted from the previous model
`SAVE` = *REML save structure*	Save structure for the `REML` analysis to put into the summary; default uses the save structure from the most recent `REML`

`VRACCUMULATE` allows you to accumulate results from a sequence of random models, so that you can view them all at once. You can do this by giving the command

```
VRACCUMULATE [PRINT=*]
```

following all except the last analysis. Then, after the last analysis, give another `VRACCUMULATE` command, but with the `PRINT` option now set to request the desired output, using the following settings:

`deviance`	prints the deviances;
`aic`	prints the Akaike information coefficients;
`bic` or `sic` (synonyms)	print the Schwarz (Bayesian) information coefficients;
`dffixed`	prints the number of parameters fitted in the fixed models;
`dfrandom`	prints the number of parameters fitted in the random models (and any covariance models);

`change` prints changes in the deviance and number of random d.f. between successive lines of the summary and their (chi-square) probabilities; and

`exit` exit codes (from `VKEEP`) indicating whether each analysis was fitted successfully (the deviance and information coefficients are set to missing values for unsuccessful fits).

The output indicates any point during the sequence of analyses where the fixed model has changed. It is not valid to compare random models unless one of the models is an extension of the other one, and the fixed model remained unchanged; if `VRACCUMULATE` detects that a comparison is invalid, the change in deviance is set to a missing value. It also flags any lines where it detects that there have been changes in the variance models (defined by `VSTRUCTURE`; see 5.4.1); before you use the change in deviance between these lines, you should check that the variance model defined in one of the lines is an extension of the model defined in the other one.

To print the information without adding another line to the summary, you can set option `METHOD=printonly`. Setting `METHOD=restart` reinitializes the summary before adding the current analysis. The default, `METHOD=add`, continues the existing summary by adding another line. The `INCLUDE` and `DMETHOD` options control how the deviance is calculated, as in `VAIC` (see above).

By default, the first line of the summary is labelled by the list of random terms in the model; subsequent lines list the random terms added or deleted from the previous model. Alternatively, you can supply your own labels using the `DESCRIPTION` parameter.

`VRACCUMULATE` usually adds a line to the summary for the most recent `REML` analysis. However, you can use the `SAVE` parameter to specify the save structure from an earlier analysis.

The `ACCUMULATED` option allows you to save the summary in a pointer, with elements labelled 'description', `'deviance'`, `'aic'`, `'sic'`, `'dffixed'`, `'dfrandom'`, `'deviance change'`, `'d.f. change'`, `'fixed changed'`, `'var-mod. changed'` and `'exit'`. `ACCUMULATED['description']` is a text. The other elements are variates. The saved values of the deviances and information coefficients all take account of the settings of the `INCLUDE` option.

Example 5.3.8c uses `VAIC` and `VRACCUMULATE` to print the Akaike information coefficient and changes in deviance for the models fitted in Examples 5.3.8a and 5.3.8b. Notice that we have set the first parameter (`DESCRIPTION`) in line 24 to define a narrower label for the first model. The default label would list the terms explicitly, as

```
Teststat + Setstat + Teststat.Setstat + Setstat.Regulator
```

The results confirm that the random term `Teststat.Setstat` should be retained in the model.

Example 5.3.8c

```
 22  VCOMPONENTS [ABSORB=Setstat] Teststat*(Setstat/Regulator)
 23  REML [PRINT=*] Voltage
 24  VRACCUMULATE [PRINT=*] 'Teststat*(Setstat/Regulator)'
 25  VAIC

Akaike information coefficient       68.06

Note: omits constant, -log(det(X'X)), that depends only on the fixed model.

 26  VCOMPONENTS [ABSORB=Setstat] Teststat+(Setstat/Regulator)
 27  REML [PRINT=*] Voltage
 28  VAIC

Akaike information coefficient       70.17

Note: omits constant, -log(det(X'X)), that depends only on the fixed model.

(based on the residual log-likelihood)
```

```
  29  VRACCUMULATE [PRINT=deviance,change,aic]

Accumulated summary of REML random models
-----------------------------------------


                               Deviance      AIC  Change in    Change in
                                                  deviance  random d.f.
 Teststat*(Setstat/Regulator)     58.06    68.06         *            *
           - Teststat.Setstat     62.17    70.17      4.11            1

                              Change
                            chi-prob
 Teststat*(Setstat/Regulator)       *
           - Teststat.Setstat   0.043

Note: omits constant, -log(det(X'X)), that depends only on the fixed model.

(based on the residual log-likelihood)
```

The original analysis, in Example 5.3.8a, used `METHOD=Fisher` in order to obtain estimates of the approximate stratum variances. These can be used to interpret the information on the variance components available from the experiment. They are calculated from a Cholesky decomposition of the information matrix of the variance components $E(-\partial^2 RL/\partial\hat{\sigma}^2)$, the expected value of the second derivative of the residual likelihood *RL*, using the vector $\hat{\gamma}$ of estimated variance components. This decomposition is motivated by analogy with the structure of orthogonal designs. Since the decomposition is based on the residual likelihood *RL* it can give no direct information on the fixed model terms, and therefore effectively gives a decomposition of a random effects model with a grand mean only, ignoring any other fixed model terms.

In an orthogonal design, the information matrix I_ξ for the independent stratum variances $\{ \xi_s \}$ is diagonal with elements $df_s/2\xi_s^2$ where df_s is the degrees of freedom of stratum *s*. Furthermore, these stratum variances are linear combinations of the variance components which always include the term σ^2, so $\xi = L\sigma$ where *L* is the matrix mapping the components onto the stratum variances and has value 1 for all elements in the final row (corresponding to σ^2). The information matrix for the variance components I_σ can then be calculated from the information matrix of the stratum variances by $I_\sigma = L' I_\xi L$.

From the results of a REML analysis, the Cholesky decomposition of the information matrix *I* of the estimated variance components can be written as $I = TDT'$, where *T* is the lower triangular Cholesky decomposition of *I*, standardized so that all values in the last row of *T* are 1 and *D* is a diagonal matrix containing the squares of the scaling factor for each column. This decomposition gives the information matrix in a form similar to that which occurs naturally in an orthogonal design. T' is then analogous to the matrix of coefficients used to construct the stratum variances from the variance components and *D* is analogous to the information matrix of the stratum variances.

The components of the decomposition can then be interpreted as if they had arisen from a hypothetical orthogonal experiment which gives information on the variance components equivalent to that available in the actual experiment. In other words, if it was carried out, the hypothetical experiment would be expected to give estimates of the variance components with precision similar to those in the actual experiment. This information can be useful for the planning of future experiments.

For orthogonal experiments, the decomposition will give the stratum variances expected from analysis of variance. As the model becomes non-orthogonal (either through the structure of the fixed or random model) the relationship breaks down, although the decomposition is usually fairly easy to interpret.

It should be remembered that the information matrix *I* represents the information on the variance components available from the data projected to remove all the treatment contrasts and

hence all the information on treatments. There is, however, no information about where the treatment degrees of freedom would have been, and this may lead to a slightly unexpected allocation of degrees of freedom where treatment efficiency factors are not all zero or one. The decomposition can indirectly give information on the fixed model terms. The change in structure when a fixed model term is dropped may give useful information about where the term was estimated and the variation in the data it accounted for.

It should also be noted that the information matrix is evaluated at the estimated value of the variance components, and thus depends on these values. For this reason, two experiments with the same structure may give slightly different decompositions.

In some circumstances the decomposition cannot be interpreted. If any of the variance components has been constrained in a `VCOMPONENTS` statement, using either `CONSTRAINTS` or a `RELATIONSHIP` matrix, there is no information directly available on the constrained components: the information on associated components is pooled, and the approximate stratum variances cannot be related back to the individual random model terms. Also, since the Cholesky decomposition works sequentially, for non-orthogonal random terms the decomposition will depend on the order of the random model. In particular, results may be difficult to interpret if the structure of the random model is non-hierarchical. Occasionally in these circumstances, the Cholesky decomposition yields negative coefficients leading to negative stratum variances which cannot be interpreted.

Example 5.3.8 gives a good illustration of how to interpret the decomposition in terms of the underlying structure of the data. The data for the voltage regulators is not quite balanced, since the number of regulators tested at each setting station varies between four and eight.

The structural information is contained in the matrix of coefficients (T' above) and the degrees of freedom (the diagonal of D above). Within an orthogonal design, the coefficients would indicate the replication of each level of the factors.

In Example 5.3.8a, the `Setstat.Regulator` stratumvariance (variation between regulators) has equation $\xi_{S.R}=4\sigma^2_{S.R}+\sigma^2$ indicating 4 readings for each regulator, which matches the experiment since each regulator was measured on each of the 4 testing stations. Similarly, the equation for the `Teststat.Setstat` stratum indicates 7 readings for each combination of setting station and testing station. This again matches the structure of the data since 64 regulators were tested on 10 setting stations, giving on average 6.4 regulators at each station. The equation for the `Setstat` stratum disagrees with this slightly, suggesting 25 readings at each setting station consisting of 6 regulators read 4 times each. Then the `Teststat` stratum again indicates 7 regulators and 62 readings at each testing station, which implies 9 setting stations. Putting these results together gives a structure consisting of 4 testing stations, 9 setting stations and 6-7 regulators used at each setting station. The degrees of freedom more or less correspond to this structure, suggesting 10 instead of 9 setting stations. This is the structure of a hypothetical orthogonal experiment which would have given the same amount of information on the variance components. Since the original experiment is nearly balanced, this hypothetical experiment is quite similar.

There are no hard and fast rules for interpreting this decomposition. In Example 5.3.8a, there was no fixed model. In general, the removal of treatment contrasts may affect both the coefficients and the degrees of freedom, making interpretation less straightforward.

5.3.9 Technical details of the Fisher method and absorbing factors

For large data sets and models with many parameters, the REML algorithm may take a large amount of computing time and/or data space when `METHOD=Fisher` is used. For this reason, the sparse (AI) algorithm, is used by default. However, as some results are available only when `METHOD=Fisher`, it may be helpful to understand the factors influencing the use of workspace.

The Fisher method estimates the variance components iteratively using Fisher scoring to solve the normal equations. For the model

$$y = X\alpha + Z\beta + \varepsilon$$

described in detail in Section 5.1, the residual-log-likelihood *RL* can be written as

$$RL(y) = -\frac{1}{2}\log|V| - \frac{1}{2}\log|X'V^{-1}X| - \frac{1}{2}(y-X\hat{\alpha})'V^{-1}(y-X\hat{\alpha})$$

ignoring terms independent of the variance parameters. The first derivatives of *RL* with respect to the gammas $\{\gamma_i\}$, where $\gamma_i=\sigma_i^2/\sigma^2$, and the residual variance σ^2 are

$$\frac{\partial RL}{\partial \gamma_i} = -\frac{1}{2}\text{trace}(Z_i'PZ_i) + (\hat{\beta}'\Gamma^{-1}D_i\Gamma^{-1}\hat{\beta})/2\sigma^2$$

$$\frac{\partial RL}{\partial \sigma^2} = -\frac{1}{2}(n-p^*)/\sigma^2 + (y-X\hat{\alpha})'V^{-1}(y-X\hat{\alpha})/2\sigma^4$$

$$\text{where} \quad P = V^{-1} - V^{-1}X(X'V^{-1}X)^{-1}X'V^{-1}\ ; \quad \frac{\partial \Gamma}{\partial \gamma_i} = D_i\ ; \quad p^* = \text{rank}(X).$$

As well as the unknown variance parameters, these equations involve the estimates of the fixed and random effects. At each iteration, these parameters can be estimated using current estimates of the variance parameters and inverting the mixed model equations

$$\begin{pmatrix} X'X & X'Z \\ Z'X & Z'Z+\hat{\Gamma}^{-1} \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} X'y \\ Z'y \end{pmatrix}.$$

Then, defining

$$Q = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} = \begin{pmatrix} X'X & X'Z \\ Z'X & Z'Z+\hat{\Gamma}^{-1} \end{pmatrix}^{-1}$$

It can be shown that trace($Z_i'PZ_i$) = trace(UD_i) where $U = \Gamma^{-1} - \Gamma^{-1}Q_{22}\Gamma^{-1}$. The information matrix *I* can also be written in terms of *U*, the estimates of β, and the residual sum of squares. The REML algorithm implemented in Genstat takes the following steps at each iteration:

0) Obtain initial estimates of the variance parameters.
1) Calculate estimates of α and β by inverting the mixed model equations using current estimates of the variance parameters. Form *U*.
2) Using $\hat{\alpha}$, $\hat{\beta}$ and *U* calculated in step 1, form the first derivatives of the likelihood *RL* and the information matrix *I*. Then use Fisher scoring (see equation below) to obtain updated estimates of the variance parameters.

$$\begin{pmatrix} \gamma_{new} \\ \sigma^2_{new} \end{pmatrix} = \begin{pmatrix} \gamma_{old} \\ \sigma^2_{old} \end{pmatrix} + I^{-1} \begin{pmatrix} \frac{\partial RL}{\partial \gamma_{old}} \\ \frac{\partial RL}{\partial \sigma^2_{old}} \end{pmatrix}$$

3) Check for convergence of variance parameter estimates: exit algorithm on convergence; otherwise, return to step 1.

The inversion of the mixed model equations at step 1 involves inversion of a symmetric matrix with number of rows equal to the number of fixed effects (n_f) plus the number of random effects (n_r) in the model. For models specifying a large number of effects, the inversion of this matrix can be time-consuming and requires $(n_f+n_r)^2$ units of double precision data space.

Since the size of the mixed model equations can limit the speed of the algorithm, it is sensible

to try and reduce the size of this matrix. Use of an absorbing factor is one way of tackling the problem. An absorbing factor is a factor from either the fixed or random model, which is used to define a partition of the mixed model equations and hence decrease the size of matrices which must be inverted and stored. However, the information required to calculate estimated errors for some of the tables of means and effects will no longer be available (see Section 5.3.3). When an absorbing factor is specified, the model terms are reordered into two groups: the first contains all the model terms involving the absorbing factor; and the second contains all the other model terms. Each part of the model may include both fixed and random terms. The general mixed model above can be partitioned in this way, so that α_1 and β_1 denote the elements of α and β that are associated with the absorbing factor model, with associated design matrices X_1 and Z_1, and α_2 and β_2 are the remaining fixed and random parameters, with design matrices X_2 and Z_2. The mixed model equations can be reordered to give

$$\begin{pmatrix} U'U+\Gamma_1^{-1} & U'W \\ W'U & W'W+\Gamma_2^{-1} \end{pmatrix} \begin{pmatrix} \theta \\ \varphi \end{pmatrix} = \begin{pmatrix} U'y \\ W'y \end{pmatrix}$$

$$\text{where} \quad U = (X_1 \,|\, Z_1)'; \quad W = (X_2 \,|\, Z_2)'; \quad \theta' = (\alpha_1' \beta_1'); \quad \varphi' = (\alpha_2' \beta_2')$$

and Γ_1 and Γ_2 are the parts of Γ relating to β_1 and β_2 respectively, with zero rows added to correspond to α_1 and α_2.

The first set of equations can be *absorbed* into the second set, giving the matrix

$$\begin{pmatrix} (U'U+\Gamma_1^{-1})^{-1} & U'M^{-1}W \\ W'M^{-1}U & W'M^{-1}W+\Gamma_2^{-1} \end{pmatrix} \qquad \text{where} \quad M = I-U'(U'U+\Gamma_1^{-1})^{-1}U$$

It is possible to write most of the expressions in the iterative REML algorithm in terms of the matrices $U'U+\Gamma_1^{-1}$ and $W'M^{-1}W+\Gamma_2^{-1}$ and their inverses. The inversion of the whole set of mixed model equations can be avoided by working with these two matrices separately. Since the inverse sum of squares matrix Q is the estimated variance-covariance matrix for the parameter estimates, this separation means that estimates of covariances between the two sets of parameters are not calculated. By reordering the parameters within the absorbing factor model by level of the absorbing factor, the matrix $U'U+\Gamma_1^{-1}$ becomes block diagonal, which means that any expression involving the matrix $U'U+\Gamma_1^{-1}$ can be calculated using each of these blocks in turn and accumulating the result. This results in a further reduction in the size of matrices that have to be stored, but since the same workspace is used for each block of $U'U+\Gamma_1^{-1}$ and the whole matrix is not stored, the covariances and variance estimates for parameters in the absorbing factor model are not available.

The calculations for comparing different choices of absorbing factor are quite straightforward.

1) Choose an absorbing factor A with v levels.
2) Split the model terms into two groups and count the number of parameters defined by the factor combinations in each group: (a) model terms containing the absorbing factor (n_1 parameters) and (b) model terms not containing the absorbing factor (n_2 parameters).
3) The matrices that must be inverted using absorbing factor A are then: one matrix of order n_2 plus v matrices of order n_1/v.

As well as considering the numerical advantages of an absorbing factor, it is also important to check that the choice of absorbing factor does not mean that the estimates of error are lost for important comparisons. It should also be noted that the inversion of very many smaller matrices

can sometimes take longer than the inversion of a few matrices of intermediate size.

Further details are given by Thompson (1977).

5.3.10 Controlling advanced features of the REML algorithm

VCYCLE directive

Controls the operation of the REML algorithm.

Options

CONVERGENCE = *string token*	Type of criterion for assessing convergence (deviance, parameter); default * uses the deviance with the average-information algorithm, and the variance parameter values for the Fisher scoring algorithm
CRITERIONVALUE = *scalar*	Sets the convergence criterion value; default * i.e. determined automatically
STEPLENGTH = *scalar*	Sets the default relative step size for the average-information algorithm; default * i.e. determined automatically
NDENSE = *scalar*	Number of equations to use as dense in the average-information algorithm; default * uses all fixed model terms as dense
EQORDER = *string token*	Method to use to reorder the mixed model equations for fitting (none, a, b); default b

No parameters

VCYCLE allows you to control various aspects of the REML algorithm. The CONVERGENCE option specifies the type of criterion to use to assess convergence. There are two possibilities, each of which is used as the default for one of the fitting algorithms. For the average-information algorithm the default is to check for convergence in deviance, whereas the Fisher scoring method checks the variance parameter values. The criterion value can be specified by the CRITERIONVALUE option. The defaults differ according to the type of criterion. For assessing changes in variance parameter values a multiplier of 0.005 is used. So, for convergence, the change in every variance parameter s must be less than $0.005 \times s$. When assessing change in deviance, convergence occurs when the absolute change in the deviance is less than 0.0001.

The STEPLENGTH option allows you to change the default step size for the average-information algorithm. Valid values are between zero and one, and the value is the proportion of the average-information step taken. The default is to start with small steps and work up to full steps.

The NDENSE option allows you to manipulate the number of equations used as dense in the average-information algorithm (see Gilmour *et al*. 1995). The default includes all the fixed model terms. This option is likely to be used only by advanced users. If NDENSE is set, the value may be modified by the algorithm so that model terms are not split between the dense and sparse sections. Note that Wald tests (dropping terms) are not available for terms in the sparse section.

The EQORDER option controls the order in which the mixed model equations are solved, with settings:

none	processes the equations in the order in which they are specified in the model;
a	method A; and
b	method B (default).

This option needs to be set only rarely as method B, which corresponds to the ASReml option

setting !EQORDER 3 (introduced to become the default in ASReml Release 2), is generally the best. Method A corresponds to the ASReml option setting !EQORDER 1 (which was the default in ASReml Release 1). For further details, see *ASReml User Guide Release 2*.

5.4 Modelling variance structures

REML estimates parameters in mixed models of the form

$$y = X\alpha + \Sigma_i Z_i u_i + e$$

where α is a vector of fixed effects with design matrix X, the u_i are vectors of random effects with design matrices Z_i and variances $\text{var}(u_i)=\sigma_i^2 G_i$ and by default $\text{cov}(u_i,u_j)=0$, and e is a vector of random error (usually called the residual) with $\text{var}(e)=\sigma^2 R$, $\text{cov}(u,e)=0$. The variance model V for the data y is then

$$V = \Sigma_i \sigma_i^2 Z_i G_i Z_i' + \sigma^2 R \qquad \text{(sigma parameterization)}$$

or

$$= \sigma^2 (\Sigma_i \gamma_i Z_i G_i Z_i' + R) \qquad \text{(gamma parameterization)}$$

In the earlier sections of this chapter, the matrices G_i and R are simply the identity matrix I. The VSTRUCTURE directive can specify a wide range of parametric forms (including auto-regressive, moving average, ante-dependence, unstructured or distance-based models) for the G_i and R matrices to enable the modelling of covariance patterns within the data. This section describes the range of models available, using examples from repeated measurements, spatial analysis of field experiments, random coefficient regression and multivariate data. Output from specific examples can be found for repeated measurements analysis in Section 5.4.3, spatial analysis in Section 5.4.4 and random coefficient regression in Section 5.4.5.

5.4.1 The VSTRUCTURE directive

The directive VSTRUCTURE can be used to define the form of covariance structure for any term in the random model defined for REML by VCOMPONENTS.

VSTRUCTURE directive

Defines a variance structure for random effects in a REML model.

Options

TERMS = *formula*	Model terms for which the covariance structure is to be defined
FORMATION = *string token*	Whether the structure is formed by direct product construction or by definition of the whole matrix (direct, whole); default dire
CORRELATE = *string token*	Whether to impose correlation across the model terms if several are specified (none, positivedefinite, unrestricted); default none
CINITIAL = *scalars*	Initial values for covariance matrix across terms
COORDINATES = *matrix* or *variates*	Coordinates of the data points to be used in calculating distance-based models

Parameters

MODELTYPE = *string tokens*	Type of covariance model associated with the term(s), or with individual factors in the term(s) if FORMATION=direct (identity, fixed, AR, MA, ARMA, power, boundedlinear, circular, spherical, linearvariance, banded, correlation, antedependence, unstructured,

	`diagonal`, `uniform`, `FA`, `FAequal`) default `iden`
`ORDER` = *scalar*	Order of model
`HETEROGENEITY` = *string token*	Heterogeneity for correlation matrices (`none`, `outside`); default `none`
`METRIC` = *string token*	How to calculate distances when `MODELTYPE=power` (`cityblock`, `squared`, `euclidean`); default `city`
`FACTOR` = *factors*	Factors over which to form direct products
`MATRIX` = *symmetric matrices*, *diagonal matrices* or *pointers*	Defines matrix values for a term or the factors when `MODELTYPE=fixed`
`INVERSE` = *symmetric matrices*, *diagonal matrices* or *pointers*	Define values for matrix inverses (instead of the fixed matrices themselves) when `MODELTYPE=fixed`
`DISTANCES` = *symmetric matrices*	Symmetric matrix of pre-formed distances to be used in distance-based models of order one
`COORDINATES` = *matrices*, *variates* or *pointers*	Specifies coordinates of each factor level to be used in calculating distance-based models
`INITIAL` = *scalars*, *variates*, *matrices*, *symmetric matrices* or *pointers*	Initial parameter values for each correlation matrix (supplied in the structures appropriate for the model concerned)
`CONSTRAINTS` = *texts*	Texts containing strings `none`, `fix` or `positive` to define constraints for the parameters in each model
`EQUALITYCONSTRAINTS` = *variates*	Non-zero values in the variate indicate groups of parameters whose values are to be constrained to be equal

`VSTRUCTURE` can be used only after `VCOMPONENTS` has defined the fixed and random models. It can be used more than once to define different structures for different random terms. The information is accumulated within Genstat, and it will all be used by subsequent `REML` commands. You can check on the model and covariance structures defined at any time by using the `VSTATUS` directive. To cancel a covariance structure for a term you simply need to use `VSTRUCTURE` to change the model back to the default identity matrix. To cancel all covariance structures you can give a new `VCOMPONENTS` command and redefine the fixed and random models.

For a random term constructed from more than one factor, the covariance matrix can be formed either as a single matrix for the whole term, or as the direct product of several matrices corresponding to the factors. (For a more general discussion about the models that can be generated using direct products, and limitation of this method, see Section 5.4.6). Below we illustrate these concepts using a range of standard models.

Repeated measurements data

For example, consider an analysis of repeated measurements where data have been taken weekly over 5 weeks from a set of 14 subjects. It is likely that data taken from the same subject will be correlated, with correlation decreasing over time, but that subjects will be independent. If we define factors `Subject` and `Week` to represent individual subjects and times of measurement, the term `Subject.Week` will represent the residual vector e (since it indexes every unit in the dataset). This can be written in terms of sub-vectors e_i for subject i at times 1...5. We can then impose some common covariance structure C on the sub-vectors e_i to model correlation over

time, and insist on independence between subjects, i.e. between the e_i, giving var(e_i)=C and cov(e_i,e_j)=0. The resulting variance matrix on e can be written as a direct product of an identity matrix and the covariance matrix C:

$$e = \begin{pmatrix} e_1 \\ e_2 \\ \cdot \\ \cdot \\ \cdot \\ e_{14} \end{pmatrix}, \quad \text{var}(e) = R = \begin{pmatrix} C & 0 & 0 & \ldots & 0 \\ 0 & C & 0 & \ldots & 0 \\ 0 & 0 & C & \ldots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \ldots & C \end{pmatrix} = I_{14} \otimes C$$

So the variance model for the residual can be constructed by considering the components of the term: independence between subjects combined with correlation within subjects. In this case, no other random terms are required to describe the structure of the variance model. If we take C to be an auto-regressive process of order 1, this can be defined and the model fitted to data held in variate `Y` as follows:

```
VCOMPONENTS [FIXED=Tmt] RANDOM=Subject.Week
VSTRUCTURE [TERM=Subject.Week] MODELTYPE=I,AR; ORDER=1; \
  FACTOR=Subject,Week
REML Y
```

The `TERM` option is used to specify the term to which the covariance structure is to be applied. By default, a direct product form is assumed. You then specify the covariance model to be applied to each factor in the term (see below for list of available models). However, it is not necessary to specify factors for which the default identity model is required, so the following is an equivalent specification:

```
VCOMPONENTS [FIXED=Tmt] RANDOM=Subject.Week
VSTRUCTURE [TERM=Subject.Week] MODELTYPE=AR; ORDER=1;
FACTOR=Week
```

To cancel the covariance structure for the term, a null setting is sufficient:

```
VSTRUCTURE [TERM=Subject.Week]
```

It is instructive to compare the auto-regressive model fitted above with the standard split-plot analysis:

```
VCOMPONENTS [FIXED=Tmt] RANDOM=Subject/Week
REML Y
```

The random model `Subject/Week` expands to `Subject + Subject.Week`, i.e. random effects for subjects plus the residual. Although the covariance structure for the random subject term here is of the form $G = \sigma_s^2 I$, the variance matrix for the data is of the form

$$V = \sigma_s^2 ZZ' + \sigma^2 I$$

In this case the random subject term generates correlations that are equal across all the times within subjects. It is important to remember that including a random term in the model will generate uniform correlations between units with the same values of the random factor(s). It is often necessary to exclude these terms when the object is to model the correlations explicitly. In fact, this model could alternatively be specified as

```
VCOMPONENTS [FIXED=Tmt] RANDOM=Subject.Week
VSTRUCTURE [TERM=Subject.Week] FACTOR=Week; MODELTYPE=uniform
```

Spatial analysis of field experiments

The repeated measurements example above naturally generates a block diagonal variance matrix V, but it is easy to find examples where more complex structures arise by combining variance

models. For example, consider the analysis of a field experiment laid out as 10 rows of 15 columns, where the object is to model spatial variation across the experiment to obtain more accurate standard errors. The standard ANOVA model for this data can be specified as

```
VCOMPONENTS [FIXED=Cv] RANDOM=Row+Column+Row.Column
```

which assumes equal correlation within rows and within columns, plus an independent residual error. However, for large experiments, equal correlation might not be a reasonable assumption and an auto-regressive model over rows and over columns separately might be tried instead:

```
VCOMPONENTS [FIXED=Cv] RANDOM=Row.Column+'*units*'
VSTRUCTURE [TERM=Row.Column] MODELTYPE=ar,ar;  \
  ORDER=1; FACTOR=Row,Column
```

Here, the Row and Column terms have been removed from the random model, as they are superceded by the correlation from the composite term Row.Column. The term '*units*' has been retained to provide an estimate of independent random error in addition to that predicted by the AR(1) $\otimes$ AR(1) structure. This model might be interpreted as the correlation structure describing the underlying spatial trend in the field, with the extra residual accounting for experimental and measurement error in the data.

In situations where rows (or columns) were spaced irregularly, the correlation between units might depend on the distances between them, and a distance-based covariance structure would be more appropriate (see Sections 5.4.4 and 5.4.6).

Random coefficient regression

In some longitudinal data sets, individual profiles appear to increase linearly over time, but with obvious variation in slope between subjects within treatment groups. In this case, a natural model for the data consists of a common linear trend over time for treatment groups plus random variation about the intercept and slope for subjects. If such a dataset is defined using variates Time and Y to hold times of measurement and responses, and factors Subject and Tmt to code for individuals and treatment groups, this model can be specified by

```
VCOMPONENTS [FIXED=Tmt*Time] RANDOM=Subject+Subject.Time
```

To make the fitted variance model invariant to the scale of time measurement, it is customary to impose correlation between the intercept and slope for each subject, i.e. if we write a_i and b_i to be the random deviation about the common intercept and slope for subject i

$$\mathrm{cov}\begin{pmatrix} a_1 \\ . \\ a_n \\ b_1 \\ . \\ b_n \end{pmatrix} = \begin{pmatrix} \sigma_a^2 I_n & \sigma_{ab} I_n \\ \sigma_{ab} I_n & \sigma_b^2 I_n \end{pmatrix} = \begin{pmatrix} \sigma_a^2 & \sigma_{ab} \\ \sigma_{ab} & \sigma_b^2 \end{pmatrix} \otimes I_n$$

and this can also be specified via the direct product construction. Where correlation is to be specified across terms in this way, the CORRELATE parameter of VSTRUCTURE is used to impose the correlation across terms, and then the form of the within term correlation is specified using the parameters of VSTRUCTURE as usual. Here the within term correlation is independence, the default:

```
VCOMPONENTS [FIXED=Tmt] RANDOM=Subject+Subject.Time
VSTRUCTURE [TERMS=Subject+Subject.Time; \
  CORRELATE=unrestricted; CINITIAL=!(1,0.5,0.05)]
```

It is often helpful to get initial values for the parameters by fitting the model without correlations first. (See Section 5.4.5.)

Multivariate analysis

In some circumstances, it is desirable to analyse two (or more) variables simultaneously to investigate correlation between the variables and their response to treatments. For example, suppose the number of leaves and average leaf area per plant has been measured from plants in an experiment done using a randomised block design with four blocks of seven plots. The model specification for a univariate analysis would be

```
VCOMPONENTS [FIXED=Tmt] RANDOM=Block/Plot
```

To analyse the two variates together, it is necessary to concatenate the two variables into a single variate, to define new block and treatment factors to match, and to define a new factor `Variable` to indicate which variate is in which units. If the block effects and residuals for variate i are b_i and e_i respectively, then the variances of the two random terms can then be written as

$$\mathrm{var}\begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} \sigma^2_{b1} I_r & \sigma_{b12} I_r \\ \sigma_{b12} I_r & \sigma^2_{b2} I_r \end{pmatrix} = \begin{pmatrix} \sigma^2_{b1} & \sigma_{b12} \\ \sigma_{b12} & \sigma^2_{b2} \end{pmatrix} \otimes I_r$$

$$\mathrm{var}\begin{pmatrix} e_1 \\ e_2 \end{pmatrix} = \begin{pmatrix} \sigma^2_{1} I_n & \sigma_{12} I_n \\ \sigma_{12} I_n & \sigma^2_{2} I_n \end{pmatrix} = \begin{pmatrix} \sigma^2_{1} & \sigma_{12} \\ \sigma_{12} & \sigma^2_{2} \end{pmatrix} \otimes I_n$$

and the model defined by

```
FACTOR [LEVELS=2; VALUES=28(1,2)] Variable
FACTOR [LEVELS=4; VALUES=(#Block)2] Mblock
&      [LEVELS=7; VALUES=(#Plot)2] Mplot
&      [LEVELS=7; VALUES=(#Tmt)2] Mtmt
VCOMPONENTS [FIXED=Mtmt*Variable] \
            Variable.Mblock + Variable.Mblock.Mplot
VSTRUCTURE  [TERM=Variable.Mblock] FACTOR=Variable; \
            MODELTYPE=unstructured
VSTRUCTURE  [TERM=Variable.Mblock.Mplot] FACTOR=Variable; \
            MODELTYPE=unstructured
```

In future releases, facilities will be provided so that multivariate problems like this can be specified simply by giving a set of variates in parallel with the block and treatment factors.

Model definitions

The examples above suggested various different structures that might be used to model covariance patterns. The possible settings for the `MODELTYPE` parameter, generating symmetric covariance matrices C ($C_{i,j} = C_{j,i}$ for all i,j), are as follows:

`identity`	identity matrix	$C_{i,i} = 1$, $C_{i,j} = 0$, for $i \neq j$
`fixed`	fixed matrix	$C_{i,j}$ specified
`AR`	auto-regressive	$C_{i,i} = 1$
	order **1** or 2 (φ_2=0 for order 1)	$C_{i+1,i} = \varphi_1 / (1-\varphi_2)$ $C_{i,j} = \varphi_1 C_{i-1,j} + \varphi_2 C_{i-2,j}$, $i > j+1$, $-1 < \varphi_1, \varphi_2 < 1$, $\lvert\varphi_1+\varphi_2\rvert<1$, $\varphi_2-\varphi_1<1$, $\varphi_2>-1$
`MA`	moving average	$C_{i,i} = 1$
	order **1** or 2	$C_{i+1,i} = -\theta_1(1-\theta_2)/(1+\theta_1^2+\theta_2^2)$

	(θ_2=0 for order 1)	$C_{i+2,\,i} = -\theta_2 / (1+\theta_1^2+\theta_2^2)$ $C_{i,\,j} = 0,\ i>j+2$ $-1 < \theta_1, \theta_2 < 1,\ \theta_2 \pm \theta_1 < 1$
`ARMA`	auto-regressive	$C_{i,\,i} = 1$
	moving-average	$C_{i+1,\,i} = (\theta-\varphi)(1-\varphi\theta) / (1+\theta^2-2\varphi\theta)$
	order 1	$C_{i,\,j} = \varphi C_{i-1,\,j},\ i>j+1$ $-1 < \varphi, \theta < 1$
`power`	based on distance	$C_{i,\,i} = 1$
	order **1** or 2 ($\varphi_1 = \varphi_2$ for order 1)	$C_{i,\,j} = \varphi_1^{d_1}\varphi_2^{d_2}$ d_1, d_2 = distance in 1st and 2nd dimensions $0 < \varphi_1, \varphi_2 < 1$
`boundedlinear`	based on distance order 1	$C_{i,\,j} = 1 - d/\varphi$ for $d \le \varphi$, $C_{i,\,j} = 0$ for $d > \varphi$ $0 < \varphi$
`circular`	based on distance order 1	$C_{i,\,j} =$ $1 - (2/\pi)\ \{(d/\varphi)\sqrt{(1-(d/\varphi)^2)} + \sin^{-1}(d/\varphi)\}$ for $d \le \varphi$, $C_{i,\,j} = 0$ for $d > \varphi$ $0 < \varphi$
`spherical`	based on distance order 1	$C_{i,\,j} = 1 - 1.5\,(d/\varphi) + 0.5\,(d/\varphi)^3$ for $d \le \varphi$, $C_{i,\,j} = 0$ for $d > \varphi$ $0 < \varphi$
`linearvariance`	based on distance order 1	$C_{i,\,j} = 1 - 2\varphi\, d / \max(d)$ $0 < \varphi < 1$
`banded`	equal bands	$C_{i,\,j} = 1$
	1 < order < nrows−1	$C_{i+k,\,i} = \theta_k$, $1 < k <$ order $-1 < \theta_k < 1$ $C_{i+k,\,i} = 0$, otherwise
`correlation`	general correlation matrix	$C_{i,\,i} = 1$
	1 < order < **nrows**−**1**	$C_{i,\,j} = \theta_{ij}$, $1 < \lvert i-j \rvert \le$ order $C_{i,\,j} = 0,\ \lvert i-j \rvert >$ order $-1 < \theta_{ij} < 1$
`uniform`	uniform matrix	$C_{i,\,j} = \theta$ for all i,j
`diagonal`	diagonal matrix	$C_{i,\,i} = \theta_i$ $C_{i,\,j} = 0,\ i \ne j$
`antedependence`	ante-dependence model	$C^{-1} = UD^{-1}U'$
	1 < order < nrows−1	$D_{i,\,i}^{-1} = d_i^{-1}$, $D_{i,\,j} = 0$ for $i \ne j$ $U_{i,\,i} = 1$, $U_{i,\,j} = u_{ij}$, $1 \le j-i \le$ order $U_{i,\,j} = 0$, for $i>j$
`unstructured`	general covariance matrix	$C_{i,\,j} = \theta_{ij}$,
	1 < order < **nrows**−**1**	$0 < \lvert i-j \rvert \le$ order

		$C_{i,j} = 0, \quad \lvert i-j \rvert >$ order
FA	factor analytic	$C = \Lambda\Lambda' + \Psi$
	order = **1** or 2	Λ is an nrows × q matrix order=q $\Psi_i = \psi_i$ for i=1...nrows
FAequal	factor analytic with common variance	
	order = **1** or 2	$C = \Lambda\Lambda' + \Psi$ Λ is an nrows × q matrix order=q $\Psi_i = \psi$ for i=1...nrows

Where more than one model order can be used, the default is shown in bold and can be changed by using the ORDER option. For the AR, MA, ARMA, power and banded models, the order is the same as the number of parameters to be fitted. For the banded, correlation, ante-dependence and unstructured models, the order is the number of non-zero off-diagonal bands in the matrix. For the FA models, the order is the number of columns in the matrix Λ.

Initial parameter values can be specified using the INITIAL parameter. For most models, the number of initial values required is the number of parameters, and default values can be generated. However, for unstructured models, a full covariance matrix of initial values must be given, and for the correlation model a full correlation matrix must be provided. For the ante-dependence model, either a full covariance matrix can be provided, or a pointer to a U and a D^{-1} matrix of the correct forms. For the FA and FAequal models, a pointer can be used to give the initial Λ and Ψ matrices, otherwise default initial values are generated. The FAequal model can be used to get initial values for the FA model. Initial values are required for these models because the algorithm may not converge when many parameters are fitted if the starting values are not realistic. Initial values might be generated from covariance matrices estimated by fitting simpler models (for an example see Section 5.4.3), or from residuals from a null variance model. A missing value in the initial values is taken to mean that the value is inestimable and it will be fixed at a small value for the analysis. Alternatively, a parameter can be fixed at its initial value using the CONSTRAINTS parameter. For each model defined, a text vector of constraint codes can be given in parallel with the initial values. The codes (not case sensitive and able to be abbreviated) may take value fix to indicate the parameter is to be fixed at its initial value, positive to indicate it is to remain positive or none to indicate no constraints. The default is a positive constraint or no constraint depending on context; for example, scaling parameters are always constrained to remain positive. The EQUALITYCONSTRAINTS parameter allows you to constrain some of the parameters to have the same value. The variate that it specifies contains a zero value if there is no constraint, and an identical integer value for any set of parameters whose values are to be equal. So, a variate containing the values (0,1,2,1,2) would constrain the second parameter to be equal to the fourth parameter, and the third parameter to be equal to the fifth parameter.

It may sometimes be desirable to allow for unequal variances for the models defined in terms of correlation matrices: that is, for the AR, MA, ARMA, uniform, power, boundedlinear, circular, spherical, linearvariance, banded and correlation models. This can be done using option setting HETEROGENEITY=outside. This means a diagonal matrix D of variances will be applied to the correlation matrix C to generate a matrix $D^{0.5}CD^{0.5}$. In this case, a number of extra parameters (equal to the number of effects in the factor or term) should be added to the vector of initial values. These models allow investigation of a structured correlation pattern for changing variances and are particularly useful in the analysis of repeated measurements data when variance increases over time. For example, to allow for changing variance over time in the repeated measurements example above, we specified

```
VCOMPONENTS [FIXED=Tmt] RANDOM=Subject.Week
```

```
VSTRUCTURE [TERM=Subject.Week] MODELTYPE=AR; FACTOR=Week; \
  HETEROGENEITY=outside
REML Y
```

In some circumstances, you may wish to define a single model to apply to the whole term, instead of using the direct product form illustrated above. In this case, you should set option FORMATION=whole. Note that when a term consists of a single factor, it is not necessary to set the FACTOR or FORMATION options.

If you set MODELTYPE=fixed, you must either give the values of the covariance matrix using the MATRIX option, or give the inverse matrix using the INVERSE option. Values for the matrix or its inverse can be supplied as diagonal matrices or symmetric matrices. In addition, values for the inverse matrix can be supplied in sparse form as a pointer. The output from VPEDIGREE (5.6.1) is designed for input here, but you can also define the inverse matrix explicitly. The second element of the pointer should then be a variate containing the non-zero values of the inverse in lower triangular order. The first element should be a factor, with number of levels equal to $n(n+1)/2$ where n is the number of rows of the matrix. This factor must contain first a block of n values giving the position in the variate of the first value stored for each row. This must be followed by a list indicating in which column each non-zero value of the matrix occurs, ordered by row.

When MODELTYPE=power is used to define a distance-based model, the model can be of order 1 (isotropic) or 2 (anisotropic). For models with ORDER=1, a single set of distances must be formed. The necessary information can be supplied using either the COORDINATES option, or the COORDINATES parameter, or the DISTANCES parameter. With the COORDINATES option you can specify either a matrix, or a list of variates, to define multi-dimensional coordinates for each unit of the data. The length of the variates, or the number of rows of the matrix, must be equal to the number of data values. The number of variates, or the number of columns of the matrix, is equal to the number of dimensions. If FORMATION=direct is used, the coordinates for each factor level are then calculated as the mean value of the units in the analysis with that level. In this case, it is essential that the set of coordinates corresponding to levels of other factors in the term is repeated for each level of the factor being processed. For example, a field experiment with row coordinates 1...12 and column coordinates 1,3,5,7,11 for all rows can use direct product formation. If one row had column coordinates 1,2,5,7,11 then direct product construction is not possible (since the covariance matrix C would then change between rows) and in this case, FORMATION=whole should be used (with constraints to restrict parameters to zero where necessary).

Alternatively, you can use the COORDINATES parameter to specify a single variate, a pointer to several variates or a matrix to define multi-dimensional coordinates for each level of the FACTOR. This parameter takes precedence over the COORDINATES option. The length of the variates, or the number of rows of the matrix, must be equal to the number of levels of the FACTOR. The number of variates, or the number of columns of the matrix, is again equal to the number of dimensions.

The distance calculation is defined by the METRIC option. For levels i and j with n-dimensional coordinates $\{c_{ik}: k=1...n\}$ and $\{c_{jk}: k=1...n\}$ the distance d_{ij} is defined as

$d_{ij} = \Sigma_k |c_{ik} - c_{jk}|$ for METRIC=cityblock (the default);
$d_{ij} = \Sigma_k (c_{ik} - c_{jk})^2$ for METRIC=squared; and
$d_{ij} = \{\Sigma_k (c_{ik} - c_{jk})^2\}^{1/2}$ for METRIC=euclidean.

Finally, you can supply a symmetric matrix of pre-calculated distances, using the DISTANCES parameter, and this takes precedence over the COORDINATES parameter and option. The number of rows of the DISTANCES matrix must be equal to the number of levels of the FACTOR.

When MODELTYPE=power and ORDER=2, the DISTANCES parameter cannot be used, and only two-dimensional coordinates are allowed. The coordinates must be specified using either the COORDINATES option or parameter, as described above. The distances are calculated within each

dimension separately, according to the setting of the METRIC option. In this case the Euclidean and city-block distances are equivalent.

The spherical family of geostatistical models correspond to the MODELTYPE settings boundedlinear (for one-dimensional distances), circular (for one or two dimensions) and spherical (for one or two dimensions). For further details, see Webster & Oliver (2007). These models are based on distances, and require coordinates to be supplied using either the COORDINATES option (to give coordinates for each data value), or the COORDINATES parameter (to give coordinates for each factor level), as described for MODELTYPE=power above. The parameter φ is interpreted as the range at which the correlation is considered to have decayed to zero. A small value therefore indicates weak correlation, and a large value indicates stronger correlation. These models do not have continuous second derivatives, and their log-likelihood may be multi-modal. To detect this potential problem, it is therefore important to start their estimation from several different initial values; this can be done using the INITIAL parameter as described above. To ensure that the estimated correlation matrix differs from the identity matrix, it is necessary for the range parameter to be larger than the minimum distance specified by the coordinates; any initial value smaller than this will be adjusted.

The setting MODELTYPE=linearvariance specifies the linear variance model of Williams (1986), extended by Piepho & Williams (2010). This model is parameterized so that the parameter φ lies in the range [0,1], which allows correlations in the range [-1,1]. Values of φ close to one indicate weak correlation and values close to zero indicate strong correlation between neighbouring observations.

The CORRELATE option allows you to specify correlations between model terms that have equal numbers of effects. A common correlation will then be fitted between parallel effects as in the random coefficient regression example described above. Correlations between terms can be cancelled using CORRELATE=none (the default). The CORRELATE option setting positivedefinite can be used to ensure that the correlation matrix between the terms remains positive definite. This constraint can be relaxed using the setting unrestricted (an unstructured covariance matrix is then used to describe covariance across the terms). The model fitting is done here in terms of a covariance matrix, where the diagonal elements are the gammas for the correlated terms. The CINITIAL option is used to give initial values for this matrix. If no initial values are given, the initial values are taken from initial gamma values given in VCOMPONENTS when the model is declared. A missing value in the initial values is taken to mean that the value is inestimable and it will be fixed at a value close to zero during the analysis. When correlations are declared between terms, you must set FORMATION=whole. In the random coefficient regression model above, no correlation structure is declared within terms since the subjects are independent. However, it is possible to declare correlation/covariance models within terms as usual. For example, an animal model might use VPEDIGREE to set up an inverse relationship matrix A^{-1}, then use this matrix to model covariances within terms:

```
VPEDIGREE INDIVIDUALS=animal; FEMALE=dam; MALE=sire;\
  INVERSE=Ainv
VCOMPONENTS [FIXED=Trt] RANDOM=animal+dam+env
VSTRUCTURE [TERM=animal+dam; CORRELATE=unrestricted; \
  FORM=whole] MODELTYPE=fixed; INVERSE=Ainv
```

These declarations set up random terms with covariance structures of the form: cov(animal) = $\sigma_a^2 A$, cov(dam) = $\sigma_a^2 A$, cov(animal,dam) = $\sigma_{ad} A$.

5.4.2 Displaying the model: the VSTATUS directive

The VSTATUS directive can be used to print out, and hence check, the fixed and random models and covariance structures as set up by the VCOMPONENTS and VSTRUCTURE directives, prior to using REML to run an analysis.

VSTATUS directive

Prints the current model settings for REML.

Option

PRINT = *string tokens*	What to print (model); default mode

No parameters

5.4.3 A repeated measurements example

The example in this section uses data generated from an experiment at Rothamsted by J. Lamptey. The data analysed here consists of five measurements of growth after 1, 3, 5, 7 and 10 weeks for 14 plants in a glasshouse, and the measurements over time are shown for each plant in Figure 5.4.3. (For details of the DREPMEASURES procedure that produced the plot, see Section 8.1.1) Individual plants were either diseased or healthy, i.e. two treatments, with seven replicates of each, and the plants were arranged in a completely randomised design. An analysis is illustrated in Example 5.4.3 below, using the saturated treatment model Plant.Treatment throughout in order to investigate the structure of the residual variance. The first analysis (Example 5.4.3a) is the standard ANOVA split-plot analysis, specified by setting random model Plant/Week, or equivalently, Plant+Plant.Week. As explained earlier (Section 5.4.1), this generates a uniform correlation over time, plus extra measurement error, and is equivalent to the second analysis, which uses random model Plant.Week and specifies the uniform correlation structure explicitly.

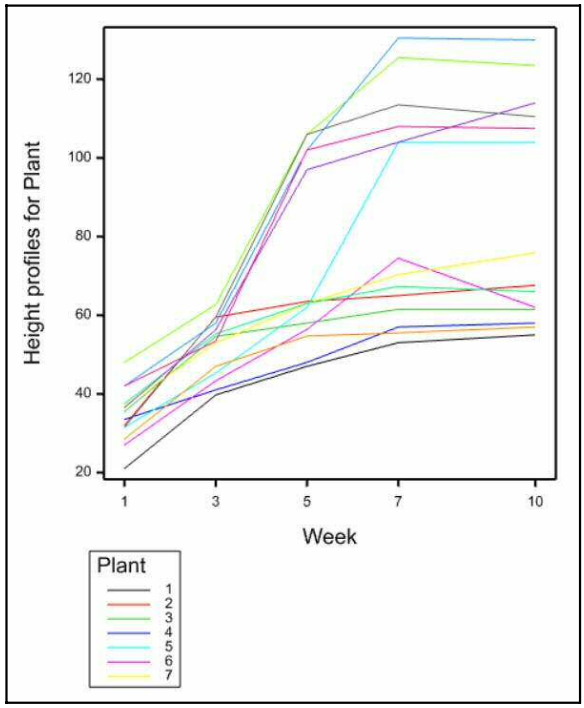


Figure 5.4.3

Example 5.4.3a

```
  2  " Repeated measurements analysis:
 -3    growth of 14 plants measured after 1,3,5,7,10 weeks."
  4  FACTOR       Plant
  5  &            [LABELS=!T(HC,MAV)] Treatment
  6  OPEN         '%GENDIR%/Examples/GuidePart2/Plant.dat'; CHANNEL=2
  7  READ         [CHANNEL=2] Plant,Treatment,Time,Height; \
  8               FREPRESENTATION=levels,labels,levels,levels

 Identifier   Minimum      Mean   Maximum    Values   Missing
       Time     1.000     5.200     10.00        70         0
     Height     21.00     66.04     130.5        70         0

 Identifier    Values   Missing    Levels
      Plant        70         0        14
  Treatment        70         0         2
```

```
   9  CLOSE        2
  10  GROUPS       Time; FACTOR=Week
  11  DREPMEASURES [GROUPS=Plant; TIMEPOINTS=Week] Height
  12  " Anova split-plot analysis."
  13  VCOMPONENTS [FIXED=Treatment*Week] Plant/Week
  14  REML        Height

REML variance components analysis
=================================

Response variate:  Height
Fixed model:       Constant + Week + Treatment + Week.Treatment
Random model:      Plant + Plant.Week
Number of units:   70

Plant.Week used as residual term

Sparse algorithm with AI optimisation


Estimated variance components
-----------------------------

Random term               component        s.e.
Plant                         159.8        75.7


Residual variance model
-----------------------

Term                          Model(order)  Parameter        Estimate      s.e.
Plant.Week                    Identity      Sigma2              126.5      25.8


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term                 Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Week                               217.83       4         54.46    48.0  <0.001
Treatment                            9.41       1          9.41    12.0   0.010
Week.Treatment                      20.41       4          5.10    48.0   0.002

Dropping individual terms from full fixed model

Fixed term                 Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Week.Treatment                      20.41       4          5.10    48.0   0.002

* MESSAGE: denominator degrees of freedom for approximate F-tests are calculated
using algebraic derivatives ignoring fixed/boundary/singular variance
parameters.


  15  " Equivalent analysis using uniform correlation structure"
  16  VCOMPONENTS [FIXED=Treatment*Week] Plant.Week
  17  VSTRUCTURE  [TERM=Plant.Week] MODELTYPE=uniform; FACTOR=Week
  18  REML        [PRINT=model,components] Height

REML variance components analysis
=================================

Response variate:  Height
Fixed model:       Constant + Week + Treatment + Week.Treatment
Random model:      Plant.Week
Number of units:   70

Plant.Week used as residual term with covariance structure as below

Sparse algorithm with AI optimisation
```

```
Covariance structures defined for random model
----------------------------------------------

Covariance structures defined within terms:

Term                    Factor      Model                        Order  No. rows
Plant.Week              Plant       Identity                         1        14
                        Week        Uniform                          1         5


Residual variance model
-----------------------

Term             Factor        Model(order)  Parameter        Estimate      s.e.
Plant.Week                                   Sigma2              286.3      78.3
                 Plant         Identity      -                       -         -
                 Week          Uniform       theta1             0.5582    0.1304
```

To verify equivalence of the two analyses, it is necessary to take into account the different parameterisations of the variance model. For the split-plot analysis, the variance model for y_{ij} (plant i, week j) is

$\text{var}(y_{ij}) = \sigma_p^2 + \sigma_1^2$, $\text{cov}(y_{ij}, y_{ik}) = \sigma_p^2$ for $j \neq k$, $\text{cov}(y_{ij}, y_{kl}) = 0$ for $i \neq k$

where σ_p^2 and σ_1^2 are the `Plant` and `Plant.Week` (or residual) variance components. For the uniform correlation model, with residual σ_2^2 and correlation parameter θ:

$\text{var}(y_{ij}) = \sigma_2^2$, $\text{cov}(y_{ij}, y_{ik}) = \theta\sigma_2^2$ for $j \neq k$, $\text{cov}(y_{ij}, y_{kl}) = 0$ for $i \neq k$

Hence $\theta\sigma_2^2 = \sigma_p^2$ and $\sigma_2^2 = \sigma_p^2 + \sigma_1^2$.

Where estimated parameters for covariance models are given, the labelling of the parameters corresponds to the model definitions given in Section 5.4.1.

Rather than uniform correlation over time, a more realistic model might decrease the correlation as the time between measurements increases. For equally spaced data, the auto-regressive model is often used. In Example 5.4.3a, the measurements are not equally spaced, so Example 5.4.3b fits a power model where correlation depends on the distance between time points calculated from the coordinates specified using the `COORDINATES` option of `VSTRUCTURE`. Since distances are calculated across only one dimension (time), it is sufficient to specify only one variate of coordinates corresponding to the different times of measurement. The same answer would be obtained by also specifying plant values (as a variate `vplant`, say) for the second dimension, i.e. `COORDINATES=vplant,Time`.

Note that this analysis would not be suitable if each plant was measured at different times, as the direct product structure would not hold: see Section 5.4.6 for further details.

Example 5.4.3b

```
  19  " Power model:
 -20    - correlation decreases as time between measurements increase
 -21    - takes account of unequally spaced measurements
 -22    - co-ordinates must be specified as a list of variates or a matrix."
  23  VCOMPONENTS [FIXED=Treatment*Week] Plant.Week
  24  VSTRUCTURE  [TERM=Plant.Week; COORDINATES=Time] MODELTYPE=power; FACTOR=Week
  25  REML        [PRINT=model,components,wald,deviance] Height

* MESSAGE: Ordering of units in COORDINATES option expected to match ordering of
data values.

REML variance components analysis
=================================

Response variate:  Height
Fixed model:       Constant + Week + Treatment + Week.Treatment
Random model:      Plant.Week
Number of units:   70

Plant.Week used as residual term with covariance structure as below
```

```
Sparse algorithm with AI optimisation


Covariance structures defined for random model
----------------------------------------------

Covariance structures defined within terms:

Term                    Factor      Model                          Order  No. rows
Plant.Week              Plant       Identity                           1        14
                        Week        Power - city block distance        1         5


Residual variance model
-----------------------

Term             Factor        Model(order)  Parameter        Estimate      s.e.
Plant.Week                                   Sigma2              301.6      96.8
                 Plant         Identity      -                     -         -
                 Week          Power(1)      phi_1              0.9190    0.0312



Deviance: -2*Log-Likelihood
---------------------------

                  Deviance   d.f.
                    365.96    58

Note: deviance omits constants which depend on fixed model fitted.


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Week                              159.27       4         39.75    46.3  <0.001
Treatment                           6.87       1          6.87    11.6   0.023
Week.Treatment                     24.51       4          6.12    46.3  <0.001

Dropping individual terms from full fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Week.Treatment                     24.51       4          6.12    46.3  <0.001

* MESSAGE: denominator degrees of freedom for approximate F-tests are calculated
using numerical derivatives ignoring fixed/boundary/singular variance
parameters.
```

The calculation of the deviance omits the terms $(n-p)\log 2\pi - \log|X'X|$, and is the same as the deviance printed out in monitoring information. The deviance cannot be used to compare models with different fixed effects (Welham & Thompson 1997), but it can be used to compare different nested random models.

Residuals from the analysis indicate variance increasing over time. This can be modelled directly by specifying that heterogeneity is to be introduced into the power model, using parameter `HETEROGENEITY=outside`, which means that a separate scaling parameter will be estimated for each time point.

Example 5.4.3c

```
 26  " Heterogeneous power model - correlations follow power model,
-27    variance allowed to change over time."
 28  VSTRUCTURE  [TERM=Plant.Week; COORDINATES=Time] MODELTYPE=power; \
 29              ORDER=1; FACTOR=Week; HETEROGENEITY=outside
 30  REML        [PRINT=model,components,wald,deviance] Height
```

```
* MESSAGE: Ordering of units in COORDINATES option expected to match ordering of
data values.

REML variance components analysis
=================================

Response variate:  Height
Fixed model:       Constant + Week + Treatment + Week.Treatment
Random model:      Plant.Week
Number of units:   70

Plant.Week used as residual term with covariance structure as below

Sparse algorithm with AI optimisation


Covariance structures defined for random model
----------------------------------------------

Covariance structures defined within terms:

Term                    Factor      Model                          Order  No. rows
Plant.Week              Plant       Identity                           0        14
                        Week        Power - city block distance (het)
                                                                       1         5


Residual variance model
-----------------------

Term             Factor        Model(order)  Parameter        Estimate      s.e.
Plant.Week                                   Sigma2              1.000     fixed
                 Plant         Identity      -                     -         -
                 Week          Power(1) het  phi_1               0.9068    0.0415
                                             Scale row 1          60.79     28.50
                                             Scale row 2          73.18     36.87
                                             Scale row 3          308.7     138.6
                                             Scale row 4          435.5     172.5
                                             Scale row 5          381.8     139.2


Deviance: -2*Log-Likelihood
---------------------------

                  Deviance   d.f.
                    342.99    54

Note: deviance omits constants which depend on fixed model fitted.


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Week                              227.48       4         51.82    21.0  <0.001
Treatment                           0.00       1          0.00    13.2   0.971
Week.Treatment                     19.08       4          4.35    21.0   0.010

Dropping individual terms from full fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Week.Treatment                     19.08       4          4.35    21.0   0.010

* MESSAGE: denominator degrees of freedom for approximate F-tests are calculated
using numerical derivatives ignoring fixed/boundary/singular variance
parameters.
```

The variance model for term `Plant.Week` now takes the form $\sigma^2 R = \sigma^2 D^{0.5} C D^{0.5}$ where C is the

power correlation matrix and D is a matrix of scaling parameters. In this specification, there are too many scaling parameters, and either σ^2 or D must be constrained. If, as here, the sigma parameterization (5.3.1) is being used, the constraint $\sigma^2=1$ is imposed. Alternatively, with the gamma parameterization, Genstat constrains the first scaling parameter $d_1=1$. The estimated variance for measurements at the time point i is then the variance component for the term (here the residual, σ^2) multiplied by the scaling parameter, $\sigma^2 d_i$.

The change in deviance of 22.97 on 4 df between the two fits (as a likelihood ratio test, compared to a χ^2 distribution on 4 df) indicates that the heterogenous power model gives a better fit to the variance structure. Note that tests based on change in deviance can be used only to compare nested random models which use the same fixed model. Here, the first model can be considered to have $d_i=1$ for all i.

This model can be compared to the fit given by an unstructured variance model. Initial values must be specified for unstructured or ante-dependence models. These might either be saved from a simpler model, such as the power model fitted above, or calculated using residuals obtained after fitting a null variance model. Both methods are illustrated in Example 5.4.3d.

Example 5.4.3d

```
  31  " Save fitted covariance model as initial values for unstructured model."
  32  VKEEP       TERM=Plant.Week; COVARIANCEMODEL=Covpower
  33  PRINT       Covpower['Week']

            Covpower['Week']
           1        60.8
           2        54.8        73.2
           3        92.6       123.6       308.7
           4        90.4       120.7       301.5       435.5
           5        63.1        84.3       210.5       304.0       381.8
                       1           2           3           4           5

  34  " Unstructured model."
  35  VSTRUCTURE  [TERM=Plant.Week] MODELTYPE=unstructured; FACTOR=Week; \
  36              INITIAL=Covpower['Week']
  37  REML        [PRINT=model,components,wald,deviance] Height

REML variance components analysis
=================================

Response variate:  Height
Fixed model:       Constant + Week + Treatment + Week.Treatment
Random model:      Plant.Week
Number of units:   70

Plant.Week used as residual term with covariance structure as below

Sparse algorithm with AI optimisation


Covariance structures defined for random model
----------------------------------------------

Covariance structures defined within terms:

Term                    Factor      Model                         Order  No. rows
Plant.Week              Plant       Identity                          0        14
                        Week        Unstructured                      4         5


Residual variance model
-----------------------

Term             Factor        Model(order)  Parameter        Estimate      s.e.
Plant.Week                                   Sigma2              1.000     fixed
                 Plant         Identity      -                       -         -
                 Week          Unstructured  v_11                37.23     15.20
                                             v_21                23.39     13.21
```

```
v_22                41.52     16.95
v_31                51.65     32.03
v_32                61.92     34.87
v_33                259.1     105.8
v_41                70.81     46.14
v_42                57.61     46.74
v_43                331.8     145.2
v_44                551.5     225.2
v_51                73.79     46.21
v_52                62.57     46.93
v_53                330.9     144.3
v_54                533.8     220.6
v_55                542.2     221.3


Deviance: -2*Log-Likelihood
---------------------------

                  Deviance   d.f.
                    316.07    45

Note: deviance omits constants which depend on fixed model fitted.


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Week                              215.86       4         40.47     9.0  <0.001
Treatment                           1.71       1          1.71    12.0   0.215
Week.Treatment                     17.84       4          3.34     9.0   0.061

Dropping individual terms from full fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Week.Treatment                     17.84       4          3.34     9.0   0.061

* MESSAGE: denominator degrees of freedom for approximate F-tests are calculated
using algebraic derivatives ignoring fixed/boundary/singular variance
parameters.


  38  " Alternatively, generate initial values using residuals generated after
 -39    fitting with no variance model."
  40  VCOMPONENTS [FIXED=Treatment*Week]
  41  REML        [PRINT=*] Height; RESIDUALS=r
  42  " Residuals are in order plants within weeks, so matrix rows correspond
 -43    to weeks and columns to plants."
  44  MATRIX      [ROWS=5; COLUMNS=14; VALUES=#r] mres
  45  SYMMETRIC   [ROWS=5] vcov
  46  CALCULATE   vcov = mres *+ TRANSPOSE(mres)
  47  " Dividing by number of replicates within each week gives an easy but
 -48    conservative estimate as it takes no account of treatment d.f."
  49  CALCULATE   vcov = vcov / 14
  50  " Unstructured model from new initial values."
  51  VCOMPONENTS [FIXED=Treatment*Week] Plant.Week
  52  VSTRUCTURE  [TERM=Plant.Week] MODELTYPE=unstructured; FACTOR=Week; \
  53              INITIAL=!(#vcov)
  54  REML        [PRINT=deviance] Height

Deviance: -2*Log-Likelihood
---------------------------

                  Deviance   d.f.
                    316.07    45

Note: deviance omits constants which depend on fixed model fitted.
```

Again, the matrix takes the form $\sigma^2 C$, where C is an unstructured covariance matrix, so the identifiability constraint $\sigma^2=1$ is imposed. With the gamma parameterization, the constraint would be $c_{1,1}=1$.

For an ante-dependence analysis, a similar constraint is required. In this case the variance structure is $\sigma^2 R$, where $R^{-1}=UD^{-1}U'$ for some upper triangular U and diagonal matrix D^{-1}, and $\sigma^2=1$. With gamma parameterization, d_1 would be fixed at an arbitrary value.

Example 5.4.3e

```
  55  " Ante-dependence model order 1 - also requires initial values."
  56  VSTRUCTURE  [TERM=Plant.Week] antedependence; FACTOR=Week; \
  57              INITIAL=Covpower['Week']; ORDER=1
  58  REML        [PRINT=model,components,deviance] Height

REML variance components analysis
=================================

Response variate:  Height
Fixed model:       Constant + Week + Treatment + Week.Treatment
Random model:      Plant.Week
Number of units:   70

Plant.Week used as residual term with covariance structure as below

Sparse algorithm with AI optimisation


Covariance structures defined for random model
----------------------------------------------

Covariance structures defined within terms:

Term                    Factor      Model                         Order  No. rows
Plant.Week              Plant       Identity                          0        14
                        Week        Antedependence                    1         5


Residual variance model
-----------------------

Term             Factor        Model(order)  Parameter        Estimate      s.e.
Plant.Week                                   Sigma2              1.000     fixed
                 Plant         Identity      -                       -         -
                 Week          Antedependence(1)
                                             dinv_1            0.02686   0.01102
                                             dinv_2            0.03729   0.01547
                                             dinv_3           0.005996  0.002468
                                             dinv_4           0.007897  0.003233
                                             dinv_5            0.03906   0.01595
                                             u_12              -0.6284    0.2459
                                             u_23               -1.491     0.586
                                             u_34               -1.280     0.207
                                             u_45              -0.9678    0.0628


Deviance: -2*Log-Likelihood
---------------------------

                  Deviance   d.f.
                    320.74    51

Note: deviance omits constants which depend on fixed model fitted.

  59  " Ante-dependence model order 2."
 -60  - use initial values from power model again ."
  61  VSTRUCTURE  [TERM=Plant.Week] antedependence; FACTOR=Week; \
  62              INITIAL=Covpower['Week']; ORDER=2
  63  REML        [PRINT=deviance] Height
```

```
Deviance: -2*Log-Likelihood
---------------------------

                  Deviance   d.f.
                    317.30    48

Note: deviance omits constants which depend on fixed model fitted.
```

In this example, the ante-dependence model of order 1 appears to give a good fit to the data. The ante-dependence model can be regarded as a generalisation of the auto-regressive model. In this context, σ^2/d_i is analogous to the AR process variance at each time point, and U_{ji} is the regression coefficient for time i on time j ($i>j$).

5.4.4 An example of spatial analysis of a field experiment

Example 5.4.4a shows the layout and standard analysis of a field experiment (at Slate Hall Farm in 1976, previously analysed by Gilmour *et al*. 1995) laid out as a lattice square in 6 replicates.

Example 5.4.4a

```
   2  " Slate Hall Farm 1976:
  -3    data from Gilmour et al. (1995) Biometrics 51, 1440-1450.
  -4
  -5    Balanced Lattice Design with Replicates laid out as:
  -6
  -7              1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
  -8              1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
  -9              1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
 -10              1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
 -11              1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
 -12              4 4 4 4 4 5 5 5 5 5 6 6 6 6 6
 -13              4 4 4 4 4 5 5 5 5 5 6 6 6 6 6
 -14              4 4 4 4 4 5 5 5 5 5 6 6 6 6 6
 -15              4 4 4 4 4 5 5 5 5 5 6 6 6 6 6
 -16              4 4 4 4 4 5 5 5 5 5 6 6 6 6 6
 -17  "
  18  FACTOR      Replicate,Rowblock,Colblock,Variety; DECIMALS=0
  19  OPEN        'Slatehfm.dat'; CHANNEL=2
  20  READ        [CHANNEL=2] Replicate,Rowblock,Colblock,Variety,Yield

    Identifier   Minimum      Mean   Maximum    Values   Missing
         Yield     917.0      1470      2119       150         0

    Identifier    Values   Missing    Levels
     Replicate       150         0         6
      Rowblock       150         0        30
      Colblock       150         0        30
       Variety       150         0        25

  21  CLOSE       2
  22  CALCULATE   Yield = Yield * 0.01
  23  FACTOR      [NVALUES=150; LEVELS=10] Row
  24  &           [LEVELS=15] Column
  25  GENERATE    Row,Column
  26  VARIATE     Vrow,Vcolumn; VALUES=Row,Column
  27  " Analysis using design blocking factors and AI method."
  28  VCOMPONENTS [FIXED=Variety] Replicate/(Rowblock*Colblock)
  29  REML        [PRINT=model,components,wald; METHOD=ai] Yield

REML variance components analysis
=================================

Response variate:  Yield
Fixed model:       Constant + Variety
Random model:      Replicate + Replicate.Rowblock + Replicate.Colblock
                   + Replicate.Rowblock.Colblock
Number of units:   150
```

```
Replicate.Rowblock.Colblock used as residual term

Sparse algorithm with AI optimisation


Estimated variance components
-----------------------------

Random term               component        s.e.
Replicate                    0.4262      0.6890
Replicate.Rowblock           1.5595      0.5091
Replicate.Colblock           1.4812      0.4865


Residual variance model
-----------------------

Term                           Model(order)  Parameter  Estimate      s.e.
Replicate.Rowblock.Colblock    Identity      Sigma2        0.806    0.1340


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term             Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Variety                        212.26      24          8.84    79.3  <0.001

Dropping individual terms from full fixed model

Fixed term             Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Variety                        212.26      24          8.84    79.3  <0.001

* MESSAGE: denominator degrees of freedom for approximate F-tests are
calculated using algebraic derivatives ignoring fixed/boundary/singular
variance parameters.
```

This is the conventional analysis and assumes uniform correlation separately across rows and columns within replicates, which is also an assumption of the design when the experiment is laid out. An alternative approach considers the layout as a two-dimensional array and attempts to model the underlying variance patterns. With the assumption that the error process is separable, i.e. correlation across rows is independent of columns and vice versa, the two-dimensional variance structure can be modelled as a direct product of a correlation model across rows with a correlation model across columns (see Cullis & Gleeson 1991).

Example 5.4.4b illustrates three methods of specifying the same analysis, via a two-dimensional power model, a direct product of two one-dimensional power models, and as a direct product of auto-regressive models. The direct product specification is more natural and more efficient in this context, but it could not be used for plots laid out in an irregular pattern, in which case the two-dimensional power model would be used.

Example 5.4.4b

```
  30  " Two-dimensional power model based on (Row,Column) co-ordinates."
  31  VCOMPONENTS [FIXED=Variety] Row.Column
  32  VSTRUCTURE  [TERM=Row.Column; FORMATION=whole; \
  33              COORDINATES=Vrow,Vcolumn] power; ORDER=2
  34  REML        [PRINT=model,components] Yield

* MESSAGE: Ordering of units in COORDINATES option expected to match ordering of
data values.

REML variance components analysis
=================================

Response variate:  Yield
```

```
Fixed model:       Constant + Variety
Random model:      Row.Column
Number of units:   150

Row.Column used as residual term with covariance structure as below

Sparse algorithm with AI optimisation


Covariance structures defined for random model
----------------------------------------------

Covariance structures defined within terms:

Term        Factor      Model                                  Order  No. rows
Row.Column  Whole term  Power - city block distance (+ scalar)    2       150


Residual variance model
-----------------------

Term             Factor        Model(order)  Parameter   Estimate      s.e.
Row.Column                                   Sigma2         3.876     0.775
                 Whole term    Power(2)      phi_1          0.4586    0.0826
                                             phi_2          0.6838    0.0633


  35  " Equivalent - more efficient - specification as power x power
 -36    using separability in layout."
  37  VCOMPONENTS [FIXED=Variety] Row.Column
  38  VSTRUCTURE  [TERM=Row.Column; COORDINATES=Vrow,Vcolumn] \
  39               MODELTYPE=power,power; FACTOR=Row,Column
  40  REML        [PRINT=model,components] Yield

* MESSAGE: Ordering of units in COORDINATES option expected to match ordering of
data values.

* MESSAGE: Ordering of units in COORDINATES option expected to match ordering of
data values.

REML variance components analysis
=================================

Response variate:  Yield
Fixed model:       Constant + Variety
Random model:      Row.Column
Number of units:   150

Row.Column used as residual term with covariance structure as below

Sparse algorithm with AI optimisation


Covariance structures defined for random model
----------------------------------------------

Covariance structures defined within terms:

Term         Factor   Model                                  Order  No. rows
Row.Column      Row   Power - city block distance (+ scalar)    1        10
             Column   Power - city block distance               1        15


Residual variance model
-----------------------

Term             Factor        Model(order)  Parameter  Estimate      s.e.
Row.Column                                   Sigma2        3.876     0.775
                 Row           Power(1)      phi_1         0.4586    0.0826
                 Column        Power(1)      phi_1         0.6838    0.0633


  41  " AR1 x AR1 - equivalent to power model for equal spacing."
```

```
  42  VCOMPONENTS [FIXED=Variety] Row.Column
  43  VSTRUCTURE  [TERM=Row.Column] AR,AR; FACTOR=Row,Column
  44  REML        [PRINT=model,components,deviance,wald] Yield

REML variance components analysis
=================================

Response variate:  Yield
Fixed model:       Constant + Variety
Random model:      Row.Column
Number of units:   150

Row.Column used as residual term with covariance structure as below

Sparse algorithm with AI optimisation


Covariance structures defined for random model
----------------------------------------------

Covariance structures defined within terms:

Term        Factor   Model                          Order  No. rows
Row.Column     Row   Auto-regressive (+ scalar)         1        10
            Column   Auto-regressive                    1        15


Residual variance model
-----------------------

Term             Factor        Model(order)  Parameter  Estimate      s.e.
Row.Column                                   Sigma2        3.876     0.775
                 Row           AR(1)         phi_1         0.4586    0.0826
                 Column        AR(1)         phi_1         0.6838    0.0633


Deviance: -2*Log-Likelihood
---------------------------

                  Deviance   d.f.
                    249.35   122

Note: deviance omits constants which depend on fixed model fitted.


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Variety                         313.04      24         13.04    80.0  <0.001

Dropping individual terms from full fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Variety                         313.04      24         13.04    80.0  <0.001

* MESSAGE: denominator degrees of freedom for approximate F-tests are
calculated using algebraic derivatives ignoring fixed/boundary/singular
variance parameters.
```

The AR(1) ⊗ AR(1) structure used in Example 5.4.4b models any underlying trend over the field, but does not allow for any extra measurement error. This can be added to the model explicitly, either by generating a new units factor, or by specifying the term `'*units*'` in the random model to indicate that an extra residual term is to be added. The analysis is shown in Example 5.4.4c. Genstat produces a warning about the two residual terms and tells you which one is to be used to provide the *R* matrix.

Example 5.4.4c

```
  45  " AR1 x AR1 + independent error."
  46  VCOMPONENTS [FIXED=Variety] Row.Column+'*units*'
  47  VSTRUCTURE  [TERM=Row.Column] AR,AR; FACTOR=Row,Column; \
  48              INITIAL=!(.45),!(.68)
  49  REML        [PRINT=model,monitoring,components,deviance,wald] Yield

******** Warning, code VC 53, statement 1 on line 49

Command: REML [PRINT=model,monitoring,components,deviance,wald] Yield
More than one residual term specified - first term found will be used as R.


REML variance components analysis
=================================

Response variate:  Yield
Fixed model:       Constant + Variety
Random model:      Row.Column + '*units*'
Number of units:   150

Row.Column used as residual term with covariance structure as below

Sparse algorithm with AI optimisation


Covariance structures defined for random model
----------------------------------------------

Covariance structures defined within terms:

Term         Factor   Model                        Order  No. rows
Row.Column      Row   Auto-regressive (+ scalar)       1        10
             Column   Auto-regressive                  1        15


Convergence monitoring
----------------------

Cycle   Deviance   Current variance parameters: gammas, sigma2, others
    0    274.471   1.00000   1.35490  0.680000  0.450000
    1    267.202  0.691570   1.58926  0.671743  0.443532
    2    249.772  0.181262   2.52092  0.678574  0.458822
    3    244.705  0.177878   3.00935  0.779718  0.567184
    4    242.428 0.0975166   4.52996  0.834529  0.662769
    5    242.354  0.107328   4.54878  0.843781  0.680285
    6    242.353  0.106086   4.57634  0.843545  0.682242
    7    242.353  0.106171   4.57955  0.843782  0.682631
    8    242.353  0.106153   4.58026  0.843793  0.682685
    9    242.353  0.106153   4.58036  0.843798  0.682695


Estimated variance components
-----------------------------

Random term               component        s.e.
Extra units term              0.486       0.179


Residual variance model
-----------------------

Term             Factor        Model(order)  Parameter  Estimate      s.e.
Row.Column                                   Sigma2        4.580     1.670
                 Row           AR(1)         phi_1        0.6827    0.1023
                 Column        AR(1)         phi_1        0.8438    0.0684
```

```
Deviance: -2*Log-Likelihood
---------------------------

                  Deviance   d.f.
                    242.35   121

Note: deviance omits constants which depend on fixed model fitted.


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Variety                         245.39      24         10.21    75.7  <0.001

Dropping individual terms from full fixed model

Fixed term              Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Variety                         245.39      24         10.21    75.7  <0.001

* MESSAGE: denominator degrees of freedom for approximate F-tests are
calculated using algebraic derivatives ignoring fixed/boundary/singular
variance parameters.
```

In this example, there is a relatively large reduction in the deviance (7.0 on 1 df) after adding the extra random error term. The values of the auto-regressive coefficients also increase, indicating that these had been artificially depressed in the absence of the random error term. However, the auto-regressive coefficients still give a substantial decrease in correlation in both directions across the layout, indicating that this may be a more realistic model than the lattice analysis, although in practice there is little difference in the estimates of fixed effects from models in Examples 5.4.4a and 5.4.4c.

Example 5.4.4d shows the two formats in which the estimated covariance models can be printed: either as the component matrices (the default) or as their parameters (option `CFORMAT=parameters`). Here there is a vector with two elements for each direction. The first is the parameter of the AR1 process, and the other is zero (it would be non-zero if we had an AR2 process).

Example 5.4.4d

```
  50  VDISPLAY    [PRINT=covariancemodels]

Estimated covariance models
---------------------------

Variance of data estimated in form:

V(y) = Sigma2( gZZ' + R )

where: V(y) is variance matrix of data
       Sigma2 is the residual variance
       g is the gamma for the random term
       Z is the incidence matrix for the random term
       R is the residual covariance matrix

Note: a gamma is the ratio of a variance component to the residual (Sigma2)


Random Term: Extra units term

Scalar Sigma2*g: 0.4862


Residual term: Row.Column
```

```
Sigma2: 4.580

R uses direct product construction

Factor: Row
Model:  Auto-regressive

Covariance matrix:


 1  1.000
 2  0.683  1.000
 3  0.466  0.683  1.000
 4  0.318  0.466  0.683  1.000
 5  0.217  0.318  0.466  0.683  1.000
 6  0.148  0.217  0.318  0.466  0.683  1.000
 7  0.101  0.148  0.217  0.318  0.466  0.683  1.000
 8  0.069  0.101  0.148  0.217  0.318  0.466  0.683  1.000
 9  0.047  0.069  0.101  0.148  0.217  0.318  0.466  0.683  1.000
10  0.032  0.047  0.069  0.101  0.148  0.217  0.318  0.466  0.683  1.000
        1      2      3      4      5      6      7      8      9     10

Factor: Column
Model:  Auto-regressive

Covariance matrix (first 10 rows only):


 1  1.000
 2  0.844  1.000
 3  0.712  0.844  1.000
 4  0.601  0.712  0.844  1.000
 5  0.507  0.601  0.712  0.844  1.000
 6  0.428  0.507  0.601  0.712  0.844  1.000
 7  0.361  0.428  0.507  0.601  0.712  0.844  1.000
 8  0.305  0.361  0.428  0.507  0.601  0.712  0.844  1.000
 9  0.257  0.305  0.361  0.428  0.507  0.601  0.712  0.844  1.000
10  0.217  0.257  0.305  0.361  0.428  0.507  0.601  0.712  0.844  1.000
        1      2      3      4      5      6      7      8      9     10

  51  VDISPLAY    [PRINT=covariancemodels; CFORMAT=parameters]

Estimated covariance models
---------------------------

Variance of data estimated in form:

V(y) = Sigma2( gZZ' + R )

where: V(y) is variance matrix of data
       Sigma2 is the residual variance
       g is the gamma for the random term
       Z is the incidence matrix for the random term
       R is the residual covariance matrix

Note: a gamma is the ratio of a variance component to the residual (Sigma2)


Random Term: Extra units term

Scalar Sigma2*g: 0.4862


Residual term: Row.Column

Sigma2: 4.580

R uses direct product construction

Factor: Row
Model:  Auto-regressive
```

```
Vector of parameters, vector of variances (if heterogeneous)

Parameters

  0.6827
  0.0000

Factor: Column
Model:  Auto-regressive

Vector of parameters, vector of variances (if heterogeneous)

Parameters

  0.8438
  0.0000
```

The variogram can be a useful diagnostic tool in these circumstances. Procedure `F2DRESIDUALVARIOGRAM` can produce a two-dimensional variogram as described in Gilmour *et al*. (1997), or the `FVARIOGRAM` directive can form one-dimensional variograms calculated in specific directions (e.g. over rows or over columns).

5.4.5 An example of random coefficient regression

Random coefficient regression seeks to model individual profiles over time using linear models with common parameters within treatment groups, allowing for random variation about these parameters for individuals. We illustrate an analysis using the plant growth data of Example 5.4.3. The plant profiles in Figure 5.4.3 indicate that linear plus quadratic terms over time may be required to model the profiles. We fit fixed effects model `Treatment*(Time+Timesqrd)` to allow a separate quadratic profile over time for each treatment. The analysis in Example 5.4.5a fits random terms `Plant` to generate a random intercept (or constant) for each plant, and `Plant.Time` to generate a random slope for each plant, without correlation.

Example 5.4.5a

```
  64  " Random coefficient regression
 -65    Growth of 14 plants measured after 1,3,5,7,10 weeks.
 -66    Profiles suggest use of quadratic functions:
 -67    fit random intercept and slope for plants - no correlation."
  68  CALCULATE   Timesqrd = Time * Time
  69  VCOMPONENTS [FIXED=Treatment*(Time+Timesqrd)] Plant+Plant.Time
  70  REML        [PRINT=model,components,deviance] Height

REML variance components analysis
=================================

Response variate:  Height
Fixed model:       Constant + Time + Treatment + Timesqrd + Time.Treatment +
Treatment.Timesqrd
Random model:      Plant + Plant.Time
Number of units:   70

Residual term has been added to model

Sparse algorithm with AI optimisation
All covariates centred


Estimated variance components
-----------------------------

Random term               component        s.e.
Plant                        173.05       75.62
Plant.Time                     6.43        3.14
```

```
Residual variance model
-----------------------

Term                            Model(order)  Parameter        Estimate      s.e.
Residual                        Identity      Sigma2             60.30     13.48


Deviance: -2*Log-Likelihood
---------------------------

                   Deviance   d.f.
                     415.57    61

Note: deviance omits constants which depend on fixed model fitted.
```

The analysis without correlation between random intercept and slope terms can be used to find initial values for an analysis with correlation. The correlation is imposed by specifying the two terms in `VSTRUCTURE` while also setting the `CORRELATE` option to `unrestricted`. The initial values are entered as gamma values, i.e. the variance components divided by the residual variance, using option `CINITIAL` to specify initial values for correlations across terms. Note that in this situation covariates are centred by default. This centring can be switched off using `VCOMPONENTS` option setting `CADJUST=none`, but in this case the initial values obtained from the first fit are likely to be less useful. Example 5.4.5b shows estimation of the correlation between the random intercept and slope terms for plants.

Example 5.4.5b

```
  71  " Fit random intercept and slope (with correlation) for plants,
 -72    using previous estimates as initial values."
  73  VCOMPONENTS [FIXED=Treatment*(Time+Timesqrd)] Plant+Plant.Time
  74  VSTRUCTURE  [TERMS=Plant+Plant.Time; FORMATION=whole; \
  75              CORRELATE=unrestricted; CINITIAL=!(3,0.1,0.1)]
  76  REML        [PRINT=#,deviance] Height

REML variance components analysis
=================================

Response variate:  Height
Fixed model:       Constant + Time + Treatment + Timesqrd + Time.Treatment +
Treatment.Timesqrd
Random model:      Plant + Plant.Time
Number of units:   70

Residual term has been added to model

Sparse algorithm with AI optimisation
All covariates centred


Covariance structures defined for random model
----------------------------------------------

Correlated terms:

Set  Correlation across terms
  1  Unstructured

Set  Terms                  Covariance model within term
  1  Plant                  Identity
  1  Plant.Time             Identity
```

```
Estimated parameters for covariance models
------------------------------------------

Random term(s)   Factor        Model(order)  Parameter        Estimate      s.e.
Plant + Plant.Time
                 Across terms  Unstructured  v_11               2.870     1.429
                                             v_21              0.5682    0.2749
                                             v_22              0.1066    0.0591
                 Within terms  Identity      -                     -         -

Note: the covariance matrix for each term is calculated as G or R where
      var(y) = Sigma2( ZGZ'+R ), i.e. relative to the residual variance, Sigma2.


Residual variance model
-----------------------

Term                           Model(order)  Parameter        Estimate      s.e.
Residual                       Identity      Sigma2             60.30     13.48


Estimated covariance models
---------------------------

Variance of data estimated in form:

V(y) = Sigma2( gZGZ' + I )

where: V(y) is variance matrix of data
       Sigma2 is the residual variance
       g is a gamma for the random term
       Z is the incidence matrix for the random term
       G is the covariance matrix for the random term
       I is the residual (identity) covariance matrix

Note: a gamma is the ratio of a variance component to the residual (Sigma2)


Correlated terms: Plant + Plant.Time

Across terms
Model: Unstructured

Covariance matrix:


1  2.870
2  0.568  0.107
       1      2


Within terms
Model: Identity (14 rows)



Residual term: added to model

Sigma2: 60.30

I is an identity matrix (70 rows)



Deviance: -2*Log-Likelihood
---------------------------

                   Deviance   d.f.
                     394.48    60

Note: deviance omits constants which depend on fixed model fitted.
```

```
Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Time                               63.44       1         63.44    51.2  <0.001
Treatment                           6.89       1          6.89    12.0   0.022
Timesqrd                           57.95       1         57.95    40.0  <0.001
Time.Treatment                      4.72       1          4.72    51.2   0.034
Treatment.Timesqrd                  5.41       1          5.41    40.0   0.025

Dropping individual terms from full fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
Time.Treatment                      9.77       1          9.77    51.2   0.003
Treatment.Timesqrd                  5.41       1          5.41    40.0   0.025

* MESSAGE: denominator degrees of freedom for approximate F-tests are calculated
using algebraic derivatives ignoring fixed/boundary/singular variance
parameters.
```

The model information lists sets of correlated terms. In the output, the parameter values for the unstructured matrix are ratios of the residual, i.e. they must be multiplied by σ^2 to get values comparable with the variance components in Example 5.4.5a.

More than two terms may be correlated; they must all be specified together and the number of initial values increased accordingly. For example, to include random variation in the quadratic component

```
VCOMPONENTS [FIX=Treatment*(Time+Timesqrd)] \
  Plant/(Time+Timesqrd)
VSTRUCTURE [TERMS=Plant/(Time+Timesqrd); CORR=unr; \
  FORM=whole; CINITIAL=!(2.9,0.3,0.11,0.01,0.01,0.01)]
```

Also, more than one set of correlated terms may be defined by repeated use of `VSTRUCTURE`. However, each set of correlated terms must be distinct. To remove correlations between terms, you should repeat the `VSTRUCTURE` statement with option setting `CORRELATE=no`.

5.4.6 Direct products

We now discuss the issues that are relevant when considering direct product construction of covariance models. Firstly, we explain the use of direct product construction in unbalanced data sets. Secondly, we discuss how a model term may be identified either as a random effect u_i with associated variance G_i, or as the residual e with associated variance R.

With unbalanced data there are several cases to consider, and it is easiest to do this via an example. Consider a set of repeated measurements, where data have been taken from subjects on five occasions. The following scenarios are possible:

i) all subjects had measurements taken on the same five dates;
ii) there were six (or more) dates on which measurements were taken, and each subject was measured on five of these dates;
iii) the sample dates for each subject were different.

(Note that it may be the *intervals* between samples rather than the sample dates that are recorded, but the same principles apply.)

In case (i), the structure of the experiment can be described in direct product terms, as discussed earlier in Section 5.4.1, using the term `Subject.Sample` where `Subject` and `Sample` are factors representing subjects and sample dates respectively.

In fact, whenever a random model term is defined in terms of an interaction of two factors with, say, l_1 and l_2 levels, this generates a set of effects of size $l_1 \times l_2$, which matches the size of the covariance matrix generated by direct product. If some combinations of the two factors are missing from the data, as in case (ii), the random effects for the missing combinations will not

be estimable. However, they remain in the model, so that the size of the random term is still compatible with the size of the matrix direct product allowing this construction still to be used. The parameters will then be estimated from covariances generated by the combinations that are present.

In case (iii), the `Sample` factor will have different levels for each subject, so most of the `Subject.Sample` combinations will be missing. In this case, a more efficient solution would usually be to provide (subject,time) coordinates for each sample and fit a two-dimensional power model over the whole term, with the subject parameter constrained to be zero to impose independence between subjects:

```
VSTRUCTURE [TERM=Subject.Week; FORMATION=whole; \
  COORD=subject,time] MODELTYPE=power; ORDER=2; \
  INITIAL=!(0,0.1); CONSTRAIN=!T(Fix,None)
```

The size of the correlation matrix generated here is equal to the number of effects in the `Subject.Week` term, which in this case is the number of data values. Note that the parameters run in the order of the coordinates vectors, which must be variates not factors. The option setting `FORMATION=whole` must be used because the values of `time` change within each level of the `Subject` factor, so direct product construction is not possible (see the description of the `COORDINATES` option in Section 5.4.3).

The rules for the allocation of a random model term to be either a random effect u_i with var(u_i)=G_i or to be the residual e with var(e)=R (see Section 5.4.1) are fairly straightforward. The form of the variance model is

$$V = \sigma^2 (\Sigma_j \gamma_j Z_j G_j Z_j' + R)$$

where matrix R corresponds to the residual term and must have n rows. To be used as the residual, a term must satisfy the following criteria:

1) the replication of each effect in the term is either one or zero;

and either

2a) there is no covariance model defined for the term, and it has n or more effects,

or

2b) there is a covariance model defined for the term, and it has n effects.

A term with no covariance model is valid even with more than n effects, since removal of the missing rows does not change the structure of the variance matrix $\sigma^2 I$. However, no term with more than n effects can be used as the residual if it has a non-identity covariance structure defined, since this matrix will also have more than n rows and would lose the structure that the algorithm expects if rows are deleted.

The first term in the random model that satisfies the criteria will be used as the residual. If no such term is found, a residual term with an independent error will automatically be added to the model. This may result in an extra independent error term being fitted unintentionally. An example of this occurs in case (ii) above: some of the `Subject.Sample` combinations are missing, so the size of the `Subject.Sample` term is larger than the number of data points and thus cannot be used as the residual. One work-around is to include missing values in the data set for the missing combinations to give equal replication, and then use `REML` option `MVINCLUDE=yvariate` to retain these missing values in the analysis. Alternatively, you could fix the unwanted extra residual component at a small value using the `INITIAL` and `CONSTRAIN` options of `VCOMPONENTS`. Note that you can ascertain whether one of the random model terms has been used as the residual or whether a residual has been added to the model by setting option `PRINT=model` in `REML`. Alternatively, you can use the `VRESIDUAL` directive to define a correlation structure on the residual term of an experiment. This works in the same way as `VSTRUCTURE`, but with the advantage that the algorithm then checks that the correct residual term is used. Details are given in Section 5.8.2, together with a description of how `VRESIDUAL` can be used to define separate residual terms for different experiments within a multi-experiment (or meta-) analysis.

5.5 Predictions from a REML analysis

This section describes the VPREDICT directive which can be used to form predictions of the values of the response variate at particular values of the variables in the fixed or random models, and the VTCOMPARISONS procedure which can calculate comparison contrasts of REML predictions.

5.5.1 The VPREDICT directive

VPREDICT directive

Forms predictions from a REML model.

Options

PRINT = *string tokens*	What to print (description, predictions, se, sed, avesed, vcovariance); default desc, pred, se, aves
CHANNEL = *scalar*	Channel number for output; default * i.e. current output channel
MODEL = *formula*	Indicates which model terms (fixed and/or random) are to be used in forming the predictions; default * includes all the fixed terms and relevant random terms
OMITTERMS = *formula*	Specifies terms to be excluded from the MODEL; default * i.e. none
FACTORIAL = *scalar*	Limit on the number of factors or variates in each term in the models specified by MODEL or OMITTERMS; default 3
PRESENTCOMBINATIONS = *identifiers*	Lists factors for which averages should be taken across combinations that are present
WEIGHTS = *tables*	One-way tables of weights classified by factors in the model; default *
PREDICTIONS = *table* or *scalar*	To save the predictions; default *
SE = *table* or *scalar*	To save standard errors of predictions; default *
SED = *symmetric matrix*	To save standard errors of differences between predictions; default *
VCOVARIANCE = *symmetric matrix*	To save variances and covariances of predictions; default *
SAVE = *REML save structure*	Specifies the save structure from which to predict; default * i.e. that from most recent REML

Parameters

CLASSIFY = *vectors*	Variates and/or factors to classify table of predictions
LEVELS = *variates*, *scalars* or *texts*	To specify values of variates and/or levels of factors for which predictions are calculated
PARALLEL = *identifiers*	Specifies variables in the classifying set whose values change in parallel (rather than in all combinations)
NEWFACTOR = *identifiers*	Identifiers for new factors that are defined when LEVELS are specified

The VPREDICT directive can be used to produce predictions of the values of the response variate at particular values of the variables in the fixed or random models. By default the predictions are from the most recent REML analysis, but you can use another analysis by supplying its save

structure using the SAVE option. However, VPREDICT is available only for analyses produced using the average-information method: i.e. the REML statement must have option METHOD=AI (see 5.3.1).

The CLASSIFY parameter specifies those variates or factors to be included in the table of predictions, and the LEVELS parameter supplies the values at which the predictions are to be made. For a factor, you can select some or all of the levels, while for a variate you can specify any set of values. A single level or value is represented by a scalar; several levels or values must be combined into a variate (which may of course be unnamed). Alternatively, if the factor has labels, you can use these to select the levels for prediction by setting LEVELS to a text. A missing value in the LEVELS parameter is taken to stand for all the levels of a factor, or the mean value of a variate.

The PARALLEL parameter allows you to indicate that a factor or variate should change in parallel with another factor or variate. both of these should have the same number of values specified for it by the LEVELS parameter of VPREDICT. The predictions are then formed for each set of corresponding values rather than for every combination of these values. For example, you could put

```
VPREDICT Treatment,Timesqrd,Time; PARALLEL=*,Time,*;\
         LEVELS=*,!(0,1,9,25,49,81),!(0,1,3,5,7,9)
```

to produce predictions at times 0, 1, 3, 5, 7 and 9 for the treatments in Example 5.4.5. The model contained both time and time squared, but you would want predictions only for matching values of time and time squared. So the PARALLEL parameter specifies that Timesqrd should change in parallel to Time.

When you specify LEVELS, VPREDICT needs to define a new factor to classify that dimension of the table. By default this will be an unnamed factor, but you can use the NEWFACTOR parameter to give it an identifier. The EXTRA attribute of the factor is set to the name of the corresponding factor or variate in the CLASSIFY list; this will then be used to label that dimension of the table of predictions.

The prediction calculations consist of two steps. The first step is to calculate a table of fitted values. The MODEL, OMITTERMS and FACTORIAL options specify the model to use for this. The formula specified by MODEL is expanded into a list of model terms, deleting any that contain more variates of factors than the limit specified by the FACTORIAL option. Then, any terms in the formula specified by OMITTERMS are removed. You can specify

```
OMITTERMS='Constant'
```

to omit the constant, e.g. if you want to obtain BLUPs for random terms.

The second step averages the fitted values over the classifications that are not in the list that was supplied by the CLASSIFY parameter. The WEIGHTS option can supply one-way tables classified by any of the factors in the model. These are used to calculate the weight to be used for each fitted value when calculating the averages. Equal weights are assumed for any factor for which no table of weights has been supplied. (for which no table of weights has been supplied. (Note, this differs from the default in PREDICT, which uses *marginal weights*; see 3.3.4.) In the averaging all the fitted values are generally used. However, if you define a list of factors using the PRESENTCOMBINATIONS option, any combination of levels of these factors that does not occur in the data will be omitted from the averaging. Where a prediction is found to be inestimable, i.e. not invariant to the model parameterization, a missing value is given.

Printed output is controlled by settings of the PRINT option with settings:

description	describes the terms and standardization policies used when forming the predictions,
predictions	prints the predictions,
se	produces predictions and standard errors,
sed	prints standard errors for differences between the predictions,

`avesed` prints the average standard error of difference of the predictions, and

`vcovariance` prints the variance and covariances of the predictions.

By default descriptions, predictions, standard errors and an average standard error of differences are printed. You can also save the results, using the `PREDICTIONS`, `SE`, `SED` and `VCOVARIANCE` options. You can send the output to another channel, or to a text structure, by setting the `CHANNEL` option.

Example 5.5.1 forms predictions for the split-plot in Section 5.3.1. Notice the `REML` statement in line 35, which reruns the analysis using the average-information method.

Example 5.5.1

```
  35  REML     [PRINT=*] Yield
  36  VPREDICT [PRINT=description,prediction,avesed] Nitrogen

Predictions from REML analysis
------------------------------

Model terms included for prediction: Constant + Nitrogen + Variety
+ Nitrogen.Variety
Model terms excluded for prediction: Blocks + Blocks.Wplots

Status of model variables in prediction:

  Variable           Type        Status
  Variety            factor      Averaged over - equal weights
  Nitrogen           factor      Classifies predictions
  Constant           factor      Included in prediction
  Blocks             factor      Ignored
  Wplots             factor      Ignored


Response variate: Yield of oats

Predictions

Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
             79.4     98.9    114.2    123.4

Approximate average standard error of difference: 4.436 (calculated on
variance scale)

  37  VPREDICT [PRINT=description,prediction,avesed] Variety

Predictions from REML analysis
------------------------------

Model terms included for prediction: Constant + Nitrogen + Variety
+ Nitrogen.Variety
Model terms excluded for prediction: Blocks + Blocks.Wplots

Status of model variables in prediction:

  Variable           Type        Status
  Variety            factor      Classifies predictions
  Nitrogen           factor      Averaged over - equal weights
  Constant           factor      Included in prediction
  Blocks             factor      Ignored
  Wplots             factor      Ignored


Response variate: Yield of oats

Predictions

Variety      Victory  Golden rain   Marvellous
                97.6        104.5        109.8
```

```
Approximate average standard error of difference: 7.079 (calculated on
variance scale)

  38  VPREDICT [PRINT=description,prediction,sed] Variety,Nitrogen

Predictions from REML analysis
------------------------------

Model terms included for prediction: Constant + Nitrogen + Variety
+ Nitrogen.Variety
Model terms excluded for prediction: Blocks + Blocks.Wplots

Status of model variables in prediction:

  Variable           Type        Status
  Variety            factor      Classifies predictions
  Nitrogen           factor      Classifies predictions
  Constant           factor      Included in prediction
  Blocks             factor      Ignored
  Wplots             factor      Ignored


Response variate: Yield of oats

Predictions

     Nitrogen    0 cwt  0.2 cwt  0.4 cwt  0.6 cwt
      Variety
      Victory     71.5     89.7    110.8    118.5
  Golden rain     80.0     98.5    114.7    124.8
   Marvellous     86.7    108.5    117.2    126.8

Standard error of differences


        Variety Victory Nitrogen 0 cwt   1       *
      Variety Victory Nitrogen 0.2 cwt   2   7.683       *
      Variety Victory Nitrogen 0.4 cwt   3   7.683   7.683       *
      Variety Victory Nitrogen 0.6 cwt   4   7.683   7.683   7.683       *
    Variety Golden rain Nitrogen 0 cwt   5   9.715   9.715   9.715   9.715
  Variety Golden rain Nitrogen 0.2 cwt   6   9.715   9.715   9.715   9.715
  Variety Golden rain Nitrogen 0.4 cwt   7   9.715   9.715   9.715   9.715
  Variety Golden rain Nitrogen 0.6 cwt   8   9.715   9.715   9.715   9.715
     Variety Marvellous Nitrogen 0 cwt   9   9.715   9.715   9.715   9.715
   Variety Marvellous Nitrogen 0.2 cwt  10   9.715   9.715   9.715   9.715
   Variety Marvellous Nitrogen 0.4 cwt  11   9.715   9.715   9.715   9.715
   Variety Marvellous Nitrogen 0.6 cwt  12   9.715   9.715   9.715   9.715
                                             1       2       3       4


    Variety Golden rain Nitrogen 0 cwt   5       *
  Variety Golden rain Nitrogen 0.2 cwt   6   7.683       *
  Variety Golden rain Nitrogen 0.4 cwt   7   7.683   7.683       *
  Variety Golden rain Nitrogen 0.6 cwt   8   7.683   7.683   7.683       *
     Variety Marvellous Nitrogen 0 cwt   9   9.715   9.715   9.715   9.715
   Variety Marvellous Nitrogen 0.2 cwt  10   9.715   9.715   9.715   9.715
   Variety Marvellous Nitrogen 0.4 cwt  11   9.715   9.715   9.715   9.715
   Variety Marvellous Nitrogen 0.6 cwt  12   9.715   9.715   9.715   9.715
                                             5       6       7       8


     Variety Marvellous Nitrogen 0 cwt   9       *
   Variety Marvellous Nitrogen 0.2 cwt  10   7.683       *
   Variety Marvellous Nitrogen 0.4 cwt  11   7.683   7.683       *
   Variety Marvellous Nitrogen 0.6 cwt  12   7.683   7.683   7.683       *
                                             9      10      11      12
```

5.5.2 The VTCOMPARISONS procedure

VTCOMPARISONS procedure

Calculates comparison contrasts within a multi-way table of predicted means from a REML analysis (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (contrasts, Waldtests); default cont
MODEL = *formula*	Indicates which model terms (fixed and/or random) are to be used in forming the predictions; default * includes all the fixed terms and relevant random terms
OMITTERMS = *formula*	Specifies terms to be excluded from the MODEL; default * i.e. none
FACTORIAL = *scalar*	Limit on the number of factors or variates in each term in the models specified by MODEL or OMITTERMS; default 3
PRESENTCOMBINATIONS = *identifiers*	Lists factors for which averages should be taken across combinations that are present
WEIGHTS = *tables*	One-way tables of weights classified by factors in the model; default *
GROUPS = *factors*	Groups for which to estimate each contrast
DFMETHOD = *string token*	Specifies which degrees of freedom to use for the comparisons (fddf, given, tryfddf, none); default fddf
DFGIVEN = *scalar*	Specifies the number of degrees of freedom to use for the comparisons when DFMETHOD=given, or if d.d.f. are unavailable when DFMETHOD=tryfddf
FMETHOD = *string token*	Controls how to calculate denominator degrees of freedom for the F-statistics, if these are not already available in the REML save structure (automatic, algebraic, numerical); default auto
SAVE = *identifier*	REML save structure for the analysis from which the comparisons are to be calculated

Parameters

CONTRAST = *tables*	Defines the comparisons to be estimated
ESTIMATES = *scalars* or *variates*	Saves the estimated contrasts
SE = *scalars* or *variates*	Saves standard errors of the contrasts
VCOVARIANCE = *symmetric matrices*	Save the variance-covariance matrices of contrasts estimated for GROUPS
STATISTIC = *scalars* or *variates*	Saves saves the test statistic (t or Wald)
DF = *scalars* or *variates*	Saves estimated numbers of residual degrees of freedom of the contrasts
PROBABILITY = *scalars* or *variates*	Saves the probabilities of the contrasts
WALD = *scalars*	Wald statistic for each comparison, combining the tests within groups
FSTATISTIC = *scalars*	F statistics for each comparison, if available, combining the tests within groups

NDF = *scalars*	Numerator d.f. for FSTATISTIC
DDF = *scalars*	Denominator d.f. for FSTATISTIC

VTCOMPARISON makes comparisons within multi-way tables of predicted means from a REML analysis. The data should previously have been analysed by the REML directive in the usual way. The SAVE option can be used to specify the save structure from the analysis for which the comparisons are to be calculated (see the SAVE option of REML). If SAVE is not specified, the comparisons are calculated from the most recent REML analysis.

The means are calculated using the VPREDICT directive (5.5.1), with options MODEL, OMITTERMS, FACTORIAL, PRESENTCOMBINATIONS and WEIGHTS all operating as in VPREDICT. Each comparison is specified in a table supplied by the CONTRAST parameter.

The GROUPS option is useful if you want to calculate the same comparisons for several groups, defined by the combimations of levels of one or more factors in the REML analysis. You can then use the CONTRAST parameter to define the comparison-definition tables ignoring the groups, and the GROUPS option to specify the factors defining the groups.

The DFMETHOD option specifies how to obtain the numbers of residual degrees of freedom for the comparisons. The default is to use the numbers of denominator degrees of freedom printed by REML in the d.d.f. column in the table of tests for fixed tests (produced by setting option PRINT=wald). These degrees of freedom are relevant for assessing the fixed term as a whole, and may differ over the various comparisons amongst its means, or for predictions produced with different models or weightings from those used in REML and VDISPLAY. So the t-probabilities should be used with caution. If you want a more exact probability for a comparison, you should set up a covariate to fit this explicitly in the analysis. The FMETHOD option controls how the denominator degrees of freedom should be calculated, if they are not already available in the REML save structure (e.g. because they were printed in the original analysis). The settings are the same as in the REML and VKEEP directives, except that there is no none setting. (You would set this option only if you really do want to calculate them.)

In some of the more complicated analyses, REML may be unable to calculate the denominator degrees of freedom. You might then want to supply the number of degrees of freedom yourself, using the DFGIVEN option, rather than having no probabilities at all. For example, you could use the number of denominator degrees of freedom from the analysis of an earlier similar design. However, the results will only be as good as the degrees of freedom that you have supplied, and thus should be used with caution! You can set option DFMETHOD=tryfddf to use the denominator degrees of freedom, if these can be calculated, or those specified by DFGIVEN otherwise. The setting DFMETHOD=given always uses the degrees of freedom specified by DFGIVEN.

If no d.d.f. are available, VTCOMPARISONS forms Wald statistics instead of t-statistics, and calculates their probabilities using the fact that, asymptotically, they have chi-square distributions with one degree of freedom. The Wald probabilities tend to be biased (giving too many significant results), and should thus be used with caution. You can set DFMETHOD=none to enforce the use of Wald statistics.

The PRINT option controls printed output, with settings:

contrasts	to print the contrasts (default).
Waldtests	when GROUPS is set this prints Wald tests combining the tests of each contrast in the various groups, F tests are also given provided REML has been able to estimate the d.d.f.

The ESTIMATE parameter allows you to save the estimates for the comparisons. If the GROUPS option is not set, each comparison will have a single estimate which will be saved in a scalar. Alternatively, if there are groups, there will be an estimate for each group, and these will be saved in a variate defined with unit labels that identify the groups. Similarly, the SE parameter can save the standard errors of the comparisons, the DF parameter can save their estimated

number of residual degrees for freedom, the STATISTIC parameter can save their test statistics (t or Wald), and the PROBABILITY parameter can save their probabilities.

When there are groups, the variances and covariances of the estimates for each contrast can be saved in a symmetric matrix, using the VCOVARIANCE parameter. The WALD, FSTATISTIC, NDF and DDF parameters can save the results of the tests combining the tests for each contrast in the various groups.

Example 5.5.2 estimates the comparison between the zero nitrogen level and the mean of the non-zero nitrogen levels in Examples 5.3.1 and 5.5.1.

Example 5.5.2

```
 39  TABLE     [CLASSIFICATION=Nitrogen; VALUES=-3,1,1,1] Ncomp
 40  CALCULATE Ncomp = Ncomp / 3
 41  VTCOMPARISONS Ncomp

Comparisons between REML means
------------------------------

Response variate:  Yield of oats

Contrast    estimate        s.e.           t        d.f.         pr.
   Ncomp      32.778       3.622        9.05       45.00      <0.001
```

5.6 Generating an inverse relationship matrix from a pedigree

5.6.1 The VPEDIGREE directive

VPEDIGREE directive

Generates an inverse relationship matrix for use when fitting animal or plant breeding models by REML.

Options

SEX = *string token*	Possible sex categories of parents (fixed, either); default fixe
UNKNOWN = *scalar*	Value to be treated as unknown

Parameters

INDIVIDUALS = *factors*	Individuals on which data has been measured
MALEPARENTS = *factors*	Male parents of the progeny
FEMALEPARENTS = *factors*	Female parents of the progeny
INVERSE = *pointer*	Inverse relationship matrix in sparse matrix form
POPULATION = *variates*	Full list of identifiers generated from the individuals and parents

VPEDIGREE is used to generate a sparse inverse relationship matrix for use when fitting animal (or plant) breeding models by REML. It takes as input sets of three factors, specified in parallel by the parameters INDIVIDUALS, MALEPARENTS and FEMALEPARENTS. The numerical levels of these factors must give identifiers for the individuals from which data are available (INDIVIDUALS) and the identifiers for the male and female parents of each individual (MALEPARENTS and FEMALEPARENTS), with missing values where the parent is unknown. The numerical codes for parents must be smaller than those for their progeny, and duplicate lines must not appear in the pedigree factors.

An individual may appear as both progeny and a parent (for example, when data have been taken from several generations). Conversely, if a code appears in more than one list, it is

assumed to refer to a single individual.

The algorithm does not take account of any factor labels. So, if labels are to be used, the labels vectors of the three factors should be identical in order to generate matching levels vectors and thus avoid errors. A complete list of all individuals in the three factors is compiled and can be saved using the POPULATION option and, on output, the three factors will be redefined with this list as their levels vector.

The inverse relationship matrix that is generated is held in a special sparse matrix form (that is, only non-zero values are stored), using a pointer. This is usable in the VSTRUCTURE directive but not, currently, elsewhere in Genstat. The second element of the pointer is a variate storing the non-zero values of the inverse matrix in lower-triangular order. The first element of the pointer is an integer index vector. This vector is not a standard Genstat data structure, and so cannot be used except by VSTRUCTURE.

By default, it is assumed that an individual can act as either a male or female parent but not as both. Option SEX=either can be used to specify that individuals can act as both male and female parents. This may be useful, for example, in plant breeding analyses.

Missing values in any of the factors will be regarded as representing unknown individuals. Option UNKNOWN allows you to specify an additional scalar value used to represent unknown individuals.

You might use VPEDIGREE, for example, to set up an inverse relationship matrix A^{-1}, and then use this matrix to model covariances within terms of an animal model. In cases where individuals appear several times in the data set, the pedigree must be constructed from a shorter list in which each individual appears only once. Given factors animal, dam and sire representing individuals, female and male parents respectively, a reduced list could be set up as follows:

```
DUPLICATE [ATTRIBUTE=levels] animal,dam,sire; \
  NEWSTRUCTURE=ran,rdam,rsire
TABULATE [CLASS=animal] !(#animal),!(#dam),!(#sire); \
  MEAN=tan,tdam,tsire
FACTOR [MODIFY=yes; VALUES=#tan] ran
&      [VALUES=#tdam] rdam
&      [VALUES=#tsire] rsire
```

The factors ran, rdam and rsire then hold the reduced lists (i.e. without duplication) for animals, dams and sires respectively. The relationship matrix can be constructed from these lists using VPEDIGREE:

```
VPEDIGREE INDIVIDUALS=ran; FEMALE=rdam; MALE=rsire; \
  INVERSE=Ainv; POPULATION=List
```

The variate List holds a combined list of parents and progeny. The length of this list matches the number of rows of the inverse relationship matrix Ainv, and this must also be the number of levels of the factors using Ainv in the analysis. It is therefore necessary to modify the levels vectors of the parent and progeny factors before proceeding with the analysis:

```
FACTOR [LEVELS=List] animal,dam,sire; \
  VALUES=animal,dam,sire
VCOMPONENTS [FIXED=Trt] RANDOM=animal+dam+env
VSTRUCTURE [animal+dam; CORRELATE=unr; FORMATION=whole] \
  MODELTYPE=fixed; INVERSE=Ainv
```

These declarations set up random terms with covariance structures of the form: cov(animal) = $\sigma_a^2 A$, cov(dam) = $\sigma_d^2 A$, cov(animal, dam) = $\sigma_{ad} A$.

5.6.2 The VFPEDIGREE procedure

VFPEDIGREE procedure

Prepares pedigree information to generate an inverse relationship matrix for use when fitting animal or plant breeding models by REML (S.A. Gezan & R.W. Payne).

Options

FREPRESENTATION = *string token*	Whether to match factor values by their levels or their labels (levels, labels); default leve
SEX = *string token*	Possible sex categories of parents (fixed, either); default fixe
UNKNOWN = *scalar* or *string*	Value to be treated as unknown in the pedigree factors
INVMETHOD = *string token*	How to represent the INVERSE (full, sparse); default spar

Parameters

INDIVIDUALS = *factors*	Individuals on which data have been measured
MALEPARENTS = *factors*	Male parents (or sires) of the progeny
FEMALEPARENTS = *factors*	Female parents (of dams) of the progeny
NEWINDIVIDUALS = *factors*	New individuals factor, with levels standardized for use in VPEDIGREE
NEWMALEPARENTS = *factors*	New males factor, with levels standardized to match those in the NEWINDIVIDUALS factor
NEWFEMALEPARENTS = *factors*	New females factor, with levels standardized to match those in the NEWINDIVIDUALS factor
OTHERFACTORS = *pointers*	Pointer containing additional factors, that may be used in the REML models, whose levels must also be standardized to match those in the NEWINDIVIDUALS factor
NEWOTHERFACTORS = *pointers*	Pointer containing new additional factors, with standardized levels
INVERSE = *pointer*	Inverse relationship matrix in sparse matrix form
POPULATION = *variates*	Full list of identifiers generated from the individuals and parents

The VPEDIGREE directive is rather stringent about its input parameters. It can use only levels to match the male and female factors with the individuals, those levels must be in ascending order, and the parents must be defined in the individuals factor before their offspring. So VFPEDIGREE has been provided to allow sets of pedigree factors to be checked and modified, so that they obey those constraints. It also allows you to match the factors to be matched by their labels, instead of their levels. Once it has standardized the factors, VFPEDIGREE calls VPEDIGREE to form the sparse inverse relationship matrix. If you are confident that your factors are already standardized, you can of course call VPEDIGREE direct (and then use VFPEDIGREE instead if that fails).

The factors defining the individuals, the male parents (or sires) and, optionally, the female parents (or dams) in the pedigree data set are specified by the INDIVIDUALS, MALEPARENTS and FEMALEPARENTS parameters, respectively. The OTHERFACTORS parameter can specify a pointer containing additional factors, involving the individuals in the pedigree, that may also be needed in the REML models. You can use the NEWINDIVIDUALS, NEWMALEPARENTS, NEWFEMALEPARENTS and NEWOTHERFACTORS parameters to save the new standardized factors. Otherwise, the original factors are redefined.

The FREPRESENTATION option indicates whether the factor values are to be matched by their

levels (the default) or their labels. If the INDIVIDUALS, MALEPARENTS and FEMALEPARENTS factors are being matched by levels, and the number corresponding to each level needs to be redefined, the factors will be given labels to help identify the original values. If INDIVIDUALS has labels, these will be used. Otherwise the labels will be textual forms of the original levels.

The POPULATION option can save the levels of the standardized factors when FREPRESENTATION=levels, or their labels when FREPRESENTATION=labels.

By default, it is assumed that an individual can act as either a male or female parent but not both. Option SEX=either can be used to specify that individuals can act as both male and female parents. This may be useful, for example, in plant breeding analyses.

Missing values in any of the factors will be treated as coding for unknown individuals. Option UNKNOWN allows you to specify an additional code to represent unknown individuals. This should be a scalar (e.g. 0 or −1) when FREPRESENTATION=levels, or a single-valued text (e.g. '*' or '0') when FREPRESENTATION=labels.

The inverse relationship matrix can be saved by the INVERSE parameter. By default, this is held in a special sparse matrix form (that is, only non-zero values are stored), using a pointer. This is usable in the VSTRUCTURE directive but not elsewhere in Genstat. The second element of the pointer is a variate storing the non-zero values of the inverse matrix in lower-triangular order. The first element of the pointer is an integer index vector. Alternatively, you can set option INVMETHOD=full to store the full matrix as a symmetric matrix (which can also be used by VSTRUCTURE). However, this is not recommended for large pedigrees.

5.7 Including cubic spline terms in the random model

The cubic smoothing spline can be formulated as a linear mixed model with the smoothing parameter as a variance ratio. This has been noted by many authors, but an accessible account is given in Verbyla (1995) and Verbyla *et al*. (1999). REML provides the estimation of smoothing parameters for cubic splines. This allows the inclusion of smoothing splines into models with random terms and/or correlated errors, and is useful for investigating nonlinearity in the data. In this formulation, the linear trend is estimated separately from the nonlinear trend, and so the linear trend must be specified separately in the model.

Terms for which cubic splines are to be generated are specified using the SPLINE option of VCOMPONENTS. Each term must contain one variate from which the cubic spline is to be calculated. The terms may be interactions of factors with variates, in which case a separate cubic spline is generated for each level of the combined factors. For example, consider the repeated measurements example in Section 5.4.3 and 5.4.5 with treatments Treatment and times of measurement indicated by variate Time. To investigate nonlinear patterns in treatments over time, we might start by fitting a linear random coefficient regression model, as in Section 5.4.5, but omitting the quadratic term so that we can look at all the nonlinear trend in the later parts of the example. (The quadratic model is still useful, though, to provide initial values for the spline model.)

Example 5.7a

```
 77  " Save random coefficient regression matrix for use as initial values "
 78  VKEEP       TERM=Plant; COVARIANCEMODEL=CovRCR
 79  PRINT       CovRCR['Across terms']

           CovRCR['Across terms']
          1       2.870
          2       0.568       0.107
                      1           2

 80  " Random cubic spline models:
-81    growth of 14 plants measured after 1,3,5,7,10 weeks.
-82    Baseline model without spline terms."
 83  VCOMPONENTS [FIXED=Treatment*Time] Plant+Plant.Time
```

```
  84  VSTRUCTURE  [TERMS=Plant+Plant.Time; FORMATION=whole; \
  85              CORRELATE=unrestricted; CINITIAL=CovRCR['Across terms']]
  86  REML        [PRINT=components,deviance] Height

Estimated parameters for covariance models
------------------------------------------

Random term(s)   Factor        Model(order)  Parameter        Estimate      s.e.
Plant + Plant.Time
                 Across terms  Unstructured  v_11               1.047     0.577
                                             v_21              0.2309    0.1111
                                             v_22             0.03114   0.02390
                 Within terms  Identity      -                      -         -

Note: the covariance matrix for each term is calculated as G or R where
      var(y) = Sigma2( ZGZ'+R ), i.e. relative to the residual variance, Sigma2.


Residual variance model
-----------------------

Term                           Model(order)  Parameter        Estimate      s.e.
Residual                       Identity      Sigma2              148.4      32.4


Deviance: -2*Log-Likelihood
---------------------------

                   Deviance   d.f.
                     426.73    62

Note: deviance omits constants which depend on fixed model fitted.
```

A random cubic spline term over time could then be introduced to model common deviations about the linear trend, as in Example 5.7b.

Example 5.7b

```
  87  " Include a random cubic spline term over time."
  88  VCOMPONENTS [FIXED=Treatment*Time; SPLINE=Time] Plant+Plant.Time
  89  VSTRUCTURE  [TERMS=Plant+Plant.Time; FORMATION=whole; \
  90              CORRELATE=unrestricted; CINITIAL=CovRCR['Across terms']]
  91  REML        [PRINT=model,components,deviance] Height

REML variance components analysis
=================================

Response variate:  Height
Fixed model:       Constant + Time + Treatment + Time.Treatment
Random model:      Plant + Plant.Time
Spline model:      Spline(Time)
Number of units:   70

Residual term has been added to model

Sparse algorithm with AI optimisation
All covariates centred


Covariance structures defined for random model
----------------------------------------------

Correlated terms:

Set  Correlation across terms
  1  Unstructured

Set  Terms                  Covariance model within term
  1  Plant                  Identity
  1  Plant.Time             Identity
```

```
Estimated variance components
-----------------------------

Random term                component        s.e.
Spline(Time)                   36.55       36.43


Estimated parameters for covariance models
------------------------------------------

Random term(s)   Factor         Model(order)  Parameter        Estimate      s.e.
Plant + Plant.Time
                Across terms   Unstructured  v_11               2.840     1.417
                                             v_21               0.5627    0.2726
                                             v_22               0.1053    0.0587
                Within terms   Identity      -                     -         -

Note: the covariance matrix for each term is calculated as G or R where
      var(y) = Sigma2( ZGZ'+R ), i.e. relative to the residual variance, Sigma2.


Residual variance model
-----------------------

Term                           Model(order)  Parameter        Estimate      s.e.
Residual                       Identity      Sigma2             60.89     13.70


Deviance: -2*Log-Likelihood
---------------------------

                   Deviance   d.f.
                     395.21    61

Note: deviance omits constants which depend on fixed model fitted.
```

The smoothing parameter is the residual variance divided by the variance component for the spline term. The change in deviance can be used to indicate whether the model has been improved by the addition of the cubic spline term. However, because the spline variance component is constrained to be greater than zero, the deviance must be compared to a statistic with the distribution $(\chi_0^2+\chi_1^2)/2$ rather than the usual χ_1^2 distribution. The change of 31.5 here indicates a much better fit with the spline term.

You can save details of splines that have been fitted for each term using the new parameters SPLBLUP, SPLDESIGN, SPLX and SPLSMOOTH of VKEEP. The information is saved in pointers with an element for each combination of the levels of the factors in the term (i.e. for each spline that has been fitted). The pointers elements are variates for SPLBLUP (best linear unbiased predictors) and SPLX (knot points), matrices for SPLDESIGN (design matrices), and scalars for SPLSMOOTH (smoothing parameters). This is illustrated in Example 5.7c, where the SPLBLUP, SPLDESIGN and SPLX parameters are used to calculate and plot the spline.

Example 5.7c

```
" Plot the spline term."
VKEEP       Time; SPLBLUP=Tblup; SPLDESIGN=Tdes; SPLX=Tknot
CALCULATE   Tspline = Tdes[1] *+ Tblup[1]
YAXIS       1; LOWER=-50; UPPER=50
PEN         1,2; METHOD=line
DGRAPH      [TITLE='Common spline effect over time'] Tspline; Tknot[1]
```

The graph in Figure 5.7a shows the predicted deviation about the linear trend over time. The scale used is approximately the range of the data, to give a more accurate impression of the

impact of the spline term on the fitted profiles.

To investigate whether each treatment group has the same nonlinear pattern, a spline term `Treatment.Time` can be introduced, as in Example 5.7d. For this term, a cubic spline is calculated and fitted separately for each factor level. However, the two splines are fitted using a common variance component, i.e. a common smoothing parameter.

Example 5.7d

```
  98  " Fit separate splines for each treatment."
  99  VCOMPONENTS [FIXED=Treatment*Time; SPLINE=Treatment.Time] \
 100              Plant+Plant.Time
 101  VSTRUCTURE  [TERMS=Plant+Plant.Time; FORMATION=whole; \
 102              CORRELATE=unrestricted; CINITIAL=CovRCR['Across terms']]
 103  REML        [PRINT=components,deviance] Height

Estimated variance components
-----------------------------

Random term               component        s.e.
Spline(Time).Treatment        74.47       55.78


Estimated parameters for covariance models
------------------------------------------

Random term(s)   Factor        Model(order)  Parameter        Estimate      s.e.
Plant + Plant.Time
                Across terms  Unstructured  v_11               3.917     1.943
                                            v_21               0.7620    0.3733
                                            v_22               0.1499    0.0804
                Within terms  Identity      -                      -         -

Note: the covariance matrix for each term is calculated as G or R where
      var(y) = Sigma2( ZGZ'+R ), i.e. relative to the residual variance, Sigma2.


Residual variance model
-----------------------

Term                          Model(order)  Parameter        Estimate      s.e.
Residual                      Identity      Sigma2             44.96     10.64


Deviance: -2*Log-Likelihood
---------------------------

                  Deviance   d.f.
                    389.86    61

Note: deviance omits constants which depend on fixed model fitted.

 104  " Plot the splines."
 105  VKEEP       [RMETHOD=all; RESIDUALS=R1]
 106  VKEEP       [RMETHOD=notspline; RESIDUALS=R2]
 107  CALCULATE   TTspline = R1 - R2
 108  DGRAPH      [TITLE='Separate treatment splines'] TTspline; Time; \
 109              PEN=Treatment
```

The predicted profiles for treatments 1 and 2 are shown in Figure 5.7b. (This time we obtain the splines by the slightly simpler process of saving residuals including and excluding the spline term and then calculating the difference.) The change in deviance suggests that differences exist in nonlinear trend between the two groups, although the predicted trend suggests that the quadratic models used earlier provided a reasonable approximation.

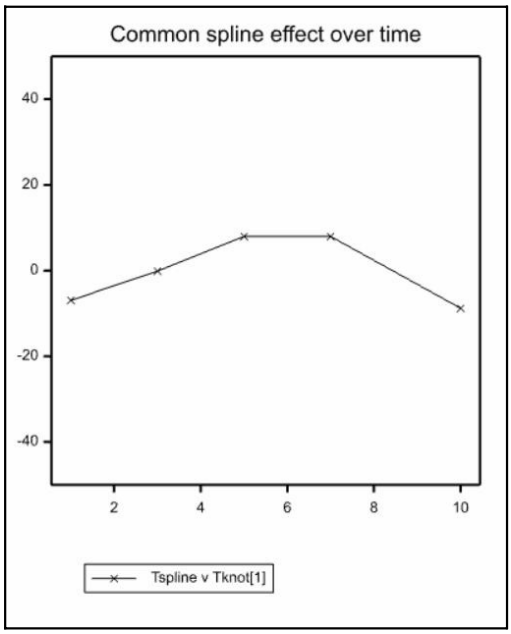


Figure 5.7a

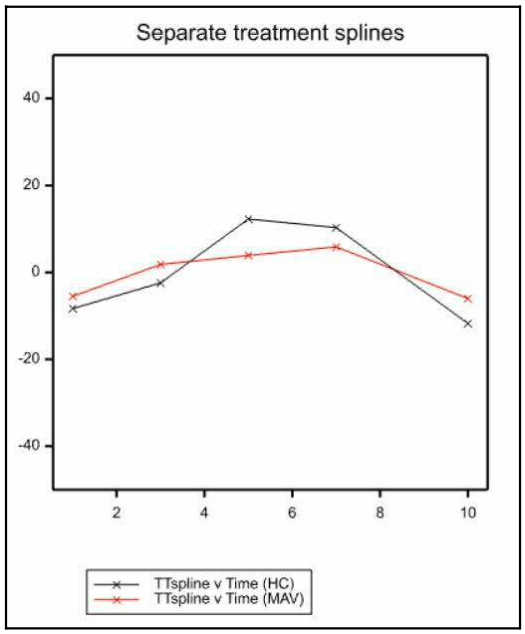


Figure 5.7b

5.8 Combined analyses of several experiments

The ability of the REML directive to handle unbalanced data sets makes it very suitable for meta-analysis. The analysis combines the original data from several experiments into a joint analysis in which all the available information is used to provide efficient estimates of the effects of interest. The REML meta-analysis differs from the meta-analyses that are often used, for example, in medical research, where the original data may not be available. So the basic data here are the effects estimated from the analyses of the individual trials. This type of meta-analysis can be done using the META procedure for a single treatment contrast. or by the VMETA procedure for several treatments; see *Genstat Reference Manual, Part 3 Procedures* for details.

There are three main issues to consider in a REML meta-analysis. Firstly the residual variance is likely to be different in each experiment. If the residual is represented by the same term in every experiment, you just need to set the EXPERIMENTS option of VCOMPONENTS (5.2.1) to a factor identifying the experiment to which each unit belongs. REML then fits a separate version of the residual term for each level of the factor, so that a different residual variance is estimated for each experiment. If you need to define a different term to act as the residual in some of the experiments, you can use the VRESIDUAL directive (5.8.2).

The second issue is that you may wish to include different terms in the random models fitted in some of the experiments. Suppose, for example, you want to include a random term for blocks in the first, but none of the other, experiments. To do this you need to generate a factor with the block levels on the first experiment, and missing values elsewhere. Then include the factor in the random model, and set option MVINCLUDE=explanatory in the REML command. The option changes the usual REML rule that any unit with a missing value in an explanatory variable is omitted from the analysis. Instead they are still included, and the missing factor values are ignored in the calculation of the model. You can also use this technique if you want to include terms for blocks in several experiments, but each with its own variance component – you just need to generate a different block factor for each one. You can define the necessary factors (or variates) using the ordinary manipulation commands, like CALCULATE (4.1.1), but procedure VRMETAMODEL (5.8.1) may be more convenient as it allows you to define the random model at the same time.

The third issue is that you may wish to specify a different residual variance model for each

experiment. This can also be done using the VRESIDUAL directive (5.8.2).

5.8.1 The VRMETAMODEL procedure

VRMETAMODEL procedure

Forms the random model for a REML meta analysis (R.W. Payne).

Options

RANDOM = *formula structure*	Saves the random model
EXPERIMENTSFACTOR = *factor*	Factor defining which units are in each experiment
TERMS = *formula*	Specifies terms, if any, to be fitted over the whole data set; default * i.e. none

Parameters

EXPERIMENT = *scalars*, *variates* or *texts*	Experiments on which additional random terms are to be fitted
LOCALTERMS = *formula structures*	Random terms that are to be fitted only on the corresponding experiment
SAVEVECTORS = *pointers*	Saves the factors (and/or any variates) defined to represent the local terms on each experiment

In REML meta analyses the designs used in the various experiments need not be identical and, even if they are all the same, the same random model may not be appropriate for every one. REML does allow you to fit different random terms in the different experiments, but their definition can be tedious. For example, if you wanted to include the term Blocks only in experiments 1 and 2 (and with a different variance component in each case), you would need to take two copies of the factor, giving them names (e.g. Blocks1 and Blocks2) that will be recognisable in the output. Then, set Blocks1 to missing except within experiment 1, and Blocks2 to missing except in experiment 2. If you now add Blocks1 + Blocks2 to the overall random model, and set option MVINCLUDE=explanatory in the REML statement, the terms Blocks1 and Blocks2 will each be fitted only in the desired experiment (1 or 2, respectively), and ignored elsewhere.

The process of forming the modified copies of the factors and devising names to label them clearly on the output can be inconvenient. So procedure VRMETAMODEL has been provided to make this clearer and more straightforward. In the output a term like Reps.Blocks, that is to be fitted only e.g. at Rothamsted, will be labelled

```
Reps@Rothamsted.Blocks@Rothamsted
```

The random model is formed automatically, and can be saved in a formula structure by the RANDOM option. The EXPERIMENTSFACTOR option must specify a factor to indicate which units of the data set belong to each experiment, and the TERMS option can specify random terms that are to be fitted over the whole data set.

The EXPERIMENT parameter lists the experiments where additional random terms are to be fitted, using either the levels or the labels of EXPERIMENTSFACTOR. You can specify a variate or a text with several values, if the terms are to be fitted with the same variance components in more than one experiment.

The LOCALTERMS parameter specifies a formula structure for each experiment to define its additional terms. The factors (and any variates) in the additional terms for each experiment are copied, the required missing values are inserted, and the terms are added to the random model.

By default, the modified copies of the factors and variates that are formed to represent the

additional random terms will be unnamed, and exist only as part of the RANDOM model. (The labels that appear in the output are attached to the factors by setting the EXTRA parameter in the FACTOR statement or VARIATE statement that defined them inside VRMETAMODEL.) The SAVEVECTORS parameter allows you to supply a pointer for each experiment, to save its factors (and any variates), so that you use them to refer to the additional random terms e.g. in the VKEEP directive (5.9.1). The elements of each pointer are labelled by the identifiers of the factors or variates in the corresponding local terms to simplify their subsequent use.

Example 5.8.1 analyses three fungicide trials that took place in different years at the same site. The data are in spreadsheet file MetaFungicide.gsh in the Genstat Data folder (line 2). There were two cultivars, one susceptible and one resistant, and ten different fungicide treatments. A split-plot design was used in each year, but the cultivars were applied to the whole-plots in 1997, and the fungicides were applied to the whole-plots in 1998 and 1999. So, we have the same treatments, but different designs in the different years, even though the blocking structures were identical. Analyses of the individual experiments, shown in Chapter 2 of the *Guide to REML in Genstat*, show that for each of experiments 1 and 2 (1997 and 1998) we need a random term for blocks, while for experiment 3 (1999) we need a random term for the combinations of whole-plots and blocks. So we set these as LOCALTERMS in line 5.

We also need to consider how to handle experiment effects and interactions between experiments and the treatment terms. If we include these as in the fixed model, the treatment terms will be tested using the within-experiment error, weighted according to precision within each experiment. Alternatively, if we include them in the random model, each treatment term will in effect be compared with its interaction with experiment (unless this is zero). In that case, a significant treatment effect would imply that the effect is consistent and large compared to its variation across experiments – thus giving a more stringent test. So in line 3, we set the TERMS option to

```
year + year.(fungicide*cultivar)
```

Example 5.8.1

```
   2  SPLOAD      [PRINT=*] '%gendir%/data/MetaFungicide.gsh'
   3  VRMETAMODEL [TERMS=year + year.(fungicide*cultivar);\
   4              EXPERIMENTSFACTOR=year; RANDOM=random] 1997,1998,1999;\
   5              LOCALTERMS=!f(block),!f(block),!f(block.wholeplot)
   6  VCOMPONENTS [FIXED=fungicide*cultivar; EXPERIMENTS=year] #random
   7  REML        [MVINCLUDE=explanatory] yield

REML variance components analysis
=================================

Response variate:  yield
Fixed model:       Constant + fungicide + cultivar + fungicide.cultivar
Random model:      year + year.fungicide + year.cultivar + year.fungicide.
cultivar + block@1997 + block@1998 + block@1999.wholeplot@1999
Number of units:   180

Separate residual terms for each level of experiment factor: year

Sparse algorithm with AI optimisation
Units with missing factor/covariate values included
   - specific effect for term(s) omitted for units with missing values in
block@1997, block@1998, block@1999, wholeplot@1999


Estimated variance components
-----------------------------

Random term                component        s.e.
year                          0.5984      0.6107
year.fungicide                0.0223      0.0147
year.cultivar                 0.0094      0.0127
year.fungicide.cultivar       0.0127      0.0114
```

```
block@1997                  0.0126      0.0150
block@1998                  0.0141      0.0166
block@1999.wholeplot@1999   0.0087      0.0271


Residual model for each experiment
----------------------------------

Experiment factor: year

Experiment Term  Factor        Model(order)  Parameter        Estimate      s.e.
1997.      Residual            Identity      Variance           0.0472    0.0112
1998.      Residual            Identity      Variance           0.0494    0.0110
1999.      Residual            Identity      Variance            0.118     0.034


Tests for fixed effects
-----------------------

Sequentially adding terms to fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
fungicide                          92.10       9         10.23    17.2  <0.001
cultivar                           41.92       1         41.92     2.1   0.021
fungicide.cultivar                 22.61       9          2.51    16.5   0.050

Dropping individual terms from full fixed model

Fixed term                Wald statistic  n.d.f.   F statistic  d.d.f.    F pr
fungicide.cultivar                 22.61       9          2.51    16.5   0.050

* MESSAGE: denominator degrees of freedom for approximate F-tests are calculated
using algebraic derivatives ignoring fixed/boundary/singular variance
parameters.
```

5.8.2 The VRESIDUAL directive

VRESIDUAL directive

Defines the residual term for a REML analysis, or the residual term for an experiment within a meta-analysis (combined analysis of several experiments).

Options

EXPERIMENT = *scalar*	Level of the EXPERIMENTS factor for which the residual is being defined
TERM = *formula*	Model term to be used as the residual
FORMATION = *string token*	Whether the structure is formed by direct product construction or by definition of the whole matrix (direct, whole); default dire
VARIANCE = *scalar*	Allows an initial estimate to be provided for the residual variance of the experiment
CONSTRAINT = *string token*	Allows the residual variance to be fixed at its initial value (fix, positive) default posi
COORDINATES = *matrix* or *variates*	Coordinates of the data points to be used in calculating distance-based models

Parameters

MODELTYPE = *string tokens*	Type of covariance model associated with the term(s), or with individual factors in the term(s) if FORMATION=direct (identity, fixed, AR, MA, ARMA, power, boundedlinear, circular,

	spherical, linearvariance, banded, correlation, antedependence, unstructured, diagonal, uniform, FA, FAequal) default iden
ORDER = *scalar*	Order of model
HETEROGENEITY = *string token*	Heterogeneity for correlation matrices (none, outside); default none
METRIC = *string token*	How to calculate distances when MODELTYPE=power (cityblock, squared, euclidean); default city
FACTOR = *factors*	Factors over which to form direct products
MATRIX = *identifiers*	To define matrix values for the term or the factors when MODELTYPE=fixed
INVERSE = *identifiers*	To define values for matrix inverses (instead of the fixed matrices themselves) when MODELTYPE=fixed
INITIAL = *identifiers*	Initial parameter values for each correlation matrix
CONSTRAINTS = *texts*	Texts containing strings none, fix or positive to define constraints for the parameters in each model
EQUALITYCONSTRAINTS = *variates*	Non-zero values in the variate indicate groups of parameters whose values are to be constrained to be equal

VRESIDUAL is used to define the residual term for a REML analysis or to define separate residual terms for different experiments within a multi-experiment (or meta-) analysis. The TERM option is used to specify the formula for the residual term. This term need not have been specified previously by the VCOMPONENTS statement.

For a single experiment, VRESIDUAL can be used to impose a covariance structure on the residual term. This could also be done by specifying the covariance structure using VSTRUCTURE (5.4.1), but VRESIDUAL has the advantage that the algorithm then checks that the term is consistent with the structure of the data.

In a multi-site experiment, VRESIDUAL can be used to specify a different residual model for each separate experiment. The EXPERIMENT option is used to specify the experiment(s) for which the model is to be used. The settings identify levels of a factor, defining the experiments, which is specified by the EXPERIMENTS option of VCOMPONENTS.

The VARIANCE option is used to give an initial value for the residual variance in the current experiment(s). You can set option CONSTRAINT=fix to fix the residual variance at the initial value rather than estimating it (as a positive value).

The definition of the residual terms then follows mainly as for the definition of correlated error terms through VSTRUCTURE The exception is that power models can be defined only in terms of the coordinates of the data points, not by specifying coordinates for the factor levels. (So the DISTANCES and COORDINATES parameters of VSTRUCTURE are not present in VRESIDUAL.)

For a multi-experiment analysis, the factors and variates for the separate experiments should be concatenated into structures which run over all the experiments. For example, consider an experiment set up at two sites to compare a set of 24 varieties in four replicates. In one site the experiment was laid out as a grid of eight rows by 12 columns, in the other a grid of 16 rows by six columns was used. In these circumstances, a single set of factors (of length 192) can be used to specify the design, using factors to describe variety, rows and columns, plus a factor expt defining the allocation of units to experiments. Note that the factor row will have 16 levels and col will have 12 levels, but REML will determine internally that site 1 has only 8 rows and site 2 only 6 columns.

```
VCOMPONENTS [FIXED=Variety; EXPERIMENTS=Expt]
```

```
VRESIDUAL [EXPERIMENT=1; TERM=Row.Col] MODELTYPE=AR,AR; \
  ORDER=1,1; FACTOR=Row,Col
VRESIDUAL [EXPERIMENT=2; TERM=Row.Col] MODELTYPE=AR,AR; \
  ORDER=1,1; FACTOR=Row,Col
```

Where some factors differ between experiments, these should be defined on the units relevant to the appropriate experiment(s) and missing elsewhere. When an EXPERIMENTS factor has been defined, the default action of the MVINCLUDE option of REML is changed to include units with missing y-values and missing factor levels.

5.9 Saving information from a REML analysis

5.9.1 The VKEEP directive

VKEEP directive

Copies information from a REML analysis into Genstat data structures.

Options

RESIDUALS = *variate*	Residuals from the analysis
FITTEDVALUES = *variate*	Fitted values from the analysis
SIGMA2 = *scalar*	Variance component for the lowest stratum
VCOVARIANCE = *symmetric matrix*	Variance-covariance matrix for the estimates of the variance components
VESTIMATES = *variate*	Saves a vector of all parameters in the variance model
VARESTIMATES = *symmetric matrix*	Variance-covariance matrix for the parameters in the variance model (as saved by VESTIMATES)
VLABELS = *text*	Vector of text labels for the VESTIMATES and VARESTIMATES structures
MVESTIMATES = *variate*	Estimates of missing values
MVSE = *variate*	Standard errors of missing-value estimates
MVUNITS = *variate*	Unit numbers of missing values
ALLEFFECTS = *variate*	Full set of estimated fixed and random effects
ALLVCOVARIANCE = *symmetric matrix*	Variance-covariance matrix for the full set of fixed and random effects not associated with the absorbing factor
DEVIANCE = *scalar*	Residual deviance from fitting the full fixed model
DF = *scalar*	Residual degrees of freedom after fitting the full fixed model
SUBDEVIANCE = *scalar*	Residual deviance after fitting the submodel of the fixed model
SUBDF = *scalar*	Residual degrees of freedom after fitting the submodel of the fixed model
RSS = *scalar*	Residual sum of squares from fitting the FIXED model by general least squares with a covariance matrix derived from the estimated variance components
INDEX = *variate*	Index of units included in the analysis
MODELS = *pointer*	Pointer to formulae giving the fixed, random, spline and residual terms fitted
RMATRIX = *pointer*	Saves details of the covariance model fitted to the residual
RMETHOD = *string token*	Which random terms to use when calculating

	RESIDUALS (final, all, notspline); default uses the setting from the REML statement
CFORMAT = *string token*	Whether the covariance matrices or the parameters are saved for a COVARIANCEMODEL (variancematrices, parameters); default vari
UVCOVARIANCE = *symmetric matrix*	Unit-by-unit variance-covariance matrix
DFFIXED = *scalar*	Number of degrees of freedom in the fixed model
DFRANDOM = *scalar*	Number of degrees of freedom in the random model
FMETHOD = *string token*	Controls how to calculate F-statistics for fixed terms (automatic, none, algebraic, numerical); default auto
WMETHOD = *string token*	Controls which Wald statistics are saved (add, drop); default drop
WORKSPACE = *scalar*	Saves the workspace setting that was used by the REML command
YVARIATE = *dummy*	Dummy to be set to the y-variate of the analysis
EXIT = *scalar*	Exit status of the fit (0 if successful)
SAVE = *REML save structure*	Save structure from the required analysis; default * takes the save structure from the latest REML statement

Parameters

TERMS = *formula*	Terms for which information is to be saved
COMPONENTS = *scalars*	Estimated variance components
COVARIANCEMODEL = *pointers*	Saves details of the covariance model fitted to a random term
MEANS = *tables*	Table of predicted means for each term
SEDMEANS = *symmetric matrices*	Standard errors of differences between the predicted means
VARMEANS = *symmetric matrices*	Variance-covariance matrix of the means
EFFECTS = *tables*	Table of estimated regression coefficients for each term
SEDEFFECTS = *symmetric matrices*	Standard errors of differences between the estimated parameters of each term
VAREFFECTS = *symmetric matrices*	Variance-covariance matrix of the effects of a term
DESIGNMATRIX = *matrices*	Saves the design matrix for the term
SPLBLUP = *pointers*	Best linear unbiased predictors for spline terms, saved in a pointer with a variate for each combination of the levels of the factors in the term
SPLDESIGN = *pointers*	Design matrices (Z) for spline terms, saved in a pointer with a matrix for each combination of the levels of the factors in the term
SPLX = *pointers*	Knot points for spline terms, saved in a pointer with a variate for each combination of the levels of the factors in the term
SPLSMOOTH = *pointers*	Smoothing parameters estimated for spline terms, saved in a pointer with a scalar for each combination of the levels of the factors in the term
CADJUSTMENT = *scalars*	For a term involving covariates, saves the adjustment made to its values during the analysis
WALD = *scalar*	Wald statistic (fixed terms only)
FSTATISTIC = *scalars*	F statistics (fixed terms only)

NDF = *scalar*	Numerator d.f. (fixed terms only)
DDF = *scalar*	Denominator d.f. (fixed terms only)

You can use the VKEEP directive to copy results from a REML analysis into Genstat data structures. Genstat automatically stores the save structure for the last y-variate that was analysed using REML, and by default this save structure provides the information for VKEEP. As for VDISPLAY, you can save the information from a REML analysis in a save structure using the SAVE parameter in the REML directive, then access the information by specifying the same structure in the SAVE option of VKEEP.

Overall information from the analysis is saved using the options of VKEEP, while the parameters are used to save information for specific model terms. The terms (fixed, random or a mixture) for which you require information are defined by a formula using the TERMS parameter. The other parameters can then be used to specify structures for saving information for each of the model terms.

Options RESIDUALS and FITTEDVALUES are used to specify variates to hold the residuals and fitted values, which are defined according to the setting of the RMETHOD option, as for the REML directive. The residual variance can be stored in a scalar using option SIGMA2. So, for example, after a REML analysis, to save the residuals and fitted values into variates called Res and Fit respectively, you can use the command

```
VKEEP [RESIDUALS=Res; FITTED=Fit]
```

The variance-covariance matrix for the estimates of the variance component can be saved using the VCOVARIANCE option. (The estimates themselves are saved using the COMPONENTS parameter, as described below.)

The VESTIMATES option is used to save a variate containing all the variance parameters estimated in the model. The VARESTIMATES option can supply a symmetric matrix to save the variance-covariance matrix for the estimates of the variance parameters, matching the ordering and contents of VESTIMATES. The vector of labels for these parameters can be saved the VLABELS option. The ALLEFFECTS option allows you to save the full set of fixed and random effects, excluding those in the absorbing factor model, and the ALLVCOVARIANCE option can be used to store their variance-covariance matrix. This matrix will often be very large, and is useful only for looking at covariances between effects associated with different model terms, since the variance-covariance matrices for individual model terms can be stored using the VAREFFECTS parameter. The unit-by-unit variance-covariance matrix can be saved using the UVCOVARIANCE option (and this may be even larger). This uses the random and residual terms, but not spline terms. It cannot be formed if the model contains sparse inverse covariance matrices, for example from VPEDIGREE (5.6.1).

The MVESTIMATES option can save a variate containing estimates of the missing values, the MVSE option saves their standard errors, and the MVUNITS option saves a list of the units that are missing.

The residual deviance from fitting the full fixed model or the submodel can be saved using options DEVIANCE and SUBDEVIANCE respectively, and the associated residual degrees of freedom can be saved using options DF and SUBDF. The degrees of freedom fitted by the (full) fixed model can be saved by the DFFIXED option, and the degrees of freedom in the random model can be saved by the DFRANDOM option. The RSS option can save the residual sum of squares from fitting the fixed model by generalized least squares.

The INDEX option saves an index of the units that were included in the analysis. (This will depend on the patterns of missing values, if any, and the setting of the MVINCLUDE option of REML.) The MODELS option can be used to save a pointer, with labels 'Fixed', 'Spline', 'Random' and 'Residual', containing formulae for the model terms fitted as fixed, spline, random or residual terms. The labels can be specified in either lower or upper case, or any mixture. The YVARIATE option can be set to a dummy to point to the variate that was analysed

(i.e. the variate defined by the Y parameter of REML).

The formula given in the TERMS parameter is expanded to give a series of model terms. The other parameters of VKEEP are taken in parallel with these terms. The string 'Constant' can be used within the formula to save structures associated with the constant term. Example 5.9.1a shows how to save information from the split-plot analysis in Section 5.3.1.

Example 5.9.1a

```
  42  VKEEP TERMS=Variety; MEANS=MV; SEDMEANS=SedV; VARMEANS=VarV
  43  & [SIGMA2=Sigma2] Blocks/Wplots; COMPONENTS=Cb,Cwp
  44  PRINT MV

                     MV
     Variety
     Victory       97.6
  Golden rain     104.5
   Marvellous     109.8

  45  PRINT [RLPRINT=integers,labels; CLPRINT=integers; RLWIDTH=20] SedV

                               SedV
 Variety     Victory   1          *
 Variety Golden rain   2      7.079          *
 Variety  Marvellous   3      7.079      7.079          *
                              1          2          3

  46  PRINT [RLPRINT=integers,labels; CLPRINT=integers] VarV

                              VarV
Variety     Victory   1      60.80
Variety Golden rain   2      35.75      60.80
Variety  Marvellous   3      35.75      35.75      60.80
                             1          2          3

  47  PRINT Cb,Cwp,Sigma2

         Cb         Cwp      Sigma2
      214.5       106.1       177.1
```

The COMPONENTS parameter allows you to save the estimated variance component for each random term in the TERMS list.

Tables of means for each term can be saved using the MEANS parameter, and standard errors of differences between the means are saved by SEDMEANS. You can also save the estimated variance-covariance matrix for the means of each term using parameter VARMEANS.

The EFFECTS parameter is used to save tables of estimated parameters. A symmetric matrix of the standard errors of differences between the effects of each term can be saved using parameter SEDEFFECTS, and the estimated variance-covariance matrix for the parameters can be saved using parameter VAREFFECTS. The DESIGNMATRIX parameter saves the design matrix used to fit the effects of each term.

You can save details of splines that have been fitted for each term using the SPLBLUP, SPLDESIGN, SPLX and SPLSMOOTH parameters. The information is saved in pointers with an element for each combination of the levels of the factors in the term (i.e. for each spline that has been fitted). The pointers elements are variates for SPLBLUP (best linear unbiased predictors) and SPLX (knot points), matrices for SPLDESIGN (design matrices), and scalars for SPLSMOOTH (smoothing parameters).

If the term involves a covariate, the CADJUSTMENT parameter can save the adjustment that will have been made to its values during the analysis. This will be zero if option CADJUST was set to none when the fixed and random models were defined by VCOMPONENTS. Alternatively, if CADJUST had its default setting of mean, each covariate will have been centred by subtracting its (weighted) mean.

Details of the covariance model fitted to each random term can be saved using the COVARIANCEMODEL parameter. The information is saved in a pointer. The contents of the pointer depend upon the complexity of the covariance model fitted and the setting of the CFORMAT parameter. First we consider the default setting: CFORMAT=variancematrices. If no covariance model has been fitted, the pointer will have two elements for the scalar (variance component) and the covariance matrix (identity – a diagonal matrix with number of rows equal to the number of levels of the term). If a covariance model has been fitted, the component matrices used to construct the model will be saved. The full covariance matrix can then be generated by taking a direct product of the component matrices and multiplying by the scalar. Alternatively, if CFORMAT=parameters, the pointer contains the component parameters of the model. The RMATRIX option provides an alternative way of saving the covariance model fitted to the residual term.

Example 5.9.1b saves details of the covariance models fitted in Example 5.4.4d.

Example 5.9.1b

```
  52  VKEEP Row.Column; COVARIANCEMODEL=Variancematrices
  53  PRINT #Variancematrices

Variancematrices['Scalar']       4.580

 Variancematrices['Row']

        1      1.0000
        2      0.6827      1.0000
        3      0.4661      0.6827      1.0000
        4      0.3182      0.4661      0.6827      1.0000
        5      0.2172      0.3182      0.4661      0.6827      1.0000
        6      0.1483      0.2172      0.3182      0.4661      0.6827
        7      0.1012      0.1483      0.2172      0.3182      0.4661
        8      0.0691      0.1012      0.1483      0.2172      0.3182
        9      0.0472      0.0691      0.1012      0.1483      0.2172
       10      0.0322      0.0472      0.0691      0.1012      0.1483
                    1           2           3           4           5

        6      1.0000
        7      0.6827      1.0000
        8      0.4661      0.6827      1.0000
        9      0.3182      0.4661      0.6827      1.0000
       10      0.2172      0.3182      0.4661      0.6827      1.0000
                    6           7           8           9          10

 Variancematrices['Column']

        1      1.0000
        2      0.8438      1.0000
        3      0.7120      0.8438      1.0000
        4      0.6008      0.7120      0.8438      1.0000
        5      0.5069      0.6008      0.7120      0.8438      1.0000
        6      0.4278      0.5069      0.6008      0.7120      0.8438
        7      0.3609      0.4278      0.5069      0.6008      0.7120
        8      0.3046      0.3609      0.4278      0.5069      0.6008
        9      0.2570      0.3046      0.3609      0.4278      0.5069
       10      0.2168      0.2570      0.3046      0.3609      0.4278
       11      0.1830      0.2168      0.2570      0.3046      0.3609
       12      0.1544      0.1830      0.2168      0.2570      0.3046
       13      0.1303      0.1544      0.1830      0.2168      0.2570
       14      0.1099      0.1303      0.1544      0.1830      0.2168
       15      0.0928      0.1099      0.1303      0.1544      0.1830
                    1           2           3           4           5

        6      1.0000
        7      0.8438      1.0000
        8      0.7120      0.8438      1.0000
        9      0.6008      0.7120      0.8438      1.0000
       10      0.5069      0.6008      0.7120      0.8438      1.0000
       11      0.4278      0.5069      0.6008      0.7120      0.8438
```

```
      12      0.3609      0.4278      0.5069      0.6008      0.7120
      13      0.3046      0.3609      0.4278      0.5069      0.6008
      14      0.2570      0.3046      0.3609      0.4278      0.5069
      15      0.2168      0.2570      0.3046      0.3609      0.4278
                   6           7           8           9          10

      11      1.0000
      12      0.8438      1.0000
      13      0.7120      0.8438      1.0000
      14      0.6008      0.7120      0.8438      1.0000
      15      0.5069      0.6008      0.7120      0.8438      1.0000
                  11          12          13          14          15

  54  VKEEP [CFORMAT=parameters] Row.Column; COVARIANCEMODEL=Parameters
  55  PRINT #Parameters

Parameters['Scalar'] Parameters['Row']['Parameters']
               4.580                          0.6827
                                              0.0000

 Parameters['Column']['Parameters']
                            0.8438
                            0.0000
```

The Wald statistic for a fixed term can be saved using the WALD parameter. The WMETHOD option controls whether these are from the table where terms are added sequentially to the model, or that where terms are dropped from the full fixed model (5.3.1). The associated F statistic, and its numerator and denominator numbers of degrees of freedom, can be saved by the FSTATISTIC, NDF and DDF parameters, respectively. The FMETHOD option specifies which algorithm to use to calculate the denominator numbers of degrees of freedom (5.3.6). The default, automatic, will use any stored values that have been calculated for this analysis by earlier REML, VDISPLAY or VKEEP statements; otherwise it will choose automatically between the two available methods.

The WORKSPACE option can save the workspace setting that was used by the REML command that performed the analysis. This may not be the same as the setting of the WORKSPACE option in the command, as REML may increase the specified value if it is found to be insufficient.

The EXIT option saves a code defining the exit status of the analysis. The codes (which are also used in the EXIT parameter of REML) are as follows:

- 0 analysis was completed successfully;
- 1 analysis did not converged within the specified number of iterations (but no fault occurred);
- 2 the fit was halted because no progress could be made;
- 3 the fit was halted the log-likelihood was diverging;
- 4 a parameter has gone out of bounds;
- 5 insufficient workspace;
- 6 no save structure is available (no REML command or a fault occurred (may be set by VKEEP but not by REML);
- 7 value of deviance at final iteration larger than at previous iteration(s);
- −1 the algorithm performed an iteration but failed for an indeterminate reason before the exit status was established;
- −2 a failure occurred prior to calling the fitting algorithm.

5.9.2 The VFRESIDUALS procedure

VFRESIDUALS procedure

Obtains residuals, fitted values and their standard errors from a REML analysis (S.J. Welham).

Options

RESIDUALS = *variate*	Saves the residuals
SERESIDUALS = *variate*	Saves standard errors of the residuals
FITTEDVALUES = *variate*	Saves the fitted values
SEFITTEDVALUES = *variate*	Saves prediction standard errors for the fitted values
RMETHOD = *string token*	Which random terms to use when calculating the residuals (final, all); default fina
MAXNUNITS = *scalar*	Maximum number of units for which the full variance-covariance matrix will be formed; default 1000
EXIT = *scalar*	Exit code set to zero if the saving was successful, one otherwise
SAVE = *REML save structure*	Save structure for the required analysis; default uses the save structure from the most recent REML

No parameters

VFRESIDUALS saves residuals, fitted values and their standard errors from a REML analysis. The residuals are formed as differences between the data and the fitted model. The RMETHOD option controls which random terms are used to calculate the residuals, with settings:

all	uses all of the random effects, and
final	uses only the final random term (default).

The final setting thus provides conditional residuals, with the fitted model is calculated from all of the fixed and random terms in the model. The all setting provides marginal residuals, with the fitted model is calculated from the fixed terms alone. VFRESIDUALS is currently unable to form standard errors for models containing spline terms.

The residuals and fitted values can be saved, in variates, using the RESIDUALS and FITTEDVALUES options, respectively. The SERESIDUALS option saves the standard errors of the residuals, and the SEFITTEDVALUES option saves the prediction standard errors of the fitted values (i.e. the square root of the prediction error variances).

The standard errors can be calculated in several different ways, and VFRESIDUALS will attempt to use the most efficient method. One method involves saving the full variance-covariance matrix for the data. This can be time-consuming for large data sets, so the MAXNUNITS option sets a limit (default 1000) on the size of data set for which this may be used.

By default, VFRESIDUALS forms the residuals etc. from the most recent REML analysis. However, you can form them from an earlier analysis, by using the SAVE option to specify its save structure (saved using the SAVE parameter of the REML command that performed the analysis).

Example 5.9.2 saves and prints the residuals and fitted values from the split-plot analysis in Section 5.3.1. The standard errors are all equal because the design is balanced.

Example 5.9.2

```
48  VFRESIDUALS [RESIDUALS=residual; SERESIDUALS=seresidual;\
49              FITTEDVALUES=fittedvalue; SEFITTEDVALUES=sefittedvalue]
50  PRINT       Yield,fittedvalue,sefittedvalue,residual,seresidual

     Yield fittedvalue sefittedvalue    residual  seresidual
     156.0       148.4         7.647       7.599       10.89
```

118.0	138.7	7.647	-20.734	10.89
140.0	130.1	7.647	9.933	10.89
105.0	108.2	7.647	-3.234	10.89
111.0	111.0	7.647	0.001	10.89
130.0	129.2	7.647	0.834	10.89
174.0	158.0	7.647	16.001	10.89
157.0	150.3	7.647	6.668	10.89
117.0	107.8	7.647	9.230	10.89
114.0	126.3	7.647	-12.270	10.89
161.0	142.4	7.647	18.564	10.89
141.0	152.6	7.647	-11.603	10.89
104.0	105.3	7.647	-1.345	10.89
70.0	74.8	7.647	-4.845	10.89
89.0	96.7	7.647	-7.678	10.89
117.0	115.0	7.647	1.988	10.89
122.0	112.8	7.647	9.207	10.89
74.0	65.8	7.647	8.207	10.89
89.0	84.0	7.647	5.040	10.89
81.0	105.1	7.647	-24.126	10.89
103.0	99.6	7.647	3.417	10.89
64.0	81.1	7.647	-17.083	10.89
132.0	115.8	7.647	16.250	10.89
133.0	125.9	7.647	7.083	10.89
108.0	105.5	7.647	2.544	10.89
126.0	121.6	7.647	4.378	10.89
149.0	131.8	7.647	17.211	10.89
70.0	87.0	7.647	-16.956	10.89
144.0	135.7	7.647	8.300	10.89
124.0	117.4	7.647	6.634	10.89
121.0	126.0	7.647	-5.033	10.89
96.0	95.5	7.647	0.467	10.89
61.0	65.0	7.647	-3.963	10.89
100.0	112.0	7.647	-11.963	10.89
91.0	83.1	7.647	7.871	10.89
97.0	104.3	7.647	-7.296	10.89
109.0	96.7	7.647	12.264	10.89
99.0	106.4	7.647	-7.402	10.89
63.0	66.2	7.647	-3.236	10.89
70.0	88.1	7.647	-18.069	10.89
80.0	70.5	7.647	9.466	10.89
94.0	105.2	7.647	-11.201	10.89
126.0	115.4	7.647	10.633	10.89
82.0	89.0	7.647	-7.034	10.89
90.0	82.6	7.647	7.418	10.89
100.0	103.7	7.647	-3.748	10.89
116.0	111.4	7.647	4.585	10.89
62.0	64.4	7.647	-2.415	10.89
96.0	110.4	7.647	-14.387	10.89
60.0	65.6	7.647	-5.554	10.89
89.0	100.2	7.647	-11.221	10.89
102.0	84.1	7.647	17.946	10.89
112.0	98.2	7.647	13.761	10.89
86.0	105.9	7.647	-19.906	10.89
68.0	58.9	7.647	9.094	10.89
64.0	77.1	7.647	-13.073	10.89
132.0	121.4	7.647	10.612	10.89
124.0	131.1	7.647	-7.054	10.89
129.0	112.7	7.647	16.279	10.89
89.0	90.9	7.647	-1.888	10.89
118.0	103.3	7.647	14.742	10.89
53.0	63.9	7.647	-10.924	10.89
113.0	110.9	7.647	2.076	10.89
74.0	82.1	7.647	-8.091	10.89
104.0	112.9	7.647	-8.936	10.89
86.0	102.8	7.647	-16.770	10.89
89.0	68.1	7.647	20.897	10.89
82.0	86.6	7.647	-4.603	10.89
97.0	84.3	7.647	12.735	10.89
99.0	106.1	7.647	-7.098	10.89
119.0	114.8	7.647	4.235	10.89
121.0	124.4	7.647	-3.431	10.89

5.9.3 The VSPREADSHEET procedure

VSPREADSHEET procedure

Saves results from a REML analysis in a spreadsheet (R.W. Payne).

Options

COMPONENTS = *variate*	Variate to contain the variance components; default components
MEANS = *pointer*	Pointer to tables to contain the means; default means
SEDMEANS = *pointer*	Pointer to matrices to contain the standard errors of differences of the means; default sedmeans
VARMEANS = *pointer*	Pointer to matrices to contain the variance-covariance matrices of the means; default varmeans
EFFECTS = *pointer*	Pointer to tables to contain the effects; default effects
SEDEFFECTS = *pointer*	Pointer to matrices to contain the standard errors of differences of the effects; default sedeffects
VAREFFECTS = *pointer*	Pointer to matrices to contain the variance-covariance matrices of the effects; default vareffects
REPLICATIONS = *pointer*	Pointer to tables of replications; default replication
WALDTABLE = *pointer*	Pointer to a text and variates containing the information in the table of tests for fixed effects; default waldtable
PTERMS = *formula*	Terms (fixed or random) for which effects or means are to be saved; default * implies all the fixed terms
FMETHOD = *string token*	Controls whether and how to calculate F-statistics for fixed terms (automatic, none, algebraic, numerical); default auto
SPREADSHEET = *string tokens*	What to include in the spreadsheet (components, waldtable, effects, sedeffects, vareffects, means, sedmeans, varmeans, replications); default comp, wald, mean, sedm, repl
OUTFILENAME = *text*	Name of Genstat workbook file (.gwb) or Excel (.xls or .xlsx) file to create
SAVE = *REML save structure*	Specifies which REML analysis to save; default * i.e. most recent one

No parameters

VSPREADSHEET puts results from a REML analysis into a spreadsheet. By default the results are from the most recent REML, but you use the SAVE option to specify the save structure from some other analysis.

The SPREADSHEET option specifies which pages of the spreadsheet to form, with settings:

components	variance components,
waldtable	tests for fixed effects,
effects	tables of effects,
sedeffects	standard errors of differences of effects,
vareffects	variance-covariance matrices of effects,
means	tables of means,
sedmeans	standard errors of differences of means,
varmeans	variance-covariance matrices of means,
replications	replication tables.

(Note: this includes only the information readily assembled from VKEEP. So, for example,

parameters of correlation models are not available.) By default, SPREADSHEET = comp, wald, mean, sedm, repl.

To help avoid clashes between the columns of the spreadsheets if you want to save results from more than one analysis, the parameters COMPONENTS, WALDTABLE, EFFECTS, SEDEFFECTS, VAREFFECTS, MEANS, SEDMEANS, VARMEANS and REPLICATIONS allow you to specify identifiers for the columns (or sets of columns) that will store the corresponding results in the current spreadsheet.

You can save the data in either a Genstat workbook (.gwb) or an Excel spreadsheet (.xls or .xlsx), by setting the OUTFILENAME option to the name of the file to create. If the name is specified without a suffix, '.gwb' is added (so that a Genstat workbook is saved). If OUTFILENAME is not specified, the data are put into a spreadsheet opened inside Genstat.

So, you could save the variance components, Wald tests, means and standard errors of differences of means in an Excel spreadsheet called Oatsresults.xlsx by giving the command

```
VSPREADSHEET [SPREADSHEET=components,waldtable,means,sedmeans;\
              OUTFILE='Oatsresults.xlsx]
```

5.9.4 The VFIXEDTESTS procedure

VFIXEDTESTS procedure

Saves fixed tests from a REML analysis (R.W. Payne).

Options

FIXEDTESTS = *pointer*	Saves the fixed tests
FMETHOD = *string token*	Controls whether and how to calculate F-statistics (automatic, none, algebraic, numerical); default auto
WMETHOD = *string token*	Controls which tests are saved (add, drop); default drop
SAVE = *REML save structure*	Specifies the save structure from the required analysis; default * i.e. most recent one

No parameters

VFIXEDTESTS saves the results of the fixed tests in a REML analysis. By default the results are from the most recent REML, but you use the SAVE option to specify the save structure from some other analysis.

The WMETHOD option controls whether the tests are from the table where terms are added sequentially to the model, or that where terms are dropped from the full fixed model.

The FMETHOD option specifies which algorithm to use to calculate the denominator numbers of degrees of freedom required for F tests. The default, automatic, will use any stored values that have been calculated for this analysis by earlier REML, VDISPLAY or VKEEP statements; otherwise it will choose automatically between the two available methods.

The tests are saved, in a pointer, using the FIXEDTESTS option. The pointer is labelled by the headings from the tests for fixed tests that appear in the REML output. If the denominator degrees of freedom are available, the labels and their corresponding vectors are as follows:

Term	text containing the names of the fixed terms,
Wald statistic	variate containing the Wald statistics,
n.d.f.	variate containing the numerator degrees of freedom,
F statistic	variate containing the F statistics,
d.d.f.	variate containing the denominator degrees of freedom,

`F pr.` variate containing the probabilities for the F tests.

If the denominator degrees of freedom are not available (either because they could not be calculated, or because `FMETHOD` has been set to `none`), the labels `F statistic`, `d.d.f.` and `F pr.` are omitted, and instead there is

`Chi pr.` variate containing the probabilities for chi-square tests for the Wald statistics.

The vectors have an element for each fixed term, with missing values if its test results are unavailable. For example, with the fixed model `Nitrogen*Variety`, tests for the main effects `Nitrogen` and `Variety` would be available only when `WMETHOD=add`. This is illustrated in Example 5.9.4, which saves and prints the fixed tests from the split-plot analysis in Section 5.3.1.

Example 5.9.4

```
  51  VFIXEDTESTS [FIXEDTESTS=Drop]
  52  PRINT       Drop[]

    Drop['Term'] Drop['Wald statistic'] Drop['n.d.f.'] Drop['F statistic']
        Nitrogen                      *              *                    *
         Variety                      *              *                    *
Nitrogen.Variety                  1.817          6.000               0.3028

 Drop['d.d.f.'] Drop['F pr.']
              *             *
              *             *
          45.00        0.9322

  53  VFIXEDTESTS [FIXEDTESTS=Add; WMETHOD=add]
  54  PRINT       Add[]

     Add['Term'] Add['Wald statistic'] Add['n.d.f.'] Add['F statistic']
        Nitrogen                113.06         3.000              37.69
         Variety                  2.97         2.000               1.49
Nitrogen.Variety                  1.82         6.000               0.30

 Add['d.d.f.'] Add['F pr.']
         45.00       0.0000
         10.00       0.2724
         45.00       0.9322
```

Acknowledgements

The Fisher-scoring algorithm was adapted from the REML program (Robinson, Thompson & Digby 1982) of the Scottish Agricultural Statistics Service, now BioSS, Edinburgh, with their kind permission. The newer algorithm, which uses sparse matrix manipulation with the average information optimization method, is also the core of the programs ASREML and ASReml-R; see http://www.vsni.co.uk/software/asreml/. The REML facilities are the result of collaboration between R. Thompson (Rothamsted), A.R. Gilmour (VSNi), S.A. Harding (VSNi), S.J. Welham (VSNi), B.R. Cullis (University of Wollongong), A.P. Verbyla (University of Wollongong), D.G. Butler (Queensland Department of Primary Industries), B.J. Gogel (University of Adelaide) and M.G. Kenward (London School of Hygiene and Tropical Medicine).

6 Multivariate and cluster analysis

In this chapter we are concerned with statistical methods for analysing more than one variable simultaneously (which correspond to the multivariate analysis menus in Genstat *for Windows*). Very often such methods initially combine information on all the given variables into a measure of association, such as a distance or dissimilarity; so, in a sense, they become univariate. Indeed in some fields of application, notably psychology and the social sciences, a single variable of associations may be observed directly, rather than calculated from more basic information. Multivariate analysis is concerned with two forms of data: (a) information on p variables for each of n samples (this can be called the data matrix); or (b) information, usually presented as a symmetric matrix, giving associations between all pairs of samples or all pairs of variables.

In the simplest cases the data matrix has no further structure, and may be regarded as the multivariate generalization of a simple random sample. Genstat does not have a special data structure for a data matrix; generally you must either list the corresponding variables, or collect them in a pointer (1:2.6). From a data matrix you can use the `FSSPM` directive to calculate the symmetric matrix of sums of squares and products, or alternatively, the correlation matrix of the variables. These are stored in a compound data structure known as an SSPM structure, which also contains the means of the variables and other information (6.1.1). However, you can easily extract the basic symmetric matrix from this more general structure.

Just as univariate samples may have structure imposed on the units, so may multivariate samples. In canonical variates analysis the units belong to a set of k mutually exclusive groups. For this Genstat lets you calculate the matrix of sums of squares and products, pooled within groups, as well as the means of all the variables in all the groups (6.1.1); these means are held as a set of p variates, each with k values, from which Genstat can calculate a matrix of between-group sums of squares and products. Sums of squares and products arising from more general sample structures are provided by `AKEEP` (1:4.6.1).

Correlations and sums of squares and products are elementary examples of how associations can be measured between variables; methods based on such measures are sometimes termed *R-techniques* and include such methods as principal components analysis and canonical variates analysis. Measures of association between units lead to methods known as *Q-techniques* which include ordination techniques, such as principal coordinates analysis and multidimensional scaling, and cluster analysis.

You can think of matrices of distances or dissimilarities as being generated by a cloud of n points in a multidimensional Euclidean space, where the distance between the points representing two samples is or is related to the corresponding distance or dissimilarity in the given matrix. To visualize such a cloud of points is difficult, and much multivariate analysis is concerned with providing approximate graphical representations that are easily interpreted by eye. These representations fall into two main classes: those depending on scatter plots of points in two or, more rarely, three dimensions; and those expressed in the form of networks, especially rooted trees. The plotted distance is usually supposed to approximate to the "true" distance in multidimensional space. Alternatively you may need to examine angle, inner product or area, rather than distance: for example, angles are used to interpret the output from biplots (Gabriel 1971). Apart from the minimum spanning tree given by the `HDISPLAY` directive (6.19.2), all other standard network-type displays in Genstat are in the form of rooted trees. An important example is the dendrogram generated as the result of a hierarchical cluster analysis (6.19.1). This represents the similarity of groupings of the units by recording the similarity levels at which they merge together. The root of the tree is the point at which they all merge into a single group. Other tree structures are generated by forming classification trees (6.21) and identification keys (6.22). These aim to provide ways of identifying the group of an object based on its observed properties. Genstat also provides regression trees, where the aim is to predict a response variate

(3.9).

Many multivariate techniques are implemented as standard Genstat directives. Others are suppled as procedures which make use of the comprehensive toolkit that Genstat provides, for example, matrix calculations (1:4.1.3 and 1:4.2.4), singular value decompositions (1:4.10.1), eigenvalue decompositions (1:4.10.2). All the main techniques (and all of those that can be performed by the menus of Genstat *for Windows*) are described in this chapter. Details of the others can be found in Part 3 of the *Genstat Reference Manual*.

FSSPM	calculates values for SSPM structures – sums of squares and products, means, etc (6.1.1)
ROBSSPM	forms robust estimates of sum-of-squares-and-products matrices (6.1.1)
FCORRELATION	forms correlations between variates (2.8.1)
FVCOVARIANCE	forms the variance-covariance matrix for a list of variates
FSIMILARITY	forms a similarity matrix or a between-group similarity matrix from a units-by-variates data matrix (6.1.2)
HREDUCE	forms a reduced similarity matrix, by groups (6.1.3)
MANTEL	assesses the association between similarity matrices (6.1.5)
ECANOSIM	does a nonparametric analysis of similarities (*ANOSIM*) to test for differences between two or more groups of sampling units (6.1.6)
PCP	principal components analysis (6.2.1)
LRVSCREE	prints a scree diagram and/or a difference table of latent roots (6.2.2)
CVA	canonical variates analysis (6.3.1)
CVASCORES	calculates scores for individual units in canonical variates analysis (6.3.2)
CVAPLOT	plots mean and unit scores from a canonical variates analysis (6.3.3)
CVATRELLIS	displays the distribution of groups over 2 dimensions from a CVA analysis using a trellis of bar or pie charts (6.3.4)
FACROTATE	rotates factor loadings from a PCP, CVA or FCA (6.4)
FCA	performs factor analysis (6.11)
DISCRIMINATE	performs discriminant analysis (6.5.1)
SDISCRIMINATE	selects the best set of variates to discriminate between groups (6.5.2)
QDISCRIMINATE	selects the best set of variates to discriminate between groups (6.5.3)
MANOVA	multivariate analysis of variance and covariance (6.6.1)
RMULTIVARIATE	multivariate linear regression (6.6.2)
MVAOD	does an analysis of distance of multivariate data
RIDGE	produces ridge regression and principal component regression analyses (6.7)
PLS	fits a partial least squares regression model (6.8)
CANCORRELATION	canonical correlation analysis (6.9)
PCO	principal coordinates analysis (6.10.1)
ADDPOINTS	adds points for new objects to a PCO (6.10.2)
PCORELATE	relates principal coordinates to original data variates (6.10.3)
MDS	non-metric multidimensional scaling (6.12)
CORANALYSIS	does correspondence analysis, or reciprocal averaging

	(6.13.1)
MCORANALYSIS	does multiple correspondence analysis (6.13.2)
CABIPLOT	plots results from correspondence analysis or multiple correspondence analysis (6.13.3)
RDA	performs redundancy analysis (6.14)
CCA	performs canonical correspondence analysis (6.15)
DBIPLOT	plots a biplot from an analysis by PCP, CVA or PCO (6.16.1)
CRBIPLOT	plots correlation or distance biplots after RDA, or ranking biplots after CCA (6.16.2)
CRTRIPLOT	Plots ordination biplots or triplots after RDA or CCA (6.16.3)
GGEBIPLOT	plots biplots to assess genotype and genotype-by-environment variation
SKEWSYMMETRY	provides an analysis of skew-symmetry for an asymmetric matrix (6.17)
ROTATE	Procrustes rotation (6.18.1)
GENPROCRUSTES	generalized Procrustes analysis (6.18.2)
PCOPROCRUSTES	performs a multiple Procrustes analysis (6.18.3)
HCLUSTER	hierarchical cluster analysis from a similarity matrix (6.19.1)
HDISPLAY	displays results associated with hierarchical clustering (6.19.2)
HLIST	lists a data matrix in abbreviated form (6.19.3)
HSUMMARIZE	summarizes data variates by clusters (6.19.4)
DDENDROGRAM	draws dendrograms with control over structure and style (6.19.5)
DMST	gives a high resolution plot of an ordination with minumum spanning tree (6.19.6)
HCOMPAREGROUPINGS	compares groupings generated, for example, from cluster analyses (6.19.7)
CLUSTER	non-hierarchical clustering from a data matrix (6.20.1)
CLASSIFY	obtains a starting classification for non-hierarchical clustering (6.20.2)
PCPCLUSTER	forms groups of units using the densities of their PCP scores (6.20.3)
BCLASSIFICATION	constructs a classification tree (6.21.1)
BCDISPLAY	displays a classification tree (6.21.2)
BCKEEP	saves information from a classification tree (6.21.5)
BCVALUES	forms values for nodes of a classification tree (6.21.3)
BPRUNE	prunes a tree using minimal cost complexity (1:4.12.8, 3.9.3, 6.21.3)
BCIDENTIFY	identifies specimens using a classification tree (6.21.4)
BCFOREST	constructs a random classification forest
BCFDISPLAY	displays information about a random classification forest
BCFIDENTIFY	identifies specimens using a random classification forest
BKEY	constructs an identification key (6.22.1)
BKDISPLAY	displays an identification key (6.22.2)
BKIDENTIFY	identifies specimens using a key (6.22.3)
BKKEEP	saves information from an identification key
IDENTIFY	identifies an unknown specimen from a defined set of

	objects (6.22.5)
IRREDUNDANT	forms irredundant test sets for the efficient identification of a set of objects (6.11.6)
AMMI	allows exploratory analysis of genotype × environment interactions
CINTERACTION	clusters rows and columns of a two-way interaction table
CONVEXHULL	finds the points of a single or a full peel of convex-hulls
DPARALLEL	displays multivariate data using parallel coordinates (2.7.2)
MULTMISSING	estimates missing values for units in a multivariate data set
NORMTEST	performs tests of univariate and/or multivariate normality
RLFUNCTIONAL	fits a linear functional relationship model

For general reading in applied multivariate analysis see for example Manly (1986), Mardia, Kent & Bibby (1979), Krzanowski (1988), Chatfield & Collins (1986) and Gower (1985a). For work in classification and cluster analysis, see Gordon (1981).

6.1 Measures of association

Section 6.1.1 describes the SSPM and FSSPM directives which form the Genstat SSPM structure (2.7.2). This contains sums of squares and products, means and associated information, and can be used as input to several multivariate commands including PCP (6.2.1) and CVA (6.3.1). It then describes the ROBSSPM procedure which can form robust estimates. Sections 6.1.2 - 6.1.4 explain how to form similarity matrices. Finally, Section 6.1.5 describes the MANTEL procedure which assessing the association between two similarity matrices. Related commands, described elsewhere, include CORRELATE directive (which forms correlation matrices: see 7.7.1), and FVCOVARIANCE (which forms variance-covariance matrices: see Part 3 of the *Genstat Reference Manual*).

6.1.1 Forming sums of squares and products

Several Genstat commands require matrices of sums of squares and products as their input. These are stored by Genstat in a data structure known as an SSPM. You first declare the structure using the SSPM directive, and then form its values using FSSPM.

SSPM directive

Declares one or more SSPM data structures.

Options

TERMS = *formula*	Terms for which sums of squares and products are to be calculated; default *
FACTORIAL = *scalar*	Maximum number of vectors in a term; default 3
FULL = *string token*	Full factor parameterization (yes, no); default no
GROUPS = *factor*	Groups for within-group SSPMs; default *
DF = *scalar*	Number of degrees of freedom for sums of squares; default *

Parameters

IDENTIFIER = *identifiers*	Identifiers of the SSPMs
SSP = *symmetric matrices*	Symmetric matrix to contain the sums of squares and products for each SSPM
MEANS = *variates*	Variate to contain the means for each SSPM
NUNITS = *scalars*	Number of units or sum of weights for each SSPM

WMEANS = *pointers*	Pointers to variates of group means for each SSPM

For a multivariate analysis, the setting of the TERMS option is simply the list of variates from which the sums of squares and products are to be calculated, and the FACTORIAL and FULL options are irrelevant (these may be used in regression, when TERMS can supply a model formula). The SSPM is a compound structure with four components (identified by their suffixes).

[1] or ['SUMS'] is a symmetric matrix containing the(corrected) sums of squares and products. The number of rows and columns of this matrix will equal the number of parameters defined by the expanded terms list: that is, the number of variates plus the number of dummy variates generated by the model formula. (See the TERMS directive: 3.2.3.)

[2] or ['MEANS'] is a variate containing the mean for each variate or dummy variate.

[3] or ['NUNITS'] is a scalar holding the total number of units used in constructing the sums of squares and products matrix. If the SSPM is weighted, this scalar will hold the sum of the weights.

The within-group SSPM (produced when the GROUPS option is set, and used for canonical variates analysis) has an additional element:

[4] or ['WMEANS'] is a pointer, pointing to variates holding within-group means. There is one variate for each row of the 'SUMS' matrix plus one extra. They are all of the same length, namely the number of levels of the GROUPS factor. The extra variate holds counts of the number of units in each group.

The first parameter of SSPM provides an identifier for the SSPM structure(s). The other parameters allow you to specify identifiers for the four components of the SSPM(s), so that you can refer to them directly. Genstat will declared them automatically as structures of the correct types and sizes. You can declared them in advance if you prefer but, if so, they must be of the correct type. You can also use them to provide values for the SSPM (instead of using the TERMS option to list the variates from which the values are to be calculated, later, by the FSSPM directive). You can then also set the DF option to indicate the degrees of freedom for the sums of squares.

Having declared the SSPM, you can form its values using FSSPM.

FSSPM directive

Forms the values of SSPM structures.

Options

PRINT = *string tokens*	Printed output required (correlations, wmeans, SSPM); default * i.e. no printing
WEIGHTS = *variate* or *symmetric matrix*	Variate of weights for weighted SSP, or symmetric matrix of weights (one row and column for each unit of data); default * i.e. all units with weight one
SEQUENTIAL = *scalar*	Used for sequential formation of SSPMs; a positive value indicates that formation is not yet complete (see READ directive); default * i.e. not sequential

Parameter

SSPMs	Structures to be formed

FSSPM forms the values for the component parts of SSPM structures, based on the information specified by the SSPM directive, when they were declared. The method used to form the SSPM is based on the updating formula for the means and corresponding corrected sums of squares and cross products (Herraman 1968).

FSSPM has one parameter which lists the SSPM structures whose values are to be formed. Genstat takes account of restrictions on any of the variates or factors forming the terms of the SSPM, or on the weights variate or grouping factor if you have specified them. If any of these vectors has a missing value, the corresponding unit is excluded from all the means and all the sums of squares and products. You can also exclude units by setting their weights to zero.

When you have very many units, you may not be able to store them all at the same time within Genstat. You can then use the SEQUENTIAL option of READ (1:3.1.10) to read the data in conveniently sized blocks, and the SEQUENTIAL option of FSSPM to control the accumulation of the sums of squares and products. The SSPM is updated for each block of data in turn until the end of data is found.

Example 6.1.1 shows the use of SSPM and FSSPM to form a within-group SSPM. The data variates are seven measurements made on 28 brooches found at the archaeological site of the cemetery at Munsingen (Doran & Hodson 1975). They have all been transformed by taking logarithms. The SSPM is used later in Section 6.3 for a canonical variates analysis. (These seven variables are also used in the first example of the CLUSTER directive in Section 6.20.1, and the grouping used here is that obtained from CLUSTER).

Example 6.1.1

```
  2  UNITS [NVALUES=28]
  3  POINTER [VALUES=Foot_lth,Bow_ht,Coil_dia,Elem_dia,Bow_wdth, \
  4    Bow_thck,Length] Data
  5  FACTOR [LEVELS=4] Groupno
  6  READ Groupno,Data[]

   Identifier   Minimum      Mean   Maximum    Values   Missing
     Foot_lth     2.398     3.278     4.554        28         0
       Bow_ht     2.079     2.842     3.296        28         0
     Coil_dia     1.792     2.166     2.833        28         0
     Elem_dia     1.099     2.026     2.708        28         0
     Bow_wdth     3.045     4.064     5.176        28         0
     Bow_thck     2.708     3.621     4.357        28         0
       Length     3.296     4.003     4.860        28         0

   Identifier    Values   Missing    Levels
      Groupno        28         0         4

 35  SSPM  [TERMS=Data[]; GROUPS=Groupno] W
 36  FSSPM [PRINT=SSPM,wmeans] W

Degrees of freedom
------------------

Sums of squares:  24
Sums of products: 23


Sums of squares and products
----------------------------

     Foot_lth   1       2.0191
       Bow_ht   2      -0.2031       1.3884
     Coil_dia   3       0.2782       0.6409       0.8659
     Elem_dia   4       0.5373       0.7506       0.7578       2.8110
     Bow_wdth   5      -0.2362       0.2028      -0.1215      -0.9082
     Bow_thck   6      -0.4963       1.1359       0.1268       0.2570
       Length   7       0.9921       0.7013       0.5380       0.2839
                             1            2            3            4

     Bow_wdth   5       2.0679
     Bow_thck   6       0.6171       2.4207
       Length   7       0.1339       0.3782       1.3242
                             5            6            7
```

```
Means over all groups
---------------------

     Foot_lth   1        3.278
       Bow_ht   2        2.842
     Coil_dia   3        2.166
     Elem_dia   4        2.026
     Bow_wdth   5        4.064
     Bow_thck   6        3.621
       Length   7        4.003


Number of units used
--------------------

           28


Within-group means
------------------

Grouping factor: Groupno

                          1           2           3           4
     Foot_lth   1        2.956       3.082       4.188       3.228
       Bow_ht   2        2.680       2.873       3.001       2.802
     Coil_dia   3        2.141       2.073       2.517       2.071
     Elem_dia   4        1.493       2.304       2.283       1.741
     Bow_wdth   5        3.457       4.078       4.059       4.762
     Bow_thck   6        3.352       3.837       3.901       3.143
       Length   7        3.775       3.899       4.592       3.938
     Constant   8        6.000      12.000       5.000       5.000
```

Alternatively, procedure ROBSSPM allows you to form robust estimates of SSPMs, and the related variance-covariance and correlation matrices, using the method of Campbell (1980). This weights the units differentially so that those that are extreme, in a multivariate sense, contribute less to the calculated means and sums of squares and products. The extremeness of a unit is judged by its Mahalanobis distance from the estimated mean.

ROBSSPM procedure

Forms robust estimates of sum-of-squares-and-products matrices (P.G.N. Digby).

Options

PRINT = *string tokens*	Controls printed output (sspm, distances, weights, vcovariance, means, correlations, outliers); default * i.e. no output
B1 = *scalar*	The value from which the threshold distance is derived (see the Method Section); default 2
B2 = *scalar*	The value indicating the decline in weight as the distance of a unit above the threshold increases, (see the Method Section); default 1.25
MAXCYCLE = *scalar*	Maximum number of iterations; default 100
TOLERANCE = *scalar*	The minimum change in the average squared-weight that has to be achieved for the iterative process to converge; default 1.0^{-8}

Parameters

DATA = *pointers*	Supplies the set of variates in each datamatrix
SSPM = *SSPMs*	SSPM structure to contain the robust estimates of the sums of squares and products, the robust estimates of the

	means, and the sum of the weights for each datamatrix
DISTANCES = *variates*	To contain the Mahalanobis distances of the units from the mean
WEIGHTS = *variates*	To contain the weights used for each unit when forming the robust estimates
VCOVARIANCE = *symmetric matrices*	To contain the robust estimates of the matrices of variances and covariances
CORRELATIONS = *symmetric matrices*	This contains on output the correlations from the robust estimates of the variances and covariances

The variates from which the sums of squares and products are to be calculated are specified, in a pointer, by the DATA parameter. They may be restricted or may contain some missing values, in which case the units concerned will be ignored.

Output is controlled by the PRINT option, with settings: sspm prints the estimated sums-of-squares-and-products, the estimated means, and the sum of the weights; distances prints the Mahalanobis distances for all the units, including any excluded by restrictions; weights prints the weights for all the units; vcovariance prints the estimated variance-covariance matrix; means prints the estimated means; correlations prints correlations derived from the variance-covariance matrix; outliers prints unit numbers, weights, and distances for outliers. By default there is no printed output.

If the outliers, weights or distances are to be printed, then an appropriate summary of the number of units, number of outliers and so on will be printed too. The outlier information consists of the unit numbers, weights and Mahalanobis distances, printed across the page.

The estimation process is iterative, with the maximum number of iterations controlled by the MAXCYCLE option (default 100). Initial (unweighted) estimates of the means and sums of squares and products are formed from all the units, subject to any restriction on the data and excluding any units with missing values for any of the variates. From the estimates, Mahalanobis distances of the units from their means are calculated, and used to determine the weights for the units. The weights are then used to reform the SSPM structure, new distances are calculated, and so on. Convergence occurs when the average change in the derived weights is less than the some tolerance. The default tolerance is 1.0^{-8}, but this can be redefined by the TOLERANCE option. Lack of convergence usually indicates some problem with the data, perhaps that the threshold has been set too low.

The weight w of each unit is given by

$$w = 1 \qquad d \le t$$
$$w = (t/d) \times \exp(-0.5 \times (d-t)^2 / \mathrm{B2}^2) \qquad d > t$$

where t, the threshold distance, is given by

$$t = \sqrt{v} + \mathrm{B1} / \sqrt{2}$$

and v is the number of means.

As explained by Campbell (1980), under Fisher's square root approximation, B1 equates to a percentage point of the standard Gaussian distribution.

The parameters in the calculation of the weights are specified by options B1 and B2. Campbell (1980) regards three possibilities as potentially most useful. If B1 is infinite, the usual (non-robust) estimates are obtained. With B1=2 and B2 infinite, the weight decreases inversely with distance ($w=t/d$); this can be obtained in the procedure by setting B2 to a missing value. Finally, there is the combination used as a default by ROBSSPM, namely B1=2 and B2=1.25.

Parameters SSPM, DISTANCES, WEIGHTS, VCOVARIANCE and CORRELATIONS allow the various components of the output to be saved.

6.1.2 Forming similarity matrices: the FSIMILARITY directive

Many forms of multivariate analysis operate on symmetric matrices that give similarities between all pairs of samples: these are termed *Q-methods*. The FSIMILARITY directive (which is used by the Form Similarity Matrix menu of Genstat *for Windows*) forms similarity matrices, essentially using the method described by Gower (1971). The similarity coefficient that is calculated allows variables to be qualitative, quantitative, or dichotomous, or mixtures of these types; values of some of the variables may be missing for some samples. The values of a similarity coefficient vary between zero and unity, though some authors express them as percentages in the range 0-100%. Two samples have a similarity of unity only when both have identical values for all variables; a value of zero occurs when the values for the two samples differ maximally for all variables. Thus similarity is the complement of dissimilarity, and to convert a similarity s_{ij} into a dissimilarity you can evaluate expressions like $1-s_{ij}$ or $\sqrt{(1-s_{ij})}$. Whether a set of dissimilarities obeys the metric axioms (particularly the triangle inequality), or can be regarded as being generated by distances between pairs of points in a multidimensional Euclidean space, depends on the particular coefficient and on the data themselves. Genstat can evaluate similarities using many of the standard similarity coefficients for qualitative and quantitative variables; Gower (1985) and Gower & Legendre (1986) discuss some of the properties of these coefficients. In Genstat the resulting similarity matrices are ordinary symmetric matrices, so you can use the standard matrix operations (1:4.10); their main use in multivariate analysis is for principal coordinates analysis (6.10.1), or other forms of metric scaling or non-metric scaling, or for hierarchical cluster analysis (6.19).

FSIMILARITY directive

Forms a similarity matrix or a between-group-elements similarity matrix or prints a similarity matrix.

Options

PRINT = *string token*	Printed output required (similarities, summary); default * i.e. no printing
STYLE = *string token*	Print percentage similarities in full or just the 10% digit (full, abbreviated); default full
METHOD = *string token*	Form similarity matrix or rectangular between-group-element similarity matrix (similarities, betweengroupsimilarities); default simi
SIMILARITY = *matrix* or *symmetric matrix*	Input or output matrix of similarities; default *
GROUPS = *factor*	Grouping of units into two groups for between-group-element similarity matrix; default *
PERMUTATION = *variate*	Permutation of units (possibly from HCLUSTER) for order in which units of the similarity matrix are printed; default *
UNITS = *text* or *variate*	Unit names to label the rows of the similarity matrix; default *
MINKOWSKI = *scalar*	Index *t* for use with TEST=minkowski

Parameters

DATA = *variates* or *factors*	The data values
TEST = *string tokens*	Test type, defining how each DATA variate or factor is treated in the calculation of the similarity between each unit (simplematching, jaccard, russellrao,

	dice, antidice, sneathsokal, rogerstanimoto, cityblock, manhattan, ecological, euclidean, pythagorean, minkowski, divergence, canberra, braycurtis, soergel); default * ignores that variate or factor
RANGE = *scalars*	Range of possible values of each DATA variate or factor; if omitted, the observed range is taken

FSIMILARITY forms a symmetric matrix of similarities, or a rectangular matrix of similarities between the units in two groups. You can save either form of similarity matrix, using the SIMILARITY option. FSIMILARITY can also be used to print the symmetric matrix of similarities after it has formed it; alternatively, you can input an existing similarity matrix for printing, using the SIMILARITY option.

The DATA parameter specifies a list of variates or factors, all of which must be of the same length. If any of the variates or factors is restricted, or if the factor in the GROUPS option is restricted, then that restriction is applied to all the variates or factors. Any restriction on any other variate or factor must be to the same set of units. The dimension of the resulting symmetric matrix of similarities is taken from the number of units that contribute to the similarity matrix. If you want to print an existing similarity matrix, the DATA parameter (and the TEST and RANGE parameters) should be omitted, and the SIMILARITY option used to input the matrix concerned.

The TEST parameter specifies a list of strings, one for each variate or factor in the DATA parameter list, that define their "types". If you want to exclude a variate or factor from contributing, you should specify an empty string (* or '').

Otherwise the similarity between units i and j is calculated as

$$\sum_k \{ w_k(x_{ik}, x_{jk})\, s_k(x_{ik}, x_{jk}) \} / \sum_k w_k(x_{ik}, x_{jk})$$

where x_{ik} is the value of the DATA variate or factor k in unit i, and the contribution functions s_k and weight functions w_k for a variate k of the available types are defined in the tables below (for further details see Gower 1971, 1985).

The first table contains the types appropriate for variates that are recording the presence or absence of a characteristic; they cannot be used with factors.

Type	Contribution s_k	Weight w_k
Jaccard	if $x_i \neq 0$ and $x_j \neq 0$, then 1	1
	if $x_i = x_j = 0$, then 0	0
	if only one of x_i or $x_j = 0$, then 0	1
RussellRao	if $x_i \neq 0$ and $x_j \neq 0$, then 1	1
	if $x_i = 0$ or $x_j = 0$, then 0	1
Dice	if $x_i \neq 0$ and $x_j \neq 0$, then 1	1
	if $x_i = x_j = 0$, then 0	0
	if only one of x_i or $x_j = 0$, then 0	0.5
antidice	if $x_i \neq 0$ and $x_j \neq 0$, then 1	1
	if $x_i = x_j = 0$, then 0	0
	if only one of x_i or $x_j = 0$, then 0	2
SneathSokal	if $x_i \neq 0$ and $x_j \neq 0$, then 1	1

	if $x_i = x_j = 0$, then 1	1
	if only one of x_i or $x_j = 0$, then 0	0.5
RogersTanimoto	if $x_i \neq 0$ and $x_j \neq 0$, then 1	1
	if $x_i = x_j = 0$, then 1	1
	if only one of x_i or $x_j = 0$, then 0	2

The simplematching type is appropriate for qualitative variables, which may be either variates or factors.

Type	Contribution s_k	Weight w_k
simplematching	if $x_i = x_j$, then 1	1
	if $x_i \neq x_j$, then 0	1

The next table shows the types that can be used for quantitative variates (but not factors). In the definitions, r is the range of the variate, t is the Minkowski index (defined by the MINKOWSKI option). Note, however, that BrayCurtis and Soergel should not be mixed with other types.

Type	Contribution s_k	Weight w_k
cityblock	$1 - \lvert x_i - x_j \rvert / r$	1
Manhattan	synonymous with cityblock	
ecological	$1 - \lvert x_i - x_j \rvert / r$	1
	unless $x_i = x_j = 0$	0
Euclidean	$1 - \{(x_i - x_j) / r\}^2$	1
Pythagorean	synonymous with Euclidean	
Minkowski	$1 - \lvert x_i - x_j \rvert^t / r^t$	1
Divergence	$1 - \{(x_i - x_j) / (x_i + x_j)\}^2$	1
Canberra	$1 - \lvert x_i - x_j \rvert / (\lvert x_i \rvert + \lvert x_j \rvert)$	1
BrayCurtis	$1 - \lvert x_i - x_j \rvert / (x_i + x_j)$	$x_i + x_j$
Soergel	$1 - \lvert x_i - x_j \rvert / \max(x_i, x_j)$	$\max(x_i, x_j)$

The RANGE parameter contains a list of scalars, one for each variate or factor in the DATA list. This allows you to check that the values of each variate or factor lie within the given range. If any variate or factor fails the range check, FSIMILARITY gives an error diagnostic and terminates without forming the similarity matrix. The range is also used to standardize quantitative variates; this allows you to impose a standard range, for example when variates are measured on commensurate scales. You can omit the RANGE parameter for all or any of the variates or factors by giving a missing identifier or a scalar with a missing value; Genstat then uses the observed range. If PRINT=summary, Genstat prints the name, the minimum value, and the range for each variate and factor.

The three parameters of the FSIMILARITY directive are also used, for the same purposes, in

the directives PCORELATE (6.10.3), HLIST (6.19.3), and HSUMMARIZE (6.19.4).

The METHOD option controls what type of matrix is produced. The default setting, similarities, gives a symmetric matrix of similarities amongst a single set of units. The betweengroupsimilarities setting gives a rectangular matrix of similarities between two sets of units. To form a rectangular matrix of similarities, you must also define the grouping of units by setting the GROUPS option (see below).

The PRINT, STYLE, and PERMUTATION options govern the printing of a symmetric matrix of similarities. You can either form the similarity matrix within FSIMILARITY, or input it by the SIMILARITY option. To print the similarity matrix you should set option PRINT=similarities. The STYLE option has two settings, full (the default) or abbreviated. The similarity matrix printed in full style has its values displayed as percentages with one decimal place. If you put STYLE=abbreviated, the values of the similarity matrix are printed as single digits with no spaces, the digit being the 10's value of the similarity as a percentage. In both cases, though, the actual similarities in the range 0-1 are stored in the similarity matrix itself. The PERMUTATION option allows you to specify a variate with values corresponding to the order in which you want the rows of the similarity matrix to be printed. The reordering of the rows is most effective when the permutation arises from a hierarchical clustering and corresponds to the dendrogram order (6.19.1).

Example 6.1.2

```
  2  " Data from Observers Book of Automobiles 1986
 -3
 -4    16 Italian cars and 12 measurements/characteristics
 -5
 -6     1.  engine capacity        c.c.        Engcc
 -7     2.  number of cylinders                Ncyl
 -8     3.  fuel tank              litres      Tankl
 -9     4.  unladen weight         kg          Weight
-10     5.  length                 cm          Length
-11     6.  width                  cm          Width
-12     7.  height                 cm          Height
-13     8.  wheelbase              cm          Wbase
-14     9.  top speed              kph         Tspeed
-15    10.  time to 100kph         secs        Stst
-16    11.  carburettor/inj/diesel 1/2/3       Carb
-17    12.  front/rear wheel drive 1/2         Drive
-18  "
 19  UNITS [NVALUES=16]
 20  VARIATE Engcc,Ncyl,Tankl,Weight,Length,Width,Height,Wbase,Tspeed,Stst,\
 21    Carb,Drive,Vct[1...3]
 22  POINTER Cd; VALUES=!P(Engcc,Ncyl,Tankl,Weight,Length, \
 23    Width,Height,Wbase,Tspeed,Stst)
 24  READ [PRINT=errors] #Cd,Carb,Drive
 41  TEXT [VALUES=Estate,'Arna1.5','Alfa2.5',Mondialqc,\
 42    Testarossa,Croma,Panda,Regatta,Regattad,Uno,\
 43    X19,Contach,Delta,Thema,Y10,Spider] Carname
 44  FACTOR [Carname; LEVELS=16] Fcar; VALUES=!(1...16)
 45  SYMMETRICMATRIX [ROWS=Carname] Carsim
 46  " Form similarity matrix between cars."
 47  FSIMILARITY [SIMILARITY=Carsim; PRINT=similarities] #Cd,Carb,Drive; \
 48    TEST=4(cityblock),4(Euclidean),2(cityblock),2(simplematch)

Similarity matrix: Carsim
-------------------------

Estate      1    ----
Arna1.5     2    97.6  ----
Alfa2.5     3    81.5  80.0  ----
Mondialqc   4    57.6  54.6  76.2  ----
Testarossa  5    38.9  35.5  56.1  82.7  ----
Croma       6    79.4  77.6  76.2  76.8  56.7  ----
Panda       7    82.0  85.5  61.8  29.6  10.3  54.6  ----
Regatta     8    98.1  96.9  82.3  58.9  39.4  82.0  80.1  ----
Regattad    9    83.9  82.2  67.5  52.5  32.9  75.6  75.6  84.4  ----
```

```
Uno         10    88.4  90.9  69.3  40.9  21.1  65.0  96.0  86.6  81.5  ----
X19         11    87.0  85.8  82.8  57.8  42.5  60.2  75.8  83.6  70.0  78.7
Contach     12    46.2  43.2  61.8  70.9  88.5  44.0  21.5  45.8  30.9  30.2
Delta       13    95.9  95.1  83.7  58.5  39.3  81.4  81.3  95.9  80.6  87.1
Thema       14    78.5  76.5  75.4  77.4  57.1  98.7  52.9  81.1  74.8  63.7
Y10         15    89.5  92.4  69.2  37.7  19.0  62.1  92.9  87.5  74.0  92.5
Spider      16    77.8  76.1  82.3  74.6  58.3  78.2  62.9  77.2  70.6  67.3
                     1     2     3     4     5     6     7     8     9    10


X19         11    ----
Contach     12    53.1  ----
Delta       13    82.6  47.1  ----
Thema       14    59.1  44.0  80.2  ----
Y10         15    83.0  30.5  88.4  60.5  ----
Spider      16    86.0  50.6  78.8  77.2  70.4  ----
                    11    12    13    14    15    16
```

You use the GROUPS option to specify a partition of the units into two groups, by giving a factor with two levels. The units with level 1 of the factor correspond to the rows of the matrix, while the units with level 2 correspond to the columns.

The UNITS option allows you to label the rows of the output similarity matrix if the variates of the DATA parameter do not have any unit labels, or if you want to use different labels from those labelling the units of the variates. This labelling also applies to the rows and columns of a matrix of similarities between group elements.

6.1.3 Forming similarities between groups: the HREDUCE directive

Sometimes you may want to regard an n-by-n similarity matrix as being partitioned into b-by-b rectangular blocks. For example, the cars in 6.1.2 could be classified by their manufacturer. You might then want to form a reduced matrix of similarities, between the different manufacturers instead of between the individual members of the full set of cars. Another example is when there are b soil samples, each with information recorded on several soil horizons, which may be different in the different samples. The n sampling units are the full set of horizons that have been observed for the soil samples. The similarity matrix can be computed for these in the usual way (6.1.2), but you may be more interested in obtaining a reduced similarity matrix between the b soil samples. To do this you have to arrange for each of the b^2 blocks of the full matrix to be replaced by a single value. Each diagonal block must be replaced by unity. Several possibilities exist for replacing the off-diagonal blocks: e.g. the maximum, minimum, or mean similarity within the block. Alternatively you could take the view that at least the first horizons of each of two soil samples should agree; you would then replace the block by its first value. Rayner (1966) suggested a more complex method, known as the *zigzag* method, which recognized that certain horizons might be absent from some soil samples. This leads to finding successive optimal matches, conditional on the constraint that one horizon cannot match a horizon that has already been assigned to a higher level; after finding these optima, an average is taken for each horizon. Again Genstat produces a symmetric similarity matrix, which you can use subsequently for matrix operations or in the appropriate multivariate directives.

HREDUCE directive

Forms a reduced similarity matrix (referring to the GROUPS instead of the original units).

Options

PRINT = *string token*	Printed output required (similarities); default * i.e. no printing
METHOD = *string token*	Method used to form the reduced similarity matrix (first, last, mean, minimum, maximum,

zigzag); default firs

Parameters

SIMILARITY = *symmetric matrices*	Input similarity matrix
REDUCEDSIMILARITY = *symmetric matrices*	Output (reduced) similarity matrix
GROUPS = *factors*	Factor defining the groups
PERMUTATION = *variates*	Permutation order of units (for METHOD = firs, last or zigz)

The SIMILARITY parameter specifies the similarity matrix for the full set of *n* observations; this must be present and have values. The REDUCEDSIMILARITY parameter specifies an identifier for the reduced similarity matrix, of order *b*; this will be declared implicitly if you have not declared it already. The factor that defines the classification of the units into groups must be specified by the GROUPS parameter. The units can be in any order, so that for example the units of the first group need not be all together nor given first. The labels of the factor label the reduced similarity matrix.

The PERMUTATION parameter, if present, must specify a variate. It defines the ordering of samples within each group, and so must be specified for methods first, last, and zigzag. Within each group, the unit with the lowest value of the permutation variate is taken to be the first sample, and so on. Genstat will, if necessary, use a default permutation of one up to the number of rows of the similarity matrix.

If you set option PRINT=similarities, the values of the reduced symmetric matrix are printed as percentages.

The METHOD option specifies how the reduced similarity matrix is to be formed. In Example 6.1.3, the similarity matrix for each car is reduced to a similarity matrix for each manufacturer as represented by the factor Maker. The METHOD option is set to mean. The resulting matrix is printed, and finally stored in the symmetric matrix Makersim.

Example 6.1.3

```
" Form reduced similarity matrix for makers."
FACTOR [LABELS=!t(Fiat,'Alfa Romeo',Lancia,Ferrari,Lamborghini,\
  Pinninfarina)] Maker; VALUES=!(2,2,2,4,4,1,1,1,1,1,1,5,3,3,3,6)
SYMMETRICMATRIX [ROWS=Maker] Makersim
HREDUCE [PRINT=similarities; METHOD=mean] Carsim; \
  REDUCEDSIMILARITY=Makersim; GROUPS=Maker
```

```
Similarity matrix reduced to groups defined by Maker,
using the mean similarity within each group
=====================================================

Reduced similarity matrix: Makersim
-----------------------------------

Fiat         1    ----
Alfa Romeo   2    82.1  ----
Lancia       3    79.5  84.0  ----
Ferrari      4    43.3  53.1  48.2  ----
Lamborghini  5    37.6  50.4  40.5  79.7  ----
Pinninfarina 6    73.7  78.7  75.5  66.5  50.6  ----
                     1     2     3     4     5     6
```

6.1.4 Forming associations using CALCULATE

An appropriate similarity coefficient can be calculated by FSIMILARITY (6.1.2) for most sets of data. However, many different coefficients of similarity, or distance, have been suggested (see, for example, Gower & Legendre 1986). FSIMILARITY does not cover all of these, but you will generally be able to form the others by using CALCULATE (1:4.1). Sometimes you may need to convert similarities to dissimilarities (distances), or vice versa. This can be done in many ways; the most common are $D=1-S$ and $D=\sqrt{(1-S)}$, but $D=-\log(S)$ can also be useful. So there are also situations where you may need to transform such matrices using CALCULATE. For example, by putting

```
FSIMILARITY [SIMILARITY=Smat] V[1...9]; TEST=Euclidean
```

the symmetric matrix Smat will contain similarities constructed from Euclidean squared distances standardized by the ranges of the variates. If you do not want standardization by range, Euclidean distances can be obtained from the PCO directive (6.10.1); but these may then have to be transformed to similarities, for example if you want to use hierarchical cluster analysis (6.19). If Smat has been obtained from the PCO directive, its values should be squared first, to get Euclidean squared distances, and then transformed to similarities:

```
CALCULATE Smat = Smat*Smat
& Smat = 1-Smat/MAX(Smat)
```

The FSIMILARITY directive allows variates of different types; for example, dichotomous variates (with values 0 or 1) can have the TEST parameter set to Jaccard or simplematching. Other variates with values on a continuous scale can have the TEST parameter set to cityblock or Euclidean. When both types of variates are present, the resulting similarities will be a weighted average of the component similarities. For example, with five dichotomous variates, Binary[1...5], and three continuous variates, Cont[1...3]

```
FSIMILARITY [SIMILARITY=Mixed] Binary[1...5],Cont[1...3]; \
  TEST=(Jaccard)5,(cityblock)3
```

will give the similarity matrix Mixed as a weighted average of the Jaccard similarity matrix constructed from Binary[1...5] and the city-block similarity matrix constructed from Cont[1...3]. If, instead of the city-block coefficient, you want to use the unstandardized Euclidean coefficient, you must construct this yourself, as shown above, and then do the averaging:

```
SYMMETRIC [ROWS=N] Jaccard,Euclid,Mixed
FSIMILARITY [SIMILARITY=Jaccard] Binary[1...5]; TEST=jaccard
PCO Cont[]; DISTANCES=Euclid
CALCULATE Euclid = Euclid*Euclid
& Euclid = 1-Euclid/MAX(Euclid)
& Mixed = (5*Jaccard+3*Euclid)/8
```

Gower (1985b) lists 15 different similarity coefficients that have been used for dichotomous variables. Of these, only the simple-matching and Jaccard coefficients can be formed directly with FSIMILARITY; these are the most commonly used. However, a further seven similarity coefficients can be formed using either, or both, of these two. For example, for the five variates Binary[1...5] the Czekanowski coefficient can be calculated from the Jaccard coefficient, using these statements:

```
FSIMILARITY [SIMILARITY=Jaccard] Binary[1...5];\
  TEST=jaccard
CALCULATE Czekanow = 2 * Jaccard / (1 + Jaccard)
```

Gower (1985b) gives details of the other relationships.

The city-block and Euclidean measures of distance are special cases of the Minkowski distance, which for some positive value of t is:

$$d_{ij} = \left[\sum_k \left(\frac{| x_{ik} - x_{jk} |}{r_k} \right)^t \right]^{1/t}$$

where r_k is usually the range of the kth variable. Although similarities derived from this distance cannot be formed with `FSIMILARITY` directly, the symmetric matrix `Minkwski` giving such similarities can be formed from the variates `X[1...p]` using these statements:

```
CALCULATE Minkwski=0
FOR Thisx=X[1...p]
  FSIMILARITY [SIMILARITY=Temp] Thisx; TEST=cityblock
  CALCULATE Minkwski = Minkwski+Temp**t
ENDFOR
CALCULATE Minkwski = EXP(LOG(Minkwski)/t)
```

6.1.5 Assessing the association between similarity matrices: the `MANTEL` procedure

`MANTEL` procedure

Assesses the association between similarity matrices (J.W. McNicol, E.I. Duff & D.A. Elston).

Options

`PRINT` = *string token*	Controls printed output (`test`); default `*` i.e. none
`METHOD` = *string token*	The type of metric by which to compare the distance matrices (`correlation`, `rankcorrelation`, `mantel`); default `corr`
`NPERMUTATIONS` = *scalar*	The number of permutations of the units in the second distance matrix `X` on which the significance of the correlation between `Y` and `X` is to be based; default 100

Parameters

`Y` = *symmetric matrices*	The first distance or similarity matrix: the order of the units of this matrix is held fixed
`X` = *symmetric matrices*	The second distance or similarity matrix: the rows of `X` are permuted to allow the significance of the correlation between `Y` and `X` to be assessed
`SEED` = *scalars*	Random number seed for the permutations; default set by `RANDOMIZE`
`M` = *scalars*	Association between `Y` and `X`
`MPERMUTED` = *variates*	Associations between `Y` and the permuted `X`'s
`CUPROB` = *scalars*	The proportion of `MPERMUTED` values greater than or equal to `M`
`YOFFDIAGONAL` = *variates*	Variate to save the off-diagonal elements of the distance/similarity matrix `Y`
`XOFFDIAGONAL` = *variates*	Variate to save the off-diagonal elements of the distance/similarity matrix `X`

The extent to which two similarity/distance matrices describe the same relationships among the units can be measured by comparing their off-diagonal elements. The metrics to be used can be selected using the `METHOD` option: product-moment correlation (`correlation`), rank correlation (`rankcorrelation`) and `SUM(X*Y)` (`Mantel`). The last of these is the metric originally proposed by Mantel (1967). If the metric `rankcorrelation` is selected, the data are restricted to non-missing units and Spearman's rank correlation is used.

The significance of the association is assessed by a permutation test. The rows/columns of the second matrix are permuted at random and the association is recalculated for each permutation. Significance is estimated by the percentage of the permutations with association less/more than or equal to that of the original association.

If the number of random permutations, specified by the NPERMUTATIONS option, is set to a number greater than or equal to the total number of distinct permutations $d!$, where d is the dimension of the symmetric matrices, the full randomization test is implemented. Otherwise the rows/columns of the second matrix are permuted at random without regard to the duplication of specific permutations. By default, 100 permutations are done. The SEED parameter can supply a seed for the random numbers used to generate the random permutations. By default SEED=0, so the random numbers will continue any existing sequence, used earlier in the Genstat program, or be initialised by the RANDOMIZE directive.

The two matrices to be compared are specified by the Y and X parameters. The M parameter allows the value of the statistic for the original matrices to be saved, the MPERMUTED parameter saves the values from the permuted matrices, and the CUPROB parameter saves the proportion of the permuted associations that are greater than the association between the original matrices. The off-diagonal elements of the matrices, on which the calculations are based, can be saved as variates using the XOFFDIAGONAL and YOFFDIAGONAL parameters.

The PRINT option can be set to test to print the values of M and CUPROB; by default there is no output.

Example 6.1.5

```
 2  " Data from Tables 1.1, 1.2 and 1.3 of Manly (1991)."
 3  SYMMETRIC [ROWS=8] Assoc,Dist1,Dist2
 4  READ      Assoc

  Identifier   Minimum      Mean   Maximum    Values   Missing
       Assoc   -0.1600    0.3533     1.000        36         0

13  READ      Dist1

  Identifier   Minimum      Mean   Maximum    Values   Missing
       Dist1    0.0000     2.000     5.000        36         0

22  READ      Dist2

  Identifier   Minimum      Mean   Maximum    Values   Missing
       Dist2    0.0000     1.389     4.000        36         0

31  PRINT     Assoc,Dist1,Dist2; FIELD=7; DECIMALS=2

    Assoc

 1   1.00
 2   0.30   1.00
 3   0.14   0.50   1.00
 4   0.23   0.50   0.54   1.00
 5   0.30   0.40   0.50   0.61   1.00
 6  -0.04   0.04   0.11   0.03   0.15   1.00
 7   0.02   0.09   0.14  -0.16   0.11   0.14   1.00
 8  -0.09  -0.06   0.05  -0.16   0.03  -0.06   0.36   1.00
        1      2      3      4      5      6      7      8

    Dist1

 1   0.00
 2   1.00   0.00
 3   2.00   1.00   0.00
 4   1.00   2.00   3.00   0.00
 5   2.00   3.00   4.00   1.00   0.00
 6   3.00   4.00   5.00   2.00   1.00   0.00
 7   2.00   3.00   4.00   3.00   4.00   5.00   0.00
 8   1.00   2.00   3.00   2.00   3.00   4.00   1.00   0.00
```

```
          1      2      3      4      5      6      7      8

      Dist2

   1   0.00
   2   1.00   0.00
   3   2.00   1.00   0.00
   4   1.00   1.00   1.00   0.00
   5   2.00   1.00   1.00   1.00   0.00
   6   3.00   2.00   2.00   2.00   1.00   0.00
   7   2.00   1.00   2.00   2.00   2.00   3.00   0.00
   8   1.00   2.00   3.00   2.00   3.00   4.00   1.00   0.00
          1      2      3      4      5      6      7      8

  32  MANTEL    [PRINT=test; NPERMUTATIONS=25] Y=Assoc; X=Dist1; SEED=615023

Mantel test based on product-moment correlations
================================================

  25 permutations performed
Association between the original matrices:                  -0.2170
Percent permutations with equal or greater association:       84.00

  33  MANTEL    [PRINT=test; NPERMUTATIONS=25] Y=Assoc; X=Dist2; SEED=712378

Mantel test based on product-moment correlations
================================================

  25 permutations performed
Association between the original matrices:                  -0.6054
Percent permutations with equal or greater association:      100.00
```

6.1.6 Nonparametric analysis of similarities: the ECANOSIM procedure

ECANOSIM procedure

Performs an analysis of similarities i.e. *ANOSIM* (D.A. Murray).

Options

PRINT = *string token*	Controls printed output (test); default test
PLOT = *string token*	Type of plot (boxplot, histogram); default hist
NTIMES = *scalar*	Number of permutations to make; default 999
BLOCKS = *factor*	Factor specifying groups for a stratified test; default * i.e. none
SEED = *scalar*	Seed for the random number generator used to make the permutations; default 0 continues from the previous generation or (if none) initializes the seed automatically

Parameters

DATA = *symmetric matrices*	Similarity matrix
GROUPS = *factors*	Specify the different groups for each matrix
STATISTIC = *scalars*	Save the *R* statistics
PROBABILITY = *scalars*	Save the probabilities

Analysis of similarities (*ANOSIM*) is a nonparametric method to test whether there is a significant difference between two or more groups of sampling units (Clarke 1993). The method performs a permutation test based on the ranks of measures of similarity between sampling units. The data should be supplied as a similarity matrix using the DATA parameter. The GROUPS parameter specifies a factor containing the groups for each corresponding row of the similarity matrix.

The *ANOSIM* statistic *R* is calculated by the difference of the between-group (r_b) and within-group (r_w) mean rank similarities:

$$R = (\text{mean}(r_b) - \text{mean}(r_w)) / (n \times (n - 1) / 4)$$

The denominator is chosen so the *R* lies in the range (−1, 1) where 0 represents no difference between the groups. The similarites are ranked where a rank of 1 corresponds to the highest similarity.

The statistical significance of the *R* statistic is assessed by a permutation test. `ECANOSIM` performs 999 random permutations (made using a default seed), and calculates the *R* statistic for each permutation. The probability for the *R* statistic is then determined from its distribution over the randomly permuted datasets. The `NTIMES` option of `ECANOSIM` allows you to request another number of permutations, and the `SEED` option allows you to specify another seed. For designs with no blocking `ECANOSIM` checks whether `NTIMES` is greater than the number of possible permutations available for the data set. If so, `ECANOSIM` does an exact test instead, which uses each possible permutation once.

The `histogram` setting of the `PLOT` option can be used to produce a distribution of the *R* values. *ANOSIM* assumes under the null hypothesis that distances within groups are smaller than those between groups, and that the ranked dissimilarities within groups have equal median and range. The `boxplot` setting for the `PLOT` option can be used to help check these assumptions.

The *R* statistic can be saved using the `STATISTIC` parameter, and the probability can be saved using the `PROBABILITY` parameter. By default the the *R* statistic and probability are printed, but this can be suppressed by setting option `PRINT=*`.

The analysis of similarities is illustrated in Example 6.1.6 and Figure 6.1.6.

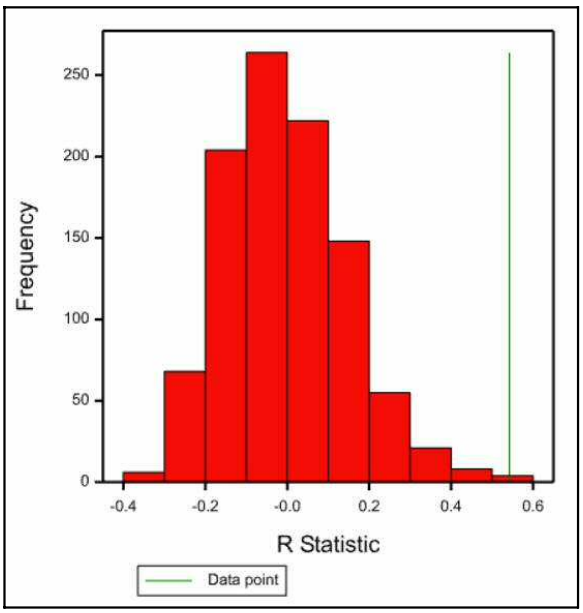


Figure 6.1.6

Example 6.1.6

```
  2  FACTOR      [LEVELS=5; VALUES=1,1,1,2,2,2,3,3,3,4,5,5,5,5] Groups
  3  VARIATE     [NVALUES=14] Data[1...5]; VALUES=!(5(1),9(0)),\
  4              !(8(1),4(0),1,0),!(0,4(1),0,4(1),0,0,1,0),\
  5              !(0,5(1),0,3(1),4(0)),!(3(0),3(1),0,1,1,0,4(1))
  6  FSIMILARITY [SIMILARITY=Sim] Data[]; TEST=jaccard
  7  ECANOSIM    [SEED=26351; PLOT=histogram] Sim; GROUPS=Groups

Analysis of Similarities (ANOSIM)
---------------------------------

R Statistic: 0.542
Probability: 0.003
Based on 999 random permutations
```

6.2 Principal components analysis

Principal components analysis finds linear combinations of a set of variates that maximize the variation contained within them, thereby displaying most of the original variability in a smaller number of dimensions. Principal components analysis operates on sums of squares and products, or a correlation matrix, or a matrix of variances and covariances, formed from the variates.

6.2.1 The PCP directive

PCP directive

Performs principal components analysis.

Options

PRINT = *string tokens*	Printed output required (loadings, roots, residuals, scores, tests); default * i.e. no printing
NROOTS = *scalar*	Number of latent roots for printed output; default * requests them all to be printed
SMALLEST = *string token*	Whether to print the smallest roots instead of the largest (yes, no); default no
METHOD = *string token*	Whether to use sums of squares, correlations or variances and covariances (ssp, correlation, vcovariance, variancecovariance); default ssp

Parameters

DATA = *pointers* or *matrices* or *SSPMs*	Pointer of variates forming the data matrix, or matrix storing the variate values by columns, or SSPM giving their sums of squares and products (or correlations) etc
LRV = *LRVs*	To store the principal component loadings, roots and trace from each analysis
SSPM = *SSPMs*	To store the computed sum-of-squares-and-products or correlation matrix
SCORES = *matrices*	To store the principal component scores
RESIDUALS = *matrices* or *variates*	To store residuals from the dimensions fitted in the analysis (i.e. number of columns of the SCORES matrix, or as defined by the NROOTS option)
SAVE = *pointers*	Saves details of the analysis; if unset, an unnamed save structure is saved automatically (and this can be accessed using the GET directive)

You supply the input for PCP using the first parameter; this list may have more than one entry, in which case Genstat repeats the analysis for each of the input structures. Instead of supplying an SSPM, you can supply a pointer containing the set of variates, or a matrix storing the variate values by columns. Genstat will then calculate the sums of squares and products, or correlations, or variances and covariances for the analysis (see option METHOD below).

For example, these two forms of input are equivalent:

```
SSPM [TERMS=Height,Length,Width,Weight] S
FSSPM S
PCP [PRINT=roots] S
```

and

```
PCP [PRINT=roots] !P(Height,Length,Width,Weight)
```

But the first form does mean that you have the sums of squares and products available for later use, in the SSPM S. Here the pointer is unnamed (1.6.3). But you may wish to use a named pointer. For example:

```
POINTER [VALUES=Height,Length,Width,Weight] Dmat
PCP [PRINT=roots] Dmat
```

By default the PCP directive does not print any results: you use the PRINT option to specify what output you require. The printed output is in five sections, each with a corresponding setting, as illustrated in the examples below.

The columns of the matrices of principal component loadings and scores correspond to the latent roots. Each latent root corresponds to a single dimension, and gives the variability of the scores in that dimension. The loadings give the linear coefficients of the variables that are used to construct the scores in each dimension. Example 6.2.1a shows a principal components analysis of four variates of length 12.

Example 6.2.1a

```
  2  UNITS   [NVALUES=12]
  3  POINTER [VALUES=Height,Length,Width,Weight] Dmat
  4  READ    [PRINT=data,errors,summary] Dmat[]

  5  4.1 5.2 1.2 3.1 4.2 1.5 3.2 5.6 2.3 0.2 0.1 0.2
  6  6.2 4.1 4.1 4.1 2.3 6.2 6.3 5.1 0.2 0.9 4.9 7.3
  7  10.1 5.6 3.2 9.4 1.2 9.8 1.0 1.0 6.1 9.7 1.0 3.7
  8  6.1 9.6 9.7 5.5 2.3 5.0 9.4 8.1 4.5 4.9 0.3 1.8 :

 Identifier   Minimum      Mean   Maximum    Values   Missing
     Height    0.2000     4.133     10.10        12         0
     Length    0.2000     5.225     9.800        12         0
      Width    0.1000     3.700     9.700        12         0
     Weight    0.2000     4.575     9.400        12         0

  9  PCP [PRINT=roots,scores,loadings,tests] Dmat

Principal components analysis
=============================

Latent roots
------------

          1          2          3          4
      181.8      130.2       82.5       18.5


Percentage variation
--------------------

          1          2          3          4
      44.01      31.52      19.98       4.49


Trace
-----

     413.2


Latent vectors (loadings)
-------------------------

                    1          2          3          4
    Height    0.21529    0.37981    0.78747    0.43506
    Length    0.25623    0.86524   -0.34389   -0.25970
     Width    0.74104   -0.21726   -0.37937    0.50964
    Weight    0.58211   -0.24474    0.34308   -0.69537
```

```
Significance tests for equality of final k roots
------------------------------------------------

      k  Chi-square      df
      2        4.52       2
      3        7.35       5
      4       10.11       9


Principal component scores
--------------------------

                    1          2          3          4
           1     -2.725      0.870      0.425     -0.256
           2     -0.714     -3.340      1.875      0.029
           3     -6.897     -3.191      0.149      1.715
           4      0.177     -0.159      1.700      1.725
           5      2.087     -0.546     -2.585     -0.091
           6      0.521     -6.164     -1.130     -1.871
           7      3.819      1.518      6.415     -1.112
           8     -3.541      4.306     -4.085     -1.354
           9     -0.940      5.420      0.734     -1.074
          10      6.529      3.002     -1.915      2.134
          11      5.824     -2.992     -2.319     -0.285
          12     -4.139      1.276      0.738      0.441
```

The significance tests are for equality of the k smallest roots: l_i (i = 1, 2, ... k). The test statistic is

$$n - \frac{(2p+11)}{6} \left[\log\left(1/k \sum_{i>k} l_i \right) - 1/k \sum_{i>k} \log l_i \right]$$

where n is the number of units and p is the number of variables. Asymptotically, the statistics have a chi-square distribution with $(k+2)(k-1)/2$ degrees of freedom. If any latent roots are zero, Genstat excludes them from the calculation of the test statistic; the effective value of p is reduced accordingly.

If you omit the NROOTS option, Genstat prints by default the results corresponding to all the latent roots. The number of latent roots is the number of variates involved in the input to PCP. The NROOTS option allows you to print only part of the results, corresponding to the first or last r latent roots. You may then want to print the residuals. Example 6.2.1b prints the results corresponding to the first two latent roots; the residuals are formed from the remaining two columns of scores.

Example 6.2.1b

```
  10  PCP [PRINT=scores,residuals; NROOTS=2] Dmat

Principal components analysis
=============================

Principal component scores
--------------------------

                    1          2
           1     -2.725      0.870
           2     -0.714     -3.340
           3     -6.897     -3.191
           4      0.177     -0.159
           5      2.087     -0.546
           6      0.521     -6.164
           7      3.819      1.518
           8     -3.541      4.306
           9     -0.940      5.420
          10      6.529      3.002
```

```
          11       5.824      -2.992
          12      -4.139       1.276


Residuals
---------

           1       0.496
           2       1.875
           3       1.721
           4       2.422
           5       2.587
           6       2.186
           7       6.510
           8       4.304
           9       1.301
          10       2.867
          11       2.337
          12       0.860
```

To print results corresponding to the r smallest latent roots, you must set option `NROOTS` to r and option `SMALLEST` to `yes`. Now if residuals are printed they will be formed from the scores corresponding to the largest roots. The `NROOTS` and `SMALLEST` options apply to the latent roots and vectors, the principal component scores and the residuals. So you cannot print directly, for example, the first two columns of scores and the last three columns of loadings. This is rarely required but, if necessary, it can be done by saving the relevant results and printing them separately.

In Example 6.2.1c the three smallest roots are printed, together with the residuals. These correspond to the first column of scores, and can be compared with the scores in Example 6.2.1a. You can see that all the residuals are positive: this is because residuals from multivariate analyses generally occupy several dimensions, so they represent distances in multidimensional space and signs cannot be attached to them.

Example 6.2.1c

```
  11  PCP [PRINT=roots,residuals; NROOTS=3; SMALLEST=yes] Dmat

Principal components analysis
=============================

Latent roots
------------

             1           2           3
        130.23       82.54       18.55


Percentage variation
--------------------

             1           2           3
         31.52       19.98        4.49


Trace
-----

       413.2


Residuals
---------

           1       2.725
           2       0.714
           3       6.897
```

```
 4       0.177
 5       2.087
 6       0.521
 7       3.819
 8       3.541
 9       0.940
10       6.529
11       5.824
12       4.139
```

By default, the PCP directive operates on the SSPM but you can set the METHOD option to correlations to operate on a derived matrix of correlations, as shown in Example 6.2.1d, or to vcovariance (or its synonym variancecovariance) to use variances and covariances. Note that when correlations are analysed the significance-test statistics no longer have asymptotic chi-square distributions.

The LRV parameter allows you to save the principal component loadings, the latent roots and their sum (the trace) in an LRV structure, while the SCORES parameter saves the principal component scores in a matrix. If you have declared the LRV already, its number of rows must be the same as the number of variates supplied in an input pointer or implied by an input SSPM. The number of rows of the SCORES matrix, if previously declared, must be equal to the number of units.

The number of columns of the LRV and of the SCORES matrix corresponds to the number of dimensions to be saved from the analysis, and this must be the same for both of them. If the structures have been declared already, Genstat will take the larger of the numbers of columns declared for either, and declare (or redeclare) the other one to match. If neither has been declared and option SMALLEST retains the default setting no, Genstat takes the number of columns from the setting of the NROOTS option. Otherwise, Genstat saves results for the full set of dimensions. The trace saved as the third component of the LRV structure, however, will contain the sums of all the latent roots, whether or not they have all been saved. Procedure LRVSCREE can be used to produce a "scree" diagram which can be helpful in deciding how many dimensions to save; see Section 6.2.2.

The SSPM parameter can save the SSPM structure used for the analysis. A particularly convenient instance is when you have supplied an SSPM structure as input but, for example, have set METHOD=correlation: the SSPM that is saved will then contain correlations instead of sums of squares and products.

The RESIDUALS parameter allows you to save the principal component residuals, in a matrix with number of rows equal to the number of units and one column. If the latent roots and vectors (loadings) are saved from the analysis, the residuals will correspond to the dimensions not saved; the same applies if you save scores. If neither the LRV nor scores are saved, the saved residuals will correspond to the smallest latent roots not printed.

Example 6.2.1d

```
 12  LRV [ROWS=Dmat; COLUMNS=2] Latent
 13  SSPM [TERMS=Dmat[]] Corrmat
 14  MATRIX [ROWS=12; COLUMNS=1] Res
 15  PCP [PRINT=roots,scores,tests; METHOD=correlation] Dmat; \
 16    LRV=Latent; SSPM=Corrmat; RESIDUALS=Res

Principal components analysis
=============================

Latent roots
------------

             1           2           3           4
         1.748       1.209       0.855       0.188
```

```
Percentage variation
--------------------

             1           2           3           4
         43.70       30.23       21.37        4.70


Trace
-----

       4.000


Significance tests for equality of final k roots
------------------------------------------------

* Note: correlation matrix used - test statistics are not asymptotically
chi-square.

      k  Chi-square      df
      2        5.78       2
      3        8.55       5
      4       11.87       9


Principal component scores
--------------------------

                        1           2           3           4
            1     -0.8216      0.3398      0.1748     -0.1050
            2     -0.0629     -0.8009      0.8807      0.0190
            3     -2.2460     -0.7633      0.6032      0.5641
            4      0.1569      0.2320      0.5762      0.5669
            5      0.4327     -0.4906     -0.8664     -0.0032
            6      0.1126     -2.1210      0.0571     -0.5723
            7      1.8402      1.0603      1.7467     -0.4016
            8     -1.4535      0.8414     -1.5076     -0.4720
            9     -0.1915      1.6947     -0.1622     -0.4031
           10      1.8470      0.7249     -1.0655      0.7278
           11      1.6518     -1.2754     -0.7506     -0.0374
           12     -1.2657      0.5581      0.3136      0.1169

  17  PRINT Latent[],Res

              Latent['Vectors']
                        1           2

       Height      0.3476      0.6121
       Length      0.1981      0.6896
        Width      0.6201     -0.3067
       Weight      0.6749     -0.2359


                          1           2
Latent['Roots']       1.748       1.209


Latent['Trace']       4.000

                      Res
                        1

            1      0.2039
            2      0.8809
            3      0.8259
            4      0.8083
            5      0.8664
            6      0.5752
            7      1.7923
            8      1.5798
            9      0.4345
           10      1.2904
           11      0.7516
```

```
12      0.3347
```

The SAVE parameter can supply a pointer to save a multivariate save structure contining all the details of the analysis. If this is unset, an unnamed save structure is saved automatically (and this can be accessed using the GET directive). Alternatively, you can set SAVE=* to prevent any save structure being formed if, for example, you have a very large data set and want to avoid committing the storage space.

If the variables used to form the SSPM structure are restricted, then the analysis will be subject to that restriction. Similarly, if a pointer to a set of variates is used as input to PCP, then any restriction on the variates will be taken into account by the analysis. If you want principal component scores or residuals to be printed or saved from the analysis, the original data must be available. The matrices to save such results must have been declared with as many rows as the variates have values, ignoring the restriction. You can calculate the analysis from one subset of units, but calculate the scores and residuals for all the units, by using as input to PCP an SSPM structure formed using a weight variate with zeros for the excluded sampling units and unity for those to be included. For example, to exclude a known set of outliers from an analysis, but to print scores for them, these statements could be used:

```
POINTER [NVALUES=5] V
FACTOR [LABELS=!T(No,Yes)] Outlier
READ [CHANNEL=2] Outlier,V[]
CALCULATE Wt = Outlier .IN. 'No'
SSPM [TERMS=V] S
FSSPM [WEIGHT=Wt] S
PCP [PRINT=scores] S
```

Principal component regression is provided by procedure RIDGE (6.7).

6.2.2 Scree diagrams of latent roots: the LRVSCREE procedure

LRVSCREE procedure

Prints a scree diagram and/or a difference table of latent roots (P.G.N. Digby).

Options

PRINT = *string tokens*	Printed output (scree, differences); default scre
PLOT = *string token*	What to plot in high-resolution graphics (scree); default scre
TITLE = *text*	Title for the graph; default * i.e. none
WINDOW = *scalar*	Window to use for the graph; default 1

Parameters

ROOTS = *LRVs* or *any numerical structures*	Latent roots to be displayed; if an LRV is supplied the trace will also be extracted from it
TRACE = *scalars*	Supplies or saves the total of the latent roots
DIFFERENCES = *pointers*	Contains 3 variates to save the difference table

Procedure LRVSCREE displays a set of latent roots in a convenient form. The input to the procedure is a set of latent roots (ROOTS), either as an LRV or any structure with numerical values. Optionally a scalar (TRACE) can be specified, either to supply or to save the total of the latent roots.

Printed output is controlled by the PRINT option. The setting scree produces a scree diagram, annotated with the latent roots on their original scale and expressed both as per-thousandths of

the total and as cumulated per-thousandths. The setting `differences` prints these quantities as a table, together with the first three differences among the per-thousandth values; i.e. the first difference column gives the differences from each per-thousandth to the next, the second difference column gives differences among the first-difference values, and so on. Large first-difference values indicate latent roots ocurring prior to large declines in the scree diagram. Large second and third differences mark the locations of series of two or more latent roots of similar magnitude, which can be thought of as plateaus on the scree diagram. Large positive, or negative, second differences indicate the first, or last, latent root of a plateau. Large negative third differences occur at the last latent root of one plateau that is followed by another plateau. See the example for illustration.

By default the scree diagram is also plotted in high-resolution graphics but this can be suppressed by setting option `PLOT=*`. The `TITLE` option can supply a title for the plot, and the `WINDOW` option specifies which window is used (by default window 1).

The `DIFFERENCES` parameter allows a pointer to be specified to contain three variates storing the columns of the difference table.

Example 6.2.2 shows a scree diagram for the latent roots from the principal components analysis in Example 6.2.1a. The resulting graph is in Figure 6.2.2. Here there are only four roots, so the diagram is not especially informative. Example 6.10.1a shows the use of `LRVSCREE` following a principal coordinates analysis with ten roots.

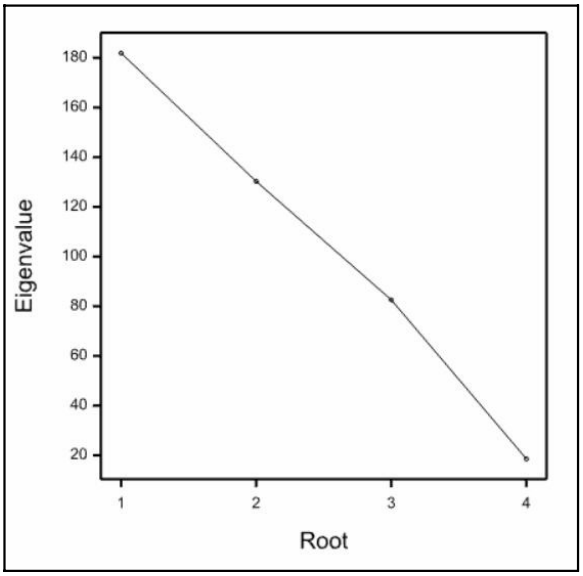


Figure 6.2.2

Example 6.2.2

```
 18  PCP [PRINT=*] Dmat; LRV=Lrv
 19  LRVSCREE Lrv

 No     Root   %%  Cum   %  Scree Diagram (* represents 2%)

  1    181.8  440  440  44 **********************
  2    130.2  315  755  32 ****************
  3     82.5  200  955  20 **********
  4     18.5   45 1000   4 **

Scale:  1 asterisk represents 2 units.
```

6.3 Canonical variates analysis

The CVA directive, for canonical variates analysis, operates on a within-group SSPM (6.1.1). This structure contains information on the within-group sums of squares and products, pooled over all the groups; it also contains the group means and group sizes, from which Genstat can derive the between-group sums of squares and products. The directive finds linear combinations of the original variables that maximize the ratio of between-group to within-group variation, thereby giving functions of the original variables that can be used to discriminate between the groups. The squares of the distances between group means are Mahalanobis D^2 statistics when all the dimensions are used; otherwise they are approximations. You can form exact Mahalanobis distances with the PCO directive (6.10.1).

6.3.1 The CVA directive

CVA directive

Performs canonical variates analysis.

Options

PRINT = *string tokens*	Printed output required (roots, loadings, means, residuals, distances, tests); default * i.e. no printing
NROOTS = *scalar*	Number of latent roots for printed output; default * requests them all to be printed
SMALLEST = *string token*	Whether to print the smallest roots instead of the largest (yes, no); default no

Parameters

WSSPM = *SSPMs*	Within-group sums of squares and products, means etc (input for the analyses)
LRV = *LRVs*	Saves loadings, roots and trace from each analysis
SCORES = *matrices*	Saves canonical variate means
RESIDUALS = *matrices*	Saves distances of the means from the dimensions fitted in each analysis
DISTANCES = *symmetric matrices*	Saves inter-group-mean Mahalanobis distances
ADJUSTMENTS = *matrices*	Saves the adjustment terms
SAVE = *pointers*	Saves details of the analysis; if unset, an unnamed save structure is saved automatically (and this can be accessed using the GET directive)

You specify the input for CVA using its first parameter, WSSPM, this may contain a list of structures, in which case Genstat repeats the analysis for each of them. The input must be an SSPM structure, declared with the GROUPS option of the SSPM directive (6.1.1) set to a factor giving the grouping of the units. If the variates used to form this SSPM structure are restricted, then the SSPM is restricted in the same way, and so the CVA directive takes account of the restriction. The other four parameters can be used to save the results.

The three options of the CVA directive control the printed output. By default there is no printed output, and so you should set the PRINT option to indicate which sections you want.

Example 6.3.1a uses the within-group SSPM formed in Example 6.1.1. This is based on data from Doran & Hodson (1975) who gave some measurements made on 28 brooches found at the archaeological site of the cemetery at Munsingen. Seven of these variables are used in the example, and have been transformed by taking logarithms. For a grouping of the 28 brooches

into four groups (formed by the CLUSTER directive in Example 6.20.1 below), canonical variates analysis is used to determine possible differences among the groups, and which variables contribute to such differences.

Example 6.3.1a

```
  37  CVA [PRINT=roots,loadings,means,tests] WSSPM=W

Canonical variates analysis
===========================

Latent roots
------------

             1           2           3
         4.543       3.777       2.537


Percentage variation
--------------------

             1           2           3
         41.85       34.79       23.37


Trace
-----

       10.86


Latent vectors (loadings)
-------------------------

                        1           2           3
     Foot_lth      -1.130       2.656       3.397
       Bow_ht       0.633      -1.631       4.799
     Coil_dia      -3.501       1.708       1.450
     Elem_dia       2.669       0.623      -2.802
     Bow_wdth       3.468       0.758       0.757
     Bow_thck      -1.859       2.028      -2.478
       Length       1.279       0.110      -3.598


Significance tests for dimensionality greater than k
----------------------------------------------------

      k     Chi-square        df
      0          97.60        21
      1          60.78        12
      2          27.16         5


Canonical variate means
-----------------------

                        1           2           3
           1      -2.967      -1.998       0.613
           2       0.825      -0.122      -1.584
           3      -1.254       3.545       0.825
           4       2.835      -0.856       2.241


Adjustment terms
----------------

                        1           2           3
           1        8.40       19.90        1.94
```

The CVA directive (line 37) specifies that the latent roots, the vectors (loadings), and the means of the canonical variate groups are to be printed, together with values for the significance tests for the latent roots that indicate the number of dimensions required.

If there are g groups, at most $g-1$ independent combinations of the variables can be found to discriminate amongst them. However, if there are fewer than $g-1$ variables, v say, then at most v independent combinations can be calculated. Thus there will be at most $\min(g-1, v)$ non-zero latent roots, with associated loadings and canonical variate scores for the group means. In the example above $\min(g-1, v)$ is 3.

The significance tests that are printed are for a significant dimensionality greater than k, that is for the joint significance of the first, second, ..., $(k+1)$th latent roots. This test is printed for $k=0, 1, \ldots \min(g-1, v)-1$. If the test is non-significant for $k=r$, then the values of chi-square for $k>r$ should be ignored as the indication is that the remaining dimensions have no interesting structure. The test statistic (Bartlett 1938) is

$$[\, n - g - \tfrac{1}{2}(v - g)\,][\, \sum_{i} \log(\, l_i + 1\,) - \sum_{i>k} \log(\, l_i + 1\,)\,]$$

which is asymptotically distributed as chi-square with $(v-k)\times(g-k-1)$ degrees of freedom. Here n is the number of units, g is the number of groups, v is the number of variables, and l_i is the ith latent root. If the coefficient $[n-g-\frac{1}{2}(v-g)]$ is less than zero, there are too few units for the statistics to be calculated and a message is printed to this effect. In any case, the tests should be treated with caution unless $n-g$ is very much larger than v.

The latent vectors, or loadings, are scaled in such a way that the average within-group variability in each canonical variate dimension is 1: thus the within-group variation is equally represented in each dimension. Since the latent roots are the successive maxima of the ratio of between-group to within-group variation, loadings corresponding to roots less than 1 are for dimensions in the canonical variate space that exhibit more within-group variation than between-group variation. In the example, all three roots are greater than 1, suggesting that differences between the four groups exist in all three dimensions; this is in accordance with the significance tests, which indicate a dimensionality greater than 2. It may not be easy to interpret the latent vectors but, for example, the second latent vector here contrasts the second variable (the height of the bow of the brooch) with the others. This suggests that the second canonical variate distinguishes brooches with a relatively narrow shape. The FACROTATE directive (6.4) may help you to interpret the loadings. However, canonical variates analysis and principal components analysis can still be useful, even if the loadings cannot be interpreted.

The scores for the means are arranged so that their centroid, weighted by group size, is at the origin. This is done by subtracting a constant (or adjustment) term, for each canonical variate dimension, from the scores initially formed as a linear combination of the group means of the original variables. For example, the constant term of −19.90 occurs in the second score for the third mean, −3.545, formed as:

$$-2.656\bar{v}_{13} + 1.631\bar{v}_{23} - 1.708\bar{v}_{33} - 0.623\bar{v}_{43} - 0.758\bar{v}_{53} - 2.028\bar{v}_{63} - 0.110\bar{v}_{73} + 19.90$$

where $\bar{v}_{ij}$ is the mean of the ith variable for the jth group. If you ask for the group mean scores to be printed, then the corresponding constant terms are also printed under the heading "Adjustment terms", as shown in Example 6.3.1a above. You can see from the canonical variate means that the second canonical variate separates the third group from the other three.

Results can be printed for a subset of the latent roots by setting the NROOTS and SMALLEST options of CVA. NROOTS specifies the number of roots for which you want the results to be printed. By default these will be the largest roots, unless you set SMALLEST=yes; then the results will be printed for the smallest non-zero roots. When you print a subset of the results, residuals can be formed and printed from the dimensions that are not displayed.

If you ask for distances, they are formed from the group mean scores for the canonical variate dimensions that are printed. If results are printed for the full dimensionality, the distances will be Mahalanobis distances between the groups.

The LRV parameter allows you to save the loadings, latent roots, and their sum (the trace) in an LRV structure, while the SCORES parameter saves the canonical variate means. If you have declared the LRV already, its number of rows must be the same as the number of variates involved in forming the input SSPM. The number of rows of the SCORES matrix, if previously declared, must be equal to the number of groups.

The number of columns of the LRV and of the SCORES matrix corresponds to the number of dimensions to be saved from the analysis, and this must be the same for both of them. If the structures have been declared already, Genstat will take the larger of the numbers of columns declared for either, and declare (or redeclare) the other one to match. If neither has been declared and option SMALLEST retains the default setting no, Genstat takes the number of columns from the setting of the NROOTS option. Otherwise, Genstat saves results for the full set of dimensions. The trace saved as the third component of the LRV structure, however, will contain the sums of all the latent roots, whether or not they have all been saved. Procedure LRVSCREE (6.2.2) can be used to produce a "scree" diagram which can be helpful in deciding how many dimensions to save.

The RESIDUALS parameter allows you to save the distances of the means from the dimensions fitted in the analysis in a matrix with number of rows equal to the number of groups and one column. If the latent roots and vectors (loadings) are saved from the analysis, the residuals will correspond to the dimensions not saved; the same applies if you save scores. If neither the LRV nor scores are saved, the saved residuals will correspond to the smallest latent roots not printed.

The DISTANCES parameter allows you to save the inter-group-mean Mahalanobis distances in a symmetric matrix, and the ADJUSTMENTS parameter saves the adjustment terms in a matrix with one row and *g* columns.

In Example 6.3.1b the NROOTS option specifies that the results to be printed are for the two largest latent roots. The residuals that are printed thus correspond to the remaining roots, here only the third. Likewise, the printed distances are formed from the first two canonical variate means. The structure Lrv saves the latent roots and vectors for these two dimensions; this is used by the CVASCORES procedure in Example 6.3.2, below, to calculate scores for the individual units for these two dimensions.

Example 6.3.1b

```
  38  CVA [PRINT=residuals,distances; NROOTS=2] W; LRV=Lrv

Canonical variates analysis
===========================

Residuals
---------

            1       0.613
            2       1.584
            3       0.825
            4       2.241


Inter-group distances
---------------------

   1       0.000
   2       4.231       0.000
   3       5.802       4.215       0.000
   4       5.913       2.140       6.007       0.000
               1           2           3           4
```

The SAVE parameter can supply a pointer to save a multivariate save structure contining all the details of the analysis. If this is unset, an unnamed save structure is saved automatically (and this can be accessed using the GET directive). Alternatively, you can set SAVE=* to prevent any save structure being formed if, for example, you have a very large data set and want to avoid committing the storage space.

6.3.2 Canonical variate scores: the CVASCORES procedure

CVASCORES procedure

Calculates scores for individual units in canonical variates analysis (S.A. Harding).

Option

PRINT = *string tokens*	What output to print (scores, adjustments); default scor

Parameters

WSSPM = *SSPMs*	Within-group sums of squares and products structure
LRV = *LRVs*	Loadings, roots and trace saved from CVA of the WSSPM
SCORES = *matrices*	Unit scores
ADJUSTMENTS = *matrices*	Mean Adjustments

Procedure CVASCORES calculates coordinates of the individual data points projected into the canonical variate space of a canonical variates analysis. The WSSPM parameter must be set to the within-group SSP matrix that was used as input to the CVA directive when calculating the analysis, and the LRV parameter must supply the LRV structure formed by CVA. The scores can be saved using the SCORES parameter, and the mean adjustments can be saved using the ADJUSTMENTS parameter (these can be printed, but not saved, by CVA). The PRINT option allows the scores and adjustments to be printed, with the default to print just the scores.

Example 6.3.2 continues Example 6.3.1b, and prints the scores of the individual brooches in the first two dimensions.

Example 6.3.2

```
  39  CVASCORES W; LRV=Lrv

Canonical variate scores
========================

                     1          2

            1   -2.537      3.908
            2   -2.819     -2.101
            3   -0.494      0.205
            4    0.671      1.149
            5    1.900     -0.255
            6   -2.888     -0.688
            7   -2.766     -1.733
            8    4.899     -0.473
            9    1.046      0.249
           10    0.733      2.780
           11   -0.942      2.557
           12    2.531      0.672
           13    1.622     -0.766
           14    1.364     -0.446
           15    0.622     -1.241
           16    0.007     -0.657
           17    0.701     -0.560
           18    1.070     -1.226
           19   -2.632     -0.612
```

```
20      -1.834       4.682
21       1.843      -0.888
22       3.831      -2.364
23       1.005       0.317
24       0.592      -0.736
25       0.863       1.280
26      -2.092      -2.536
27      -4.606      -4.318

28      -1.690       3.801
```

6.3.3 Plotting canonical variate scores: the CVAPLOT procedure

CVAPLOT procedure

Plots the mean and unit scores from a canonical variates analysis (D.A. Murray).

Options

PLOT = *string tokens*	Type of plot to be drawn (meanscores, unitscores, confidenceregion); default mean, conf
GROUPS = *factor*	Group allocations in the CVA
MSCORES = *matrix*	Mean scores from the CVA; if unset these are calculated using the CVA directive
USCORES = *matrix*	Unit scores from the CVA; if unset these are calculated using the CVASCORES procedure
WSSPM = *SSPM*	Within-group sums of squares and products, means etc. for the CVA; must be supplied if the scores and groupings are not provided
CREGION = *string tokens*	Type of confidence region to be drawn (mean, population); default mean
CIPROBABILITY = *scalar*	The probability level for the confidence region; default 0.95
TAREA = *scalar*	Defines the transparency to use to shade the confidence regions; default 255 i.e. no shading

Parameters

YDIMENSION = *scalars*	Dimensions to be plotted in the y direction of each graph
XDIMENSION = *scalars*	Dimension to be plotted in the x direction
TITLE = *texts*	Title for each plot
WINDOW = *scalars*	Window for each graph; default 1
SCREEN = *string tokens*	Whether to clear the screen before plotting (clear, keep); default clea

Procedure CVAPLOT plots information from a canonical variates analysis. The type of graph to be displayed is controlled by the PLOT option with settings meanscores to draw mean scores, unitscores to display the unit scores and confidenceregion to display confidence regions about the means or the tolerance region for a population.

The CREGION option specifies the type of confidence region that is drawn. The setting mean will draw the confidence region about the population means, and population plots the tolerance region for the populations. By default a 95% confidence region is calculated, but this can be changed by setting the CIPROBABILITY option to the required value (between 0 and 1).

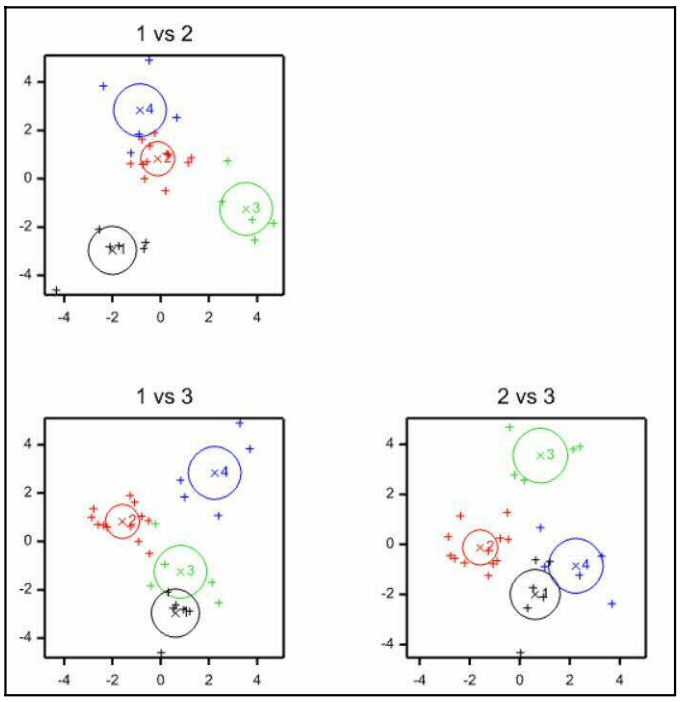


Figure 6.3.3

You can shade the confidence regions by setting the TAREA option. This defines a transparency value (between 0 and 255) for the shaded regions, in a similar way to the TAREA option of PEN. The default value of 255 indicates that the regions are completely transparent (i.e. completely unshaded); a line is then drawn around each region.

Matrices containing the mean scores and units scores (saved from CVA and CVASCORES) can be supplied directly, using options MSCORES and USCORES respectively; option GROUPS should then supply a factor defining the groupings of the units in the canonical variates analysis. Alternatively, you can supply a within-group SSPM and the scores will be calculated within the procedure, using the CVA directive and the CVASCORES procedure, and the groups will be accessed from within the SSPM.

The YDIMENSION and XDIMENSION parameters specify which dimensions are to be plotted in the y and x directions; by default these are dimensions 1 and 2 respectively. The WINDOW parameter indicates the window to be used for each plot (default 1), the TITLE parameter provides a title for each plot, and the SCREEN parameter indicates whether existing plots on the screen are to be kept or cleared each time (the default being to clear the screen).

Figure 6.3.3 contains a graph of the scores for the brooches discussed in Sections 6.1.1, 6.3.1 and 6.3.2, plotted by the statement

```
CVAPLOT [PLOT=mean,unit,confidence; WSSPM=W] \
  YDIMENSION=1,1,2; XDIMENSION=2,3,3; \
  TITLE='1 vs 2','1 vs 3','2 vs 3'; \
  WINDOW=5,7,8; SCREEN=clear,keep,keep
```

6.3.4 Plotting large numbers of canonical variate scores: the CVATRELLIS procedure

CVATRELLIS procedure

Displays the distribution of groups over 2 dimensions from a CVA analysis using a trellis of bar or pie charts (R.W. Payne).

Options

PLOT = *string tokens*	What to plot (barchart, piechart, scaledpiechart, key); default barc, key
NPARTITIONS = *scalar*	Number of partitions along each axis; default 8
COLOURS = *variate* or text	Colours for the groups; default uses the colours defined for pens 2 upwards
KEYHEIGHT= *scalar*	Height of the key; default 0.1
SAVE = *pointer*	Save structure from the CVA analysis to display; default displays the most recent analysis

Parameters

YDIMENSION = *scalars*	Dimension for the y-axis for each graph; default 1
XDIMENSION = *scalars*	Dimension for the x-axis for each graph; default 2
TITLE = *texts*	Title for each graph

When there are many units in a CVA analysis, the points in the graphs of scores plotted by CVAPLOT (6.3.3) may become too dense to interpret. Instead CVATRELLIS divides the plane into a trellis of squares, and calculates and plots the replications of the groups in each of the squares using the D2GROUPS procedure (1:6.8.5).

The PLOT option specifies how the replications are plotted, with settings:

barchart	plots a trellis with a bar chart in each square,
piechart	plots a trellis with a pie chart in each square, with a title showing the number of points in the square,
scaledpiechart	plots a trellis with a pie chart in each square, each one scaled according to the number of points in the square, and
key	includes a key showing the colour used for each group.

The default is to plot a bar chart with a key.

The y and x dimensions to display are specified by the YDIMENSION and XDIMENSION parameters, respectively. The default displays dimensions 1 and 2.

The NPARTITIONS option specifies the number of partitions (i.e. the number of trellis boxes) along each dimension; default 8. The KEYHEIGHT option specifies the height of the key. This must not be less than 0.1 or greater than 0.5. The default is 0.1. The COLOURS option can specify either a variate or a text to define the colours to be used for the groups; for details see 1:6.9.9. The default is to use the colours defined for pens 2 upwards. The TITLE parameter can supply a title for the plots; by default the is none. The SAVE option can supply a save structure from the CVA analysis to display. The default is to display scores from the most recent CVA.

The command below displays the scores for the brooches, discussed in Sections 6.1.1, 6.3.1 and 6.3.2, in a bar chart (Figure 6.3.4a) and a pie chart (Figure 6.3.4b).

```
CVATRELLIS [PLOT=barchart,piechart,key; NPARTITIONS=6;\
           COLOURS=!t(black,red,green,blue)]
```

The specified colours are chosen to match those used for the groups in Figure 6.3.3.

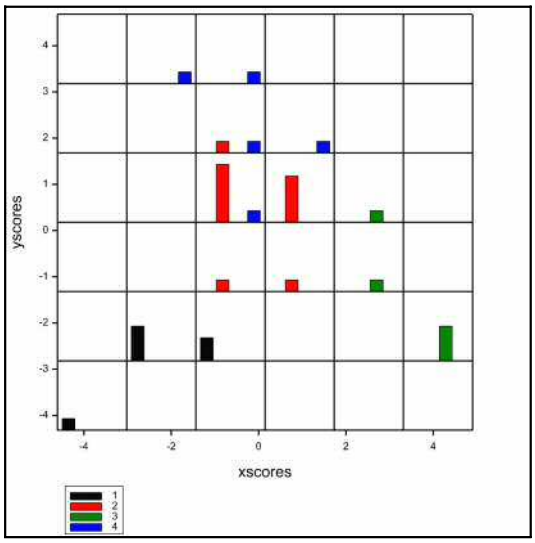


Figure 6.3.4a

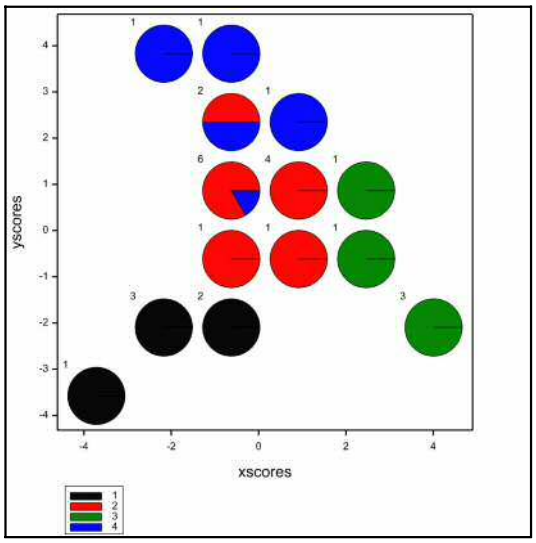


Figure 6.3.4b

6.4 Factor rotation: the FACROTATE directive

Principal components analysis (6.2), canonical variates (6.3) and factor analysis (6.11) all define a set of dimensions (sometimes called axes) that are linear combinations of the original variables. The individual coefficients of these combinations are called loadings, and can be used to interpret the dimensions. With principal components analysis, the loadings must lie in the range [−1, 1]; this is the situation that we discuss in the initial part of this subsection. The situation with canonical variates and factor analysis is slightly different and is described at the end of this subsection.

When several dimensions are considered it is possible to define an equivalent set of new dimensions, whose loadings are linear combinations of the original loadings. If the absolute values of the loadings for a new dimension are either close to 0 or close to 1, you can interpret the dimension as mainly representing only those original variables with large positive (or negative) loadings. You may sometimes want new dimensions determined by loadings like these, because they are easier to interpret. The methods by which these new dimensions can be obtained are generally known collectively as *factor rotation* because the new dimensions represent a rotation of the axes of the original dimensions. The `FACROTATE` directive provides two methods of orthogonal factor rotation: varimax rotation and quartimax rotation (Cooley & Lohnes 1971). The default method, varimax rotation, maximizes the variance of the squares of the loadings within each new dimension: the effect of this rotation should be to spread out the squared-loadings to the extremes of their range. Quartimax rotation uses the fourth power of the loadings instead of the second power.

FACROTATE directive

Rotates factor loadings from a principal components, canonical variates or factor analysis.

Options

`PRINT` = *string tokens*	Printed output required (`communalities`, `loadings`, `orthogonalrotationmatrix`, `rotation`); default `*` i.e. no printing
`METHOD` = *string token*	Criterion (`varimax, quartimax`); default `vari`
`NROOTS` = *scalar*	Sets the number of dimensions to rotate from the original loadings; default `*` i.e. all

Parameters

OLDLOADINGS = *matrices*	Original loadings
NEWLOADINGS = *matrices*	Rotated loadings for each set of OLDLOADINGS
COMMUNALITIES = *matrices*	Communalities of the variables in each rotation
ROTATION = *matrices*	Saves the orthogonal rotation from the original solution to the rotated space

The first parameter, OLDLOADINGS, specifies a list of matrices, that contain the loadings for the original dimensions. These can be obtained from the first element of the LRV structures, that can be saved by the LRV parameter of PCP, CVA and FCA. The matrices to save the new loadings are specified by the NEWLOADINGS parameter. The ROTATION parameter can save the orthogonal rotations from the original solutions to the rotated spaces.

One way of supplying the loadings for the original variables is by saving the latent roots and vectors from a principal components analysis (6.2) using the LRV parameter. You can then either supply the whole LRV, or just the first structure of the LRV (which is the matrix of loadings). Example 6.4a is similar to Example 6.2.1a; however, here the first two latent roots and vectors are saved and used as input to the FACROTATE directive.

Example 6.4a

```
   2  UNITS [NVALUES=12]
   3  POINTER [VALUES=Height,Length,Width,Weight] Dmat
   4  READ [PRINT=errors] Dmat[]
   9  LRV [ROWS=Dmat; COLUMNS=2] Latent
  10  PCP [PRINT=loadings] Dmat; LRV=Latent

Principal components analysis
=============================

Latent Vectors (Loadings)
-------------------------

                        1           2           3           4
       Height     0.21529     0.37981     0.78747     0.43506
       Length     0.25623     0.86524    -0.34389    -0.25970
        Width     0.74104    -0.21726    -0.37937     0.50964
       Weight     0.58211    -0.24474     0.34308    -0.69537

  11  FACROTATE [PRINT=rotation,communalities] Latent[1]

Factor rotation
===============

Communalities
-------------

                        1
       Height      0.1906
       Length      0.8143
        Width      0.5963
       Weight      0.3988


Rotated factors
---------------

                        1           2
       Height      0.0630      0.4320
       Length     -0.0747      0.8993
        Width      0.7694      0.0660
       Weight      0.6312     -0.0172
```

The LRV structure `Latent` is declared 1n line 9, and is used 1n line 10 to save the latent roots and vectors. The full set of latent vectors is printed from the `PCP` directive to allow you to compare the original loadings with those after rotation. The original loadings seem to tell us that the first new axis is some negative measure of overall size, and that the second is a contrast between the first two variables (`Height` and `Length`) and last two (`Width` and `Weight`). The new loadings give the first axis as largely consisting of `Width` and `Weight`, and the second as largely consisting of `Height` and `Length`.

Note that under either method of factor rotation, the total contribution of each of the original variables always remains the same as in the input set of loadings (for mathematical reasons). These contributions are called the *communalities* of the variables, and can be expressed as the sum of the squared loadings: they indicate how much of the variation of each of the original variables is retained in either set of dimensions (whether the original set from the principal component analysis, or the new set from the rotation). For example, the communality for the first variable can be calculated from the set of new dimensions as follows

$$0.1906 = (-0.0630)^2 + (0.4320)^2$$

Equivalently, from the original set, it is

$$0.1906 = (-0.2153)^2 + (0.3798)^2$$

The communalities can be saved using the `COMMUNALITIES` parameter.

If you keep all the loadings from a principal components analysis, each of the variables will have communality 1. Factor rotation in this case will simply give a set of new loadings, each of which will represent just one of the variables, with loading 1. Thus factor rotation is sensible only if you keep merely the higher-dimensional loadings.

The loadings from canonical variates analysis (6.3) are not constrained to lie in the range (−1, +1). The factor rotation methods operate in a similar manner as for principal component loadings. Again, the objective is to obtain loading values, such that each is either relatively small or relatively large. Also the communalities of the variables remain the same in the rotated loadings as in the original loadings, and the new loadings are obtained as an orthogonal rotation of the old loadings. However, the complete set of loadings can generally be retained from canonical variates analysis and used for factor rotation, without giving meaningless results. This is because the original dimensions from the canonical variates analysis do not contain all the dimensionality of the original variables, unless the number of variables is less than the number of groups. So a factor rotation of all the dimensions will not merely recover the original variables, as would happen with loadings from principal components analysis. Likewise, loadings from the full set of available dimensions in a factor analysis (6.11) can be also be retained for rotation without recovering the original variables.

Printed output is controlled by the `PRINT` option, with the following settings:

`communalities`	to print the communalities;
`loadings`	to print the rotated loadings, under the caption "Rotated factors";
`orthogonalrotationmatrix`	to print the rotation matrix;
`rotation`	this is the original setting used to print the rotated loadings. It is retained as a synonym of `loadings` to allow earlier programs to run. However, in view of the confusion with the `ROTATION` parameter, it may be deleted in a future release.

By default, nothing is printed.

The `NROOTS` option sets the number of dimensions to rotate from the original loadings (the other dimensions are left unchanged). The default is to rotate them all.

This is illustrated in Example 6.4b, which rotates the loadings as produced by Example 6.3.1a.

Example 6.4b

```
  2  UNITS [NVALUES=28]
  3  POINTER [VALUES=Foot_lth,Bow_ht,Coil_dia,Elem_dia,Bow_wdth, \
  4     Bow_thck,Length] Data
  5  FACTOR [LEVELS=4] Groupno
  6  READ [PRINT=errors] Groupno,Data[]
 35  SSPM [TERMS=Data[]; GROUPS=Groupno] W
 36  FSSPM W
 37  LRV [ROWS=Data; COLUMNS=3] L
 38  CVA [PRINT=loadings] WSSPM=W; LRV=L

Canonical variates analysis
===========================

Latent Vectors (Loadings)
-------------------------

                         1           2           3
        Data
    Foot_lth        -1.130       2.656       3.397
      Bow_ht         0.633      -1.631       4.799
    Coil_dia        -3.501       1.708       1.450
    Elem_dia         2.669       0.623      -2.802
    Bow_wdth         3.468       0.758       0.757
    Bow_thck        -1.859       2.028      -2.478
      Length         1.279       0.110      -3.598

 39  FACROTATE OLDLOADINGS=L[1]; NEWLOADINGS=L[1]
 40  PRINT L[1]

             L['Vectors']
                         1           2           3
        Data
    Foot_lth         0.135       4.381       0.810
      Bow_ht         0.210       1.670       4.823
    Coil_dia        -2.513       3.254      -0.612
    Elem_dia         2.560      -2.192      -2.001
    Bow_wdth         3.530      -0.111       0.837
    Bow_thck        -1.074       0.471      -3.512
      Length         1.041      -2.606      -2.593
```

Rather than print the rotated loadings directly from the analysis (line 39), the program saves and prints them separately (line 40). This might be appropriate if you intend to calculate canonical variate scores for the units, in the rotated factor space. If you do intend to do this, you will also have to calculate new canonical variate means in the rotated factor space; however, this is easy to do as they are simply the group means of the rotated scores for the units.

6.5 Discriminant analysis

Linear discriminant analysis uses a "training" set of data to find the best dimensions to distinguish between a set of groups. It can then use this information to allocate some new observations to the groups (i.e. to identify the group to which each new observation belongs). The `DISCRIMINATE` procedure (6.5.1) can be used if you want to use all the available variates that provide information about the attributes of the data units (or if you have already selected the best variates to use). Alternatively, you can use the `SDISCRIMINATE` procedure (6.5.2) to select the best set of variates from those available. `DISCRIMINATE` assumes that the groups share a common variance-covariance matrix. The `QDISCRIMINATE` procedure is available for situations where this is not a reasonable assumption (6.5.3).

6.5.1 The DISCRIMINATE procedure

DISCRIMINATE procedure

Performs discriminant analysis (L.H. Schmitt & P.G.N. Digby).

Options

PRINT = *string tokens*	Printed output from the analysis (counts, lrv, tests, ccorrelations, icorrelations, correlations, adjustments, means, gdistances, scores, distances, newgroups, table, validation); default coun
NROOTS = *scalar*	The number of dimensions to be used for printed and saved output, and used in calculating the distances and the allocation of units; default is to use the full dimensionality
REALLOCATE = *string token*	Whether units from the training set are to be reallocated to groups (no, yes); default no
PLOT = *string tokens*	Features for the plots (means, mlabels, scores, polygons, confidencecircle); default mean, scor, poly (Note: * suppresses plotting)
VALIDATIONMETHOD = *string token*	Validation method to use to calculate error rates (bootstrap, crossvalidation, jackknife); default cros
NSIMULATIONS = *variate*	Number of bootstraps or cross-validation sets to use for selection and for validation; default !(10,50)
NCROSSVALIDATIONGROUPS = *scalar*	Number of groups for cross-validation, default 10
SEED = *scalar*	Seed for random number generation; default 0
YROOT = *scalars*	Specifies roots for plotting on y-axes
XROOT = *scalars*	Specifies roots for plotting on x-axes
TITLE = *string tokens*	Titles for plots
WINDOW = *scalars*	Windows for plots
SCREEN = *string tokens*	Action before each plot (keep, clear); default clea

Parameters

DATA = *pointers*	Each pointer contains a set of variates to be analysed
GROUPS = *factors*	Define groupings for the units in each training set, or missing values for the units to be allocated
NEWGROUPS = *factors*	Saves allocations (and reallocations)
ALLOCATION = *factors*	Saves allocations to groups including those not present in the training set
MEANS = *matrices* or *pointers*	Saves scores for group means
SCORES = *matrices* or *pointers*	Saves scores for units
DISTANCES = *matrices*	Saves unit to group-mean squared distances
LRV = *LRVs*	Saves the LRVs from the canonical variates analyses
ADJUSTMENTS = *matrices*	Saves adjustments to the canonical variates analyses
GDISTANCES = *symmetric matrices*	Saves the distances between groups
CCORRELATIONS = *matrices*	Saves canonical correlation coefficients
ICORRELATIONS = *symmetric matrices*	Saves within-group correlation matrices of the input variates

CORRELATIONS = matrices	Saves within-group correlations between the input and canonical variates

DISCRIMINATE performs discriminant analysis (see, for example, Mardia, Kent & Bibby 1979).

The input for the procedure is given by a pointer and a factor, specified by the DATA and GROUPS parameters, respectively. The pointer contains a set of variates defining the attributes of the units. Any unit with a missing value in any of the variates is excluded from the analysis. Units can also be excluded from the analysis by restricting the factor or variates; any such restrictions must be consistent (the rules here are exactly as used by the FSSPM directive). The factor specifies the pre-defined groupings of the units from which the allocation is derived (the "training set"); the units to be allocated by the analysis have missing factor values.

A canonical variates analysis (CVA) is used to obtain the scores for the group means and the LRV containing the loadings (L), roots and trace. Scores are then calculated for all the units (i.e. ignoring any restrictions or missing values), using the formula

$$(XL) - (JA)$$

where X is a matrix containing the full set of units-by-variables data, J is a column vector of one's, and A is a row vector of adjustments required to place the scores for the units onto the same scale as those for the group means.

Mahalanobis squared distances between the units and the group means are calculated from the canonical variate scores. Each unit is then allocated to the group for which it has the smallest Mahalanobis squared distance to the group mean.

Printed output is controlled by the option PRINT with settings:

counts	tables of the number of units in each group with a complete set of observations;
lrv	canonical variate loadings, latent roots and trace;
tests	chi-square tests (as given by CVA);
ccorrelations	canonical correlation coefficients (see Klecka 1980);
icorrelations	within-group correlation matrix of the input variates;
correlations	within-group correlations between the input and canonical variates;
adjustments	adjustments required to the canonical variate scores;
means	canonical variate scores for the group means;
gdistances	inter-group distances (as given by CVA);
scores	canonical variate scores for the units;
distances	Mahalanobis squared distances between the units and the group means;
newgroups	initial grouping and the allocation of units to groups;
table	tables of counts of allocations; and
validation	estimated error rates (see the VALIDATION option below).

The NROOTS option specifies how many dimensions are to be printed and retained for the latent roots and vectors, and for the scores of the means and units. The distances of the units from the group means, and thus the allocation of units, are also formed from the scores in the number of dimensions specified by NROOTS. By default results are for the full dimensionality, i.e. the smaller of the number of variates and one less than the number of groups.

The REALLOCATE option specifies whether the units in the training set are to be reallocated to groups by the procedure. If the default setting no is used then their group values, either printed or saved, will be missing.

The VALIDATIONMETHOD option specifies the validation method, with settings for cross-validation, jackknife and bootstrap. Cross-validation works by randomly splitting the units into a number of groups specified by the NCROSSVALIDATIONGROUPS option (default 10). It then omits each of the groups, in turn, and predicts how the the omitted units are allocated to the

discrimination groups. Jackknifing leaves the units out one at a time, and uses the rest of the data to predict the group of the omitted unit. The bootstrap method works by drawing a bootstrap sample of units (a random sample of units with replacement of the same size as the original sample), and predicting the units that are not present in the random sample. The resulting bootstrap error rate is then calculated as a weighted average of the error rate of the omitted observations and the predictive error rate of the bootstrap sample. The weights used are 0.632 and 0.368 respectively, and so this is known as the *632 rule*.

The NSIMULATIONS option sets the number of simulations for cross-validation or bootstrapping. It should be set to a variate with two values: the first value defines the number of simulations to use during selection (default 10), and the second sets the number to use in the estimation of the error rates (default 50).

The SEED option provides the seed for the random numbers used for the randomizations during in the simulations. The default value of 0 continues an existing sequence of random numbers, if none have been used in the current Genstat job, it initializes the seed automatically using the computer clock.

The PLOT option provides for group means, labels for group means, unit scores, group polygons enclosing units, and 95% confidence circles around group means. The YROOT and XROOT options specify the roots for the axes. The TITLE, WINDOW and SCREEN options allow further control of the plots. More than one plot can be output by having a list of scalars for YROOT. In this case, the values of XROOT, TITLE, WINDOW and SCREEN are cycled in parallel. A rug-like plot is drawn if only one root is extracted or if YROOT is set to a missing value.

Results from the analysis can be saved using the parameters NEWGROUPS, ALLOCATION, MEANS, SCORES, DISTANCES, LRV, ADJUSTMENTS, GDISTANCES, CCORRELATIONS, ICORRELATIONS and CORRELATIONS. The structures specified for these parameters need not be declared in advance. The default is to save MEANS and SCORES in matrices. However, if you declare either as a pointer, it will instead store the results as a data matrix (i.e. a pointer of variates corresponding to the columns of the matrix). The results correspond to p dimensions, where p is the smaller of either the number of variates, or the number of groups minus one.

Example 6.5.1 performs a discriminant analysis for two of the species in Fisher's Iris data (see Table 1.2.2 in Mardia, Kent & Bibby 1979). DISCRIMINATE reallocates the observations to the closest group (according to its Mahalanobis squared distance from the group mean). As the output shows, this results in the reallocation of one observation from *Septosa* to *Versicolour*. An analysis of the whole of Fisher's Iris data, including graphs, can be accessed within Genstat using procedure LIBEXAMPLE, or the Example Programs menu of Genstat *for Windows*.

Example 6.5.1

```
  2  SPLOAD       [PRINT=*] '%GENDIR%/Data/Iris.gsh'
  3  POINTER      [VALUES=Sepal_Length,Sepal_Width] Measurements
  4  " Take a subset of the sepal data with species Setosa and Versicolour."
  5  SUBSET       [Species.IN.!t(Setosa,Versicolor); SETLEVELS=yes]\
  6               Species,Measurements[]
  7  " Use DISCRIMINATE: allowing training set to be reallocated;
 -8    printing LRV and adjustments from CVA."
  9  DISCRIMINATE [PRINT=lrv,adjustments; PLOT=*; REALLOCATE=yes]\
 10               Measurements; GROUPS=Species; NEWGROUPS=New_Species

Discriminant analysis
=====================

Latent vectors, roots, and trace from CVA
-----------------------------------------
```

```
Vectors:

       Scores   Scores[1]
        Sepal
       Length       2.561
        Width      -3.167

Roots:

       Scores
    Scores[1]       5.087

    Trace:       5.087


Adjustments applied to columns of scores
----------------------------------------

                        1
            1       4.196

  30  "Tabulate the original grouping and the reallocation of units."
  31  TABULATE [PRINT=counts; CLASSIFICATION=Species,New_Species; MARGIN=yes]

                   Count
  New_Species      Setosa Versicolour       Count
      Species
       Setosa          49           1          50
  Versicolour           0          50          50
        Count          49          51         100
```

6.5.2 The SDISCRIMINATE procedure

SDISCRIMINATE procedure

Selects the best set of variates to discriminate between groups (D.B. Baird, L.H. Schmitt & J.W. McNicol).

Options

PRINT = *string tokens*	Printed output from the analysis (summary, steps, validation, specificity, discrimination, monitoring); default summ, vali, spec, disc
PLOT = *string tokens*	What plots to produce (errorrate, steps, specificity, discriminant); default erro, steps, spec, disc
DDISCRIMINANT = *string tokens*	What to display on the discriminant plot (means, mlabels, scores, polygons, confidencecircle); default means, mlabels, scores, conf
METHOD = *string token*	The variable selection method to use (forward, backward); default forw
NSELECT = *scalar*	Number of variates to select; default 4
CRITERION = *string token*	Criterion to use to select variables (wilkslambda, crossvalidation, bootstrap, jackknife); default wilk
MODELCHOICE = *string token*	Which model to save (optimal, nselect); default opti
VALIDATIONMETHOD = *string token*	Validation method to use to calculate error rates (bootstrap, crossvalidation, jackknife, prediction); default cros
NSIMULATIONS = *variate*	Number of bootstraps or cross-validation sets to use for

	selection and for validation; default `!(10,50)`
NCROSSVALIDATIONGROUPS = *scalar*	Number of groups for cross-validation, default 10
SEED = *scalar*	Seed for random number generation; default 0
YROOT = *scalar*	Specifies the root for plotting on the y-axis
XROOT = *scalar*	Specifies the root for plotting on the x-axis

Parameters

DATA = *pointers*	Each pointer contains a set of variates that are available to be selected
GROUPS = *factors*	Define groupings for the units in each training set
FORCED = *pointers*	Variates that must be included in the model
SELECTED = *pointers*	Saves the variates in the final model
STEPS = *pointers*	Saves the criterion values for each step in the model selection
ERRORRATE = *scalars*	Saves the validation error rate for the final model
SPECIFICITY = *tables*	Saves the specificity table for the final model
ALLOCATION = *factors*	Saves the groups allocated by the final model
LRV = *LRVs*	Saves the LRVs from the final discriminant analysis
SCORES= *matrices* or *pointers*	Save discriminant scores for unit from the final model

SDISCRIMINATE uses forward selection or backwards elimination to search for the best set of variates to discriminate between groups. The variates that are available for the discrimination must be specified, in a pointer, by the DATA parameter. The membership of the groups must be specified, in a factor, by the GROUPS parameter. If there are some variates that must always be included in the model, these can be specified, in a pointer, by the FORCED parameter.

Printed output is controlled by the option PRINT, with settings:

summary	summary of the model fitting,
steps	criterion values evaluated at each step of the model fitting,
validation	error rates at each model step,
specificity	specificity of allocation (i.e. the proportion of each group that is assigned correctly),
discrimination	the standard discriminant analysis output for the final model, and
monitoring	criterion values for each model tried.

The default is `PRINT=summ,vali,spec,disc`.

The PLOT option controls what plots are displayed, with settings:

errorrate	error rate at each selection step,
steps	criterion values at each step of the model fitting,
specificity	specificity at each selection step, and
discriminant	the standard discriminant plot from the final model.

By default these are all plotted. The DDISCRIMINANT option allows group means, labels for group means, unit scores, group polygons enclosing units, and 95% confidence circles around group means to be included on the discriminant plot. The YROOT and XROOT options specify the roots for the axes.

The selection method is defined by the METHOD option. The forward setting starts with the FORCED model and then, at each step, looks to see which of DATA variates not already in the model gives the best improvement; this is the default. The backward setting starts with the model, and looks to see which variate in model (other than those in FORCED) gives the least reduction in the criterion when eliminated at that step.

The criterion for evaluating the model is defined by the CRITERION option, with settings:

`wilkslambda`	uses the ratio of the determinant of the within-group sums of squares and products to the determinants of the total sums of squares and products (default),
`crossvalidation`	uses the cross-validation error rate,
`bootstrap`	uses the bootstrap error rate, and
`jackknife`	uses jackknifing.

Cross validation and bootstrapping take much longer than the use of Wilks' lambda.

The number of variates in the final model (excluding those in the `FORCED` model) is set by `NSELECT` option. The `MODELCHOICE` option indicates how to choose the final model. The default setting `optimal` takes the model from the step with the minimum validation error. Alternatively, the `nselect` setting takes the model with the number of variates specified by the `NSELECT` option.

The `VALIDATIONMETHOD` option specifies the validation method, with settings for prediction, cross-validation, jackknife and bootstrap. Cross-validation works by randomly splitting the units into a number of groups specified by the `NCROSSVALIDATIONGROUPS` option (default 10). It then omits each of the groups, in turn, and predicts how the the omitted units are allocated to the discrimination groups. Jackknifing leaves the units out one at a time, and uses the rest of the data to predict the group of the omitted unit. The bootstrap method works by drawing a bootstrap sample of units (a random sample of units with replacement of the same size as the original sample), and predicting the units that are not present in the random sample. The resulting bootstrap error rate is then calculated as a weighted average of the error rate of the omitted observations and the predictive error rate of the bootstrap sample. The weights used are 0.632 and 0.368 respectively, and so this is known as the *632 rule*.

The `NSIMULATIONS` option sets the number of simulations for cross-validation or bootstrapping. It should be set to a variate with two values: the first value defines the number of simulations to use during selection (default 10), and the second sets the number to use in the estimation of the error rates (default 50).

The `SEED` option provides the seed for the random numbers used for the randomizations during in the simulations. The default value of 0 continues an existing sequence of random numbers, if none have been used in the current Genstat job, it initializes the seed automatically using the computer clock.

The `SELECTED` parameter can save the contents of the chosen model, in a pointer. The `STEPS` parameter can save a pointer with a variate for each step of the selection, containing the criterion evaluated for each `DATA` variate at then step. The variates contain a missing value if the `DATA` variate had already been included or excluded from the model. The `ERRORRATE` parameter can save a variate with the minimum value of the validation error rate after each step. The `SPECIFICITY` parameter can save the specificity table for the final model. The `LRV` parameter can save the latent roots, vectors and trace from the final discriminant analysis, and the `ALLOCATION` and `SCORES` parameters can save the assigned groups and discriminant scores.

Example 6.5.2 finds the three variates that give the best discrimination for all the species in Fisher's Iris data.

Example 6.5.2

```
  2  SPLOAD        [PRINT=*] '%GENDIR%/Data/Iris.gsh'
  3  " Use SDISCRIMINATE to find the best 3 variates for discrimination."
  4  POINTER       [VALUES=Sepal_Length,Sepal_Width,Petal_Length,Petal_Width]\
  5                Vars
  6  SDISCRIMINATE [PRINT=summary,validation,specificity; PLOT=*; NSELECT=3;\
  7                SEED=719122] Vars; GROUPS=Species

Stepwise discriminant analysis
==============================
```

```
Summary information for stepwise selection of variables
-------------------------------------------------------

Forward selection
Selection criterion: Wilks' lambda

Best 3 variables:
         Variable   Criterion
     Petal_Length     0.05863
      Sepal_Width     0.03688
      Petal_Width     0.02498


Optimal variables selected
--------------------------

Petal_Length
 Sepal_Width
 Petal_Width


Validation error rate
---------------------

Using 10-fold cross-validation to calculate errors
Error: 3.15%


Percentage of each group allocated to groups
--------------------------------------------

Decision           True group
                 Setosa  Versicolor   Virginica
     Setosa      100.00        0.00        0.00
 Versicolor        0.00       95.08        4.52
  Virginica        0.00        4.92       95.48
```

6.5.3 The QDISCRIMINATE procedure

QDISCRIMINATE procedure

Performs quadratic discrimination between groups i.e. allowing for different variance-covariance matrices (D.B. Baird).

Options

PRINT = *string tokens*	Printed output from the analysis (allocation, counts, distance, probabilities, specificity, summary, table, validation, vcovariance); default spec, summ, vali
VALIDATIONMETHOD = *string token*	Validation method to use to calculate error rates (bootstrap, crossvalidation, jackknife, prediction); default cros
NSIMULATIONS = *scalar*	Number of bootstraps or cross-validation sets; default 50
NCROSSVALIDATIONGROUPS = *scalar*	Number of groups for cross-validation, default 10

Parameters

DATA = *pointers*	Each pointer contains a training set of variates to be used to form a quadratic discrimination
GROUPS = *factors*	Define groupings for the units in each training set
PRIORPROBABILITIES = *variates*	Prior probabilities of group membership; default * i.e. equal

SEED = *scalars*	Seed for the random numbers used in bootstrapping or cross-validation; default 0 continues from the previous generation or (if none) initializes the seed automatically
ERRORRATE = *scalars*	Saves the validation error rate
SPECIFICITY = *tables*	Saves the specificity table
ALLOCATION = *factors*	Saves the groups allocated by the discriminant rule
PROBABILITIES = *matrices* or *pointers*	Save posterior probabilities of membership of the groups (in the columns of a matrix or the variates in a pointer) for the units in the training set (in the rows)

QDISCRIMINATE performs a quadratic discrimination analysis to identify members of a set of groups using their observations on a set of variates. The quadratic discrimination rule assumes that the values of the variates within each group are distributed with a multi-variate Normal distribution, and that the variance-covariance matrix of the distributions are different for each group. This differs from the more familiar linear discriminant analysis, performed by procedure DISCRIMINATE, where the groups are assumed to have the same variance-covariance matrix.

The variates to be used to discriminate between the groups are specified in a pointer by the DATA parameter, and the membership of the groups is specified in a factor by the GROUPS parameter. The non-missing units of the GROUPS factor provide a training set to estimate the discriminant rule. Units that you would like to allocate to groups using the discriminant rule should be included in the data set with missing values in the GROUPS factor.

You can specify prior probabilities for the groups using the PRIORPROBABILITIES option; by default the groups are all assumed to be equally likely. You can use this to allow for unequal costs of mis-allocation by weighting the prior probabilities like this:

```
PRIORPROBABILITIES = Cost * Prior / SUM(Cost * Prior)
```

where Cost is a variate defining the cost of mis-allocation for each group.

Printed output is controlled by the option PRINT, with settings:

allocation	the allocated group for each unit,
counts	number of units in each group with a complete set of observations,
distance	generalized pairwise distance between group means,
probabilities	the posterior probability of being allocated to each group,
specificity	specificity of allocation (i.e. the proportion of each group that is assigned correctly),
summary	summary of the model fitting,
table	table of counts of training units allocated to each group,
validation	the error rate, and
vcovariance	variance-covariance matrices for the groups

The default is PRINT=spec,summ,vali.

The VALIDATIONMETHOD option specifies the validation method, with settings for prediction, cross-validation, jackknife and bootstrap. Prediction calculates the error rate as the proportion of the training set that were misallocated. Cross-validation works by randomly splitting the units into a number of groups specified by the NCROSSVALIDATIONGROUPS option (default 10). It then omits each of the groups, in turn, and predicts how the the omitted units are allocated to the discrimination groups. Jackknifing leaves the units out one at a time, and uses the rest of the data to predict the group of the omitted unit. The bootstrap method works by drawing a bootstrap sample of units (a random sample of units with replacement of the same size as the original sample), and predicting the units that are not present in the random sample. The resulting bootstrap error rate is then calculated as a weighted average of the error rate of the omitted observations and the predictive error rate of the bootstrap

sample. The weights used are 0.632 and 0.368 respectively, and so this is known as the *632 rule*.

The NSIMULATIONS option sets the number of simulations for cross-validation or bootstrapping; default 50.

The SEED parameter provides the seed for the random numbers used for the randomizations during in the simulations. The default value of 0 continues an existing sequence of random numbers, if none have been used in the current Genstat job, it initializes the seed automatically using the computer clock.

The ERRORRATE parameter can save the validation error rates. The SPECIFICITY parameter can save the proportion of each group that is assigned correctly. The ALLOCATION parameter can save the assigned groups, and the PROBABILITIES parameter can save the posterior probabilities of the groups.

Example 6.5.3 continues Example 6.5.1, and finds that quadratic discrimination gives the same results as ordinary linear discrimination with the sepal measurements for the species *Septosa* and *Versicolour* in Fisher's Iris data.

Example 6.5.3

```
 13  " Use QDISCRIMINATE to perform quadratic discrimination."
 14  QDISCRIMINATE [PRINT=specificity,summary,validation;\
 15                 VALIDATIONMETHOD=bootstrap; NSIMULATIONS=100]\
 16                 Measurements; GROUPS=Species; ALLOCATION=New_Species;\
 17                 SEED=324741

Quadratic discriminant analysis
===============================

Fitted variables: Sepal_Length, Sepal_Width
Groups: Species (Setosa, Versicolor)
Number of units in each group: 50, 50
Total number of units: 100
Number in training set: 100
Prior probabilities equal


Validation error rate, using bootstrapping with 632 rule to calculate errors
-----------------------------------------------------------------------------

Error: 0.99%


Percentage of each training group allocated to groups
-----------------------------------------------------

      Decision        Setosa    Versicolor
    True group
        Setosa         99.26          0.74
    Versicolor          1.25         98.75

Based on 100 simulations

 18  "Tabulate the original grouping and the reallocation of units."
 19  TABULATE     [PRINT=counts; CLASSIFICATION=Species,New_Species; MARGIN=yes]

                    Count
  New_Species      Setosa  Versicolor       Count
      Species
        Setosa         49           1          50
   Versicolor           0          50          50
        Count          49          51         100
```

6.6 Multivariate analysis of variance and regression

Multivariate analysis of variance, covariance and regression can be performed using procedures MANOVA and RMULTIVARIATE. MANOVA uses the ANOVA directive and is thus designed for balanced situations (see Section 4.7.2), while RMULTIVARIATE uses the Genstat regression facilities (Chapter 3) and so can be used for analyse unbalanced analyses of variance as well as ordinary regressions.

The analysis of multivariate distance (Gower & Krzanowski 1999) is another way of assessing a linear statistical model with multivariate data. It patitions the total squared distance between the units into the components that can be explained by each of the terms in the model, and assesses their significance by doing a permutation test. So, unlike multivariate analysis of variance, there is no need to assume multivariate Normality, (Note, though that you can also do permutation tests in MANOVA.)

6.6.1 The MANOVA procedure

MANOVA procedure

Performs multivariate analysis of variance and covariance (R.W. Payne & G.M. Arnold).

Options

PRINT = *string tokens*	Printed output required from the multivariate analysis of covariance (ssp, tests, permutationtest); default test
APRINT = *string tokens*	Printed output from the univariate analyses of variance of the y-variates (as for the ANOVA PRINT option); default *
UPRINT = *string tokens*	Printed output from the univariate unadjusted analyses of variance of the y-variates (as for the ANOVA UPRINT option); default *
CPRINT = *string tokens*	Printed output from the univariate analyses of variance of the covariates (as for the ANOVA CPRINT option); default *
TREATMENTSTRUCTURE = *formula*	Treatment formula for the analysis; if this is not set, the default is taken from the setting (which must already have been defined) by the TREATMENTSTRUCTURE directive
BLOCKSTRUCTURE = *formula*	Block formula for the analysis; if this is not set, the default is taken from any existing setting specified by the BLOCKSTRUCTURE directive and if neither has been set the design is assumed to be unstratified (i.e. to have a single error term)
COVARIATES = *variates*	Covariates for the analysis; by default MANOVA uses those listed by a previous COVARIATE directive (if any)
FACTORIAL = *scalar*	Limit on the number of factors in a treatment term
LRV = *pointer*	Contains elements first for the treatment terms and then the covariate term (if any), allowing the LRV's to be saved from one of the analyses; if a term is estimated in more than one stratum, the LRV is taken from the lowest stratum in which it is estimated
FPROBABILITY = *string token*	Printing of probabilities for F statistics and Chi-square variables (no, yes); default no

SELECTION = *string tokens*	Which test statistics to print when PRINT=test (lawleyhotellingtrace, pillaibartletttrace, roysmaximumroot, wilkslambda}; default lawl, pill, roys, wilk
NTIMES = *scalar*	Number of permutations to make when PRINT=perm; default 999
EXCLUDE = *factors*	Factors in the block model of the design whose levels are not to be randomized
SEED = *scalar*	Seed for the random number generator used to make the permutations; default 0 continues from the previous generation or (if none) initializes the seed automatically

Parameter

Y = *variates*	Y-variates for an analysis

Procedure MANOVA performs multivariate analysis of variance or covariance for balanced data.

The data variates are specified by the Y parameter. If any of the y-variates is restricted, the analysis will involve only the units not excluded by the restriction.

The model for the design is specified by options of the procedure. TREATMENTSTRUCTURE specifies a model formula to define the treatment terms in the analysis; if this is unset, MANOVA will use the model already defined by the TREATMENTSTRUCTURE directive (4.1.1), or will fail if that too has not been set. BLOCKSTRUCTURE defines the underlying structure of the design, and MANOVA will use the model (if any) previously defined by the BLOCKSTRUCTURE directive (4.2.1) if this is not set; this can be omitted if there is only one error term (i.e. if the design is unstratified). The COVARIATES option specifies any covariates; by default MANOVA will take those already listed (if any) by the COVARIATE directive. The FACTORIAL option can be used to set a limit on the number of factors in the terms generated from the treatment formula.

The LRV option allows a pointer to be saved containing an LRV structure for each treatment term, storing its canonical variate loadings, roots and trace. When covariates have been specified, the pointer will also contain a final LRV structure for the covariate term. If a term is estimated in more than one stratum, the LRV is taken from the stratum that occurs last in the BLOCKTERMS pointer.

The PRINT option indicates the output required from the multivariate analysis of covariance, with settings ssp to print the sums of squares and products matrices, tests to print the various test statistics, and permutationtest to calculate probabilities for the test statistics using a permutation test.

The SELECTION option controls which test statistics are given when PRINT=tests. The available statistics are Wilks' Lambda (with approximate F test), the Pillai-Bartlett trace, Roy's maximum root test and the Lawley-Hotelling trace. The default is to print them all.

By default, when PRINT=perm, MANOVA makes 999 random permutations and determines the probability of each test statistic from its distribution over these randomly generated datasets. The NTIMES option allows you to request another number of allocations, and the SEED option allows you to specify the seed to use for the random numbers used to make the permutations. The permutations are done by the RANDOMIZE directive, using the block model defined by the BLOCKSTRUCTURE option. The EXCLUDE option allows you to restrict the randomization so that one or more of the factors in the block model is not randomized. The most common situation where this is required is when one of the treatment factors involves time-order, which cannot be randomized.

The APRINT, UPRINT and CPRINT options control output from the univariate analyses of each of the y-variates, corresponding to ANOVA options PRINT, UPRINT and CPRINT, respectively (see 4.1.2, 4.1.3 and 4.3.1). FPROBABILITY controls whether or not probabilities are produced for

F-ratios and for Chi-square variables in the analysis; by default these are omitted.

Example 6.6.1

```
  2  " Data from Chatfield & Collins (1986) pages 142, 147, 149, 156."
  3  FACTOR [LEVELS=3; VALUES=3(1...3)] Block
  4  & [VALUES=(1...3)3] Treat,Plot
  5  VARIATE [NVALUES=9] V[1...3]
  6  READ [PRINT=errors] V[]
 10  MANOVA [PRINT=ssp,tests; TREATMENTSTRUCTURE=Treat; \
 11    BLOCKSTRUCTURE=Block/Plot; LRV=!p(TLRV)] V[]


Multivariate analysis of variance
=================================

Y-variates: V[1], V[2], V[3].


SSP matrices
============

Block  stratum
--------------


Residual
--------

         V[1]      0.7800
         V[2]      0.0300      0.7800
         V[3]      0.9600      0.6600      1.6800
                     V[1]        V[2]        V[3]

Degree of freedom: 2


Block.Plot  stratum
-------------------


Treat
-----

         V[1]       1.680
         V[2]       1.380       1.140
         V[3]      -1.260      -1.080       1.260
                     V[1]        V[2]        V[3]

Degree of freedom: 2


Residual
--------

         V[1]      0.4600
         V[2]      0.0300      0.3000
         V[3]     -0.4000     -0.4800      1.0600
                     V[1]        V[2]        V[3]

Degree of freedom: 4


Test statistics
===============


Block.Plot  stratum
-------------------
```

```
 Term  d.f.  Wilks' lambda  Rao F  n.d.f.  d.d.f. F prob.
Treat     2       0.004313   9.48       6       4   0.024

 Term  d.f.  Pillai-Bartlett  Roy's maximum  Lawley-Hotelling
                      trace      root test             trace
Treat     2            1.361         0.9932             146.2
```

```
  12  " Print the canonical variates information stored from the MANOVA."
  13  PRINT TLRV[]
```

```
            TLRV['Vectors']
                          1           2           3

        V[1]         10.846       0.575      -2.111
        V[2]         21.135       1.558       2.955
        V[3]         13.538       2.857       0.422


                          1           2           3
TLRV['Roots']        145.61        0.58        0.00


TLRV['Trace']         146.2
```

6.6.2 The RMULTIVARIATE procedure

RMULTIVARIATE procedure

Performs multivariate linear regression with accumulated tests (H. van der Voet).

Options

PRINT = *string tokens*	Controls printed output (model, summary, accumulated); default mode, summ, accu
RPRINT = *string tokens*	Controls printed output from the univariate regression analyses (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring); default *
FACTORIAL = *scalar*	Limit for expansion of model terms; default 3
NOMESSAGE = *string tokens*	Which warning messages to suppress when fitting the complete model - messages are always suppressed when fitting models for individual tests (aliasing, marginality); default *
RESULTS = *pointer*	To save results from accumulated and summary tests in a pointer containing terms, degrees of freedom of terms, Wilks' Lambda, Rao's F-statistic, degrees of freedom for numerator and denominator of Rao's F and P-value of Rao's F

Parameter

TERMS = *formula*	List of explanatory variates and factors, or model formula

RMULTIVARIATE calculates hierarchical tests, based on Wilks' Lambda, for the terms in a multivariate linear regression model. The use of RMULTIVARIATE must be preceded by a MODEL statement (3.1.1) to define the response variables and, if required, a vector of weights and an offset. Generalized linear models are not allowed. Note that the FIT directive (3.1.2) performs a regression analysis for each of the response variables in turn, whereas RMULTIVARIATE performs multivariate modelling and testing.

The TERMS parameter specifies the model terms to be assessed. The FACTORIAL option sets a limit on the number of factors and variates in each term, similarly to the FACTORIAL option of FIT; by default this is 3. Printed output from the multivariate analysis is controlled by the PRINT option: model gives a description of the model, summary prints test results for the full model, while accumulated gives accumulated test results for each term in the model formula. The RPRINT option controls output from univariate regressions of the individual variates, which are performed (by FIT) in order to calculate the multivariate analysis. The NOMESSAGE option can be used to suppress aliasing and marginality warning messages when fitting the full model.

The RESULTS option can be used to save both accumulated and summary test results in a pointer. This pointer contains a text structure saving the individual model terms and six variates saving the number of degrees of freedom associated with each term, Wilks' Lambda, Rao's F-statistic, degrees of freedom for numerator and denominator of Rao's F-statistic and the calculated P-value. Directives RDISPLAY and RKEEP can be used subsequent to RMULTIVARIATE, to display further output and store results from the univariate regressions of each response variate.

Units with one or more missing values in any term are excluded from the analysis. This implies that successive calls of RMULTIVARIATE may give different test results if terms with missing values are dropped or added. Any restriction applied to vectors used in the regression model will apply also to the results from RMULTIVARIATE.

Example 6.6.2

```
  2  " Data from Chatfield & Collins (1986) pages 143 and 176."
  3  FACTOR   [NVALUES=18; LEVELS=!(4,20,34)] temp
  4  FACTOR   [NVALUES=18; LABELS=!T(Male,Female)] sex
  5  GENERATE temp,sex,3
  6  VARIATE  [NVALUES=18] initweight,finalweight,tumourweight
  7  READ     initweight,finalweight,tumourweight

   Identifier   Minimum      Mean   Maximum    Values   Missing
    initweight     17.20     19.67     21.56        18         0
   finalweight     15.90     19.51     23.30        18         0
  tumourweight    0.1600    0.2633    0.4500        18         0

 14  MODEL    finalweight,tumourweight
 15  RMULTIVARIATE [RPRINT=accumulated] initweight + temp * sex

Multivariate regression analysis
================================

Response variates:  finalweight, tumourweight
Terms: initweight + temp*sex

Regression analysis
===================

Accumulated analysis of variance
--------------------------------

Response variate: finalweight

Change                          d.f.         s.s.         m.s.      v.r.
+ initweight                       1       14.045       14.045      6.40
+ temp                             2       22.630       11.315      5.15
+ sex                              1        1.553        1.553      0.71
+ temp.sex                         2        2.624        1.312      0.60
Residual                          11       24.157        2.196

Total                             17       65.007        3.824
```

```
Response variate: tumourweight

Change                          d.f.         s.s.         m.s.      v.r.
+ initweight                       1     0.000166     0.000166      0.05
+ temp                             2     0.025480     0.012740      3.76
+ sex                              1     0.055261     0.055261     16.29
+ temp.sex                         2     0.006772     0.003386      1.00
Residual                          11     0.037320     0.003393

Total                             17     0.125000     0.007353


Summary of multivariate analysis
--------------------------------

      term  df  WLambda     RaoF  df1  df2  pvalue
 All terms   6   0.1025     3.54   12   20   0.006


Accumulated multivariate tests
------------------------------

      term  df  WLambda     RaoF  df1  df2  pvalue
initweight   1   0.5977     3.37    2   10   0.076
      temp   2   0.3123     3.95    4   20   0.016
       sex   1   0.3464     9.43    2   10   0.005
  temp.sex   2   0.7830     0.65    4   20   0.633
```

6.6.3 The MVAOD procedure

MVAOV procedure

Does an analysis of distance of multivariate data (R.W. Payne & R.P. White).

Options

PRINT = *string tokens*	Controls printed output (aodtable, permutationtest); default aodt
TERMS = *formula*	Model terms to fit in the analysis; must be specified
FACTORIAL = *scalar*	Limit on the number of factors or variates in a term for it to be included in the analysis; default 3
NTIMES = *scalar*	Number of permutations to use in the permutation test; default 999
SEED = *scalar*	Seed for the random number generator used to make the permutations; default 0 continues from the previous generation or (if none) initializes the seed automatically

Parameters

DATA = *symmetric matrices*	Supplies the squared distances between the data points
SSD = *variates*	Saves the sums of squared distances
DF = *variates*	Saves the numbers of degrees of freedom
PRPERMUTATION = *variates*	Saves probabilities from the permutation test
DISTANCES = *pointers*	Contains a symmetric matrix of distances for each model term

MVAOD implements the analysis of multivariate distance devised by Gower & Krzanowski (1999). This is useful when you have units whose positions in multi-dimensional space may be explained by a linear statistical model. It provides a breakdown of the sums of squared distances between the units, similar to that provided for sums of squares in an analysis of variance. So, the total squared distance between the units is partitioned into the components that can be explained by

each of the terms in the model. These cannot be tested directly as in an analysis of variance, as it is unclear what probability distributions would be appropriate. Instead the importance of the terms can be assessed by doing a permutation test, in which the several permutations of the units are made, and the significances of the sums of squared distances from the observed data are calculated by seeing where they lie in the distribution of values obtained from all the analyses (the original analysis and those of the permuted data sets).

The squared distances between the units must be supplied in a symmetric matrix, using the DATA parameter. In some situations, these may be actual distances. Alternatively, the units may often be described by a collection of attribute ranging from continuous measurements to categorical variables, like the presence or absence of a particular feature. In these circumstances, the FSIMILARITY directive (6.1.2) can be used combine these attributes to give a symmetric matrix that represents the similarity between each pair of units. This can then be converted into a squared distance matrix, for example, by subtracting the similarities from one. (So MVAOD can be regarded as providing an alternative to multivariate analysis of variance, for units whose attributes are not all continuous variables.)

The model to fit in the analysis is specified by the TERMS option. The FACTORIAL option sets a limit on the number of factors of variates that the terms can contain; any terms with more factors of variates are deleted from the analysis.

Printed output is controlled by the PRINT option, with settings:

aodtable	for an analysis-of-distance table, giving the sums of squared distances and numbers of degrees of freedom for each model term; and
permutationtest	adds a column to the analysis-of-distance table containing probabilities from the permutation test.

The NTIMES option specifies the number of permutations to perform; the default is 999. The SEED option specifies the seed to use to generate the random numbers that are used to select the permutations; the default of zero continues the sequence of random numbers from a previous generation or, if none have yet been used in this Genstat job, it initializes the seed automatically. MVAOD checks whether NTIMES is greater than the number of possible permutations available for the data set. If so, it does an exact test instead, which uses each possible permutation once.

The SSD, DF and PRPERMUTATION parameters allow you to save the sums of squared distances, degrees of freedom and permutation probabilities. These are each saved in a variate, with each unit labelled by the name of the model term concerned. There are also have two final units in each variate to save the corresponding information for residual and the total.

The DISTANCES parameter can save a pointer containing a symmetric matrix for each model term. Each matrix has a row for each combination of levels of the factors in the corresponding term, and its values are the distances between the factor combinations in the multi-dimensional space defined by the possible effects of the term. So, to investigate the relationships between the effects of the term, you could convert the DISTANCES to similarities, and then use them as input for a principal coordinates analysis (6.10.1).

Example 6.6.3 analyses the data set in Gower & Krzanowski (1999). Note that the analysis here differs from theirs, as they do an unweighted analysis that ignores the differenses in group size. The analysis shows evidence for main effects of the factors N and S.

Example 6.6.3

```
" Data from Gower & Krzanowski 1999, Applied Statistics, 48, 505-519."
SPLOAD       [PRINT=*] FILE='%gendir%/examples/Publicbad.gsh'
" Form similarity matrix using city-block metric."
FSIMILARITY  [SIMILARITY=pbsimilarity] publicbad[]; TEST=cityblock
" Convert to squared distances."
CALCULATE    pbdistances = 1 - pbsimilarity
" Factorial model - note: this is on a different scale and gives a
  slightly different breakdown from Table 2 of Gower & Krzanowski,
```

```
 -10    as their analysis ignored differences in group size.
 -11    Only 99 permutations are made, to save computing time."
  12  MVAOD        [PRINT=aod,permutation; TERMS=N*T*S*G; NTIMES=99; SEED=629856]\
  13               pbdistances

Analysis of distance
====================


Term        d.f.  Sum of squared     pr.
                        distance
N              1           1.644   0.010
T              1           0.115   0.520
S              1           0.559   0.010
G              1           0.165   0.340
N.T            1           0.173   0.270
N.S            1           0.188   0.170
T.S            1           0.143   0.450
N.G            1           0.111   0.590
T.G            1           0.140   0.420
S.G            1           0.084   0.830
N.T.S          1           0.183   0.310
N.T.G          1           0.069   0.860
N.S.G          1           0.097   0.760
T.S.G          2           0.058   1.000
Residual     224          30.818
Total        239          34.547

Probabilities determined from 99 random permutations
```

6.7 Ridge and principal component regression: the RIDGE procedure

RIDGE procedure

Produces ridge regression and principal component regression analyses (A.J. Rook & M.S. Dhanoa).

Options

PRINT = *string token*	What to print (correlation, pcp, ridge); default corr
PLOT = *string token*	Graphical output required (ridgetrace); default *

Parameters

Y = *variates*	Response variate in regression model
X = *pointers*	Containing explanatory variates in regression model

Procedure RIDGE produces analyses for identifying and overcoming collinearity among the independent variates in a multiple regression analysis. The correlation matrix, variance inflation factors (the diagonal elements of the inverse of the correlation matrix) and the ratio of the squared error in the least squares regression coefficients to the expected squared error in orthogonal data are calculated. Principal component regressions excluding 1, 2 or 3 minor principal axes are calculated and transformed back to the original variables on either the original or standardized scale. The "Positive correlation spread association" (PCSA) (Vinod 1976) is also calculated. This is an overall measure of the suitability of the data for the application of principal component regression and ridge regression. Ridge regressions (Hoerl & Kennard 1970) are calculated and the ridge coefficients are printed together with 2 indices of stability proposed by Vinod (1976): the index of stability of relative magnitudes (ISRM) and the numerical largeness of more significant regression coefficients (NLMS). These are 0 and 1 respectively in orthogonal data. High-resolution graphs of the ridge trace can be plotted against Hoerl & Kennard's *k* scale

and Vinod's *m* scale.

The parameters of the procedure are used to input the data: the `Y` parameter supplies the y-variate, and the `X` parameter specifies a pointer containing the x-variates. None of these variates must be restricted nor contain missing values.

Printed output is controlled by the `PRINT` option: `correlation` prints the correlation matrix, variance inflation factors and ratio of squared error to that in orthogonal data, `pcp` prints principal component analysis and principal component regression, and `ridge` prints ridge coefficients and stability parameters.

Graphical output is controlled by the `PLOT` option: `ridgetrace` produces ridge traces.

Example 6.7

```
  2  " Data on French economy from Chatterjee & Price
 -3    (1991, pages 182, 185, 213, 218 and 220)."
  4  VARIATE [NVALUES=11] Import,Doprod,Stock,Consum
  5  READ    Import,Doprod,Stock,Consum

   Identifier   Minimum      Mean   Maximum    Values   Missing
       Import     15.90     21.89     28.10        11         0
       Doprod     149.3     194.6     239.0        11         0
        Stock    0.7000     3.300     5.600        11         0
       Consum     108.1     139.7     167.6        11         0

 17  RIDGE [PRINT=correlation,pcp] Import; !p(Doprod,Stock,Consum)

Regression analyses for multicollinear data
===========================================

Correlation matrix among predictor variables
--------------------------------------------

       Doprod       1.000
        Stock       0.026       1.000
       Consum       0.997       0.036       1.000
                   Doprod       Stock      Consum


Variance inflation factors
--------------------------

      Doprod       Stock      Consum
      186.00        1.02      186.11


Ratio of squared error in OLS estimates of regression coefficients
to error if data were orthogonal
-------------------------------------------------------------------

       124.4


Principal components analysis
=============================

Latent roots
------------

          1           2           3
       19.99        9.98        0.03


Percentage variation
--------------------

          1           2           3
       66.64       33.27        0.09
```

```
Trace
-----

       30.00


Latent vectors (loadings)
-------------------------

                        1           2           3
       Doprod     0.70633     0.03569    -0.70698
        Stock     0.04350    -0.99903    -0.00697
       Consum     0.70654     0.02583     0.70720


Regression analysis
===================

 Response variate: stany
     Fitted terms: pcp[1], pcp[2], pcp[3]


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.  F pr.
Regression       3      9.91897      3.30632    326.41  <.001
Residual         8      0.08103      0.01013
Total           11     10.00000      0.90909

Percentage variance accounted for 99.0
Standard error of observations is estimated to be 0.101.

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
           11        0.970        0.70


Estimates of parameters
-----------------------

Parameter      estimate         s.e.      t(8)  t pr.
pcp[1]           0.6900       0.0225     30.65  <.001
pcp[2]          -0.1913       0.0319     -6.01  <.001
pcp[3]           -1.160        0.614     -1.89  0.095


Regression coefficients of original variables on standardized scale
-------------------------------------------------------------------

                 Ordinary        Smallest    Two smallest  Three smallest
                    least       principal       principal       principal
                  squares       component      components      components
                                 excluded        excluded        excluded
    Doprod         -0.3393          0.4805          0.4874               0
     Stock          0.2130          0.2211          0.0300               0
    Consum          1.3027          0.4826          0.4875               0


Regression coefficients of original variables on original scale
---------------------------------------------------------------

                 Ordinary        Smallest    Two smallest  Three smallest
                    least       principal       principal       principal
                  squares       component      components      components
                                 excluded        excluded        excluded
  Constant        -10.128          -9.130          -7.746          21.891
    Doprod         -0.051           0.073           0.074           0.000
     Stock          0.587           0.609           0.083           0.000
    Consum          0.287           0.106           0.107           0.000
 R-squared          0.990           0.987           0.952           0.091
```

```
Correlation of standardised response variable with principal component scores
------------------------------------------------------------------------------

      pcp[1]      pcp[2]      pcp[3]
      0.9756     -0.1911     -0.0602


Positive correlation spread association
---------------------------------------

      0.8180
```

6.8 Partial least squares: the PLS procedure

PLS procedure

Fits a partial least squares regression model (I. Wakeling & N. Bratchell).

Options

PRINT = *string tokens*	Printed output required (data, xloadings, yloadings, ploadings, scores, leverages, xerrors, yerrors, scree, xpercent, ypercent, predictions, groups, estimates, fittedvalues); default esti, xper, yper, scor, xloa, yloa, ploa
NROOTS = *scalar*	Number of PLS dimensions to be extracted
YSCALING = *string token*	Whether to scale the Y variates to unit variance; (yes, no); default no
XSCALING = *string token*	Whether to scale the X variates to unit variance; (yes, no); default no
NGROUPS = *scalar*	Number of cross-validation groups into which to divide the data; default 1 (i.e. no cross-validation performed)
SEED = *scalar* or *factor*	A scalar indicating the seed value to use when dividing the data randomly into NGROUPS groups for the cross-validation or a factor to indicate a specific set of groupings to use for the cross-validation; default 0
LABELS = *text*	Sample labels for X and Y that are to be used in the printed output; defaults to the integers 1...*n* where *n* is the length of the variates in X and Y
PLABELS = *text*	Sample labels for XPREDICTIONS that are to be used in the printed output; default uses the integers 1, 2 ...

Parameters

Y = *pointers*	Pointer to variates containing the dependent variables
X = *pointers*	Pointer to variates containing the independent variables
YLOADINGS = *pointers*	Pointer to variates used to store the *Y* component loadings for each dimension extracted
XLOADINGS = *pointers*	Pointer to variates used to store the *X* component loadings for each dimension extracted
PLOADINGS = *pointers*	Pointer to variates used to store the loadings for the bilinear model for the *X* block
YSCORES = *pointers*	Pointer to variates used to store the *Y* component scores for each dimension extracted
XSCORES = *pointers*	Pointer to variates used to store the *X* component scores for each dimension extracted

B = *matrices*	A diagonal matrix containing the regression coefficients of YSCORES on XSCORES for each dimension
YPREDICTIONS = *pointers*	A pointer to variates used to store predicted *Y* values for samples in the prediction set
XPREDICTIONS = *pointers*	A pointer to variates containing data for the independent variables in the prediction set
ESTIMATES = *matrices*	An n_X+1 by n_Y matrix (where n_X and n_Y are the numbers of variates contained in X and Y respectively) used to store the PLS regression coefficients for a PLS model with NROOTS dimensions
FITTEDVALUES = *pointers*	Pointer to variates used to store the fitted values for each Y variate
LEVERAGES = *variates*	Variate used to store the leverage that each sample has on the PLS model
PRESS = *variates*	Variate used to contain the Predictive Residual Error Sum of Squares for each dimension in the PLS model, available only if cross-validation has been selected
RSS = *variates*	Variate used to store the Residual Sum of Squares for each dimension extracted
YRESIDUALS = *pointers*	Pointer to variates used to store the residuals from the *Y* block after NROOTS dimensions have been extracted, uncorrected for any scaling applied using YSCALING
XRESIDUALS = *pointers*	Pointer to variates used to store the residuals from the *X* block after NROOTS dimensions have been extracted, uncorrected for any scaling applied using XSCALING
XPRESIDUALS = *pointers*	Pointer to variates used to store the residuals from the XPREDICTIONS block after NROOTS dimensions have been extracted
FTEST = *pointers*	Pointer to save the results from the Osten F test (when NGROUPS > 1)

The regression method of Partial Least Squares (PLS) was initially developed as a calibration method for use with chemical data. It was designed principally for use with overdetermined data sets and to be more efficient computationally than competing methods such as principal components regression. If *Y* and *X* denote matrices of dependent and independent variables respectively, then the aim of PLS is to fit a bilinear model having the form $T=XW$, $X=TP'+E$ and $Y=TQ'+F$, where *W* is a matrix of coefficients whose columns define the PLS factors as linear combinations of the independent variables. Successive PLS factors contained in the columns of *T* are selected both to minimise the residuals in *E* and simultaneously to have high squared covariance with a single *Y* variate (PLS1) or a linear combination of multiple *Y* variates (PLS2). The columns of *T* are constrained to be mutually orthogonal. See Helland (1988) or Hoskuldsson (1988) for a more comprehensive description of the method.

The PLS procedure allows the calculation of PLS1 and PLS2 models with cross-validation to assist in the determination of the correct number of dimensions to include in the model. If the NGROUPS option is set, the data are randomly divided into groups; samples in each group are then modelled from the remaining samples only. The sum of squares of differences between these "leave out predictions" and the observed values of *Y* are called PRESS. Many tests of significance for determining the correct number of dimensions are based on comparing values of PRESS for PLS models of varying rank. Values of PRESS are used in the procedure to perform Osten's (1988) test of significance, and may also be plotted in a scree diagram. In addition to the factor scores, factor loadings and residuals, the procedure also calculates a

leverage measure (Naes & Martens 1989, page 276) and a single linear combination of the *X* variables (ESTIMATES) which summarises the entire PLS model.

To use a PLS model to make predictions from new observations on the *X* variables, two methods are available. Either the user may do this manually by using the model as specified in the estimates matrix, or the new *X* data may be specified beforehand as the pointer to variates XPREDICTIONS and the corresponding predictions obtained as YPREDICTIONS.

The data for PLS are supplied using the X and Y parameters, as pointers to variates containing the columns of the *X* and *Y* matrices. Other parameters allow output to be saved in appropriate data structures. The procedure will fail if there are missing values present in either the X or Y variates.

The procedure will work with restricted variates, fitting a PLS model to the subset of objects indicated by the restriction. If there are different restrictions on different data variates then these restrictions will be combined and the analysis performed on the subset of samples that is common to all the restrictions. Note that the unrestricted length of all of the data variates must be the same and the number of samples in the common subset must be at least three. Any restrictions on a text supplied for the LABELS option or a factor for the SEED option will be ignored. On exit from the procedure all the data variates, and if supplied the SEED factor and LABELS text, will all be returned restricted to the common subset of samples. Output data structures that correspond to the samples (i.e. XSCORES, YSCORES, FITTEDVALUES, LEVERAGES, YRESIDUAL and XRESIDUAL) will also be returned restricted to the common subset, and missing values will be used for those values that have been restricted out.

When restricted data are supplied and LABELS are also given then the appropriate subset of labels will be appear in the output; if LABELS are not defined then default labels reflecting the position of the restricted data in the unrestricted variate will be used instead.

No restrictions are allowed in the variates supplied in the XPREDICTIONS parameter or the PLABELS option.

Output from PLS is selected using the following settings of the PRINT option.

data	the unscaled data values (with labels).
xloadings	*X*-component loadings (columns of the matrix *W* – see above).
yloadings	variable loadings for the bilinear model of the matrix of dependent variables. Note that these are standardized to unit length and are not the same as the columns of the matrix *Q* above. To obtain *Q*, form the matrix *C*, whose columns are the standardized loadings, and post-multiply by the diagonal matrix supplied as the output parameter *B*.
ploadings	variable loadings for the bilinear model of the matrix of independent variables (columns of the matrix P – see above).
scores	*X* and *Y* component scores. The *X* component scores are the columns of the matrix *T* and are mutually orthogonal. The *Y* component scores, usually given the symbol *u*, are not in fact needed in the calculation of the PLS model unless an iterative algorithm is used (see method section). They are provided here for completeness, as sometimes it is useful to plot the *Y* component scores against the *X* component scores to give a visual indication of the degree of fit for each PLS dimension.
leverages	measure of leverage.
xerrors	residual sum of squares and residual standard deviations for all the independent variables. When NGROUPS>1

	additional statistics are calculated from the cross-validated residuals, derived when each object is left out. The PRESS value is equal to the sum of squares of cross-validated standard deviations for each X variable multipled by $N-1$, where N is the total number of observations. The cross-validated standard deviations may therefore be used to measure the predictive ability of the model for each of the variables.
`yerrors`	residual sum of squares and residual standard deviations for all the dependent variables (see xerrors above).
`scree`	scree diagram of PRESS.
`xpercent`	percentage variance explained for the X variables.
`ypercent`	percentage variance explained for the Y variables.
`predictions`	predicted values for any observations that were not included in the PLS model but were supplied using the `XPREDICTIONS` parameter.
`groups`	details of groupings used for cross-validation.
`estimates`	estimated PLS regression coefficients.
`fittedvalues`	fitted values from the PLS regressions.

The default settings are `estimates`, `xpercent`, `ypercent`, `scores`, `xloadings`, `yloadings`, `ploadings`.

Example 6.8

```
  2  " 24 calibration samples used to determine the protein content of
 -3    wheat from spectroscopic readings at six different wavelengths
 -4    (Fearn, T., 1983, Applied Statistics 32, 73-79)."
  5  VARIATE [NVALUES=24] L[1...6],%Protein[1]
  6  READ    L[1...6],%Protein[1]

   Identifier   Minimum      Mean   Maximum    Values   Missing
         L[1]     450.0     487.4     592.0        24         0
         L[2]     111.0     140.5     229.0        24         0
         L[3]     233.0     264.5     360.0        24         0    Skew
         L[4]     352.0     390.6     484.0        24         0
         L[5]     340.0     400.3     524.0        24         0
         L[6]    -16.00    0.2083     51.00        24         0    Skew
  %Protein[1]     7.750     9.966     12.55        24         0

 19  " Fit a 3 dimensional PLS model to the standardized data using
-20    leave-one-out cross-validation. All three dimensions are
-21    significant using Osten's test"
 22  PLS [PRINT=estimates,xpercent,ypercent,xloadings,yloadings,ploadings;\
 23      NROOTS=3; NGROUPS=24; SEED=708003; XSCALING=yes; YSCALING=yes]\
 24      Y=%Protein; X=L

Partial least-squares regression analysis
=========================================

PRESS and Osten's F-test for significance of a dimension
--------------------------------------------------------

                  PRESS         F  d.f. 1  d.f. 2  Prob > F
     Dim  1      18.897      7.48       6     138    <0.001
     Dim  2      10.307     18.34       6     132    <0.001
     Dim  3       0.981    199.71       6     126    <0.001
```

```
Estimates of PLS regression coefficients
----------------------------------------

    YLAB %Protein[1]
   CXLAB
Constant     40.5744
    L[1]     -0.0370
    L[2]      0.1524
    L[3]      0.1247
    L[4]     -0.1846
    L[5]      0.0129
    L[6]     -0.0653


Percentage of the Y variances explained
---------------------------------------

          %Protein[1]
   Dim  1        22.5
        2        40.3
        3        35.0


Percentage of the X variances explained
---------------------------------------

             L[1]     L[2]     L[3]     L[4]     L[5]     L[6]
   Dim  1    99.4     98.8     99.3     98.8     91.4     99.0
        2     0.1      1.1      0.7      0.3      7.0      0.0
        3     0.3      0.0      0.0      0.8      1.6      0.0


X component loadings
--------------------

         X   Dim  1   Dim  2   Dim  3
      L[1]   0.4109   0.0348  -0.2988
      L[2]   0.4857  -0.4823   0.1272
      L[3]   0.4732  -0.3908   0.1395
      L[4]   0.3371   0.5289  -0.5292
      L[5]   0.3159   0.5648   0.7540
      L[6]   0.3974   0.1210  -0.1630


P loadings
----------

         X   Dim  1   Dim  2   Dim  3
      L[1]   0.4160  -0.1415  -0.3155
      L[2]   0.4148  -0.4073   0.1318
      L[3]   0.4157  -0.3085   0.1283
      L[4]   0.4148   0.2167  -0.5365
      L[5]   0.3989   1.0096   0.7508
      L[6]   0.4152   0.0249  -0.1294


Y component loadings
--------------------

                Y   Dim  1   Dim  2   Dim  3
      %Protein[1]   1.0000  -1.0000   1.0000
```

Orthogonal partial least squares regression can be performed by the `OPLS` procedure.

6.9 Canonical correlation analysis: the CANCORRELATION procedure

CANCORRELATION procedure

Does canonical correlation analysis (P.G.N. Digby).

Option

PRINT = *string tokens*	Printed output from the analysis (correlations, pcoeff, qcoeff, pscores, qscores); default * i.e. no output

Parameters

PVARIATES = *pointers*	Pointer to P-set of variates to be analysed
QVARIATES = *pointers*	Pointer to Q-set of variates to be analysed
CORRELATIONS = *diagonal matrices*	Stores the canonical correlations from each analysis
PCOEFF = *matrices*	Stores the coefficients for the P-set of variates
QCOEFF = *matrices*	Stores the coefficients for the Q-set of variates
PSCORES = *matrices*	Stores the unit scores from the P-set of variates
QSCORES = *matrices*	Stores the unit scores from the Q-set of variates

Procedure CANCORRELATION provides canonical correlation analysis (see, for example, Mardia, Kent & Bibby 1979 or Digby & Kempton 1987). The data for the procedure consists of two pointers specified by the PVARIATES and QVARIATES parameters; these contain two sets of variates. The variates may have missing values, or be restricted. Any unit for which any of the variates is missing will be excluded from the analysis, and any restrictions on the variates must be consistent. The other parameters allow results to be saved from the analysis.

Printed output is controlled by the option PRINT with settings: correlations to print the canonical correlations (also expressed as percentages, and cumulative percentages, of their total), pcoeff to print the canonical correlation coefficients for the P-set of variates, qcoeff to print the canonical correlation coefficients for the Q-set of variates, pscores to print the canonical correlation scores for the units calculated from the P-set of variates, and qscores to print the canonical correlation scores for the units calculated from the Q-set of variates.

Example 6.9

```
  2  " Data from Table 3.7 of Digby & Kempton (1987)."
  3  TEXT [VALUES='1d','3a','3d','4a','4d','7a','7d','8a','8d','9a','9d', \
  4    '10a','10d','11/1a','11/1d','11/2a','11/2d','14a','14d','16a','16d',\
  5    '17a','17d','18d'] Plot
  6  POINTER [VALUES=N,Nstar,P,K,Lime] Treatments
  7  & [VALUES=Axis_1,Axis_2,Axis_3,Axis_4] Species
  8  VARIATE [NVALUES=Plot] Treatments[],Species[]
  9  READ [PRINT=errors] Treatments[]
 14  READ [PRINT=errors] Species[]
 23  CALCULATE Species[] = Species[] / 100
 24  CANCORRELATION [PRINT=correlations,pcoeff,qcoeff] Treatments; Species

Canonical correlation analysis
==============================
```

```
Canonical correlations
----------------------

                CA_Corrs      %Corrs   Cum%Corrs

            1      0.9804       35.99       35.99
            2      0.8994       33.02       69.01
            3      0.5907       21.69       90.70
            4      0.2533        9.30      100.00


Loadings for the P-set of variates
----------------------------------

                        1           2           3           4
   Treatments
            N      0.1515      0.0031      0.0813      0.0857
        Nstar      0.0264     -0.1443      0.0232      0.3538
            P      0.0409     -0.1077      0.1249      0.1487
            K      0.0794     -0.1956     -0.3124     -0.3109
         Lime     -0.1112     -0.2150      0.2632     -0.1681


Loadings for the Q-set of variates
----------------------------------

                        1           2           3           4
      Species
       Axis_1    -0.01003     0.06995     0.01411     0.02015
       Axis_2     0.09108     0.00145     0.00622     0.02793
       Axis_3     0.03738     0.00317     0.15526    -0.03726
       Axis_4    -0.03252    -0.01647     0.07699     0.11913
```

6.10 Principal coordinates analysis

Principal coordinates analysis (or metric scaling) is a method of generating an "ordination" of a set of objects. The term *ordination* is used mainly in biometrics, particularly in ecology, where it usually refers to attempts to order a set of objects along some environmental gradient. Archaeologists use the term *seriation* to refer to the same set of techniques, whilst the phrase *multidimensional scaling* is used in some other areas. There is no fixed statistical terminology for these methods; however, they have in common an attempt to "order" a set of objects in one dimension with a generalization to give some useful distribution of the objects in multidimensional space. Other ordination methods available in Genstat include principal components analysis (6.2.1) and correspondence analysis (6.13). These methods operate with data in the form of a data matrix or a two-way table. Principal coordinates analysis operates on a symmetric matrix measuring the associations between a set of objects, which can be produced using the methods in Sections 6.1.2 - 6.1.4.

Suppose that symmetric matrix, A, contains values representing the associations amongst a set of n units. Principal coordinates analysis (Gower 1966) attempts to find a set of points for the n units in a multidimensional space so that the squared distance between the ith and jth points is given by:

$$d_{ij} = a_{ii} + a_{jj} - 2a_{ij}$$

If A is a similarity matrix (see 6.4.1) then a_{ii} and a_{jj} are both equal to 1 (as every unit is completely similar to itself). So this is equivalent to:

$$d_{ij} = 2 \times (1 - a_{ij})$$

Thus similar units are placed close together and dissimilar units are further apart.

Often the data consist of distances rather than similarities (6.1.4). If B is a distance matrix (i.e. element b_{ij} is the observed distance between the ith and jth units), then the preliminary transformation

$$A = - B \times B / 2$$

will give points with inter-point squared distance

$$\begin{aligned} d_{ij} &= a_{ii} + a_{jj} - 2a_{ij} \\ &= 0 + 0 - 2 \times (-b_{ij} \times b_{ij} / 2) \\ &= b_{ij}^2 \end{aligned}$$

Therefore the analysis will give points whose inter-point distances match the supplied distances.

The coordinates of the points are arranged so that their centroid, or mean position, is at the origin. Furthermore they are arranged relative to their principal axes, so that the first dimension of the solution gives the best one-dimensional fit to the full set of points, the first two dimensions give the best two-dimensional fit, and so on. The analysis also gives the distances of the points from their centroid, the origin. Associated with each dimension of the set of coordinates is a latent root which is the sum of squares of the coordinates of all the points in that dimension.

For n units, if there is an exact solution it will be in at most $n-1$ dimensions. However, such a solution may not always be available, because the matrix of distances derived from the associations may not be Euclidean: that is, the distances may not be reproducible by points in a Euclidean space of any number of dimensions. If an incomplete solution results, either because the Euclidean property does not hold or because not all the dimensions are to be used, then a residual can be calculated for each unit; this residual is the difference between (a) the distance from the point for that unit in the incomplete solution to the centroid, and (b) the equivalent distance derived from the original data. When the Euclidean property does not hold, some of the residuals may be complex numbers; Genstat represents these as missing values.

If you regard a set of p variables of length n as giving the coordinates of a set of n points in p dimensions, then you can construct the symmetric matrix with values that give the Euclidean distance between the n points (for example B above). If this matrix is then transformed to an association matrix as

$$A = -B \times B/2$$

the principal coordinates analysis of the association matrix will give identical results to a principal components analysis of the original set of variables.

Another special case of principal coordinates analysis occurs when a within-group SSPM structure is to be analysed. Now you can calculate Mahalanobis squared distances amongst the group means as

$$d_{ij}^2 = (x_i - x_j)\, W^{-1}\, (x_i - x_j)'$$

where x_i is the row vector of means for the ith group, and W is the pooled within-group covariance matrix. These squared distances can be transformed to associations, and used as input to principal coordinates analysis to obtain an ordination of the groups. In general, results from this will be different from those of canonical variates analysis, since the ordination operates on a Mahalanobis distance matrix unweighted by group size, whereas the `CVA` directive (6.3.1) operates on a matrix of between-group sums of squares and products, weighted by group size.

Having obtained an ordination, you may sometimes want to add points to the ordination for additional units. For example, with canonical variates analysis, Genstat gives the scores for the group means; you may want to add points to the group-mean ordination for each of the units. It is easy to take the data for the new units, apply the centring of the analysis, and use the loadings matrix to get coordinates for the new units.

When you use principal coordinates analysis to analyse an association matrix, there is no loadings matrix. However, if you know the squared distances of the new units from the old, the technique of Gower (1968) can be used to add points to the ordination for the new units. You can do this in Genstat by using the `ADDPOINTS` directive (6.10.2), together with results saved from the preceding `PCO` directive.

The assumption that the squared inter-point distance is directly related to the values in the association matrix may be too strict with some types of data, for example in psychology. This has led to a family of methods known as *non-metric scaling* or *multidimensional scaling*, several variants of which are provided by the `MDS` directive in Section 6.12.

6.10.1 The PCO directive

PCO directive

Performs principal coordinates analysis, also principal components and canonical variates analysis (but with different weighting from that used in CVA) as special cases.

Options

PRINT = *string tokens*	Printed output required (roots, scores, loadings, residuals, centroid, distances); default * i.e. no printing
NROOTS = *scalar*	Number of latent roots for printed output; default * requests them all to be printed
SMALLEST = *string token*	Whether to print the smallest roots instead of the largest (yes, no); default no

Parameters

DATA = *identifiers*	These can be specified either as a symmetric matrix of similarities or transformed distances or, for the canonical variates analysis, as an SSPM containing within-group sums of squares and products etc or, for principal components analysis, either as a pointer containing the variates of the data matrix or as a matrix storing the variates by columns
LRV = *LRVs*	Latent vectors (i.e. coordinates or scores), roots and trace from each analysis
CENTROID = *diagonal matrices*	Squared distances of the units from their centroid
RESIDUALS = *matrices* or *variates*	Distances of the units from the fitted space
LOADINGS = *matrices*	Principal component loadings, or canonical variate loadings
DISTANCES = *symmetric matrices*	Computed inter-unit distances calculated from the variates of a data matrix, or inter-group Mahalanobis distances calculated from a within-group SSPM
SAVE = *pointers*	Saves details of the analysis; if unset, an unnamed save structure is saved automatically (and this can be accessed using the GET directive)

In its simplest form, the PCO directive needs to be supplied with a symmetric matrix, with values giving the associations amongst a set of objects. This could, for example, be a similarity matrix (6.1.2). The DATA parameter provides the symmetric matrix of associations and the PRINT option specifies what is to be printed, using the following settings of the PRINT option:

roots	prints the latent roots and trace;
scores	prints the principal coordinate scores;
loadings	when the directive is being used for principal components analysis or canonical variates analysis, this specifies that the loadings from the analysis are to be printed;
residuals	prints the residuals, this is relevant only if results are to be printed corresponding to only some of the latent roots;
centroid	prints the distances (not squared distances) of each unit from their overall centroid;
distances	prints the matrix of inter-unit distances (not squared

distances).

The NROOTS and SMALLEST options control the printed output of roots, scores, loadings, and residuals. By default, results are printed for all the roots, but you can set the NROOTS option to specify a lesser number. If option SMALLEST has the default setting no these are taken to be the largest roots, but if you set SMALLEST=yes the results are for the smallest non-zero roots. The inter-unit distances are unaffected by the setting of the NROOTS option.

Nathanson (1971) gives squared distances amongst ten types of galaxy: those of an elliptical shape, eight different types of spiral galaxy, and irregularly-shaped galaxies. The spiral types vary from those that are mainly made up of a central core (coded as types SO and SBO) to those that are extremely tenuous (Sc and SBc). Example 6.10.1a below uses these data to form an ordination of the ten galaxy types. It also illustrates the use of the LRVSCREE procedure (6.2.2) to produce a "scree" diagram of the latent roots (this time only as a printed histogram rather than as a graph like Figure 6.2.2), to help determine how many roots to consider. The final part of the example produces a graph of the ordination, shown in Figure 6.10.1.

Example 6.10.1a

```
  2  TEXT [VALUES=E,SO,SBO,Sa,SBa,Sb,SBb,Sc,SBc,I] Galaxies
  3  SYMMETRICMATRIX [ROWS=Galaxies] Galaxy
  4  READ [PRINT=data,errors] Galaxy
  5  0
  6  1.87 0
  7  2.24 0.91 0
  8  4.03 2.05 1.51 0
  9  4.09 1.74 1.59 0.68 0
 10  5.38 3.41 3.15 1.86 1.27 0
 11  7.03 3.85 3.24 2.25 1.89 2.02 0
 12  6.02 4.85 4.11 3.00 2.13 1.71 1.45 0
 13  6.88 5.70 5.12 3.72 3.01 2.97 1.75 1.13 0
 14  4.12 3.77 3.86 3.93 3.27 3.77 3.52 2.79 3.29 0 :
 15  CALCULATE Galaxy = -Galaxy/2
 16  PCO [PRINT=roots,scores,centroid] Galaxy; LRV=PCOlrv

Principal coordinates analysis
==============================

Latent roots
------------

             1           2           3           4           5           6
         6.662       3.058       1.267       1.171       0.737       0.516

             7           8           9          10
         0.381       0.291       0.109       0.000


Percentage variation
--------------------

             1           2           3           4           5           6
         46.94       21.55        8.93        8.25        5.19        3.64

             7           8           9          10
          2.69        2.05        0.77        0.00


Trace
-----

       14.19
```

```
Latent vectors (coordinates)
----------------------------

                1          2          3          4          5
 1         1.3965    -0.6742     0.4808     0.2564     0.0072
 2         1.0082     0.1916    -0.2521     0.0488     0.2665
 3         0.8176     0.3197    -0.2581     0.2306     0.1209
 4         0.1744     0.6571    -0.0324    -0.0699    -0.5732
 5         0.0114     0.5111     0.0315    -0.1844    -0.2450
 6        -0.4237     0.4417     0.5654    -0.5320     0.2897
 7        -0.8244     0.3341    -0.5082     0.2136     0.3103
 8        -0.9375    -0.2451     0.3141     0.0592     0.1534
 9        -1.1167    -0.4324     0.1205     0.5713    -0.2104
10        -0.1057    -1.1036    -0.4615    -0.5937    -0.1195

                6          7          8          9
 1         0.0422     0.1080     0.1334    -0.1166
 2        -0.3960    -0.1314     0.0950     0.1501
 3         0.3759    -0.0046    -0.3260     0.0324
 4         0.1177     0.1796     0.1790     0.0944
 5        -0.1582    -0.3563    -0.0828    -0.1802
 6        -0.0839     0.2260    -0.1229     0.0145
 7         0.0376     0.1792     0.1915    -0.1381
 8         0.3087    -0.3187     0.1693     0.0968
 9        -0.2703     0.0838    -0.1915     0.0410
10         0.0263     0.0344    -0.0450     0.0058

* MESSAGE: vectors corresponding to zero or negative roots are not printed


Centroid distances
------------------

                    1          2          3          4          5
                1.657      1.181      1.074      0.940      0.740

                    6          7          8          9         10
                1.065      1.132      1.140      1.392      1.346

  17  LRVSCREE [PLOT=*] PCOlrv

  No     Root   %%  Cum   %  Scree Diagram (* represents 2%)

   1    6.662  469  469  47 ************************
   2    3.058  215  685  22 ***********
   3    1.267   89  774   9 *****
   4    1.171   83  857   8 ****
   5    0.737   52  909   5 ***
   6    0.516   36  945   4 **
   7    0.381   27  972   3 **
   8    0.291   21  992   2 *
   9    0.109    8 1000   1 *
  10    0.000    0 1000   0

Scale:  1 asterisk represents 2 units.

  18  CALCULATE PCOscore[1,2] = PCOlrv[1]$[*; 1,2]
  19  FRAME     3; SCALING=xyequal
  20  XAXIS     3; YORIGIN=0
  21  YAXIS     3; XORIGIN=0
  22  PEN       1; SYMBOLS=0; LABELS=Galaxies
  23  DGRAPH    [TITLE='Principal coordinate analysis'; WINDOW=3; KEY=0] \
  24            PCOscore[2]; PCOscore[1]
```

Line 3 declares a symmetric matrix to hold the galaxy data; the rows (and columns) are labelled by the codes from Nathanson (1971). Line 15 transforms the data from squared distances to associations, as explained at the start of Section 6.10. Line 16 specifies that the PCO directive is to print the latent roots, the scores for the 10 galaxy types, and their distances from their centroid. The first two latent roots are much larger than the others, and so we can infer that a good ordination of the galaxy types can be found from the first two columns of scores (or dimensions).

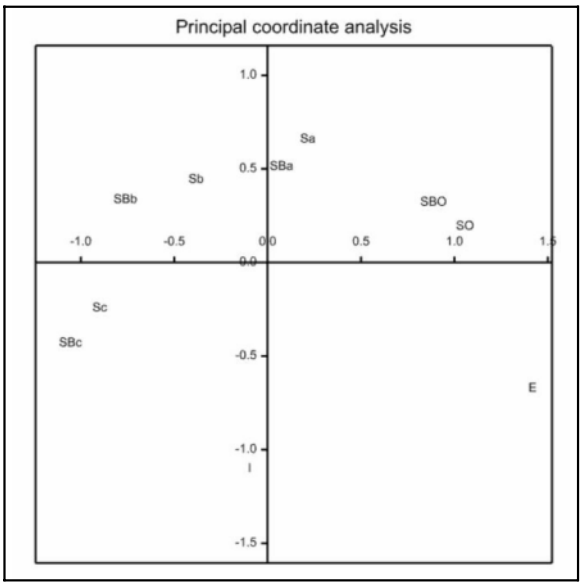


Figure 6.10.1

Ignoring for the moment the score for the irregular galaxies (0.1057), the first column of scores follows a trend from the elliptical galaxies, through the densely packed spiral types, to the tenuous spiral types. The irregularly shaped galaxies are placed somewhere near the middle of the others on this first principal axis.

The second axis places the irregular galaxies at the top of the ordination; the other types again roughly follow a trend, but now it is curved. Remember that at most nine dimensions are needed to obtain an exact solution for 10 points; so here the last latent root is zero, and only nine columns of scores are printed.

Instead of a symmetric matrix of associations, the input to PCO can be a pointer whose values are the identifiers of a set of variates, or a matrix storing the variates by columns. Now the PCO directive will construct the matrix of inter-unit squared distances, and will base the analysis on associations derived from this. As described above, this is equivalent to a principal components analysis; however, the results are derived by analysing the distance matrix rather than an SSPM. When there are more units than variates, using PCO for principal components analysis is less efficient than using the PCP directive; however, if there are more variates than units the PCO directive is more efficient.

When PCO is used for principal components analysis, all the variates must be of the same length and none of their values may be missing; any restrictions on the variates are ignored.

Suppose that we have data, as parts per million, for 12 chemical elements measured on eight insects. Analysing the 12 variates with the PCP directive will form the matrix of sums of squares and products for the 12 variates, and use that for the analysis. In Example 6.10.1b the more efficient approach is adopted, analysing the 8-by-8 inter-insect distance matrix instead.

Example 6.10.1b

```
 2  UNITS [NVALUES=8]
 3  POINTER Elements; VALUES=!P(Na,Mg,P,S,Cl,K,Ca,Zn,Fe,Si,Al,Cu)
 4  READ Elements[]

 Identifier   Minimum      Mean   Maximum    Values   Missing
         Na     137.0     266.6     408.0         8         0
         Mg     481.0     627.2     889.0         8         0
          P      1227      1437      1740         8         0
          S     412.0     590.6     786.0         8         0
         Cl     115.0     201.8     432.0         8         0
          K      1344      1690      2352         8         0
         Ca     28.00     71.62     127.0         8         0
         Zn    0.0000     7.625     15.00         8         0
         Fe     9.000     26.12     47.00         8         0
```

```
            Si      8.000      22.00      38.00          8          0
            Al      1.000      14.50      30.00          8          0
            Cu     0.0000      13.12      30.00          8          0

  13  CALCULATE Elements[] = LOG(Elements[]+1)
  14  PCO [PRINT=roots,scores,distances] Elements

Principal coordinates analysis
==============================

Latent Roots
------------

            1          2          3          4          5          6          7
       25.960     11.437      3.795      1.549      0.790      0.617      0.056


Percentage variation
--------------------

            1          2          3          4          5          6          7
        58.73      25.87       8.58       3.50       1.79       1.39       0.13


Trace
-----

       44.20


Latent vectors (coordinates)
----------------------------

              1          2          3          4          5          6          7
   1     1.0057    -1.9782     0.8397     0.4943     0.0315     0.1066     0.0856
   2    -2.6013     0.2070     0.4511     0.4229    -0.3377    -0.1168    -0.1242
   3    -2.2071    -1.7375    -0.7367    -0.6271     0.0455    -0.0908     0.0243
   4     1.7203     0.6858     0.8343    -0.7711    -0.3456     0.0282    -0.0018
   5     1.6349    -0.0188     0.0597    -0.0440     0.6029    -0.1580    -0.1229
   6     1.3564     0.8063    -0.7473     0.3134    -0.1868    -0.4868     0.0779
   7     1.1926     0.2210    -0.9985     0.1934    -0.1663     0.5364    -0.0393
   8    -2.1015     1.8145     0.2976     0.0183     0.3565     0.1813     0.1003


Distance matrix
---------------

   1     0.000
   2     4.263     0.000
   3     3.764     2.573     0.000
   4     3.060     4.529     4.894     0.000
   5     2.361     4.388     4.362     1.607     0.000
   6     3.291     4.204     4.502     2.030     1.520     0.000
   7     2.929     4.124     4.072     2.253     1.584     1.228     0.000
   8     4.967     1.909     3.780     4.161     4.196     3.859     3.939     0.000
             1         2         3         4         5         6         7         8
```

The data are defined on lines 2 and 3, and input on line 4. You can see from the report from READ that the amounts of the 12 elements differ considerably from each other. Often with such data, logarithms are taken before any analysis; this has been done on line 13. The PRINT option in the PCO statement (line 14) requests printing of the latent roots, the scores for the eight insects, and the matrix of inter-insect distances. These are shown above. You should note that *distances* are printed not squared distances, even though the analysis has been calculated from squared distances.

The third type of input to PCO is an SSPM structure. This must be a within-group SSPM: that is, you must have set the GROUP option of the SSPM directive (6.1.1) when the SSPM was declared. Now the PCO directive will calculate the Mahalanobis distances amongst the group

means, and base the analysis on them. As described at the start of Section 6.10, this will give results similar to a canonical variates analysis. The representation of distances in four dimensions will be better than that of CVA, but CVA will be better if you are interested in loadings for discriminatory purposes. In Example 6.10.1c, we analyse the same data as in the examples of CVA (6.3). These consist of seven variables measured on 28 brooches; the brooches are classified into four groups.

Example 6.10.1c

```
  2  POINTER [VALUES=Foot_lth,Bow_ht,Coil_dia,Elem_dia,Bow_wdth, \
  3     Bow_thck,Length] Data
  4  FACTOR [LEVELS=4] Groupno
  5  READ [PRINT=errors] Groupno,Data[]
 34  SSPM [TERMS=Data[]; GROUPS=Groupno] W
 35  FSSPM W
 36  PCO [PRINT=roots,scores,distances] W

Principal coordinates analysis
==============================

Latent Roots
------------

             1           2           3
         19.91       16.85        6.98


Percentage variation
--------------------

             1           2           3
         45.52       38.52       15.95


Trace
-----

       43.73


Latent vectors (coordinates)
----------------------------

                        1           2           3
            1       1.816       2.980       0.631
            2      -0.571       0.038      -2.262
            3       2.162      -2.815       0.560
            4      -3.407      -0.204       1.071


Distance matrix
---------------

   1       0.000
   2       4.767       0.000
   3       5.806       4.855       0.000
   4       6.133       4.383       6.172       0.000
               1           2           3           4
```

The first part the example, up to line 35 calculates the within-group SSPM. The PCO statement (line 36) prints the latent roots, the scores (that is canonical variate means for the four groups), and the matrix of inter-group Mahalanobis distances. Notice again that Mahalanobis *distances* are printed, not squared distances.

The second and subsequent parameters of PCO allow you to save the results. The number of units that determine the sizes of the output structures differs according to the input to PCO. For

a matrix or a symmetric matrix the number of units is the number of rows of the matrix, for a pointer it is the number of values in the variates that the pointer contains, while for an SSPM the number of units is the number of groups.

The latent roots, scores, and trace can be saved in an LRV structure using the LRV parameter. If you have declared the LRV already, its number of rows must equal the number of units.

If the input to PCO is a pointer, a matrix, or an SSPM, the principal component or canonical variate loadings can be saved in a matrix using the LOADINGS parameter. The number of rows of the matrix is equal to the number of variates (either those specified by an input pointer or those specified in the SSPM directive for an input SSPM structure), or the number of columns in an input matrix.

The number of columns of the LRV and of the LOADINGS matrix corresponds to the number of dimensions to be saved from the analysis, and this must be the same for both of them. If the structures have been declared already, Genstat will take the larger of the numbers of columns declared for either, and declare (or redeclare) the other one to match. If neither has been declared and option SMALLEST retains the default setting no, Genstat takes the number of columns from the setting of the NROOTS option. Otherwise, Genstat saves results for the full set of dimensions. The trace saved as the third component of the LRV structure, however, will contain the sums of all the latent roots, whether or not they have all been saved.

The distances of the units from their centroid can be saved in a diagonal matrix using the CENTROID parameter. The diagonal matrix has the same number of rows as the number of units, defined above. The RESIDUALS parameter allows you to save residuals, formed from the dimensions that have not been saved, in a matrix with one column and number of rows equal to the number of units. Finally, the inter-unit distances can be saved in a symmetric matrix using the DISTANCES parameter. The number of rows of the symmetric matrix is again the same as the number of units.

The SAVE parameter can supply a pointer to save a multivariate save structure contining all the details of the analysis. If this is unset, an unnamed save structure is saved automatically (and this can be accessed using the GET directive). Alternatively, you can set SAVE=* to prevent any save structure being formed if, for example, you have a very large data set and want to avoid committing the storage space.

6.10.2 The ADDPOINTS directive

ADDPOINTS directive

Adds points for new objects to a principal coordinates analysis.

Option

PRINT = *string tokens*	Printed output required (coordinates, residuals); default * i.e. no printing

Parameters

NEWDISTANCES = *matrices*	Squared distances of the new objects from the original points
LRV = *LRVs*	Latent roots and vectors from the PCO analysis
CENTROID = *diagonal matrices*	Centroid distances from the PCO analysis
COORDINATES = *matrices*	Saves the coordinates of the additional points in the space of the original points
RESIDUALS = *matrices* or *variates*	Saves the residuals of the new objects from that space

The input to ADDPOINTS is specified by the first three parameters. The NEWDISTANCES parameter specifies an $s \times n$ matrix containing squared distances of the s new units from the n old units. The LRV and CENTROID parameters specify structures defining the configuration of old units; these have usually been produced by a PCO statement (6.10.1).

The PRINT option controls the printed output; by default nothing is printed. The option has two settings:

coordinates	prints the coordinates of the new points;
residuals	prints the residual distances of the new units from the coordinates in the space of the old units.

For example, suppose that three original objects are equidistant, with a squared distance of four units amongst them. An ordination of these squared distances will place the points at the corners of an equilateral triangle of side two units. The coordinates of the three points will be (−0.5774, 1.0000), (−0.5774, −1.0000), and (1.1547, 0.0000). Now suppose that a new object is known to be equidistant from the original objects, at some squared distance d from them. If d is 4/3 the new object can be located precisely at the centroid of the three original points (that is at the origin), and all the distances in the system will be satisfied exactly. However if d>4/3, it would be possible to satisfy all the distances in three dimensions by placing the new object at a squared distance of $d-4/3$ above, or below, the plane in which the original points lie. The fitted coordinates in the space of the original objects will be the projection of the new point onto the plane (that is, at the centroid of the original points); the residual for the new object will be the square root of $d-4/3$. If d<4/3 the new distances can be satisfied only by introducing an imaginary third dimension in which squared distance is negative: the fitted coordinates will be the same as above, but the residual will be a complex number, which the ADDPOINTS directive will print and store as a missing value.

The other parameters can be used to save the results. The COORDINATES parameter allows you to specify an $s \times k$ matrix to save the coordinates for the new units; the residuals can be saved in an $s \times 1$ matrix using the RESIDUALS parameter. The value k is determined by the dimensionality of the input coordinates from the preceding PCO statement.

In Example 6.10.2, we use the data from Example 6.10.1a on the different galaxy types, and construct an ordination of the eight spiral forms. Then points for the irregular and elliptical types are added to this ordination. First we need to extract from the data the symmetric matrix of distances for the spiral types and also a matrix giving the distances of the two other types from the spiral types (lines 26 and 28). Remember that the input distances were transformed ready for the PCO in 6.10.1; this transformation is also appropriate for the distances amongst the spiral types used as input to PCO in line 32. However, the ADDPOINTS directive requires squared distances so the reverse transformation is required for the distances of the irregular and elliptical galaxy types from the spiral types (line 30).

Example 6.10.2

```
 25  TEXT      Gname2,Gname8; VALUES=!T(E,I),!T(SO,SBO,Sa,SBa,Sb,SBb,Sc,SBc)
 26  SYMMETRICMATRIX [ROWS=Gname8] G8
 27  CALCULATE G8 = Galaxy$[!(2...9)]
 28  MATRIX    [ROWS=Gname2; COLUMNS=Gname8] G2
 29  CALCULATE G2 = Galaxy$[!(1,10); !(2...9)]
 30  &         G2 = -2 * G2
 31  LRV       [ROWS=Gname8; COLUMNS=2] L8
 32  PCO       [PRINT=roots] G8; LRV=L8; CENTROID=C8

Principal coordinates analysis
==============================
```

```
Latent Roots
------------

        1        2        3        4        5        6        7        8
    5.006    1.359    0.838    0.724    0.508    0.358    0.216    0.000


Percentage variation
--------------------

        1        2        3        4        5        6        7        8
    55.57    15.08     9.30     8.04     5.64     3.98     2.40     0.00


Trace
-----

       9.009

  33  ADDPOINTS [PRINT=coordinates,residuals] G2; LRV=L8; CENTROID=C8

Adding points to a principal coordinates analysis
=================================================

Coordinates of added points
---------------------------

                        1          2
           1       1.1003     0.5186
           2      -0.1787     0.4406


Residuals
---------

         1       1.445
         2       1.474
```

6.10.3 Relating associations to data variables: the PCORELATE directive

One way of interpreting the principal coordinates obtained from a similarity matrix is by relating them to the original variables of the data matrix. For each coordinate and each data variate, an F-statistic can be computed as if the variable and the coordinate vector were independent. This is not the case but, although the exact distribution of these pseudo F-values is not known, they do serve to rank the variables in order of importance of their contribution to the coordinate vector.

Qualitative variables (variates or factors with `TEST` settings `simplematching` - `rogerstanimoto`) are treated as grouping factors, and the mean coordinate for each group is calculated. Only 10 groups are catered for; group levels above 10 are combined. The pseudo F-statistic gives the between-group to within-group variance ratio. Missing values are excluded.

Quantitative variables (i.e. variates with other settings) are grouped on a scale of 0-10 (where zero signifies a value up to 0.05 of the range), and mean coordinates for each group are calculated. The printed pseudo F statistic is for a linear regression of the principal coordinate on the ungrouped data variable, after standardizing the data variable to have unit range; the regression coefficient is also printed.

PCORELATE directive

Relates the observed values on a set of variates or factors to the results of a principal coordinates analysis.

Options

`COORDINATES` = *matrix* Points in reduced space; no default i.e. this option must

	be specified
NROOTS = *scalar*	Number of latent roots for printed output; default * requests them all to be printed

Parameters

DATA = *variates*	The data variables
TEST = *string tokens*	Test type, defining how each variable is treated in the calculation of the similarity between each unit (simplematching, jaccard, russellrao, dice, antidice, sneathsokal, rogerstanimoto, cityblock, manhattan, ecological, euclidean, pythagorean, minkowski, divergence, canberra, braycurtis, soergel); default * ignores that variable
RANGE = *scalars*	Range of possible values of each variable; if omitted, the observed range is taken

The DATA parameter lists the variables that are to be related to the PCO results and the TEST parameter indicates their "type" as in the FSIMILARITY directive (6.1.2). The RANGE parameter contains a list of scalars, one for each variable in the DATA list, allowing you to standardize quantitative variables. Notice that you do not need to supply the complete list of data variables (with their corresponding types and ranges), only those that you wish to relate to the PCO results. In Example 6.10.3, where we analyse the similarities between the cars discussed in Section 6.1.2, we examine two of the original variables.

The COORDINATES option must be present and must be a matrix. This represents the units in reduced space. Usually the coordinates will be from a principal coordinates analysis (6.10.1). The number of rows of the matrix must match the number of units present in the variables, taking account of any restriction.

The output from PCORELATE can be extensive. You may not be interested in relating the variables to the higher dimensions of the principal coordinates analysis even though you may have saved these in the coordinate matrix. The NROOTS option can request that results for only some of the dimensions are printed, for example NROOTS=3 for the first three dimensions as in Example 6.10.3. If NROOTS is not specified, PCORELATE prints information for all the saved dimensions: that is, for the number of columns of the coordinates matrix.

Example 6.10.3

```
  2  UNITS [NVALUES=16]
  3  VARIATE Engcc,Ncyl,Tankl,Weight,Length,Width,Height,Wbase,Tspeed,Stst,\
  4    Carb,Drive,Vct[1...3]
  5  POINTER Cd; VALUES=!P(Engcc,Ncyl,Tankl,Weight,Length, \
  6    Width,Height,Wbase,Tspeed,Stst)
  7  READ [PRINT=errors] #Cd,Carb,Drive
 24  TEXT [VALUES=Estate,'Arna1.5','Alfa2.5',Mondialqc,\
 25    Testarossa,Croma,Panda,Regatta,Regattad,Uno,\
 26    X19,Contach,Delta,Thema,Y10,Spider] Carname
 27  FACTOR [Carname; LEVELS=16] Fcar; VALUES=!(1...16)
 28  SYMMETRICMATRIX [ROWS=Carname] Carsim
 29  " Form similarity matrix between cars."
 30  FSIMILARITY [SIMILARITY=Carsim; PRINT=*] #Cd,Carb,Drive; \
 31    TEST=4(cityblock),4(Euclidean),2(cityblock),2(simplematch)
 32  " Produce output from ordination of Carsim and
-33    relate matrix of coordinates to the original variates "
 34  LRV [ROWS=Carname; COLUMNS=6] Carpco; VECTORS=Carvec
 35  PCO [PRINT=roots] Carsim; LRV=Carpco
```

```
Principal coordinates analysis
==============================

Latent Roots
------------

             1           2           3           4           5           6
        2.3578      0.8407      0.4220      0.3180      0.2171      0.1795

             7           8           9          10          11          12
        0.1022      0.0559      0.0504      0.0405      0.0277      0.0207

            13          14          15          16
        0.0194      0.0127      0.0121      0.0000


Percentage variation
--------------------

             1           2           3           4           5           6
         50.42       17.98        9.02        6.80        4.64        3.84

             7           8           9          10          11          12
          2.19        1.19        1.08        0.87        0.59        0.44

            13          14          15          16
          0.41        0.27        0.26        0.00


Trace
-----

       4.677


  36  PCORELATE [COORDINATES=Carvec; NROOTS=3] Weight,Carb; \
  37    TEST=cityblock,simplematch

Relate principal coordinates to original data
=============================================

  Variate: Weight
  Minimum: 720.0         Range: 786.0     Test type: City block
  Data scaled by factor of 0.01272

           F       *       0       1       2       3       4       5
 Counts              0       1       2       2       3       2       1
 Vector 1 335.8  0.0000  0.4145  0.4600  0.1931  0.2148  0.0469 -0.1521
 Vector 2   0.1  0.0000 -0.1075 -0.1815 -0.1519  0.0847  0.0395  0.4599
 Vector 3   0.0  0.0000  0.1475  0.0786 -0.1350  0.0672 -0.1742  0.0559

             6       7       8       9      10
 Counts      2       0       0       2       1
 Vector 1 -0.1288  0.0000  0.0000 -0.6205 -0.8082
 Vector 2  0.2076  0.0000  0.0000 -0.1494 -0.1349
 Vector 3 -0.1034  0.0000  0.0000  0.0508  0.1613

Regression coefficients:    -0.0016    0.0001    0.0000


  Variate: Carb    Test type: Simple Matching

           F       *       1       2       3
 Counts              0      10       5       1
 Vector 1   4.2  0.0000  0.1567 -0.3558  0.2118
 Vector 2   4.4  0.0000 -0.1131  0.1900  0.1812
 Vector 3   0.6  0.0000 -0.0007 -0.0336  0.1745
```

In Example 6.10.3, the coordinates for the cars in a reduced space of six dimensions are saved in the matrix, `Carvec`. The first three coordinates account for 71.2% of the trace.

6.11 Factor analysis: the FCA directive

FCA directive
Performs factor analysis.

Options

PRINT = *string tokens*	Printed output required (communalities, loadings, coefficients, scores, residuals, cresiduals, vresiduals, tests); default * i.e. no printing
NDIMENSIONS = *scalar*	Number of factors to fit; no default, must be specified
METHOD = *string token*	Whether to use correlations or variances and covariances (correlation, vcovariance, variancecovariance); default vcov
MAXCYCLE = *scalar*	Maximum number of iterations; default 50
TOLERANCE = *scalar*	Minimum value to assume for the unique component ψ_i^2 of each observed variable; default 10^{-6}

Parameters

DATA = *pointers* or *matrices* or *symmetric matrices* or *SSPMs*	Pointer of variates forming the data matrix, or matrix storing the variate values by columns, or symmetric matrix storing their variances and covariances, or SSPM giving their sums of squares and products
NUNITS = *scalars*	When DATA is set to a symmetric matrix of variances and covariances, NUNITS must specify the number of units from which they were calculated if tests are required
LRV = *LRVs*	To store the loadings, latent roots and trace from each analysis
SSPM = *SSPMs*	To save the SSPM formed from a DATA matrix or pointer
COMMUNALITIES = *variates*	Saves the communalities
COEFFICIENTS = *matrices*	Saves the factor score coefficients
SCORES = *matrices* or *pointers*	Saves the factor analysis scores
RESIDUALS = *matrices* or *pointers*	Saves residuals from the dimensions fitted in the analysis
CRESIDUALS = *symmetric matrices*	Saves the residual correlation or covariance matrix
VRESIDUALS = *variates*	Saves the residual variances

Factor analysis aims to find a set of "latent" (or unobservable) variables $\{z_1...z_k\}$ that account for the variances and covariances **S** between a set of p observed variables $\{x_1...x_p\}$. In the terminology of factor analysis, the latent variables $\{z_i\}$ are known as *factors*. However, they are continuous variables, and thus are represented in Genstat by variate rather than by factor data structures. So to avoid confusion, when we refer to the latent variables below, *factor* will be printed in italic font.

The data for a factor analysis consists of observed measurements on the variables $\{x_i\}$ made on a set of subjects. The assumption is that, for each subject, the values of the observed variables are related to the *factors* by a linear model

$$\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{\Gamma}\,\boldsymbol{z} + \boldsymbol{\varepsilon}$$

where $\boldsymbol{x}$ is the vector of observed variables,

$\boldsymbol{z}$ is the vector of *factors*,
$\boldsymbol{\mu}$ is a vector of means for the observed variables,
$\boldsymbol{\Gamma}$ is a matrix of *loadings* defining the relationship between observed and latent variables, and
$\boldsymbol{\varepsilon}$ is a vector of residuals.

The elements of the residual vector $\boldsymbol{\varepsilon}$ are assumed to have mean zero and to be uncorrelated, i.e. the dispersion matrix of $\boldsymbol{\varepsilon}$ is assumed to be diagonal

$$\mathrm{cov}(\boldsymbol{\varepsilon}) = \boldsymbol{\Psi} = \mathrm{diag}(\psi_1^2, \ldots \psi_p^2)$$

(They thus differ from the residuals formed in a principal components analysis, which will be correlated; see e.g. Krzanowski 1988 Section 16.2 for more details). The *factors* themselves are assumed to have variance one and to be uncorrelated, i.e.

$$\mathrm{cov}(\boldsymbol{z}) = \mathbf{I}.$$

So the correlations between the observed variables $\{x_i\}$ arise only through their relations with the *factors*, and not because of any correlation between the residuals or between the *factors*.

The `DATA` parameter specifies the data for the factor analysis. You can supply either a pointer containing a set of variates, one for each observed variable $\{x_i\}$, or a matrix storing the observed variables by columns, or a symmetric matrix containing variances and covariances between the variables, or an SSPM structure (formed using `FSSPM` from the variates of observed measurements). When `DATA` specifies a symmetric matrix of variances and covariances, you must also set the `NUNITS` parameter to specify the number of units from which they were calculated if you want `FCA` to print tests.

The `METHOD` option has settings `vcovariance` (with synonym `variancecovariance`) and `correlation`, to control whether `FCA` forms a matrix of variances and covariances or a matrix of correlations for the analysis. The same *factors* will be obtained if you use a correlation matrix, but the loadings will be scaled to be between zero and one. The number of *factors*, q, to fit must be specified by the `NDIMENSIONS` option. Arising from the numbers of parameters in the model (see Krzanowski 1988 Section 16.2.2) this is subject to the constraint

$$(p - q)^2 \geq p + q.$$

The `PRINT` option controls printed output, with settings:

`communalities`	the proportion of variation explained by the *factors* for each observed variable, $(\mathrm{var}(x_i) - \psi_i^2) / \mathrm{var}(x_i)$;
`loadings`	the matrix of factor loadings $\boldsymbol{\Gamma}$;
`coefficients`	the factor score coefficients;
`scores`	the factor scores calculated from the model for each subject;
`residuals`	the vectors of residuals ε,
`cresiduals`	the residual correlation or covariance matrix i.e. a symmetric matrix showing the amount of unexplained correlation or covariance between each pair of variables;
`vresiduals`	the residual variances; and
`tests`	a chi-square goodness of fit test for the model.

By default nothing is printed. Note, however, that scores and residuals cannot be produced when `DATA` is set to a symmetric matrix of variances and covariances.

The communalities, factor coefficients, scores, residuals, residual correlations or covariances and residual variances can also be saved using the `COMMUNALITIES`, `COEFFICIENTS`, `SCORES`, `RESIDUALS`, `CRESIDUALS` and `VRESIDUALS` parameters, respectively. The `LRV` parameter allows an LRV structure to be saved, with the loadings in the `['vectors']` component, and the eigenvalues of the matrix $\boldsymbol{\Psi}^{-½}\,\mathbf{S}\,\boldsymbol{\Psi}^{-½}$ in the `['roots']` component; the loadings are scaled eigenvectors of $\boldsymbol{\Psi}^{-½}\,\mathbf{S}\,\boldsymbol{\Psi}^{-½}$. (Remember, $\mathbf{S}$ is the matrix of variances and covariances of the observed variables $\{x_i\}$.) The `SSPM` parameter can save the SSPM structure constructed from a `DATA` pointer for the analysis. A particularly convenient instance is when you have supplied an

SSPM structure as input but, for example, have set `METHOD=correlation`: the SSPM that is saved will then contain correlations instead of sums of squares and products.

Example 6.11 analyses a correlation matrix for nine variates, calculated from a sample of 211 subjects, used in Section 2.4 of Lawley & Maxwell (1963). This is also used as an example in the documentation for NAG subroutine G03CCF.

Example 6.11

```
 2  TEXT [VALUES=Gaelic,English,History,Arithmetic,Algebra,Geometry] Subjects
 3  SYMMETRICMATRIX [ROWS=Subjects; VALUES=\
 4  1.000,\
 5  0.439, 1.000,\
 6  0.410, 0.351, 1.000,\
 7  0.288, 0.354, 0.164, 1.000,\
 8  0.329, 0.320, 0.190, 0.595, 1.000,\
 9  0.248, 0.329, 0.181, 0.470, 0.464, 1.000] Correlation
10  FCA [PRINT=communalities,loadings,cresiduals,tests; NDIMENSION=2]\
11      Correlation; NUNITS=220

Factor analysis
===============

Factor loadings
---------------

   Subjects
     Gaelic   1      0.5533     -0.4286
    English   2      0.5682     -0.2883
    History   3      0.3922     -0.4500
 Arithmetic   4      0.7404      0.2728
    Algebra   5      0.7239      0.2113
   Geometry   6      0.5954      0.1317


Factor communalities
--------------------

     Subjects
       Gaelic   1      0.4898
      English   2      0.4059
      History   3      0.3563
   Arithmetic   4      0.6226
      Algebra   5      0.5686
     Geometry   6      0.3718


Residual correlation matrix
---------------------------

     Gaelic   1    0.000000
    English   2    0.001067    0.000000
    History   3    0.000162   -0.001551    0.000000
 Arithmetic   4   -0.004777    0.011978   -0.003627    0.000000
    Algebra   5    0.019029   -0.030347    0.001196    0.001385    0.000000
   Geometry   6   -0.024985    0.028712    0.006770   -0.006742    0.005210
                         1           2           3           4           5

   Geometry   6    0.000000
                         6


Factor analysis test statistics
-------------------------------

Log-likelihood:               -1.190
Goodness of fit statistic:     2.335
Degrees of freedom:                4
Probability:                   0.674
```

FCA estimates the parameters of the model by maximum likelihood, assuming multivariate Normality, using subroutines G03CAF and G03CCF from the NAG Library. The MAXCYCLE option sets a limit on the number of iterations (default 50). The TOLERANCE option specifies the minimum value to assume for the unique component ψ_i^2 of each observed variable so that the communality is always less than one; the default is 10^{-6}.

6.12 Multidimensional scaling: the MDS directive

MDS directive

Performs non-metric multidimensional scaling.

Options

PRINT = *string tokens*	Printed output required (coordinates, roots, distances, fitteddistances, stress, monitoring); default * i.e. no printing
DATA = *symmetric matrix*	Distances amongst a set of units
METHOD = *string token*	Whether to use non-metric scaling, or metric scaling with linear regression of the fitted distances to the actual distances (nonmetric, linear); default nonm
SCALING = *string token*	Whether least-squares, least-squares-squared, or log-stress scaling is to be used (ls, lss, logstress); default ls
TIES = *string token*	Treatment of tied data values (primary, secondary, tertiary); default prim
WEIGHTS = *symmetric matrix*	Weights for each distance value; default * i.e. all distances with weight one
INITIAL = *matrix*	Initial configuration; default * i.e. a principal coordinate solution is used
NSTARTS = *scalar*	Number of starting configurations to be used, by making random perturbations to the initial configuration; default 10
SEED = *scalar*	Seed for the random-number generator; default 0
MAXCYCLE = *scalar*	Maximum number of iterations; default 30

Parameters

NDIMENSIONS = *scalars*	Number of dimensions for each solution
COORDINATES = *matrices*	To store the coordinates of the units for each solution
STRESS = *scalars*	To store the stress value for each solution
DISTANCES = *symmetric matrices*	To store the distances amongst the points for the units in the fitted number of dimensions
FITTEDDISTANCES = *symmetric matrices*	To store the fitted distances from the monotonic (METHOD=nonmetric) or linear (METHOD=linear) regression

The MDS directive carries out iterative scaling, including metric and non-metric scaling. The input data consists of a symmetric matrix whose values may be interpreted, in a general sense, as distances between a set of objects. The matrix is specified by the DATA option; thus only one matrix can be analysed each time the MDS directive is used.

The objective of the MDS directive is to find a set of coordinates whose inter-point distances

match, as closely as possible, those of the input data matrix. When plotted, the coordinates provide a display which can be interpreted in the same way as a map: for example, if points in the display are close together, their distance apart in the data matrix was small.

The algorithm invoked by the MDS directive uses the method of steepest descent to guide the algorithm from an initial configuration of points to the final matrix of coordinates that has the minimum stress of all configurations examined.

Printed output is controlled by the PRINT option; by default nothing is printed. There are six possible settings:

coordinates	prints the solution coordinates, rotated to principal coordinates;
roots	prints the latent roots of the solution coordinates;
distances	prints the inter-unit distances, computed from the solution configuration;
fitteddistances	prints the fitted values from the regression of the inter-unit distances on the distances in the data matrix, the regression may be monotonic or linear through the origin, depending on the setting of the METHOD option;
stress	prints the stress of the solution coordinates;
monitoring	prints a summary of the results at each iteration.

The METHOD option determines whether metric or non-metric scaling is given. The algorithm involves regression of the distances, calculated from the solution coordinates, against the dissimilarities in the symmetric matrix specified by the DATA option. With the default setting, METHOD=nonmetric, monotonic regression is used; if METHOD=linear, the algorithm uses linear regression through the origin.

The stress function to be minimized can be selected using the STRESS option. There are three possibilities.

ls (least squares):
$$\sqrt{\sum_i \sum_j w_{ij}(d_{ij}-\hat{d}_{ij})^2 / (m \times \sum_i \sum_j w_{ij} d_{ij}^2)}$$

lss (least-squares-squared):
$$\sqrt{\sum_i \sum_j w_{ij}(d_{ij}^2-\hat{d}_{ij}^2)^2 / (m \times \sum_i \sum_j w_{ij} d_{ij}^4)}$$

logstress:
$$\sqrt{\sum_i \sum_j w_{ij}(\log d_{ij} - \log \hat{d}_{ij})^2 / m}$$

where the d_{ij} are the elements of the dissimilarity matrix calculated for the fitted configuration, the $\hat{d}_{ij}$ are the fitted values from the regression selected by the METHOD option, the w_{ij} are the corresponding weights and m is the number of off-diagonal elements in the dissimilarity matrix.

The TIES option allows you to vary the way in which tied data values in the input data matrix are to be treated. By default, the treatment of ties is primary, and no restrictions are placed on the distances corresponding to tied dissimilarities in the input data matrix. In the secondary treatment of ties, the distances corresponding to tied dissimilarities are required to be as nearly equal as possible. Kendall (1977) describes a compromise between the primary and secondary approaches to ties: the block of ties corresponding to the smallest dissimilarity are handled by the secondary treatment, the remaining blocks of ties are handled by the primary treatment. This tertiary treatment of ties is useful when the dissimilarities take only a few values. For example, in the reconstruction of maps from abuttal information, the dissimilarity coefficient takes only two values: zero if localities abut, and one if they do not. The block of ties associated with the dissimilarity of zero are handled by the secondary treatment, and the block of ties with dissimilarity one by the primary treatment.

The WEIGHT option can be used to specify a symmetric matrix of weights. Each element of the matrix gives the weight to be attached to the corresponding element of the input data matrix. If the option is not set, the elements of the data matrix are weighted equally: w_{ij}=1 for all i and j. The most important use of the option occurs when the matrix of weights contains only zeros and ones; the zeros then correspond to missing values in the input data matrix, allowing incomplete data matrices to be scaled. Up to about two thirds of the data matrix may be missing before the algorithm breaks down. This enables experimenters to design studies in which only a subset of all the dissimilarities need to be observed. This is particularly useful when there are a large number of units; if the number of units is m, say, a complete $m \times m$ data matrix requires $m(m-1)/2$ dissimilarities to be observed.

Since the algorithm is an iterative one, making use of the method of steepest descent, there is no guarantee that the solution coordinates found from any given starting configuration has the minimum stress of all possible configurations. The algorithm may have found a local, rather than the global, minimum. This problem may be partially overcome by using a series of different starting configurations. If several of the solutions arrive at the same lowest stress solution, then you may be reasonably confident of having found the global minimum. The NSTARTS option determines the number of starting configurations to be used. The starting configuration used on the first start can be specified by the INITIAL option; if this is not set, the default is to take the principal coordinate solution obtained from a PCO analysis of the input dissimilarity matrix. Subsequent starting configurations are found by perturbing each coordinate of the first starting configuration by successively larger amounts. This strategy generally results in at least one starting configuration that does not get entrapped in a local minimum: however there can be no guarantee that the global minimum for the stress function has been found. Experience suggests that, for safety, the NSTARTS option should be set equal to at least 10. By default NSTARTS=10.

The SEED option supplies the seed for the random numbers that are used to perturb the initial configuration. The default of zero continues the existing sequence of random numbers if MDS has already been used in the current Genstat job. If MDS has not yet been used, Genstat picks a seed at random.

The MAXCYCLES option determines the maximum number of iterations of the algorithm. The default of 30 should usually be sufficient. However, it may be necessary to set a larger value for very large data matrices or when using the logstress setting of the SCALING option. The monitoring setting of the PRINT option may be used to see how convergence is progressing.

The NDIMENSIONS parameter must be set to a scalar (or scalars) to indicate the number(s) of dimensions in which the multidimensional scaling is to be performed on the data matrix. An MDS statement with a list of scalars will carry out a series of scaling operations, all based on the same matrix of dissimilarities, but with different numbers of dimensions.

The remaining parameters of the MDS directive allow output to be saved in Genstat data structures. The COORDINATES parameter can list matrices to store the minimum stress coordinates in each of the dimensions given by the NDIMENSIONS parameter, and the STRESS parameter can specify scalars to store the associated minimum stresses. The parameters DISTANCES and FITTEDDISTANCES can specify symmetric matrices to store the distances computed from the coordinates matrix and the fitted distances computed from the monotonic or linear regressions, respectively.

Example 6.12 shows the use of non-metric multidimensional scaling with the inter-galaxy distances of Example 6.10.1a, printing the stress, the coordinates, and the roots. The remainder of the example plots the two-dimensional solution obtained as Figure 6.12a, and also the "Shepard diagram" in Figure 6.12b. This shows the distances that have been computed from the solution obtained – the distances between the points in Figure 6.12a – plotted as crosses against the actual distances in the input data, and also the fitted monotonic regression line using circles to show the fitted values. The small distances, typically of the points in Figure 6.12a from their immediate neighbours, have been fitted well, as have most of the large distances.

Example 6.12

```
   2  TEXT    [VALUES=E,SO,SBO,Sa,SBa,Sb,SBb,Sc,SBc,I] Galaxies
   3  SYMMETRICMATRIX [ROWS=Galaxies] Galaxy
   4  READ    [PRINT=errors] Galaxy
  15  MDS     [PRINT=roots,coordinates,stress; DATA=Galaxy; SEED=934306]\
  16         NDIMENSIONS=2; COORDINATES=MDScoord; DISTANCES=MDSdist; FITTED=MDSfit

Multidimensional scaling
========================


Least-squares scaling criterion
-------------------------------

Distances fitted using monotonic regression (non-metric MDS).
Primary treatment of ties.

Stress
------


      0.0469

Coordinates
-----------

                        1           2
     Galaxies
            E     -1.5717     -0.4588
           SO     -0.9217      0.1298
          SBO     -0.7469      0.4075
           Sa     -0.2278      0.5630
          SBa      0.0266      0.2905
           Sb      0.4536      0.6067
          SBb      0.9005      0.2550
           Sc      0.8588     -0.1838
          SBc      1.2015     -0.4086
            I      0.0271     -1.2012


Latent roots
------------

            1       7.126
            2       2.874

  17  CALCULATE Score[1,2] = MDScoord$[*; 1,2]
  18  VARIATE Actual,OnMDS,FitMDS; Galaxy,MDSdist,MDSfit
  19  PEN     1,2,3; SYMBOLS=0,1,2; METHOD=(point)2,line; \
  20          LABELS=Galaxies,*,*; LINESTYLE=1; SIZE=1.5,1,1
  21  FRAME   3; SCALING=xyequal
  22  DGRAPH  ['Non-metric multidimensional scaling'; WINDOW=3; \
  23          KEYWINDOW=0] Score[2]; Score[1]
  24  DGRAPH  ['Shepard diagram'; WINDOW=4; KEYWINDOW=0] \
  25          OnMDS,FitMDS; Actual; PEN=2,3
```

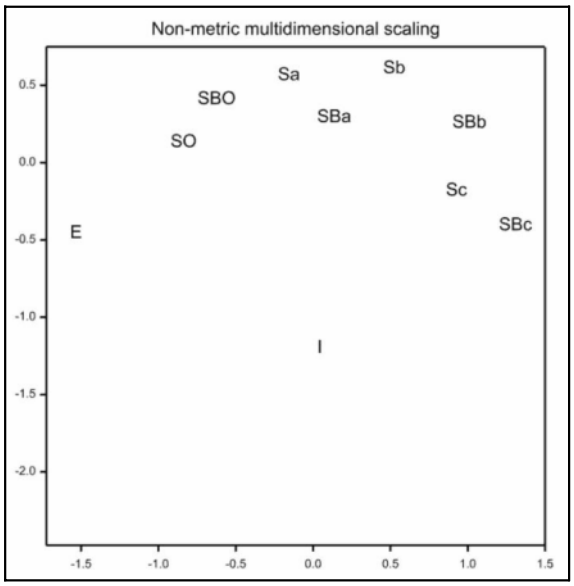


Figure 6.12a

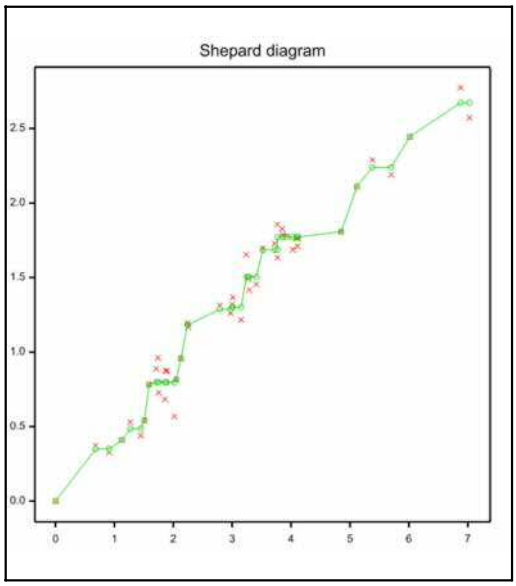


Figure 6.12b

6.13 Correspondence analysis

This Chapter describes the procedure `CORANALYSIS` (6.13.1) which does ordinary correspondence analysis, `MCORANALYSIS` (6.13.2) which does multiple correspondence analysis, and `CABIPLOT` (6.13.3) which can display biplots of their results.

6.13.1 The `CORANALYSIS` procedure

`CORANALYSIS` procedure

Does correspondence analysis, or reciprocal averaging (P.G.N. Digby & A.I. Glaser).

Options

`PRINT` = *string tokens*	Printed output from the analysis (`roots`, `rowscores`, `rowinertias`, `rowchisquare`, `rowmass`, `rowquality`, `colscores`, `colinertias`, `colchisquare`, `colmass`, `colquality`); default `*` i.e. no output
`METHOD` = *string token*	Type of analysis required (`correspondence`, `digbycorrespondence`, `biplot`, `reciprocal`); default `corr`
`NROOTS` = *scalar*	Number of latent roots for printed output; default * requests them all to be printed
`%METHOD` = *string token*	How to represent proportions or %s in quality statistics (`permills`, `percentages`, `proportions`); default `prop`
`NDIMENSIONS` = *scalar*	Number of dimensions for which quality statistics are required; default 2
`ROWSUBSET` = *scalars*	Indexes of subset rows
`COLSUBSET` = *scalars*	Indexes of subset columns
`ROWPASSIVE` = *scalars*	Indexes of passive rows
`COLPASSIVE` = *scalars*	Indexes of passive columns

Parameters

DATA = *matrices* or *data matrices*	Data to be analysed
ROOTS = *diagonal matrices*	Saves the squared singular values from each analysis
ROWSCORES = *matrices*	Saves the scores for the rows of the data matrix
COLSCORES = *matrices*	Saves the scores for the columns of the data matrix
ROWINERTIAS = *matrices*	Saves the inertias for the rows of the data matrix
COLINERTIAS = *matrices*	Saves the inertias for the columns of the data matrix
ROWQUALITY = *matrices*	Saves the quality statistics for rows of the data
COLQUALITY = *matrices*	Saves the quality statistics for columns of the data
SAVE = *pointers*	Saves details of the analysis for use by CAPLOT

Correspondence analysis is an ordination technique used to analyse two-way categorical data tables. Ordination techniques approximate relationships between variables in a reduced number of dimensions.

The type of analysis is specified by the METHOD option, with one of the following settings:

correspondence	correspondence analysis (Greenacre 1984),
digbycorrespondence	an alternative implementation of correspondence analysis described by Digby & Kempton (1987),
reciprocal	reciprocal averaging (see Digby & Kempton 1987), or
biplot	a similar biplot-style analysis (again see Digby & Kempton 1987).

The default setting is correspondence, and this should be retained if either of the options to subset rows or columns are set.

The data matrix X, is scaled to have sum one for METHOD settings correspondence and digbycorrespondence. The matrices U, S and V are taken from the singular-value decomposition of

$$Y = (X - R\,C) / \sqrt{(R\,C)}$$

for METHOD=correspondence and

$$Y = (R^{-½}\,X\,C^{-½})$$

for the other methods, where R and C are diagonal matrices of row and column totals of the data matrix X. The scores for the rows and columns from METHOD=correspondence are

$$A = (R^{-½}\,U\,S)$$

and

$$B = (C^{-½}\,V\,S)$$

The scores from METHOD=digbycorrespondence are similar, but are multiplied by the square root of S.

With the other two methods X is not scaled to total one, and the scores are given by $A = (R^{-½}\,U\,S^m)$ and $B = (C^{-½}\,V\,S^m)$: the parameter m is zero for METHOD=reciprocal, and 0.5 for METHOD=biplot.

The inertia values for the rows and columns are defined as

$$(R\,A\,A')\,S'$$

and

$$(C\,B\,B')\,S'$$

where $S' = S$ for METHOD=correspondence, and $S = 1$ for the other methods; see Greenacre (1984) for further information.

The roots are the squares of the singular values. Note that the first singular value will always be one for methods other than correspondence; this corresponds to a trivial solution given in the first column of A and B above, which is automatically removed from the results printed and saved from CORANALYSIS.

Rows and/or columns chosen as passive rows and/or columns are separated from the original data matrix before it is scaled. Rows and/or columns chosen as subset rows and/or columns are

separated from Y after this scaling.

The data for the procedure are specified by the DATA parameter as either a matrix or a datamatrix (i.e. a pointer to variates, all with the same length). The matrix must not contain any missing values; it is unchanged on exit from the procedure.

Printed output is controlled by the PRINT option with settings:

roots	to print the roots (together with the roots expressed as percentages and cumulative percentages),
rowscores	to print the scores for the rows of the data matrix,
rowinertias	to print the inertias for the rows of the data matrix,
rowmass	to print the row masses,
rowchisquare	to print the row chisquare distances,
rowquality	to print the quality statistics for the rows,
colscores	to print the scores for the columns of the data matrix,
colinertias	to print the inertias for the columns of the data matrix,
colmass	to print the column masses,
colchisquare	to print the column chisquare distances, and
colquality	to print the quality statistics for the columns.

The NROOTS option controls the printed output of roots, scores and inertias. By default, results are printed for all the roots, but you can set the NROOTS option to specify a lesser number.

The quality settings produce tables with the following columns:

- the mass of the row (or column), in proportion to the total mass;
- the "quality" of the representation i.e. how much of the inertia of a row (or column) is represented by the dimensions shown;
- the proportion of the total inertia of the row (or column) compared to the total inertia for all rows (or columns);
- principal coordinates of the rows (or columns) in the specified dimension;
- the amount of inertia for each row (or column) in the specified dimension relative to the total amount of inertia given by the value of the quality statistic – hence the sum of a specific row (or column) across the dimensions shown will be equal to the value given by the quality statistic;
- the proportion of inertia explained by a row (or column) in a dimension, compared to the total inertia in that dimension.

The representation of the columns of proportions is controlled by the %METHOD option; these can be printed either as proportions (default), percentages or as permills i.e. tenths of a percent. The NDIMENSIONS option specifies the number of dimensions for which to print quality statistics; default 2.

When carrying out correspondence analysis, there may be rows and/or columns (for example outliers with low mass) that you would like to ignore during the calculation of the roots or inertia, so that they have no influence. Instead of removing these rows and/or columns from the data before running CORANALYSIS, an alternative is to list the indexes of the rows or columns that are to be ignored using the ROWPASSIVE and/or COLPASSIVE options. These "passive" rows will still be included in the table of quality statistics, where their relative contributions will be shown and compared to total for all the passive rows or columns.

You may want to apply a correspondence analysis calculated from the whole data set onto only a subset of the rows and/or columns when some of the rows and/or columns divide into groups with common traits. This can be done by setting the ROWSUBSET and/or COLSUBSET options to the indexes of the rows and/or columns indexes in the subset of interest. If any of these options is set, the METHOD option must be set to correspondence. If ROWPASSIVE and ROWSUBSET (or COLPASSIVE and COLSUBSET) are both set, any indexes that occur in both will be removed from the ROWSUBSET (or COLSUBSET).

Results from the analysis can be saved using the parameters ROOTS, ROWSCORES, COLSCORES,

ROWINERTIAS, COLINERTIAS, ROWQUALITY and COLQUALITY. The structures specified for these parameters need not be declared in advance. The SAVE parameter can save full details of the analysis for use by the CAPLOT procedure.

Example 6.13.1 analyses a set of data from Greenacre (2007).

Example 6.13.1

```
  2  " Data from Table 9.1 of Greenacre (2007)"
  3  TEXT [VALUES=S_Manager,J_Manager,S_Employee,J_Employee,Secretary] Staff
  4  &    [VALUES=SM,JM,SE,JE,Sy] Staff2
  5  &    [VALUES=None,Light,Medium,Heavy] Smoke
  6  MATRIX [ROWS=Staff; COLUMNS=Smoke] Smoking; VALUES= \
  7    !( 4, 2, 3, 2,  4, 3, 7, 4,  25,10,12, 4,  18,24,33,13,  10, 6, 7, 2)
  8  PRINT       Smoking; FIELDWIDTH=8; DECIMALS=0

             Smoking
      Smoke    None   Light  Medium   Heavy
      Staff
   S_Manager       4       2       3       2
   J_Manager       4       3       7       4
  S_Employee      25      10      12       4
  J_Employee      18      24      33      13
   Secretary      10       6       7       2

  9  CORANALYSIS [PRINT=roots,rowscores,colscores,rowinertia,colinertia; \
 10              METHOD=correspondence] Smoking

Correspondence analysis
=======================

Squared singular values
-----------------------

                  Roots     % Roots  Cumulative
                                        % roots
          1     0.07476       87.76       87.76
          2     0.01002       11.76       99.51
          3     0.00041        0.49      100.00


Row scores
----------

                   Dim. 1      Dim. 2      Dim. 3

    S_Manager      -0.241       1.936      -3.490
    J_Manager       0.947       2.431       1.657
   S_Employee      -1.392       0.107       0.254
   J_Employee       0.852      -0.577      -0.163
    Secretary      -0.735      -0.788       0.397


Row inertias
------------

                   Dim. 1      Dim. 2      Dim. 3       Total  Proportion

    S_Manager     0.00025    0.002139  0.00028716     0.00267      0.0314
    J_Manager     0.00625    0.005521  0.00010595     0.01188      0.1395
   S_Employee     0.03828    0.000030  0.00000702     0.03831      0.4497
   J_Employee     0.02474    0.001520  0.00000498     0.02627      0.3084
    Secretary     0.00524    0.000807  0.00000846     0.00605      0.0711
```

```
Column scores
-------------

                 Dim. 1      Dim. 2      Dim. 3

        None     -1.4385      0.3047      0.0438
       Light      0.3637     -1.4094     -1.0817
      Medium      0.7180     -0.0735      1.2617
       Heavy      1.0744      1.9760     -1.2889


Column inertias
---------------

                 Dim. 1      Dim. 2      Dim. 3       Total  Proportion

        None    0.04889    0.000294   0.0000003     0.04919      0.5774
       Light    0.00231    0.004640   0.0001128     0.00706      0.0829
      Medium    0.01238    0.000017   0.0002115     0.01261      0.1480
       Heavy    0.01118    0.005066   0.0000890     0.01633      0.1917

  11  CABIPLOT    [COLSCALING=standard] LROWVARIABLES=Staff2
```

Figure 6.13.1 plots the scores in the first and second dimensions, with the rows in principal coordinates, and the columns in standard coordinates (this corresponds to Figure 9.2 of Greenacre 2007). The `CABIPLOT` procedure, which was used to produce the plot, is described in Section 6.13.3.

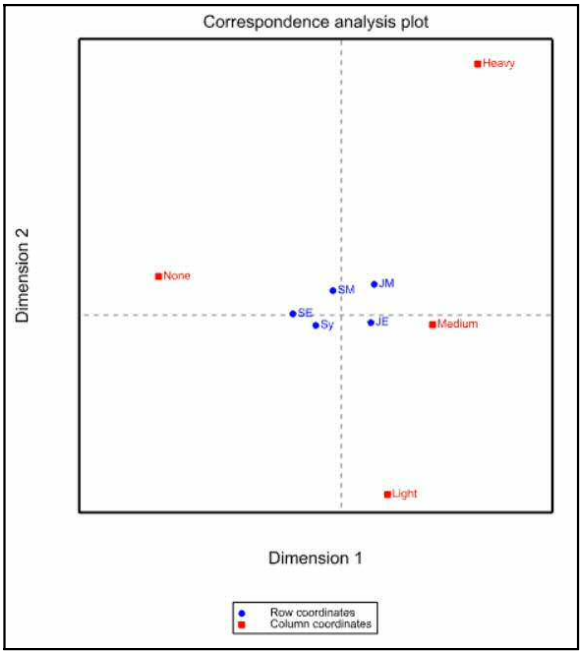


Figure 6.13.1

6.13.2 The `MCORANALYSIS` procedure

`MCORANALYSIS` procedure

Does multiple correspondence analysis (A.I. Glaser).

Options

`PRINT` = *string tokens* — Printed output from the analysis (`roots`, `rowscores`, `rowinertias`, `rowchisquare`, `rowmass`, `rowquality`, `colscores`, `colinertias`, `colchisquare`, `colmass`, `colquality`); default `*` i.e. no output

`ROWMETHOD` = *string token* — Analysis method for rows i.e. units (`indicator`); default `indi`

COLMETHOD = *string token*	Analysis method for columns i.e. factors (adjusted, burt, indicator); default adju
NROOTS = *scalar*	Number of latent roots for printed output; default * requests them all to be printed
%METHOD = *string token*	How to represent proportions or %s in quality statistics (permills, percentages, proportions); default prop
NDIMENSIONS = *scalar*	Number fo two dimensions for which quality statistics are required; default 2
TOLERANCE = *scalar*	Tolerance criteria for zero eigenvalues; default 10^{-6}

Parameters

DATA = *pointers*	Data to be analysed
ROOTS = *diagonal matrices*	Saves the squared singular values from each analysis
ROWSCORES = *matrices*	Saves the scores for the rows of the data
COLSCORES = *matrices*	Saves the scores for the columns of the data
ROWINERTIAS = *matrices*	Saves the total inertias for the rows of the data
COLINERTIAS = *matrices*	Saves the total inertias for the columns of the data
ROWQUALITY = *matrices*	Saves the quality statistics for rows of the data
COLQUALITY = *matrices*	Saves the quality statistics for columns of the data
SUBINERTIAS = *matrices*	Saves the inertias of the subtables of the Burt matrices
FREQUENCY = *variates*	Frequencies for elements of DATA
SAVE = *pointers*	Saves details of the analysis for use by CABIPLOT

Ordinary correspondence analysis is an ordination technique used to analyse relationships between two categorical variables (6.13.1). Multiple correspondence analysis provides a similar analysis for more than two variables.

The data consist of a list of factors, which are supplied in a pointer by the DATA parameter. By default, each unit of the factors is assumed to represent a single observation. However, with large data sets, you may want to use the FREQUENCY parameter to supply a variate defining frequencies (or numbers of replications) for each unit. MCORANALYSIS uses the data to form an *indicator* matrix D, with a row for each unit and a columns for each level of every factor. Each row of the matrix has the value one in the columns corresponding to the levels of the factors that occurred in that data unit and zero elsewhere. (This is equivalent to the design matrix that is used in analysis of variance or regression.) The factors must not contain any missing values.

The relationships between the rows are assessed by doing an ordinary correspondence analysis on the indicator matrix. This analysis also provides information on the relationships between the columns (i.e. the factor levels). However, an alternative method for the columns does the correspondence analysis on the Burt matrix $D'D$. A refinement of the use of the Burt matrix discards eigenvalues below a threshold $1/Q$, where Q is the number of DATA factors. This adjusts for the inflation of the eigenvalues that arises from the within-factor diagonal blocks of the Burt matrix; see Greenacre (2007) Chapter 19 for more details. The difference between the results obtained using the indicator and Burt matrices is that the singular values obtained from the Burt matrix will be the squares of those obtained from the indicator matrix. The adjusted method is the default method for the columns, but the other two methods can be requested by using the COLMETHOD option. With very large data sets it may be impractical to do the correspondence analysis on the indicator matrix for rows. So MCORANALYSIS allows this to be suppressed by setting option ROWMETHOD=*.

Printed output is controlled by the PRINT option with settings:

roots	to print the roots (together with the roots expressed as percentages and cumulative percentages),

rowscores	to print the scores for the rows of the indicator matrix,
rowinertias	to print the inertias for the rows of the indicator matrix,
rowmass	to print the row masses,
rowchisquare	to print the row chisquare distances,
rowquality	to print the quality statistics for the rows,
colscores	to print the scores for the columns of the indicator or Burt matrix (as selected by the COLMETHOD option),
colinertias	to print the inertias for the columns,
colmass	to print the column masses,
colchisquare	to print the column chisquare distances,
colquality	to print the quality statistics for the columns, and
subinertias	to print the inertias of the subtables of the Burt matrix.

The NROOTS option controls the printed output of roots, scores and inertias. By default, results are printed for all the roots greater than the limit defined by the TOLERANCE option. However, you can set the NROOTS option to specify a lesser number.

The quality settings produce tables with the following columns:

- the mass of the row (or column), in proportion to the total mass;
- the "quality" of the representation i.e. how much of the inertia of a row (or column) is represented by the dimensions shown;
- the proportion of the total inertia of the row (or column) compared to the total inertia for all rows (or columns);
- principal coordinates of the rows (or columns) in the specified dimension;
- the amount of inertia for each row (or column) in the specified dimension relative to the total amount of inertia given by the value of the quality statistic – hence the sum of a specific row (or column) across the dimensions shown will be equal to the value given by the quality statistic;
- the proportion of inertia explained by a row (or column) in a dimension, compared to the total inertia in that dimension.

The representation of the columns of proportions is controlled by the %METHOD option; these can be printed either as proportions (default), percentages or as permills i.e. tenths of a percent. The NDIMENSIONS option specifies the number of dimensions for which to print quality statistics; default 2.

Results from the analysis can be saved using the parameters ROOTS, ROWSCORES, COLSCORES, ROWINERTIAS, COLINERTIAS, ROWQUALITY and COLQUALITY. The structures specified for these parameters need not be declared in advance. The SAVE parameter can save full details of the analysis for use by the CABIPLOT procedure.

6.13.3 The CABIPLOT procedure

CABIPLOT procedure

Plots results from correspondence analysis or multiple correspondence analysis (A.I. Glaser).

Options

DIMENSIONS = *scalars*	Two numbers specifying which axes of the ordinations to plot; default 1,2
PLOT = *string tokens*	Which scores to plot (rowscores, rowactive, rowpassive, colscores, colactive, colpassive); default rows, cols for correspondence analysis and cols for multiple correspondence analysis
ROWSCALING = *string token*	Scaling to use for row coordinates (principal, standard, mass, sqrtmass); default prin

COLSCALING = *string token*	Scaling to use for column coordinates (principal, standard, mass, sqrtmass); default prin
COLOURMETHOD = *string tokens*	Whether colour of symbol should show level of inertia of rows or columns (rowinertia, colinertia); default *
SIZEMETHOD = *string tokens*	Whether size of symbol should show row or column masses (rowmass, colmass); default *
FACCOLOURS = *text*, *variate* or *scalar*	Specifies a colour or colours for the factors in a multiple correspondence analysis; if this is unset, a different colour is selected automatically for every factor
WINDOW = *scalar*	Which graphical window to use; default 1
KEYWINDOW = *scalar*	Graphical window for the key
SAVE = *pointer*	Supplies results from a analysis by CORANALYSIS or MCORANALYSIS; default uses the most recent analysis

Parameters

TITLE = *texts*	Titles for the plot
LMROWVARIABLES = *string tokens*	How to label the row scores (identifiers, labels, none, numbers); default labe if LROWVARIABLES is set, otherwise iden
LMCOLVARIABLES = *string tokens*	How to label the column scores (identifiers, labels, none, numbers); default labe if LCOLVARIABLES is set, otherwise iden
LROWVARIABLES = *texts*	Labels for row variables
LCOLVARIABLES = *texts*	Labels for column variables

CABIPLOT provides a graphical representation of results from a correspondence analysis procuced by procedure CORANALYSIS (6.13.1), or a multiple-correspondence analysis produced by procedure MCORANALYSIS (6.13.2). By default CABIPLOT plots both sets of scores (rowscores, colscores) for correspondence analysis or just columns scores for multiple correspondence analysis, but you can set option PLOT to select which ones are required. For correspondence analysis, you can also select settings that will plot only active or passive scores (see 6.13.1).

The row scores are plotted as blue circles, while the column scores are plotted as red squares; active scores have filled symbols, but passive scores are not filled. With multiple correspondence analysis, the FACCOLOURS option can be used to define the colour to use for each factor, using either RGB values (in a variate or scalar) or the standard Genstat colour names (in a text); see PEN for more details. If insufficient colours are specified, CABIPLOT will recycle the list. So you can set FACCOLOURS to a scalar or to a text with a single string if you want to use the same colour for all the factors. If FACCOLOURS is not set, CABIPLOT will select a different colour for each factor automatically.

The ROWSCALING and COLSCALING options are define the scaling to use for the row and columns coordinates respectively, with settings:

principal	plots principal coordinates (default),
standard	plots standard coordinates,
mass	plots standard coordinates multiplied by the row (or column) mass,
sqrtmass	plots standard coordinates multiplied by the square root of the row (or column) mass.

These are based on the row and column scores obtained from CORANALYSIS or MCORANALYSIS.

Principal coordinates are scaled so that they have inertia equal to the square of the singular values, whereas the weighted sum-of-squares of the standard coordinates are equal to one. At least one of ROWSCALING or COLSCALING must be set to principal, which is the default for both options. These default settings produce a plot, which is not a biplot, but which is used very often to illustrate relationships between and amongst variables. The reasoning behind multiplying the standard coordinates by the corresponding mass or its square root is to "pull" the rarer categories to be closer to the origin; see Chapter 13 of Greenacre (2007).

The COLOURMETHOD option has settings rowinertia and colinertia that plot the row or coordinates scores, respectively, at a different level of shading; the coordinates with higher inertias are plotted with darker colours then those with low inertias. The shading is proportional to the square root of the inertia relative to the row or column with the highest inertia. Symbols representing passive points will appear completely transparent on the plot as they are perceived to have zero inertia.

The SIZEMETHOD option similarly has settings rowmass and colmass that plot the row and column coordinates, respectively, in sizes that depend on the row and column mass. The sizes of the symbols are proportional to the square root of the mass compared to the square root of the row or column with the highest mass, plus a constant to ensure all symbols are visible.

By default the first two dimensions are plotted, but you can specify other dimensions to be plotted using the DIMENSIONS option.

The data used in MCORANALYSIS may have many repeated values (particularly in survey data). To avoid replotting the same points in a large data set (i.e. with more than 500 units), only one point is plotted and the label refers to the first point in the data set. If the COLOURMETHOD or SIZEMETHOD options are set, these will use the mass and/or inertia of the labelled point.

The labels for the row and column scores can be set using the LMROWVARIABLES and LMCOLVARIABLES parameters, by selecting one of the following settings:

identifiers	uses the identifiers of the row or column scores,
labels	expects labels to be supplied (in a text) using the LROWVARIABLES or LCOLVARIABLES parameter,
none	gives no labels, and
numbers	uses the row or column numbers of the original matrix.

The default for both parameters is identifiers, unless LROWVARIABLES or LCOLVARIABLES is set, when the corresponding default becomes labels. Note that the texts supplied by LROWVARIABLES or LCOLVARIABLES must have the same number of values as number of the rows or columns in the original data matrix, even if active or passive points are being omitted from the plot. Similarly, if the setting numbers is chosen, these will refer to the corresponding row or column of the original matrix, ignoring any any active or passive rows or columns, or subsetting of rows or columns in CORANALYSIS.

By default CABIPLOT uses the results from the most recent analysis from by CORANALYSIS or MCORANALYSIS. However, you can display results from an earlier analysis by saving the information about the analysis with the SAVE parameter of CORANALYSIS or MCORANALYSIS, and then using this as the setting of the SAVE option of CABIPLOT.

In Example 6.13.1, the statement

```
CABIPLOT [COLSCALING=standard] LROWVARIABLES=Staff2
```

is used to plot the scores in the first and second dimensions of a correspondence analysis of data from Table 9.1 of Greenacre (2007). The rows are plotted using principal coordinates (the default for the ROWSCALING option), while the columns are plotted in standard coordinates. The resulting graph, which corresponds to Figure 9.2 of Greenacre (2007), is shown in Figure 6.13.1.

6.14 Redundancy analysis: the RDA procedure

RDA procedure

Performs redundancy analysis (A.I. Glaser).

Options

PRINT = *string tokens*	What to print (variance, loadings, roots, evalues, evectors, speciesscores, sitescores, fitsitescores, correlations, fitcorrelations, weights); default vari, root
NROOTS = *scalar*	Number of eigenvalues and eigenvectors to include in output; default * takes all the non-zero eigenvalues
NORMALIZE = *string tokens*	Whether to normalize the Y, X and/or Z variates to have unit sums-of-squares before the analysis (x, y, z); default x, z
SCALING = *string token*	Scaling for species and site scores (none, both); default none
TOLERANCE = *scalar*	Tolerance for detecting non-zero eigenvalues; default 10^{-5}

Parameters

Y = *pointers*	Each pointer defines a set of response variates to be modelled
X = *pointers*	Explanatory variates or factors to use for each pointer of y-variates
Z = *pointers*	Conditioning variates or factors to remove ("partial out") before the analysis
LRV = *LRVs*	LRV structure from each analysis, storing the eigenvectors, eigenvalues and total variance
SPECIESSCORES = *matrices*	Saves the "species scores" from each analysis
SITESCORES = *matrices*	Save the "site scores" from each analysis
FITSITESCORES = *matrices*	Save the fitted "site scores" from each analysis
CORRELATIONS = *matrices*	Saves the correlations between the site scores and the x-variates
FITCORRELATIONS = *matrices*	Saves the correlations between the fitted site scores and the x-variates
WEIGHTS = *matrices*	Save the weights of the x-variates in the formation of the site scores
SAVE = *pointers*	Save structure which provides information for use in CRBIPLOT and CRTRIPLOT

Redundancy analysis is the direct extension of multiple regression to the modelling of multivariate response data (see Sections 11.1 and 11.3 of Legendre & Legendre 1998). The response data are a set of y-variates, specified in a pointer using the Y parameter. The explanatory variables, which may be either variates or factors, are specified in a pointer by the X parameter. Similarly, the Z parameter can be used to specify conditioning variables, which again may be either variates or factors; this gives partial RDA, in which the effect of the z-variables is removed before performing RDA. This may be useful in cases where the effects of the elements of Z on Y are well known, or we may wish to isolate the effect of an individual explanatory variable (in which case we would place all but one of the explanatory variables in

Z). If any of the variate or factors in the Y, X or Z pointers are restricted, only the defined subset of the units will be used in the analysis. If all elements of a variable are equal to zero, CCA removes the variable.

The PRINT option controls printed output, with settings:

roots	the eigenvalues of the fitted values;
evalues	synonym of roots;
loadings	the eigenvectors associated with each eigenvalue, also known as the "species scores";
evectors	synonym of loadings;
speciesscores	the "species scores" from the analysis (synonym of loadings and evectors);
variance	the fraction of the variance of the y-variates associated with each eigenvalue;
sitescores	the "site scores" of the y-variates (i.e. the ordination of the units in the y-variate space);
fitsitescores	the fitted "site scores" of the fitted values of the y-variates (i.e. the ordination of the units in the y-variate space);
correlations	the correlation between the site scores and the x-variables;
fitcorrelations	the correlation between the fitted site scores and the x-variables;
weights	the weights of the x-variables in the formation of the site scores.

By default PRINT=roots,variance. The LRV, SPECIESSCORES, SITESCORES, FITSITESCORES, CORRELATIONS, FITCORRELATIONS and WEIGHTS parameters allow this information to be saved.

The NROOTS option specifies the number of eigenvalues and eigenvectors to include in the output. By default all the non-zero eigenvalues are included. The NORMALIZE option controls whether to normalize the Y variates, or X or Z variables to have unit sums-of-squares before the analysis. The default is to normalize the x- and z-variables but not the y-variates. (Note: this normalization of the x's and z's does not affect the variances accounted for in the y-variates.) The SCALING option controls scaling for species and site scores. If both is selected, both species and site scores are multiplied by the square root of their corresponding eigenvalues. For RDA choosing none is equivalent to Scaling type 1 in Legendre & Legendre (1998), whilst both is equivalent to Scaling type 2 in the same book. The TOLERANCE option specifies a threshold for the detection of non-zero eigenvalues (default 10^{-5}). An eigenvalue is taken to be non zero if is it greater than TOLERANCE multiplied by the total variance.

The SAVE parameter allows you to save a pointer containing full details of the analysis. This can then be used to generate plots using the CRBIPLOT or CRTRIPLOT procedures; see 6.16.2 and 6.16.3. The most recent save structure is kept automatically inside Genstat to use as a default for the SAVE options of CRBIPLOT and CRTRIPLOT. So, you need save the pointer explicitly only if you want to display output from more than one analysis at a time.

Example 6.14 analyses data from Table 11.3 of Legendre & Legendre (1998). The data simulate fish observations from a beach at 10 sites with different water depths and substrates.

Example 6.14

```
" Legendre & Legendre (1998), page 590, Table 11.3."
POINTER [VALUES=Depth_m,Coral,Sand] X
VARIATE [NVALUES=10] Species[1...6],X[]; VALUES=\
        !(1, 0, 0, 11, 11, 9, 9, 7, 7, 5),\
        !(0, 0, 1, 4, 5, 6, 7, 8, 9, 10),\
        !(0, 0, 0, 0, 17, 0, 13, 0, 10, 0),\
        !(0, 0, 0, 0, 7, 0, 10, 0, 13, 0),\
        !(0, 0, 0, 8, 0, 6, 0, 4, 0, 2),\
```

```
  10          !(0, 0, 0, 1, 0, 2, 0, 3, 0, 4),\
  11          !(1, 2, 3, 4, 5, 6, 7, 8, 9, 10),\
  12          !(0, 0, 0, 0, 1, 0, 1, 0, 1, 0),\
  13          !(1, 1, 1, 0, 0, 0, 0, 0, 0, 0)
  14  " RDA for species 1-6 with Depth_m, Coral and Sand."
  15  RDA     [PRINT=variance,loadings,roots,speciesscores,sitescores,\
  16          fitsitescores,correlations,fitcorrelations] Species; X;\
  17          SAVE=SaveRDA

Variance
--------

                Variance  Proportion        Rank
  Constrained      108.34      0.9597           3
Unconstrained        4.55      0.0403           4
        Total      112.89


Eigenvalues (with respect to total variance = 112.9)
----------------------------------------------------

Canonical

       74.52       24.94        8.88

Non-canonical

       4.189       0.314       0.037       0.008


Fraction of total variance
--------------------------

Canonical eigenvalues

      0.6601      0.2209      0.0786

Non-canonical eigenvalues

      0.0371      0.0028      0.0003      0.0001


Cumulative fraction of total variance
-------------------------------------

Canonical eigenvalues

      0.6601      0.8811      0.9597

Non-canonical eigenvalues

      0.9968      0.9996      0.9999      1.0000


Eigenvectors (species scores)
-----------------------------

Canonical

                    RDA.1       RDA.2       RDA.3

   Species[1]     -0.3013     -0.6462      0.3994
   Species[2]     -0.2004     -0.4727     -0.7446
   Species[3]     -0.7410      0.1681      0.2569
   Species[4]     -0.5501      0.1684     -0.2611
   Species[5]      0.1159     -0.5059      0.2932
   Species[6]      0.0629     -0.2154     -0.2568
```

Non-canonical

	PC.1	PC.2	PC.3	PC.4
Species[1]	-0.0066	-0.4048	0.7071	-0.1669
Species[2]	0.0066	0.4048	0.7071	0.1669
Species[3]	-0.6890	-0.2667	0.0000	0.6739
Species[4]	0.5880	0.2151	0.0000	0.6863
Species[5]	0.3789	-0.6662	0.0000	0.1237
Species[6]	-0.1894	0.3331	0.0000	-0.0619

Site scores

Canonical

	RDA.1	RDA.2	RDA.3
1	6.828	5.644	1.152
2	7.129	6.290	0.753
3	6.929	5.818	0.008
4	4.004	-6.972	4.257
5	-13.634	0.855	3.962
6	4.037	-5.828	1.125
7	-12.119	1.035	-0.137
8	4.069	-4.685	-2.006
9	-11.345	1.383	-3.979
10	4.102	-3.541	-5.137

Non-canonical

	PC.1	PC.2	PC.3	PC.4
1	0.2471	1.1435	0.2357	0.0127
2	0.0000	0.0000	-0.4714	0.0000
3	-0.2471	-1.1435	0.2357	-0.0127
4	2.1425	-0.2823	0.0000	0.0014
5	-3.8092	-0.1457	0.0000	0.1036
6	0.7142	-0.0941	0.0000	0.0005
7	0.2297	0.0889	0.0000	-0.2246
8	-0.7142	0.0941	0.0000	-0.0005
9	3.5796	0.0568	0.0000	0.1210
10	-2.1425	0.2823	0.0000	-0.0014

Fitted site scores

Canonical

	RDA.1	RDA.2	RDA.3
1	6.795	5.495	2.249
2	6.962	5.917	0.638
3	7.129	6.339	-0.973
4	3.552	-6.523	4.394
5	-12.700	0.247	3.172
6	3.886	-5.679	1.171
7	-12.366	1.091	-0.051
8	4.220	-4.834	-2.051
9	-12.032	1.936	-3.273
10	4.554	-3.990	-5.274

```
Non-canonical

                    PC.1        PC.2        PC.3        PC.4

           1       0.177       0.277      -7.307      -4.953
           2       0.430       1.826      -7.307      -4.773
           3       0.684       3.374      -7.307      -4.594
           4       1.084      -5.723       2.828      -5.015
           5      -3.397      -3.486       3.536      10.382
           6       1.591      -2.626       2.828      -4.656
           7      -2.890      -0.389       3.536      10.741
           8       2.098       0.471       2.828      -4.296
           9      -2.383       2.708       3.536      11.101
          10       2.606       3.567       2.828      -3.937


Correlations
------------

Correlations of environmental variables with site scores

                    RDA.1       RDA.2       RDA.3

      Depth_m     -0.4220     -0.5572     -0.6987
        Coral     -0.9871      0.1503     -0.0115
         Sand      0.5557      0.8148      0.1447


Correlations of environmental variables with fitted site scores

                    RDA.1       RDA.2       RDA.3

      Depth_m     -0.4227     -0.5591     -0.7133
        Coral     -0.9885      0.1508     -0.0118
         Sand      0.5565      0.8176      0.1477
```

6.15 Canonical correspondence analysis: the CCA procedure

CCA procedure

Performs canonical correspondence analysis (A.I. Glaser).

Options

PRINT = *string tokens*	Controls printed output (variance, loadings, roots, evalues, evectors, speciesscores, sitescores, fitsitescores, correlations, fitcorrelations); default vari, root
NROOTS = *scalar*	Number of eigenvalues and eigenvectors to include in output; default * takes all the non-zero eigenvalues
NORMALIZE = *string tokens*	Whether to normalize the Y, X and/or Z variates to have unit sums-of-squares before the analysis (x, y, z); default x, z
SCALING = *string tokens*	Whether to scale for species or site score (species, site); default spec
TOLERANCE = *scalar*	Tolerance for detecting non-zero eigenvalues; default 10^{-5}

Parameters

Y = *pointers*	Each pointer defines a set of response variates to be modelled
X = *pointers*	Explanatory variates or factors to use for each pointer of

	y-variates
Z = *pointers*	Conditioning variates or factors to remove ("partial out") before the analysis
LRV = *LRVs*	LRV structure from each analysis, storing the eigenvectors, eigenvalues and total variance
SPECIESSCORES = *matrices*	Save the "species scores" from each analysis
SITESCORES = *matrices*	Save the "site scores" from each analysis
FITSITESCORES = *matrices*	Save the fitted "site scores" from each analysis
CORRELATIONS = *matrices*	Saves the correlations between the site scores and the x-variates
FITCORRELATIONS = *matrices*	Saves the correlations between the fitted site scores and the x-variates
SAVE = *pointers*	Save structure which provides information for use in CRBIPLOT and CRTRIPLOT

CCA performs canonical correspondence analysis and partial canonical correspondence analysis; see Sections 11.2 and 11.3 of Legendre & Legendre (1998)

Canonical correspondence analysis is the canonical form of correspondence analysis. It is similar to redundancy analysis (see RDA). However, in CCA, we apply weighted multiple regression to a transformed data matrix with the fitted values subjected to correspondence analysis.

The Y parameter specifies the response data as a pointer to a set of y-variates. Each variate contains observations of numbers of a particular species at a set of sites (the same sites and in the same order for each species). The explanatory variables, which may be either variates or factors, are specified in a pointer by the X parameter. Similarly, the Z parameter can be used to specify conditioning variables, which again may be either variates or factors. When a pointer of z-variables is supplied, CCA performs a partial canonical correspondence analysis, in which the effects of the z-variables are removed prior to the canonical correspondence analysis. This can be useful when the effects of the elements of Z on Y are well known, or if we wish to isolate the effect of an single explanatory variable (in which case we would place all but one of the explanatory variables in Z). If any of the variate or factors in the Y, X or Z pointers are restricted, only the defined subset of the units will be used in the analysis. If all elements of a variable are equal to zero, CCA removes the variable.

The PRINT option controls printed output, with settings:

roots	the eigenvalues of the fitted values;
evalues	synonym of roots;
loadings	the eigenvectors associated with each eigenvalue, also known as the "species scores";
evectors	synonym of loadings;
speciesscores	the "species scores" from the analysis (synonym of loadings and evectors);
variance	the fraction of the variance of the y-variates associated with each eigenvalue;
sitescores	the "site scores" of the y-variates (i.e. the ordination of the units in the y-variate space);
fitsitescores	the fitted "site scores" of the fitted values of the y-variates (i.e. the ordination of the units in the y-variate space);
correlations	the correlation between the site scores and the x-variables;
fitcorrelations	the correlation between the fitted site scores and the x-variables.

By default PRINT=roots,variance. The LRV, SPECIESSCORES, SITESCORES,

FITSITESCORES, CORRELATIONS and FITCORRELATIONS parameters allow this information to be saved.

The NROOTS option specifies the number of eigenvalues and eigenvectors to include in the output. By default all the non-zero eigenvalues are included. The NORMALIZE option controls whether to normalize the Y variates, or X or Z variables to have unit sums-of-squares before the analysis. The default is to normalize the x and z-variables but not the y-variates. (Note: normalization of only the x's and z's does not affect the variances accounted for in the y-variates.)

The SCALING option controls which scores are scaled by CCA: either the species scores or the site scores. The scaling is done by multiplying them by their corresponding eigenvalues. Choosing 'site' is equivalent to Scaling type 1 in Legendre & Legendre (1998), whilst 'species' is equivalent to their Scaling type 2.

The TOLERANCE option specifies a threshold for the detection of non-zero eigenvalues (default 10^{-5}). An eigenvalue is taken to be non-zero if is it greater than TOLERANCE.

The SAVE parameter allows you to save a pointer containing full details of the analysis. This can then be used to generate plots using the CRBIPLOT or CRTRIPLOT procedures. The most recent save structure is kept automatically inside Genstat to use as a default for the SAVE options of CRBIPLOT and CRTRIPLOT; see 6.16.2 and 6.16.3. So, you need save the pointer explicitly only if you want to display output from more than one analysis at a time.

Example 6.15 analyses the full data set in Table 11.3 of Legendre & Legendre (1998), originally introduced in Example 6.14.

Example 6.15

```
 18  " Define extra variables for CCA analysis."
 19  VARIATE [NVALUES=10] Species[7...9],Other; VALUES=\
 20          !(2, 5, 0, 6, 6, 10, 4, 6, 6, 0),\
 21          !(4, 6, 2, 2, 6, 1, 5, 6, 2, 1),\
 22          !(4, 1, 3, 0, 2, 4, 4, 4, 0, 3),\
 23          !(0, 0, 0, 1, 0, 1, 0, 1, 0, 1)
 24  POINTER [VALUES=Depth_m,Coral,Sand,Other] X
 25  " CCA of full data set."
 26  CCA     [PRINT=variance,loadings,roots,speciesscores,sitescores,\
 27          fitsitescores,correlations,fitcorrelations] Species; X;\
 28          SAVE=SaveCCA

Variance
--------

                Variance  Proportion        Rank
  Constrained      0.6319      0.8058           3
Unconstrained      0.1523      0.1942           6
        Total      0.7842


Eigenvalues (with respect to total variance = 0.784)
----------------------------------------------------

Canonical

      0.3661      0.1869      0.0788

Non-canonical

     0.08229     0.03513     0.02333     0.00990     0.00122     0.00042


Fraction of total variance
--------------------------

Canonical eigenvalues


      0.4669      0.2383      0.1005
```

Non-canonical eigenvalues

0.10494	0.04481	0.02975	0.01263	0.00156	0.00053

Cumulative fraction of total variance

Canonical eigenvalues

0.4669	0.7052	0.8058			

Non-canonical eigenvalues

0.9107	0.9555	0.9853	0.9979	0.9995	1.0000

Eigenvectors (species scores)

Canonical

	CCA.1	CCA.2	CCA.3
Species[1]	0.1104	0.2824	0.2030
Species[2]	0.1414	0.3035	-0.3954
Species[3]	-1.0155	0.0958	0.1983
Species[4]	-1.0362	0.1096	-0.2210
Species[5]	1.0537	0.5372	0.4381
Species[6]	0.9986	0.5740	-0.6799
Species[7]	0.2552	-0.1782	0.2041
Species[8]	0.1466	-0.8574	0.0152
Species[9]	0.4137	-0.7079	-0.2157

Non-canonical

	CA.1	CA.2	CA.3	CA.4	CA.5	CA.6
Species[1]	0.0019	0.0822	0.0857	-0.0122	-0.0425	-0.0047
Species[2]	0.1413	0.0269	0.1433	0.0430	0.0476	0.0023
Species[3]	0.1048	-0.1300	0.0244	0.0465	0.0269	-0.0350
Species[4]	-0.2236	0.2437	-0.0259	-0.0534	-0.0316	-0.0256
Species[5]	-0.2235	0.3239	0.1246	-0.1193	0.0416	0.0382
Species[6]	0.3900	-0.2991	0.3285	0.2122	-0.0830	0.0440
Species[7]	-0.4334	-0.0707	-0.1882	0.1269	0.0045	0.0123
Species[8]	-0.0528	-0.3545	-0.0417	-0.1990	-0.0021	0.0078
Species[9]	0.6903	0.1484	-0.3343	-0.0063	-0.0036	0.0123

Site scores

Canonical

	CCA.1	CCA.2	CCA.3
1	0.711	-3.082	-0.220
2	0.585	-3.007	0.947
3	0.763	-3.153	-2.139
4	1.112	1.072	1.875
5	-0.979	-0.060	0.696
6	1.043	0.459	0.640
7	-0.954	-0.085	-0.133
8	0.947	-0.108	-0.526
9	-1.148	0.490	-0.478
10	1.033	1.035	-2.747

Non-canonical

	CA.1	CA.2	CA.3	CA.4	CA.5	CA.6
1	1.2453	1.0729	-0.5062	0.2441	-3.6316	-1.1631
2	-2.6997	-2.1368	0.8135	0.4715	0.9084	1.3472
3	3.1163	2.3066	-0.6989	-1.3906	4.8412	-0.5621
4	-0.6664	1.1015	1.4352	-1.1062	0.0137	0.0372
5	0.6126	-0.9830	0.3157	0.5741	0.3286	-0.8680
6	-0.2872	0.5739	-1.4498	1.7017	0.3062	0.4421
7	0.4214	0.1116	-0.3942	-0.6740	-0.3790	1.7473
8	0.0057	-1.2627	-1.0657	-1.4633	-0.1545	-0.7781
9	-1.1702	1.0060	0.0735	0.0860	0.0418	-0.9358
10	1.2808	-0.3630	1.9865	1.0536	-0.2481	0.4632

Fitted site scores

Canonical

	CCA.1	CCA.2	CCA.3
1	0.6921	-3.0805	0.3287
2	0.6646	-3.0621	-0.2302
3	0.6370	-3.0438	-0.7892
4	1.1089	0.5004	1.5561
5	-0.9700	0.0655	1.1206
6	1.0537	0.5372	0.4381
7	-1.0252	0.1023	0.0026
8	0.9986	0.5740	-0.6799
9	-1.0803	0.1391	-1.1154
10	0.9434	0.6107	-1.7979

Non-canonical

	CA.1	CA.2	CA.3	CA.4	CA.5	CA.6
1	1.2453	1.0729	-0.5062	0.2441	-3.6316	-1.1631
2	-2.6997	-2.1368	0.8135	0.4715	0.9084	1.3472
3	3.1163	2.3066	-0.6989	-1.3906	4.8412	-0.5621
4	-0.6664	1.1015	1.4352	-1.1062	0.0137	0.0372
5	0.6126	-0.9830	0.3157	0.5741	0.3286	-0.8680
6	-0.2872	0.5739	-1.4498	1.7017	0.3062	0.4421
7	0.4214	0.1116	-0.3942	-0.6740	-0.3790	1.7473
8	0.0057	-1.2627	-1.0657	-1.4633	-0.1545	-0.7781
9	-1.1702	1.0060	0.0735	0.0860	0.0418	-0.9358
10	1.2808	-0.3630	1.9865	1.0536	-0.2481	0.4632

Correlations

Correlations of environmental variables with site scores

	CCA.1	CCA.2	CCA.3
Depth_m	-0.1861	0.6019	-0.6581
Coral	-0.9923	0.0919	0.0461
Sand	0.2128	-0.9176	-0.0376
Other	0.8796	0.4441	-0.0247

```
Correlations of environmental variables with fitted site scores

                  CCA.1       CCA.2       CCA.3

      Depth_m    -0.1864      0.6403     -0.7452
        Coral    -0.9938      0.0978      0.0522
         Sand     0.2131     -0.9761     -0.0426
        Other     0.8809      0.4724     -0.0279
```

6.16 Biplots

6.16.1 The DBIPLOT procedure

DBIPLOT procedure

Plots a biplot from an analysis by PCP, CVA or PCO (A.I. Glaser).

Options

PLOT = *string tokens*	Additional features for the plot (convexhull, means); default * i.e. none
METHOD = *string token*	Type of axes to plot (predictive, interpolative); default pred
HORIZONTAL = *identifer*	Which axis to make horizontal; default * i.e. none
PREDICTIONS = *matrix*	Saves predicted values
GROUPS = *factor*	Factor defining groupings of individuals for a PCP biplot; default * i.e. none
LMINDIVIDUALS = *string tokens*	How to label the individuals (labels, none, numbers, unitlabels); default labe if LINDIVIDUALS is set, otherwise unit
LMVARIABLES = *string tokens*	How to label the variables (identifiers, labels, none, numbers); default labe if LVARIABLES is set, otherwise iden
LINDIVIDUALS = *texts*	Labels for individuals (i.e. scores)
LVARIABLES = *texts*	Labels for variables (i.e. biplot axes)
MULTIPLIER = *scalar*	Value to multiply vector loadings; default * i.e. determined automatically
TITLE = *text*	Title for the plot; if this is unset, an appropriate title is formed auomatically
WINDOW = *scalar*	Which graphical window to use; default 1 when there are groups, otherwise 3
KEYWINDOW = *scalar*	Which graphical window to use for the key when there are groupings of individuals (0 for none); default 2
SCREEN = *string token*	Whether to clear the screen before plotting or to continue plotting on the old screen (clear, keep); default clea
SIZEMULTIPLIER = *scalar*	Multiplier used in the calculation of the size in which to draw symbols and labels; default 1
SAVE = *pointer*	Supplies results from an ordination analysis by PCP, CVA or PCO; default uses the most recent analysis

Parameters

VARIABLE = *identifiers*	Axis variables
DISPLAY = *string tokens*	Whether to show, hide or omit each axis (show, hide, omit); default show
COLOUR = *texts* or *scalars*	Colour to use to plot each axis

DBIPLOT plots biplots displaying the results from a principal components, canonical variates or principal coordinates analysis, performed by the PCP, CVA or PCO directives (6.2.1, 6.3.1 & 6.10.1). By default DBIPLOT uses the results from the most recent PCP, CVA or PCO, but you can display results from an earlier analysis by saving the information with the SAVE parameter of PCP, CVA or PCO, and then providing this to DBIPLOT using its own SAVE parameter.

Following the approach of Gower & Hand (1996), the biplot can be viewed as a multivariate analogue of the scatterplot. The information is plotted on the plane defined by the first two principal axes of the analysis (i.e. the first two principal components for a PCP, or the first two canonical variates for a CVA). The default title of the biplot contains the percentage of variance explained by the first and second dimension combined, whilst the title of the x- and y-axis shows the amount of variation explained by the first and second dimension individually (you can specify your own title using the TITLE option). The scores from the analysis are plotted, to show the positions of the individual observations. More importantly, the plot contains an oblique "axis" for each variable (its *biplot axis*) that allows you to see how each individual's projection into this plane relates to its value for the variable concerned. The type of axis to be displayed will depend on how you want to use the plot. The possibilities, selected by the setting of the METHOD option, are as follows:

predictive	plots predictive axes (default),
interpolative	plots interpolative axes.

Predictive axes show the values of the variables that are predicted by the projection into 2-dimensions that is defined for each point by the analysis; essentially this is done by taking an orthogonal projection of the point onto each the biplot axis. Interpolative axes show the values of the variables that would lead to a point being placed at the position of the selected point on the graph. So here the point is being predicted by the variables, rather than the variables by the point. This is done by taking the sum of a set of vectors, one in the direction of each variable, with lengths equal to the values of the variables for that point.

The axes are defined from the loadings from the analysis. With a PCP analysis (6.2.1) or a PCO analysis based on a data matrix (6.10.1), the directions of the axes are given by loadings calculated in the analysis (but the positions of the scale points on the axes differ between the two types of axis). For a CVA analysis (6.3.1), the loadings define the interpolative axes for the biplots, and their inverses define the predictive axes. However, no loadings are available for PCO analyses based a dissimilarity matrices, and so no axes can be plotted. For further explanation, and details of the underlying mathematics, see Gower & Hand (1996).

Arrows are plotted on the axes to represent their loadings (or inverse loadings); the loadings show the approximate contribution of each variable in the first two dimensions. If the loadings are all close to the origin, they are multiplied by a scalar to make them easier to read. By default, the multiplier is calculated automatically, but you can supply a specific value by using the MULTIPLIER option. To save the automatic value, you can set MULTIPLIER to a scalar containing a missing value.

In general, each axis will be at an angle to the traditional x-axis. However, you can arrange for one of the biplot axes to be in the direction of the x-axis, by setting the HORIZONTAL option to the identifier of its variate. It should be noted that this operation is purely cosmetic and, if HORIZONTAL is not set, then the direction of the x-axis will represent the direction of maximum variance.

By default all the axes are plotted, each in a colour chosen automatically by DBIPLOT. However, there are parameters to allow you to modify this for any axis. The VARIABLE parameter specifies the axis to change (using its identifier). The DISPLAY parameter indicates whether the axis is to be shown, hidden or omitted altogether. (The Graphics Viewer of Genstat *for Windows* allows you to toggle displayed items to become hidden, or hidden items to become displayed.) The COLOUR parameter defines the colour to be used, by supplying either a single-valued text with the name of the colour or a scalar containing the RGB value for the colour (see the PEN directive for details).

The scores from PCP analyses are plotted to identify the position of each individual as a red circle, unless you use the GROUPS option to define groupings of the individuals (the groups are then plotted in different colours). With a CVA analysis, groupings are automatically defined from the groups in the analysis itself.

Hotpoints are defined at the point for each of individual to allow you to view the values corresponding to that individual on the axes. In the Graphics viewer in Genstat *for Windows*, you can click on the hotpoint symbol and then click on any score to see how that point is represented on each of the axes. In addition, whatever axes are defined, you can use the the PREDICTIONS option to save a matrix with the predicted values of the individuals for all the variables.

The PLOT option allows you to illustrate other aspects of the scores.

convexhull	draws a convex hull around the points (or the points in each group if groupings have been defined).
means	plots the group means for a CVA, or the group means for a PCP (if the GROUPS option is set), or the overall mean for a PCO biplot. (In other situations the centroid is the origin, which is where all the oblique axes cross, so it would clutter up an already congested plot.)

The types of label for the scores and loadings can be set using the LMINDIVIDUALS and LMVARIABLES parameters respectively, by selecting one of the following settings:

identifiers	uses the identifiers of the variables,
labels	expects labels to be supplied (in a text) using the LINDIVIDUALS or LVARIABLES parameter,
none	gives no labels, and
numbers	uses the row or column numbers of the scores and variables.

If LINDIVIDUALS is set, the default for LMINDIVIDUALS is defined to be labels. Otherwise, if LMINDIVIDUALS is not set, DIBPLOT will use the unit labels of the original data variates or row labels of a data matrix if these are available, or the unit or row numbers if none have been defined. The default for LMVARIABLES is identifiers, unless LVARIABLES is set it is defined to be labels.

The WINDOW and KEYWINDOW options specify the windows to use for the plot and its key, respectively, in the usual way. The SCREEN option controls whether the graphical display is cleared before the biplot is plotted.

The SIZEMULTIPLIER option allows you to modify the sizes of the symbols and labels in the plot. The default of 0.75 works well under most circumstances, but you might want to specify a smaller value to prevent overlapping, when there are large numbers of points or axes to be displayed.

Figure 6.16.1 shows a predictive biplot, plotted following the principal components analysis in Example 6.2.1 by giving the command

```
DIPLOT
```

The figure shows the biplot displayed in the Graphics Viewer of Genstat *for Windows*. Notice that the hotpoint tool has been activated (by clicking on the button on the right-hand side of the menu bar), and a click has been made on point 10 to show its predicted values on the biplot axes.

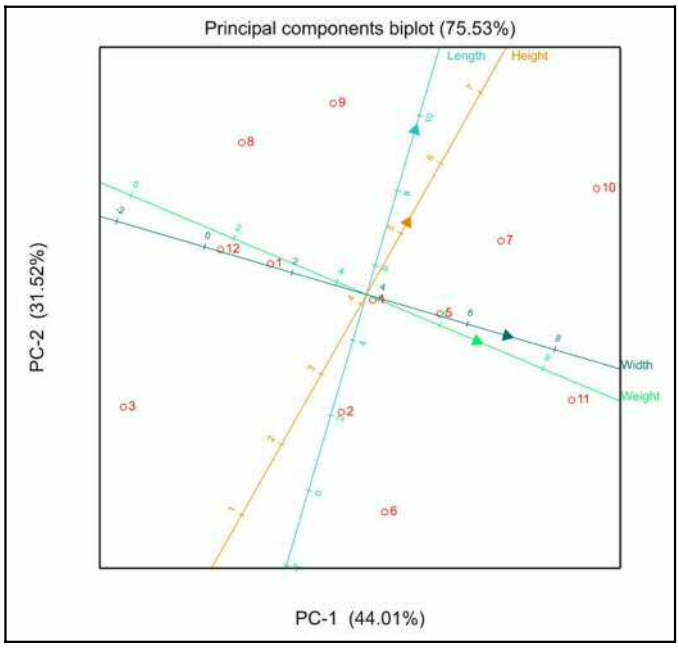


Figure 6.16.1

6.16.2 The CRBIPLOT procedure

CRBIPLOT procedure

Plots correlation or distance biplots after RDA, or ranking biplots after CCA (A.I. Glaser).

Options

DIMENSIONS = *scalars*	Two numbers specifying which axes of the ordinations to plot; default 1,2
PLOT = *string token*	Whether to plot site or species scores (sitescores, speciesscores); default spec
WINDOW = *scalar*	Which graphical window to use; default 1
KEYWINDOW = *scalar*	Which graphical window to use for the key (zero for none); default 2
SAVE = *pointer*	Supplies results from an ordination analysis by CCA or RDA; default uses the most recent analysis

Parameters

X1 = *scalars*, *variates* or *texts*	First explanatory variable to plot; default 1
X2 = *scalars*, *variates* or *texts*	Second explanatory variable to plot; default * i.e. none
LMXVARIABLES = *string tokens*	How to label the x-variables (identifiers, labels, none, numbers); default labe if LXVARIABLES is set, otherwise iden
LMSPECIES = *string tokens*	How to label the species scores (identifiers, labels, none, numbers); default labe if LSPECIES is set, otherwise numb
LMSITES = *string tokens*	How to label the site scores (labels, none, numbers); default labe if LSITES is set, otherwise numb
LXVARIABLES = *texts*	Labels for variables
LSPECIES = *texts*	Labels for species scores

LSITES = *texts*	Labels for site scores

CRBIPLOT provides biplot representations of the results from RDA (6.14) or CCA (6.15), showing projections of species or site scores onto one or two environmental variables. By default CRBIPLOT plots the species scores, but you can set option PLOT=sitescores to plot site scores instead.

CRBIPLOT usually plots the results from the most recent RDA or CCA analysis, but you can display results from an earlier analysis by saving the information about the analysis with the SAVE parameter of RDA or CCA, and then providing this to CRBIPLOT using its own SAVE option.

The type of biplot depends on the scaling method used in the analysis. In RDA, Scaling Type 1 (i.e. no scaling) produces a distance biplot, while Scaling Type 2 (which scales both species and site scores) gives a correlation biplot. Similarly, for CCA, Scaling Type 1 (species scaling) produces a biplot with the sites at the centroids of the species, and Scaling Type 2 (site scaling) plots the species at the centroids of the site.

A distance biplot has the following features:

- distances among elements of Y show approximations of their Euclidean distances in multidimensional space;
- when an element of Y is projected at right angles onto a variable this approximates the position of the object on that variable;
- since the eigenvectors have length one, the length of a projection of an element of Y onto a variable shows its contribution to the formation of that space;
- the angle amongst variables is meaningless.

Figure 6.16.2a shows a distance biplot from the RDA analysis in 6.14, produced by the statement

```
CRBIPLOT [SAVE=SaveRDA]
```

A correlation biplot has the following features:

- distances among elements of Y are *not* approximations of the Euclidean distances between objects in multidimensional space (so the distance biplot is preferable if you want to interpret relationships amongst the elements of Y);
- when an element of Y is projected at right angles onto a variable this approximates the position of the object on that variable;
- the length of a projection of an element of Y onto a variable shows its contribution to the formation of that space;
- the angles between variables approximate their correlation.

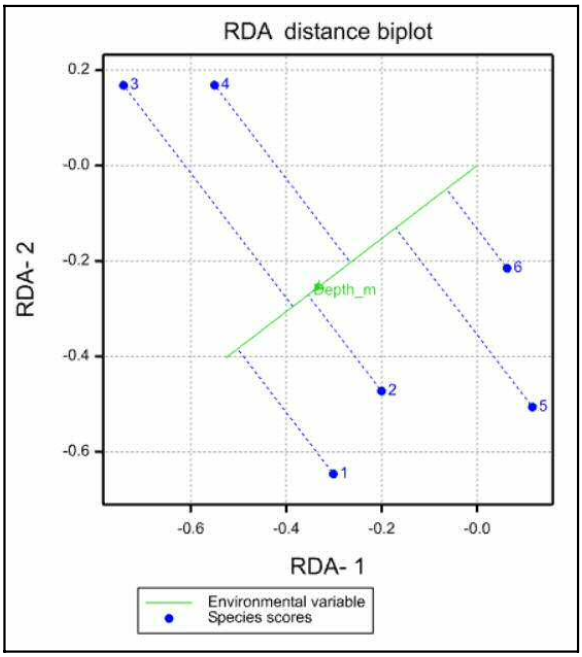


Figure 6.16.2a

In addition when we carry out CCA Scaling Type 1 (site scaling):

- distances among sites show approximations in reduced space of their chi-square distances;
- the sites are at the centroids of the species, and the centroids are calculated using weights equal to the relative frequencies of the species (see Makarenkov & Legendre 2002);
- the position of an object on an explanatory variable can be obtained by projecting the objects at right angle on the variable. This scaling is appropriate when the primary interest is the ordination of sites.

With CCA Scaling Type 2 (species scaling):

- it is the distances among species in reduced space that are approximations of their chi-square distances;
- the species are at the centroids of the sites in the graph;
- any species scores that lie close to the point representing an explanatory variable are more likely to be found with higher frequency at that site than others further away (or more likely to be in State '1' with binary data).

This scaling is appropriate when the primary interest is the relationship between species. Figure 6.16.2b shows an example, from the CCA analysis in 6.15, produced by the statement

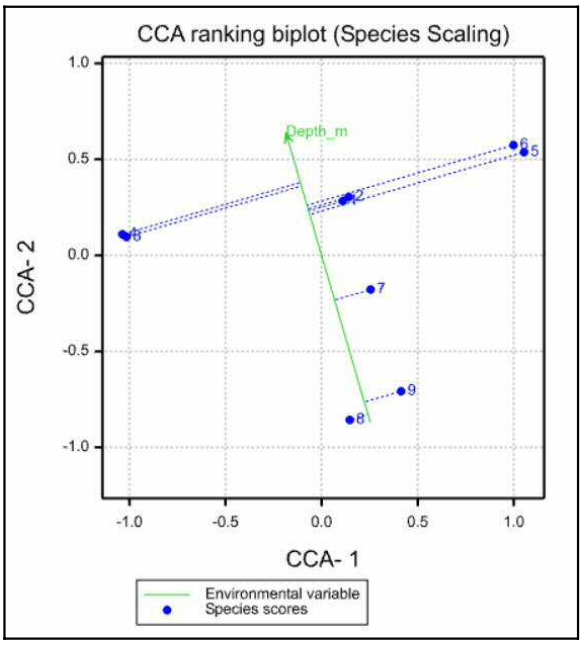


Figure 6.16.2b

```
CRBIPLOT [SAVE=SaveRDA]
```

The explanatory variables to display can be specified using the X1 and X2 parameters. If the variable is a variate, you can set them to its identifier. Alternatively, if it is either a variate or a variable representing one of the levels of a factor, you can set them to the position of the variable in the list of variables involved in the analysis. Finally, if the variable represents the level of a factor, you can set them to a text containing the label used for the variable in the analysis. (You can see the labels by looking at the row labels of the matrix showing the correlations between the environmental variables and the site scores; see Examples 6.14 and 6.15). The DIMENSIONS option lists the numbers of the two canonical axes to plot; default 1,2.

The labels for the species scores, site scores and x-variable(s) can be set using the LMSPECIES, LMSITES and LMXVARIABLES parameters respectively, by selecting one of the following settings:

identifiers	uses the identifiers of the X variates with LMXVARIABLES, or of the Y variates with LMSPECIES (not available with LMSITES),
labels	expects labels to be supplied (in a text) using the LSPECIES, LSITES or LXVARIABLES parameter,
none	gives no labels, and
numbers	uses the column numbers of X and Y.

The defaults are LMSPECIES=numbers, LMSITES=numbers and LMXVARIABLES=identifiers, unless LSPECIES, LSITES or LXVARIABLES is set when the corresponding default becomes labels.

6.16.3 The CRTRIPLOT procedure

CRTRIPLOT procedure

Plots ordination biplots or triplots after CCA or RDA (A.I. Glaser).

Options

DIMENSIONS = *scalars* Which dimensions of the ordinations to display; default

	1,2
PLOT = *string token*	What to plot (sitescores, speciesscores, xvariables); default spec, site, xvar
DGROUPS = *string token*	Features to plot for the XGROUPS variate (ellipse, hull, lines, spider); default * i.e. none
DBINARY = *string token*	What to plot for binary variables (biplot, centroid); default bipl
MULTIPLIER = *scalar*	Value to multiply species and environmental variables scores by when plotting RDA; default *, i.e. none chosen
WINDOW = *scalar*	Which graphical window to use; default 1
KEYWINDOW = *scalar*	Which graphical window to use for the key (zero for none); default 2
SAVE = *pointer*	Supplies results from an ordination analysis by CCA or RDA; default uses the most recent analysis

Parameters

LMXVARIABLES = *string tokens*	How to label the x-variables (identifiers, labels, none, numbers); default labe if LXVARIABLES is set, otherwise iden
LMSPECIES = *string tokens*	How to label the species scores (identifiers, labels, none, numbers); default labe if LSPECIES is set, otherwise numb
LMSITES = *string tokens*	How to label the site scores (labels, none, numbers); default labe if LSITES is set, otherwise numb
LXVARIABLES = *texts*	Labels for variables
LSPECIES = *texts*	Labels for species scores
LSITES = *texts*	Labels for site scores
XGROUPS = *variates*, *factors* or *scalars*	X-variate to generate grouping information to appear on the plot (see the DGROUPS option)

CRTRIPLOT plots ordination biplots or triplots following an analysis from either the RDA (6.14) or CCA (6.15) procedures. By default it uses the results from the most recent RDA or CCA, but you can display results from an earlier analysis by saving the information about the analysis with the SAVE parameter of CCA or RDA, and then providing this to CRTRIPLOT using its own SAVE option.

An ordination biplot displays the site scores, species scores and biplot scores of environmental variables in a two or three dimensional plot. The site scores are plotted as crosses, the species scores are plotted as dashed arrows. The biplot scores of non-binary variables are represented as full lines. The DBINARY option controls how any binary variables are plotted: they can be represented either by triangles plotted at the centroid of the site scores associated with the value '1', or as arrows showing the biplot scores.

The DIMENSIONS option lists the dimensions of the ordination that you want to use. You can list either two or three of these. The default is a two dimensional plot of dimensions 1 and 2. The PLOT option allows you to control what results are plotted, using the following settings:

sitescores	sites scores,
speciesscores	species scores,
xvariables	biplot scores of the environmental variables.

However, if any of the specified DIMENSIONS is higher than the number of canonical axes, the biplot scores of the environmental variables will not be plotted.

In RDA plots, the species scores and biplot scores of environmental variables are usually much smaller than the site scores. So their values are multiplied by a scalar to make them easier to read. The value is set by the procedure and displayed in the output, but you can set your own multiplier by using the MULTIPLIER option.

You can display additional information for one of the explanatory variables by setting the XGROUPS option either to the identifier of the relevant variate or factor, or to a scalar containing its position in the X pointer (see the X parameter of CCA and RDA). The information that appears is controlled by the DGROUPS option, with settings:

ellipse	draws an ellipse showing an approximate 95% confidence interval for the group centroid (2-dimensional plots only),
hull	draws an enclosing convex hull around the species scores by XGROUPS (2-dimensional plots only),
lines	links the species scores by XGROUPS, and
spider	draws lines from the group centroid to each site score.

The group centroid is the (weighted) group mean of the site scores.

The labels for the species scores, site scores and x-variable(s) can be set using the LMSPECIES, LMSITES and LMXVARIABLES parameters respectively, by selecting one of the following settings:

identifiers	uses the identifiers of the X and Y variates,
labels	expects labels to be supplied (in a text) using the LSPECIES, LSITES or LXVARIABLES parameter,
none	gives no labels, and
numbers	uses the column numbers of X and Y.

The defaults are LMSPECIES=numbers, LMSITES=numbers and LMXVARIABLES=identifiers, unless LSPECIES, LSITES or LXVARIABLES is set when the corresponding default becomes labels.

Figures 6.16.3a and 6.16.3b show plots from the RDA and CCA analyses in 6.14 and 6.15, respectively. These were produced by the statements

```
CRTRIPLOT [SAVE=SaveRDA]
CRTRIPLOT [SAVE=SaveCCA]
```

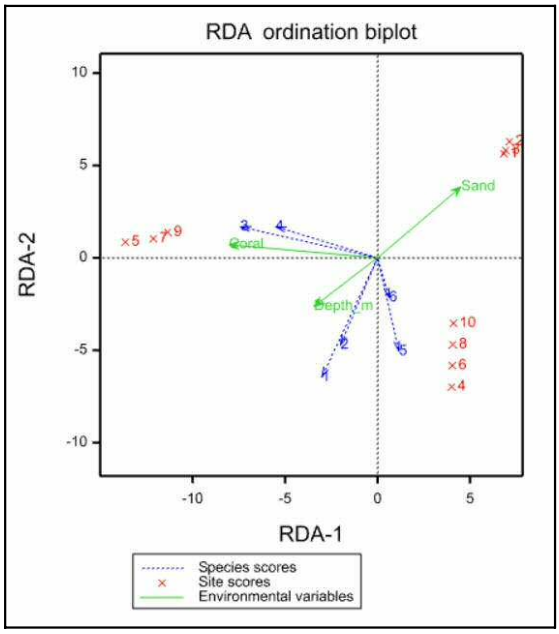


Figure 6.16.3a

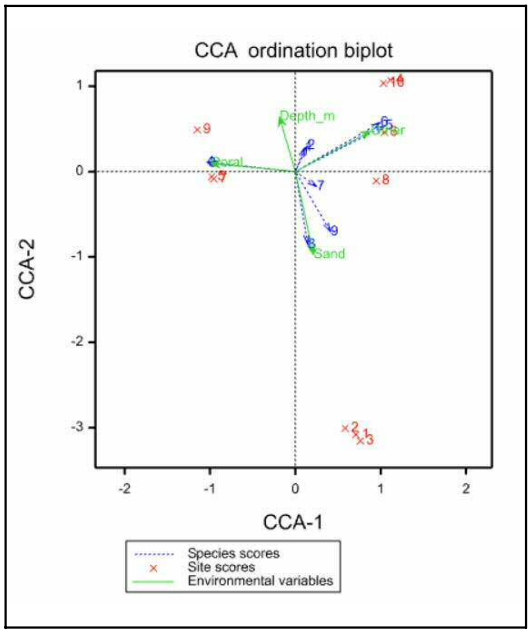


Figure 6.16.3b

6.17 Analysis of skew-symmetry: the SKEWSYMMETRY procedure

SKEWSYMMETRY procedure

Provides an analysis of skew-symmetry for an asymmetric matrix (P.G.N. Digby).

Option

PRINT = *string tokens*	Printed output from the analysis (roots, scores); default * i.e. no output

Parameters

DATA = *matrices*	Asymmetric (square) matrices to be analysed
ROOTS = *diagonal matrices*	Stores the squared singular values from the analysis; the structure has one value for each plane fitted in the analysis (e.g. if the DATA matrix has 11 rows and columns, the ROOTS diagonal matrix will have 5 values)
SCORES = *matrices*	Stores the coordinates of the points from the analysis; each matrix has the same number of rows as the corresponding DATA matrix, and has 2 columns for each plane fitted in the analysis (e.g. if the DATA matrix has 11 rows and columns, the SCORES matrix will have 11 rows and 10 columns)

Procedure SKEWSYMM provides the canonical analysis of skew-symmetry described by Gower (1977). The input to the procedure, specified by the parameter DATA, is a (square) asymmetric matrix of associations. The rows and columns of the matrix usually represent the same set of objects, but in different modes. For example, with migration data, the rows may represent the Countries or States being departed from, and the columns the same locations but being arrived at. The DATA matrix must not contain any missing values.

If A is the asymmetric matrix of associations, then $S = A - A'$ is skew-symmetric; this matrix is analysed using a singular value decomposition, followed by a reflection and rotation, to provide a set of roots and scores. The scores are coordinates for points representing the entities labelling the rows or columns of the DATA matrix. In pairs, these coordinates give positions on a series of planes, also called bimensions. So there is an even number of coordinates for each point; if the DATA matrix has an odd number of rows/columns, there will be one fewer coordinate than the number of rows or columns of the DATA matrix. The roots give the amount of (squared) skew-symmetry explained in each pair of dimensions, allowing the "importance" of each plane to be assessed.

The results are interpreted in terms of the areas of triangles. The skew symmetry between the entities in rows (or columns) p and q is proportional to the area of the triangle OPQ, where O is the origin, and P and Q are the points representing p and q respectively. (For further details see either Gower 1977 or Digby & Kempton 1987.) Within each plane the coordinates are arranged so that their centroid is at $(0,y)$, for $y>=0$, and so that positive row-to-column skew symmetry is represented in a clockwise direction. (Note that in planes other than the first it is residual skew symmetry, after fitting the preceding planes, that is being modelled).

Printed output is controlled by the strings listed for the PRINT option: roots prints the roots (also the roots expressed as percentages and cumulative percentages) and scores prints the scores. Results from the analysis can be saved using the parameters ROOTS and SCORES. The structures specified for these parameters need not be declared in advance. Column labels are provided automatically for the SCORES matrix, but any row labels (useful to identify the entities) are left unchanged.

Example 6.17 analyses some data from Table 6.7 of Digby & Kempton (1987). Figures 6.17a and 6.17b plot the first two axes of skew symmetry.

Example 6.17

```
  2  TEXT   [VALUES=Bare,Lichens,Grasses,Erica,'E/C',Calluna,'C/M', \
  3         Mosses,'C/A','Arctost.'] Vegstate
  4  &      [VALUES=B,L,G,E,EC,C,CM,M,CA,A] Labels
  5  MATRIX [ROWS=Vegstate; COLUMNS=Labels] Heath,Coords; VALUES= \
  6    !(15,18,47,15, 5, 1, 1, 1, 5, 3,  0,11,17,27, 0, 8, 1, 6, 3,14,\
  7       0, 0, 5,20, 5, 8, 1, 3, 0, 8,  0, 1, 0,10, 4,21, 3, 7, 0, 0,\
  8       0, 0, 0, 5,10, 5, 4, 0, 5, 0,  4, 1, 0, 7, 2,18,11, 1, 1, 3,\
  9       0, 3, 1, 0, 0, 0,101,29,16,3,  0, 0, 0, 0, 0, 3, 7,17, 0, 5,\
 10       0, 0, 0, 1, 0, 1, 0, 0, 6, 9,  0, 0, 0,10, 0,21, 0, 2, 5, 7)
 11  PRINT  Heath; FIELDWIDTH=6; DECIMALS=0
```

	Heath									
Labels	B	L	G	E	EC	C	CM	M	CA	A
Vegstate										
Bare	15	18	47	15	5	1	1	1	5	3
Lichens	0	11	17	27	0	8	1	6	3	14
Grasses	0	0	5	20	5	8	1	3	0	8
Erica	0	1	0	10	4	21	3	7	0	0
E/C	0	0	0	5	10	5	4	0	5	0
Calluna	4	1	0	7	2	18	11	1	1	3
C/M	0	3	1	0	0	0	101	29	16	3
Mosses	0	0	0	0	0	3	7	17	0	5
C/A	0	0	0	1	0	1	0	0	6	9
Arctost.	0	0	0	10	0	21	0	2	5	7

```
 12   " Use SKEWSYMM, saving SCORES, printing roots only "
 13  SKEWSYMM  [PRINT=roots] Heath; SCORES=Coords

Canonical analysis of Skew-Symmetry
===================================

Squared singular values for each plane
--------------------------------------
```

	Sk_Roots	%Roots	Cum%Root
1	9605	76.50	76.50
2	2095	16.69	93.19
3	773	6.16	99.35
4	81	0.64	99.99
5	1	0.01	100.00

```
 14  CALCULATE Score[1...4] = Coords$[*; 1...4]
 15  FRAME  3; SCALING=xyequal
 16  XAXIS  5,6; LOWER=-6.25,-2.75; UPPER=(5.25)2; YORIGIN=0
 17  YAXIS  5,6; LOWER=-3.25,-2.5; UPPER=8.25,5.5; XORIGIN=0
 18  PEN    3; SYMBOLS=0; LABELS=Vegstate; SIZE=1.25
 19  XAXIS  3; LOWER=-6.25; UPPER=5.25; YORIGIN=0
 20  YAXIS  3; LOWER=-3.25; UPPER=8.25; XORIGIN=0
 21  DGRAPH [TITLE='Skew-symmetry: first plane'; WINDOW=3; \
 22         KEYWINDOW=0] Score[2]; Score[1]; PEN=3
 23  XAXIS  3; LOWER=-2.75; UPPER=5.25; YORIGIN=0
 24  YAXIS  3; LOWER=-2.5; UPPER=5.5; XORIGIN=0
 25  DGRAPH [TITLE='Skew-symmetry: second plane'; WINDOW=3; \
 26         KEYWINDOW=0] Score[4]; Score[3]; PEN=3
```

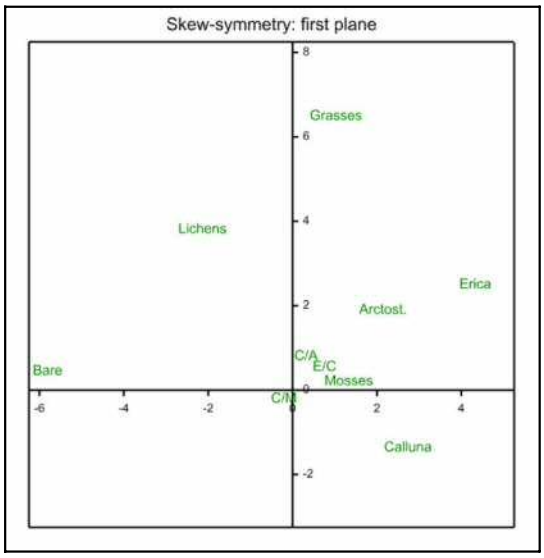


Figure 6.17a

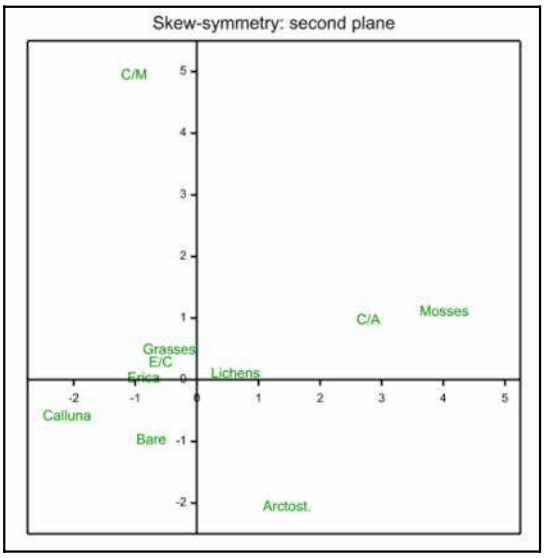


Figure 6.17b

6.18 Procrustes rotation

Multivariate analyses often give the coordinates of a set of points in some multidimensional space. Typically these are obtained so that certain features of the underlying data are represented by the distances between the points in the multidimensional space. One example is principal components analysis, where the distance amongst the principal component scores represents the Pythagorean distances between the values in the data matrix. Another example is canonical variates analysis, where the distance between the canonical variate scores for the means is the Mahalanobis distance between the groups. The distances amongst a set of points do not change if the origin of the coordinate system is shifted, nor do they change if the axes of the coordinate system are rotated.

Suppose that two sets of points are obtained for the same set of objects but with respect to different coordinate systems. For example, two sets of data concerning the same set of objects may be analysed using principal components analysis to give two sets of principal component scores. Alternatively, one set of data may be analysed using two different methods, again giving two sets of points for the same set of objects. The question that now arises is: can the two sets of points be related to each other without disturbing the relationships contained inside the sets? Since the properties of distance are unchanged by a shift of origin or a rotation of the axes, this question is equivalent to asking whether the coordinate system for one set of points can be shifted and rotated so that they match, as well as possible, the coordinates of the other set of points.

Procrustes rotation, of which there are several variants (Gower 1975b, 1985a), addresses this problem; orthogonal Procrustes rotation is the method most commonly used, and is provided by the `ROTATE` directive. Suppose that there are two sets of coordinates for n points in r dimensions contained in the $n \times r$ matrices X and Y. The X-set is arbitrarily supposed to be a fixed configuration, and the Y-configuration is to be shifted and rotated so that it best matches the X-set. Here *best* means minimizing the sum of the squared distances between the points in the X-set and the matching shifted and rotated points in the Y-set. The best translation (shift of origin) makes the centroids for the two sets of points coincide; this is easily done by translating both sets of points so that their centroids are at the origin. After translation, to find the best rotation involves doing a singular value decomposition (see, for example, Digby & Kempton 1987).

After translation and rotation the goodness of fit can be assessed by the residual sum of squares, which is the sum of squared distances between each X-point and the corresponding Y-

point, after translation and rotation. Sometimes the relationships contained inside *X* and inside *Y* are similar but are expressed on different scales. You might then want the coordinates in the *Y*-set to be stretched or contracted by a scaling factor; this can be estimated by least squares. But least-squares scaling should not be used if *X* and *Y* are known to be on comparable scales: for example, they may both have come from canonical variates analysis and thus express Mahalanobis distance.

When you cannot say which configuration of points is the fixed set, you might want to know about the results of both Procrustes rotations. The best translation remains the same: both configurations of points are translated so that their centroids coincide, typically at the origin. If the best rotation of *Y* to *X* is given by the orthogonal matrix *H*, then the best rotation of *X* to *Y* is the transpose of *H*. If least-squares scaling is not used, the two residual sums of squares will be the same, unless there is a reflection that has been suppressed. However, if scaling is used, then in general these residuals will differ; you can overcome this by arranging that the two configurations of points, after translation, have the same sum of squares: a convenient value is unity. This initial scaling is particularly desirable when several configurations are to be compared pair by pair.

In general, the best rotation of *Y* to *X* may contain a reflection. Usually this is acceptable; however, you may sometimes want to stipulate that the rotation should be a pure rotation and not contain any reflection (Gower 1975a).

Above we have assumed that the two matrices of coordinates have the same number of columns: that is, that the dimensionalities of the two multidimensional spaces are the same. If they differ, Genstat pads out the smaller matrix with columns of zero values, so that it matches the larger.

6.18.1 The ROTATE directive

ROTATE directive

Does a Procrustes rotation of one configuration of points to fit another.

Options

PRINT = *string tokens*	Printed output required (rotations, coordinates, residuals, sums); default * i.e. no printing
SCALING = *string token*	Whether or not isotropic scaling is allowed (yes, no); default no
STANDARDIZE = *string tokens*	Whether to centre the configurations (at the origin), and/or to normalize them (to unit sum of squares) prior to rotation (centre, normalize); default cent,norm
SUPPRESSREFLECTION = *string token*	Whether to suppress reflection (yes, no); default no

Parameters

XINPUT = *matrices*	Inputs the fixed configuration
YINPUT = *matrices*	Inputs the configuration to be fitted
XOUTPUT = *matrices*	To store the (standardized) fixed configuration
YOUTPUT = *matrices*	To store the fitted configuration
ROTATION = *matrices*	To store the rotation matrix
RESIDUALS = *matrices* or *variates*	To store distances between the (standardized) fixed and fitted configurations
RSS = *scalars*	To store the residual sum of squares

The ROTATE directive provides orthogonal Procrustes rotation. You must set the parameters XINPUT and YINPUT, which specify respectively the fixed configuration and the configuration that you want to be translated and rotated; these are called *X* and *Y* above. The other parameters are used for saving results from the analysis. For *X* and *Y* to refer to the same set of objects they must have the same number of rows, and each object must be represented by the same row in both *X* and *Y*. If the XINPUT matrix is $n \times p$ and the YINPUT matrix is $n \times q$, Genstat does the analysis using matrices that are $n \times r$, where r is $\max(p, q)$. The smaller matrix is expanded with columns of zeros, as explained above.

The PRINT option specifies which results you want to print; the settings are:

coordinates	specifies that the fixed and fitted configurations are to be printed; note that the fixed configuration is printed after any standardization (see below), and the fitted configuration is printed after standardization and rotation.
residuals	prints the residual distances of the points in the fixed configuration from the fitted points; this is after any standardization and rotation.
rotations	prints the orthogonal rotation matrix.
sums	prints an analysis of variance giving the sums of squares of each configuration, and the residual sum of squares; if scaling is used, the scaling factor is also printed.

The three other options of the ROTATE directive control the form of analysis. The SCALING option specifies whether you want least-squares scaling to be applied to the standardized YINPUT matrix when finding the best fit to the fixed configuration. You should set SCALING=yes if you want scaling; Genstat will then print the least-squares scaling factor with the analysis of variance. By default there is no scaling.

The STANDARDIZE option specifies what preliminary standardization is to be applied to the XINPUT and YINPUT matrices. It has settings:

centre	centre the matrices to have zero column means;
normalize	normalize the matrices to unit sums of squares.

The default is STANDARDIZE=centre,normalize. The initial centring ensures that the configurations are translated to have a common centroid, and thus automatically provides the best translation of *Y* to match *X*. The normalization arranges that the residual sum of squares from rotating *X* to *Y* is the same as that for rotating *Y* to *X*. Switching off both centring and standardization is rarely advisable, but can be requested by putting STANDARDIZE=*.

With some methods of multivariate analysis, for example the analysis of skew-symmetry (6.17), the direction of travel about the origin is important. It is then undesirable to perform a reflection as part of the rotation: the SUPPRESSREFLECTION option can be used to prevent this. The default setting is no, which allows reflection to take place.

As an example, we again consider the galaxies discussed in 6.3. Figures 6.10.1 and 6.12a show very similar relationships amongst the galaxy types even though they were produced by different methods, principal coordinates analysis and non-metric multidimensional scaling respectively. Indeed the pictures are almost identical, apart from one being the mirror image of the other. Example 6.18.1 uses Procrustes rotation to assess their similarity. Whereas the scales in Figure 6.10.1 bear a relation to the actual distances input to PCO, those in Figure 6.12a need not because in the MDS solution it is only the order of the distance values that is important. So the scaling option of the ROTATE command (lines 11-12) has been set to yes: this also ensures that the sum of squares of the fitted configuration plus that of the residual will equal the sum of squares of the fixed configuration. To assist in the comparison of the two analyses in Example 6.18.1 no normalization is done, and since both input configurations are already centred any standardization has been suppressed. The rotation matrix for a simple reflection would take the

form $\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$ and that from the ROTATE command is very similar to it, although there is also a slight rotation of arccos(0.99888), that is, about 2.7 degrees. None of the residuals is especially large or small: the second smallest is for the last galaxy type, the Irregulars, which may be because their points are remote from the points for the other galaxy types.

The least-squares scaling factor of 0.9753 is the amount by which the MDS solution has been scaled, after which the sum of squares of its points from the origin is 9.51. The sum of the squared residuals is 0.21, which is also the difference between the sums of squares of the fixed and fitted configurations. Lines 13-17 extract the fixed and fitted coordinates and plot them as Figure 6.18.1, with the larger symbols being used for the fixed points from the PCO analysis. Note that option SCALING=xyequal is used in the FRAME statement (line 14) – as is appropriate for output from many multivariate analyses.

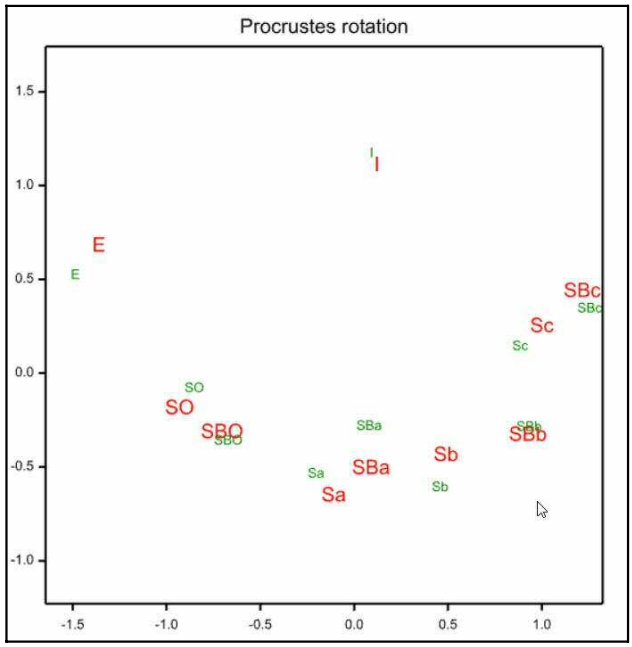


Figure 6.18.1

The second Procrustes rotation in Example 6.18.1 (lines 18-19) is similar to the first, except that reflection has been suppressed. Whilst there is no statistical reason to do this with these configurations of points, it does illustrate what can happen if reflections are suppressed unnecessarily. It is obvious with this example, where only two dimensions are being considered, but with coordinates in more dimensions the effect may be less apparent. The rotation matrix specifies rotation through 180 degrees (apart from 0.5 degree). The sums of squares for the two configurations, and also the scaling factor, are the same as with the first analysis; however the residual is now much larger so that the sums of squares do not add up, as noted below the table.

Example 6.18.1

```
  2  TEXT   [VALUES=E,SO,SBO,Sa,SBa,Sb,SBb,Sc,SBc,I] Galaxies
  3  MATRIX [ROWS=Galaxies; COLUMNS=2] Pco,Mds
  4  READ   [SERIAL=yes] Pco,Mds

   Identifier   Minimum      Mean   Maximum    Values   Missing
          Pco    -1.397 -0.50E-05     1.117        20         0
          Mds    -1.202  0.50E-05     1.572        20         0

 11  ROTATE [PRINT=rotations,residuals,sums; SCALING=yes; STANDARDIZE=*] \
 12         XINPUT=Pco; YINPUT=Mds; YOUTPUT=Mdsout

Procrustes rotation
===================

Orthogonal rotation
-------------------

                  1          2
            1    -0.99888     0.04733
            2     0.04733     0.99888
```

```
Residuals
---------

                    1
            1      0.1914
            2      0.1505
            3      0.0832
            4      0.1389
            5      0.2271
            6      0.1700
            7      0.0600
            8      0.1405
            9      0.1164
           10      0.0696


Sums of Squares
---------------

Fitted Configuration           9.5124
Residual                       0.2077
Fixed Configuration            9.7201

Least-squares scaling factor = 0.9753

  13  CALCULATE Pco1,Pco2,Mdsout1,Mdsout2 = Pco$[*; 1,2],Mdsout$[*; 1,2]
  14  FRAME  3; SCALING=xyequal
  15  PEN    1,2; COLOUR='red','green'; SYMBOLS=0; LABELS=Galaxies; SIZE=1.5,1
  16  DGRAPH [TITLE='Procrustes rotation'; WINDOW=3; KEYWINDOW=0] \
  17         Pco2,Mdsout2; Pco1,Mdsout1; PEN=1,2
  18  ROTATE [PRINT=rotations,sums; SCALING=yes; STANDARDIZE=*;\
  19         SUPPRESSREFLECTION=yes] XINPUT=Pco; YINPUT=Mds

Procrustes rotation
===================

Orthogonal rotation
-------------------

                    1           2
            1    -0.99996     0.00908
            2    -0.00908    -0.99996


Sums of Squares
---------------

Fitted Configuration           9.5124
Residual                      11.4846
Fixed Configuration            9.7201

Least-squares scaling factor = 0.9753

* MESSAGE: a reflection has been suppressed, sums of squares need not total.
```

6.18.2 Generalized Procrustes rotation: the GENPROCRUSTES procedure

GENPROCRUSTES procedure

Performs a generalized Procrustes analysis (G.M. Arnold & R.W. Payne).

Options

PRINT = *string tokens*	Printed output required (analysis, centroid, column, individual, monitoring); default anal, cent
SCALING = *string token*	Type of scaling to use (none, isotropic, separate); default none

METHOD = *string token*	Method to be used (Gower, TenBerge); default Gowe
NROOTS = *scalar*	Number of roots (i.e. dimensions) to print for the output configurations, consensus and rotation matrices, and number of dimensions to save with the XOUTPUT, CONSENSUS and ROTATIONS paramaters if their matrices have alread not been defined; default is to print and save all the dimensions
PLOT = *string tokens*	Controls which graphs to display (consensus, individuals, projections); default * i.e. none
NDROOTS = *scalar*	Number of dimensions to display in the consensus and individuals plots; default 3
TOLERANCE = *scalar*	The algorithm is assumed to have converged when (last residual sum of squares) − (current residual sum of squares) < TOLERANCE × (number of configurations); default 0.00001
MAXCYCLE = *scalar*	Limit on number of iterations; default 50

Parameters

XINPUT = *pointers*	Each pointer points to a set of matrices holding the original input configurations
XOUTPUT = *pointers*	Each pointer points to a set of matrices to store a set of final (output) configurations
CONSENSUS = *matrices*	Stores the final consensus configuration from each analysis
ROTATIONS = *pointers*	Each pointer points to a set of matrices to store the rotations required to transform each set of XINPUT configurations to their final (scaled) XOUTPUT configurations
RESIDUALS = *pointers*	Each pointer points to a set of matrices to store the distances of a set of scaled XINPUT configurations from its consensus
RSS = *scalars*	Stores the residual sum of squares from each analysis
ROOTS = *diagonal matrices*	Stores the latent roots from referring the centroid configuration to its principal axis form (consensus) for each analysis
WSS = *scalars*	Stores the initial within-configuration sum of squares from each analysis
SCALINGFACTOR = *variates*	Stores the isotropic scaling factors for configurations from each analysis
PROJECTIONS = *pointers*	Each pointer points to a set of matrices to store a set of projection matrices

Generalized Procrustes analysis is widely used in sensory analysis of food, wine etc. to match configurations of points which may arise, for example, from different assessors. The analysis iteratively matches the configurations to a common centroid configuration using the operations of translation to a common origin, rotation and reflection of axes, and possibly also scale changes. This matching seeks to minimize the sum of the squared distances between the centroid and each individual configuration summed over all points (the Procrustes statistic for each configuration and the centroid, summed over all configurations). The final centroid is referred to principal axes to give a unique consensus configuration. Two methods of scaling are available (controlled by the SCALING option). Isotropic scaling, which scales the all the dimensions of

each configuration by an equal amount, takes place during the Procrustes analysis. The alternative is to scale each configuration prior to the analysis so that the trace of each matrix is one (see Arnold 1992).

The XINPUT parameter specifies a pointer storing the configurations as matrices. The other parameters (XOUTPUT, CONSENSUS, ROTATIONS, RESIDUALS, RSS, ROOTS, WSS, SCALINGFACTOR and PROJECTIONS) save the various results. There are options for different methods to use for the matching (SCALING, METHOD), control of convergence (TOLERANCE, MAXCYCLE) and printing and plotting of results (PRINT, PLOT, NROOTS and NDROOTS).

The default method used by GENPROCRUSTES is that given by Gower (1975b). Suppose we have a set of M input configurations X_i (i=1...M) each representing a configuration of N points in V dimensions. Each matrix X_i is initially column-centred (and the individual column means for each configuration can be printed by including column amongst the settings of PRINT option). A constraint is required on the overall sum of squares to prevent the trivial solution of matching by all configurations collapsing to the origin. In GENPROCRUSTES, the constraint used is

$$\sum (\text{trace} (X_i' X_i)) = M.$$

An initial estimate of the centroid is found from these centred and scaled configurations; firstly X_2 is rotated to X_1, with the rotated X_2 saved as the new X_2 and the centroid computed as the mean of X_1 and the new X_2; X_3 is rotated to this centroid which is then recalculated as the mean of the three current configurations; and so on until all configurations X_i (i=1...M) have been included. The centroid thus found is taken as the initial centroid estimate Y, with the rotated values as the new X_i. The initial residual sum of squares S_r is calculated as

$$Sr = M \times (1 - \text{trace} (Y' Y)).$$

Each of the current configurations X_i is then rotated to Y and the rotated position saved as the new X_i. The updated estimate of the centroid Y_n is calculated as the mean of the new X_i (i=1...M) and the new residual sum of squares calculated as

$$Sr_n = M \times (1 - \text{trace} (Y_n' Y_n)).$$

If isotropic scaling has been requested (by setting option SCALING=yes) new estimates ro_i' of the individual scaling factors ro_i (originally set to 1) are now found by

$$ro_i'/ro_i = \sqrt{(\text{trace}(X_i' Y_n)/(\text{trace}(X_i' X_i) \times \text{trace}(Y_n' Y_n)))}$$

and each X_i is updated by a factor of ro_i'/ro_i. The centroid is then recalculated as the mean of the new X_i and the new residual sum of squares calculated in a similar manner to before. If the change in residual Sr is less than a preset tolerance (controlled by option TOLERANCE) the algorithm is taken to have converged. If not, the process is repeated until the tolerance is reached, up to a maximum number of iterations as set by the option MAXCYCLE (default 50) after which a message of non-convergence is printed and the procedure terminated. Monitoring information about convergence can be printed by including the monitoring setting with the PRINT option.

After convergence a unique consensus configuration is found by referring the final centroid to principal axes; the corresponding latent roots may be saved using the ROOTS parameter. Final results for the consensus and individual configurations (referred to the same principal axes) may be printed using the centroid and individual settings of the PRINT option, and/or saved using the parameters XOUTPUT, CONSENSUS and ROTATIONS. Analysis of variation for the M configurations (including the individual scaling factors) and for the N points, along with the initial within and between configurations sums of squares (*WSS* and *BSS*), the final residual sum of squares (*RSS*) and number of steps in the iteration process may be printed using the analysis setting of the PRINT option. The initial within-configuration sum of squares, final residual sum of squares and individual isotropic scaling factors may also be saved using, respectively, the WSS, RSS and SCALINGFAC parameters. (Note that the final results are still scaled by the original factor from the initial overall constraint; to return to the original scale all sums of squares need adjustment by a factor of WSS/M and configurations by the square root of that factor).

Modifications to the method described above are given in TenBerge (1975), and may be invoked by the `TenBerge` setting of the `METHOD` option. This may give considerable savings in the time to reach convergence (Arnold 1988).

Note that the special case of M=2 corresponds to the classical pairwise Procrustes matching (`ROTATE` directive) except that by fitting each configuration to a common centroid the requirement to regard one of the initial configurations as fixed is obviated.

Example 6.18.2

```
  2  " Data from Gower (1975b). Note, however, that in Table 3 the
 -3    scaling factors printed were SQRT(ro[i]) instead of ro[i],
 -4    and in Table 4 the Between and Within Judges sums of squares
 -5    were transposed."
  6  MATRIX [ROWS=9; COLUMNS=7] X[1...3]
  7  READ [PRINT=errors; SERIAL=yes] X[]
 37  GENPROCRUSTES [PRINT=analysis,centroid; SCALING=isotropic] X

Generalized Procrustes analysis
===============================

Isotropic scaling


Rotation of centroid to principal axes
======================================

Latent roots
------------

        1        2        3        4        5        6        7
    0.609    0.081    0.064    0.027    0.012    0.004    0.002


Percentage variance
-------------------

        1        2        3        4        5        6        7
    76.12    10.12     8.05     3.36     1.54     0.56     0.26


Coordinates of the consensus configuration
==========================================

                  1        2        3        4        5        6        7
            1 -0.08776  0.17976 -0.08543  0.07956 -0.00947  0.01898 -0.00739
            2  0.14525  0.01582 -0.07554 -0.09195 -0.04359 -0.01877 -0.01829
            3 -0.14286 -0.00618 -0.06175 -0.05358 -0.02329  0.01879  0.03331
            4 -0.14958 -0.04543 -0.10026 -0.00187  0.08222 -0.02304  0.00067
            5 -0.14987  0.09345  0.13294 -0.05591  0.02574  0.00807 -0.01186
            6  0.31770  0.06500  0.11896  0.01323  0.01113 -0.01746  0.01775
            7  0.09444 -0.05719  0.00439  0.07142 -0.03728 -0.03146  0.00112
            8 -0.46364 -0.12395  0.07526  0.02665 -0.02299  0.00692 -0.00669
            9  0.43631 -0.12128 -0.00856  0.01245  0.01754  0.03797 -0.00861


Analysis of variation for the configurations
============================================

         Scaling Residual    Total
     1     1.071    0.240    0.931
     2     1.222    0.177    1.033
     3     0.832    0.181    1.036
```

```
Analysis of variation for the entities
======================================

      Consensus Residual    Total
    1     0.162    0.066    0.228
    2     0.114    0.084    0.198
    3     0.087    0.079    0.166
    4     0.125    0.067    0.192
    5     0.159    0.129    0.287
    6     0.361    0.038    0.399
    7     0.059    0.026    0.085
    8     0.712    0.032    0.744
    9     0.622    0.079    0.701


Initial within-configuration sum of squares    53254.889
Initial between-configuration sum of squares   22114.815
Final residual sum of squares                      0.599
Number of steps to convergence 8
```

6.18.3 Multiple Procrustes analysis: the PCOPROCRUSTES procedure

PCOPROCRUSTES procedure

Performs a multiple Procrustes analysis (P.G.N. Digby).

Options

PROTATE = *string tokens*	Printed output required from each Procrustes rotation (rotations, coordinates, residuals, sums); default * i.e. no output
PPCO = *string tokens*	Printed output required from the PCO analysis (roots, scores, centroid); default root, score, cent
SCALING = *string token*	Whether isotropic scaling should be used for the Procrustes rotations (no, yes); default no
STANDARDIZE = *string tokens*	Whether to centre the configurations and/or normalize them to unit sums-of-squares for the Procrustes rotations (centre, normalize); default cent, norm

Parameters

DATA = *pointers*	Each pointer points to a set of matrices holding the original input configurations
LRV = *LRVs*	Stores the latent vectors (i.e. coordinates), roots and trace from the PCO analysis
CENTROID = *diagonal matrices*	Stores the squared distances of the points representing the input configurations from their overall centroid from the PCO analysis
DISTANCES = *symmetric matrices*	Stores the residual sums-of-squares from the Procrustes rotations

Multiple Procrustes analysis operates on a set of M configurations of points, each representing the coordinates of N units in V dimensions. The analysis compares them in pairs, keeping the residual sums-of-squares, and then performs a principal coordinate analysis of the residual sums-of-squares to obtain an ordination representing the individual configurations. The rows of the matrices must represent the same set of units, in the same order; however there is no need for them to have the same number of columns (although generally they will do). An example of the use of multiple Procrustes analysis is given by Digby & Kempton (1987, pages 121-123).

The configurations of points are specified using the DATA parameter. This supplies a pointer

containing a matrix with the data for each configuration. The PROTATE option controls the output from the individual Procrustes rotations, and the PPCO option controls that from the principal coordinate analysis. There are $M\times(M-1)/2$ Procrustes rotations so, by default, PROTATE=* to suppress any output. The SCALING and STANDARDIZE options control the way in which the Procrustes rotations are carried out, using the SCALING and STANDARDIZE options of ROTATE. However, the combination of SCALING=yes and STANDARDIZE=centre should not be used, because then the results will be dependent on the order of the input matrices. The LRV and CENTROID parameters can be used to save results from the principal coordinates analysis, and the DISTANCES parameter can be used to save the symmetric matrix of the residual sums-of-squares from the Procrustes analyses.

Example 6.18.3 uses multiple Procrustes analysis to compare seven different ways of generating ordinations of 16 grass species. The first two dimensions of the solution are plotted in Figure 6.18.3.

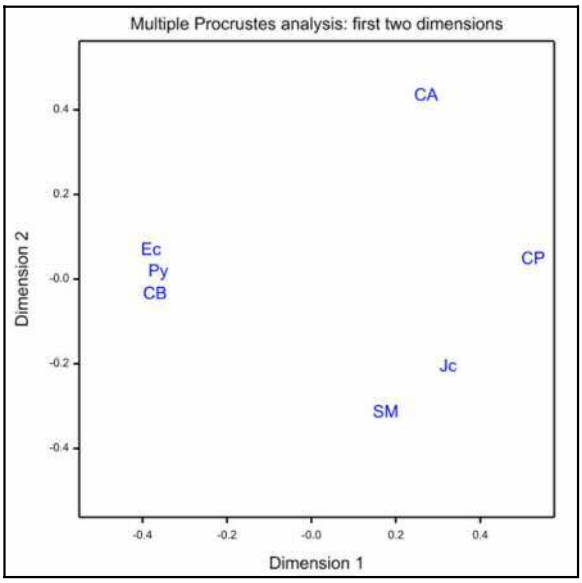


Figure 6.18.3

Example 6.18.3

```
  2  " Abundances of 16 grass species on 9 plots of land:
 -3    part of Table 1.1 in Digby & Kempton (1987)."
  4  UNITS [NVALUES=16]
  5  READ  [SERIAL=yes] Abund[1...6]

   Identifier   Minimum      Mean   Maximum    Values   Missing
     Abund[1]    0.0000     4.744     33.20        16         0    Skew
     Abund[2]    0.0000     4.206     13.10        16         0
     Abund[3]    0.0000     5.844     37.60        16         0    Skew
     Abund[4]    0.0000     5.300     37.00        16         0    Skew
     Abund[5]    0.0000     5.463     48.70        16         0    Skew
     Abund[6]    0.0000     6.250     82.70        16         0    Skew

 12  CALCULATE   LogAbund[1...6] = LOG10(Abund[1...6] + 1)
 13  &           PrsAbund[1...6] = Abund[1...6] > 0
 14  " Form similarity matrices using 5 different methods
-15    on suitably transformed copies of the data."
 16  FSIMILARITY [SIMILARITY=Sjaccard] PrsAbund[]; Jaccard
 17  &           [SIMILARITY=Ssmc]     PrsAbund[]; simplematching
 18  &           [SIMILARITY=Scity]    LogAbund[]; cityblock
 19  &           [SIMILARITY=Secol]    LogAbund[]; ecological
 20  &           [SIMILARITY=Spythag]  LogAbund[]; Pythagorean
 21  POINTER     [NVALUES=7] Config
 22  MATRIX      [ROWS=16; COLUMNS=6] Config[]
 23  LRV         [ROWS=16; COLUMNS=6] Pcol
 24  " Use PCO on each similarity matrix, to get 5 ordinations', \
-25    of 16 points in 6 dimensions."
 26  FOR         Dsim=Sjaccard,Ssmc,Scity,Secol,Spythag; Dcpco=Config[1...5]
```

```
  27    PCO       Dsim; LRV=Pcol
  28    CALCULATE Dcpco = Pcol[1]
  29  ENDFOR
  30  " Use correspondence analysis on the data, and the data
 -31    transformed to presence/absence, to get 2 more
 -32    ordinations of 16 points in 6 dimensions."
  33  MATRIX      [ROWS=16; COLUMNS=6] MatAbund
  34  CALCULATE   MatAbund$[*; 1...6] = Abund[]
  35  CORANALYSIS [METHOD=digby] MatAbund; ROW=Config[6]
  36  CALCULATE   MatAbund = MatAbund > 0
  37  CORANALYSIS [METHOD=digby] MatAbund; ROW=Config[7]
  38  TEXT        [VALUES=Jc,SM,CB,Ec,Py,CA,CP] Points
  39  SYMMETRICMATRIX [ROWS=Points] MPdist
  40  " Use multiple Procrustes analysis to compare
 -41    the 7 different ordination methods."
  42  PCOPROCRUSTES Config; LRV=MPLRV; DISTANCE=MPdist

Principal coordinates analysis
==============================

Latent Roots
------------

           1         2         3         4         5         6         7
      0.8906    0.3382    0.1222    0.0730    0.0448    0.0279    0.0000


Percentage variation
--------------------

           1         2         3         4         5         6         7
       59.50     22.60      8.17      4.88      2.99      1.86      0.00


Trace
-----

      1.497


Latent vectors (coordinates)
----------------------------

                     1         2         3         4         5         6
         1      0.3034   -0.2079    0.0634    0.0900   -0.1420   -0.0297
         2      0.1453   -0.3166    0.1253   -0.0214    0.1164    0.0527
         3     -0.3986   -0.0368    0.0100   -0.0242    0.0456   -0.1315
         4     -0.4039    0.0673   -0.1162    0.1764    0.0191    0.0568
         5     -0.3864    0.0163   -0.0035   -0.1697   -0.0861    0.0599
         6      0.2435    0.4319    0.1676    0.0110    0.0180    0.0033
         7      0.4968    0.0458   -0.2466   -0.0620    0.0290   -0.0116

* MESSAGE: vectors corresponding to zero or negative roots are not printed.


Centroid distances
------------------

          Centdist
                1        2        3        4        5        6        7
           0.4104   0.3922   0.4246   0.4647   0.4352   0.5238   0.5609


  43  PRINT MPdist; FIELD=8; DECIMALS=4

      MPdist

   1  0.0000
   2  0.1266  0.0000
   3  0.5836  0.4264  0.0000
   4  0.6492  0.5559  0.1032  0.0000
   5  0.6091  0.4731  0.0783  0.1465  0.0000
   6  0.4568  0.5849  0.6769  0.6628  0.6455  0.0000
```

```
   7  0.2505  0.4066  0.8906  0.8905  0.8700  0.3905  0.0000
         1       2       3       4       5       6       7

  44  CALCULATE   MPscore[1,2] = MPLRV[1]$[*; 1,2]
  45  FRAME       3; SCALING=xyequal
  46  XAXIS       3; TITLE='Dimension 1'; LOWER=-0.55; UPPER=0.55
  47  YAXIS       3; TITLE='Dimension 2'; LOWER=-0.55; UPPER=0.55
  48  PEN         1; SYMBOLS=0; LABELS=Points; SIZE=1.5; COLOUR='blue'
  49  DGRAPH   [TITLE='Multiple Procrustes analysis: first two dimensions';\
  50              WINDOW=3; KEY=0] MPscore[2]; MPscore[1]
  51  PRINT    !T('The 7 methods are plotted as the points:', \
  52              '   Jc  Jaccard similarity coefficient;', \
  53              '   SM  simple-matching similarity coefficient;', \
  54              '   CB  city-block similarity coefficient;', \
  55              '   Ec  ecological similarity coefficient;', \
  56              '   Py  Pythagorean similarity coefficient;', \
  57              '   CA  correspondence analysis of data;', \
  58              '   CP  correspondence analysis of presence/absence.'); \
  59           JUSTIFICATION=left

The 7 methods are plotted as the points:
   Jc  Jaccard similarity coefficient;
   SM  simple-matching similarity coefficient;
   CB  city-block similarity coefficient;
   Ec  ecological similarity coefficient;
   Py  Pythagorean similarity coefficient;
   CA  correspondence analysis of data;
   CP  correspondence analysis of presence/absence.
```

6.19 Hierarchical cluster analysis

Hierarchical cluster analysis operates on a similarity matrix and aims to arrange the *n* sampling units into homogeneous groups. Methods of constructing similarity matrices in Genstat are described in Sections 6.1.2 - 6.1.4. The HCLUSTER directive offers several possibilities. The general strategy is best appreciated in geometrical terms, with the *n* sampling units represented by points in a multidimensional space. In *agglomerative* methods, these points initially represent *n* separate clusters, each containing one member. At each of $n-1$ stages, two clusters are fused into one bigger cluster, until at the final stage all units are fused into a single cluster: this process can be represented by a hierarchical tree whose nodes indicate what fusions have occurred. The methods fuse the two closest clusters and vary in how *closest* is defined. In *single-linkage* cluster analysis, *closest* is defined as the smallest distance between any two samples from different clusters; in *centroid* clustering it is the smallest distance between cluster centroids; and so on (see Gordon 1981 for a full discussion). All these methods are in the hierarchical cluster analysis menus in Genstat *for Windows*.

Genstat can display the tree fitted to a given similarity matrix, and provides a scale to show the level of similarity at which the fusions have occurred; scaled tree like this is termed a *dendrogram*. The endpoints of the dendrogram correspond to the units in some permuted order; you can save this order, for example to use with FSIMILARITY (6.1.2). Of course, a hierarchical tree does not by itself provide a classification. This can be derived by cutting the dendrogram at some arbitrary level of similarity; each cluster then consists of those samples occurring on the same detached branch of the dendrogram. A factor can be formed to indicate cluster membership, and you can calculate indexes to assess the similarity between factors obtained from different cluster analyses (6.19.7).

To assess the reliability of the clusters, you can perform a bootstrap analysis using the HBOOTSTRAP procedure (6.19.8). For each bootstrap sample, a set of vectors is formed by sampling with replacement from the variates and factors used to form the similarity matrix for the original cluster analysis. HBOOTSTRAP does a cluster analysis using each bootstrap sample, and counts the number of times each cluster occurs in the analyses. These numbers can be printed, or plotted alongside the clusters in the dendrogram from the original analysis.

6.19.1 The HCLUSTER directive

HCLUSTER directive

Performs hierarchical cluster analysis.

Options

PRINT = *string tokens*	Printed output required (dendrogram, amalgamations); default * i.e. no printing
METHOD = *string token*	Criterion for forming clusters (singlelink, nearestneighbour, completelink, furthestneighbour, averagelink, mediansort, groupaverage); default sing
CTHRESHOLD = *scalar*	Clustering threshold at which to print formation of clusters; default * i.e. determined automatically

Parameters

SIMILARITY = *symmetric matrices*	Input similarity matrix for each cluster analysis
GTHRESHOLD = *scalars*	Grouping threshold where groups are formed from the dendrogram
GROUPS = *factors*	Stores the groups formed
PERMUTATION = *variates*	Permutation order of the units on the dendrogram
AMALGAMATIONS = *matrices*	To store linked list of amalgamations

The input for HCLUSTER is provided by the SIMILARITY parameter, as a list of symmetric matrices, one for each analysis. These matrices can be formed by FSIMILARITY (6.1.2), HREDUCE (6.1.3) or CALCULATE (6.1.4). Missing values are allowed in the similarity matrix only with the single-linkage method.

The GTHRESHOLD and GROUPS parameters must be either both present or both absent. When you are deriving a classification, the level of similarity at which the dendrogram is to be cut is specified by the scalar value in the GTHRESHOLD parameter. The level is given as a percentage similarity. The resulting cluster membership is saved in a factor, whose identifier is specified by the GROUPS parameter. The factor will be declared implicitly, if necessary, and it will have its number of levels set to the number of clusters formed and its number of values taken from the number of rows of the corresponding symmetric matrix.

The PERMUTATION parameter allows you to specify a variate to save the order in which the units appear on the printed dendrogram. Genstat will define it to be a variate automatically, if necessary, with number of values is taken from the number of rows of the corresponding similarity matrix. Conventionally, the first unit on the dendrogram is unit 1 and so the first value of the variate of permutations will be 1.

The AMALGAMATIONS parameter can specify a matrix to store information about the order in which the units form groups, and at what level of similarity. At any stage in the process of agglomeration, each group is represented by the unit with the smallest unit number: for example, a group containing units 2, 5, 17 and 22 is represented by unit 2. This means that the final merge is always between a group indexed by unit 1 and a group indexed by another unit. Since there are $n-1$ stages of agglomeration, the matrix will have a number of rows one less than the number of rows of the input similarity matrix. Each row represents a joining of two groups and consists of three values. The first two values are the numbers indexing the two groups that are joining, and the third value is the level of similarity. So the matrix has three columns. The matrix will be declared implicitly, if necessary.

HCLUSTER can print two pieces of information. The first gives details of each amalgamation, followed by a list of clusters that are formed at decreasing levels of similarity. The second is the

dendrogram. The PRINT option allows you to control which of these are printed. If METHOD=singlelink and the PRINT setting includes amalgamations, the minimum spanning tree (6.19.2) will be printed instead of the stages at which the clusters merge. This is because information from forming the minimum spanning tree is used to form the single linkage clustering.

Alternatively, if you save the AMALGAMATIONS matrix, you can use procedure DDENDROGRAM (6.19.5) to display the dendrogram using high-resolution graphics, as shown in Example 6.19.5. Also the HFCLUSTERS procedure can be used to obtain the full set of clusters constructed during the cluster analysis, and the similarity values at which they were formed; see Example 6.19.8.

The METHOD option has seven possible settings; these determine how the similarities amongst clusters are redefined after each merge. The default singlelink, which has synonym nearestneighbour, gives single linkage. The setting completelink (synonym furthestneighbour) defines the distance between two clusters as the maximum distance between any two units in those clusters. The setting averagelink defines the similarity between a cluster and two merged clusters as the average of the similarities of the cluster with each of the two. For groupaverage, an average is taken over all the units in the two merged clusters. Median sorting (Gower 1967) is best thought of in terms of clusters being represented by points in a multidimensional space; when two clusters join, the new cluster is represented by the midpoint of the original cluster points.

The CTHRESHOLD option is a scalar which allows you to define the levels of decreasing similarity at which the lists of clusters are printed with their membership. The decreasing levels of similarity are formed by repeatedly subtracting the CTHRESHOLD value from the maximum similarity of 100%. For example, setting CTHRESHOLD=10 will list the clusters formed at 90% similarity, 80%, and so on. At each level, those units that have not joined any group are also listed. If you do not set this option, the default value will be calculated from the range of similarities at which merges occur, to give between 10 and 20 separate levels.

Example 6.19.1 uses the average linkage method to cluster the cars discussed in Section 6.1.2. The amalgamations matrix is saved, in matrix Caramalg, so that we can plot the dendrogram later on, in Example 6.19.5.

Example 6.19.1

```
  2  UNITS    [NVALUES=16]
  3  VARIATE Engcc,Ncyl,Tankl,Weight,Length,Width,Height,Wbase,Tspeed,Stst,\
  4          Carb,Drive,Vct[1...3]
  5  POINTER Cd; VALUES=!P(Engcc,Ncyl,Tankl,Weight,Length, \
  6          Width,Height,Wbase,Tspeed,Stst)
  7  READ     [PRINT=errors] #Cd,Carb,Drive
 24  TEXT     [VALUES=Estate,'Arna1.5','Alfa2.5',Mondialqc,\
 25          Testarossa,Croma,Panda,Regatta,Regattad,Uno,\
 26          X19,Contach,Delta,Thema,Y10,Spider] Carname
 27  FACTOR   [NVALUES=Carname; LEVELS=16] Fcar; VALUES=!(1...16)
 28  SYMMETRICMATRIX [ROWS=Carname] Carsim
 29  " Form similarity matrix between cars."
 30  FSIMILARITY [SIMILARITY=Carsim; PRINT=*] #Cd,Carb,Drive; \
 31           TEST=4(cityblock),4(Euclidean),2(cityblock),2(simplematch)
 32  HCLUSTER [PRINT=dendrogram; METHOD=averagelink] Carsim; \
 33           GTHRESHOLD=70; GROUPS=Cargrp; PERMUTATION=Carperm; \
 34           AMALGAMATIONS=Caramalg

Average linkage cluster analysis
================================

Dendrogram
----------

  ** Levels   100.0  90.0  80.0  70.0  60.0  50.0

Estate         1  ..
Regatta        8  ..)
```

```
Arna1.5        2  ..)
Delta         13  ..).....
Panda          7  ..      )
Uno           10  ..)..   )
Y10           15  .....)..).....
Regattad       9  ..............)..
Alfa2.5        3  ...........      )
X19           11  ........   )     )
Spider        16  ........)..).....)..
Mondialqc      4  ..............      )
Croma          6  ..            )     )
Thema         14  ..)...........).....).........
Testarossa     5  ........                       )
Contach       12  ........).....................).....
```

```
  35  FSIMILARITY [PRINT=similarities; SIMILARITY=Carsim; \
  36              PERMUTATION=Carperm; STYLE=abbreviated]

Abbreviated similarity matrix: Carsim
-------------------------------------

Estate      -
Regatta     9-
Arna1.5     99-
Delta       999-
Panda       8888-
Uno         88989-
Y10         889899-
Regattad    8888787-
Alfa2.5     88786666-
X19         888877878-
Spider      7777667788-
Mondialqc   55552435757-
Croma       787856677677-
Thema       7878566775779-
Testarossa  33331213545855-
Contach     444423336557448-
```

6.19.2 Displaying and saving information from a cluster analysis: the HDISPLAY directive

HDISPLAY directive

Displays results ancillary to hierarchical cluster analyses: matrix of mean similarities between and within groups, a set of nearest neighbours for each unit, a minimum spanning tree, and the most typical elements from each group.

Option

PRINT = *string tokens*	Printed output required (`neighbours, tree, typicalelements, gsimilarities`); default `tree`

Parameters

SIMILARITY = *symmetric matrices*	Input similarity matrix for each cluster analysis
NNEIGHBOURS = *scalars*	Number of nearest neighbours to be printed
NEIGHBOURS = *matrices*	Matrix to store nearest neighbours of each unit
GROUPS = *factors*	Indicates the groupings of the units (for calculating typical elements and mean similarities between groups)
TREE = *matrices*	To store the minimum spanning tree (as a series of links and corresponding lengths)
GSIMILARITY = *symmetric matrices*	To store similarities between groups

The HDISPLAY directive prints ancillary information useful for interpreting cluster analyses, or can save information to use elsewhere in Genstat, for example for plotting.

The SIMILARITY parameter specifies a list of symmetric similarity matrices. These are operated on, in turn, to produce the output requested by the PRINT option and to save the information specified by other parameters. Since the interpretations of the remaining parameters are closely linked to the different settings of the PRINT option, each setting is discussed below with the relevant parameters.

The NNEIGHBOURS parameter gives a list of scalars indicating how many neighbours will appear in the printed table of nearest neighbours.

The NEIGHBOURS parameter can specify a list of identifiers to store details of nearest neighbours. These will be declared implicitly, if necessary, as matrices. The rows of the matrices correspond to the units; there should be an even number of columns. The values in the odd-numbered columns represent the neighbouring units in order of their similarity, while the values in the even-numbered columns are the corresponding similarities. If you have declared the matrix previously and it does not have enough columns, then NEIGHBOURS stores as many neighbours as possible. If there is an odd number of columns in the matrix, the last column is not filled. If the matrix is declared implicitly, the number of columns will be twice the value of the NNEIGHBOURS scalar.

If the PRINT option includes the setting neighbours, Genstat prints a table of nearest neighbours for every sample, together with their values of similarity. The number of neighbours printed is determined by the value of the NNEIGHBOURS scalar; if NNEIGHBOURS is not set, the table is not printed. This information is also useful for interpreting clusters and ordinations. In Example 6.19.2a, the table is printed for three nearest neighbours, and the matrix Carneig is given values corresponding to the first two nearest neighbours.

Example 6.19.2a

```
  37  MATRIX [ROWS=Carname; COLUMNS=4] Carneig
  38  HDISPLAY [PRINT=neighbours] Carsim; NNEIGHBOURS=3; NEIGHBOURS=Carneig

Neighbours table derived from Carsim
====================================

Estate         1       8  98.1       2  97.6      13  95.9
Arna1.5        2       1  97.6       8  96.9      13  95.1
Alfa2.5        3      13  83.7      11  82.8       8  82.3
Mondialqc      4       5  82.7      14  77.4       6  76.8
Testarossa     5      12  88.5       4  82.7      16  58.3
Croma          6      14  98.7       8  82.0      13  81.4
Panda          7      10  96.0      15  92.9       2  85.5
Regatta        8       1  98.1       2  96.9      13  95.9
Regattad       9       8  84.4       1  83.9       2  82.2
Uno           10       7  96.0      15  92.5       2  90.9
X19           11       1  87.0      16  86.0       2  85.8
Contach       12       5  88.5       4  70.9       3  61.8
Delta         13       8  95.9       1  95.9       2  95.1
Thema         14       6  98.7       8  81.1      13  80.2
Y10           15       7  92.9      10  92.5       2  92.4
Spider        16      11  86.0       3  82.3      13  78.8

  39  PRINT Carneig

                Carneig
                      1           2           3           4
    Carname
     Estate       8.000       0.981       2.000       0.976
    Arna1.5       1.000       0.976       8.000       0.969
    Alfa2.5      13.000       0.837      11.000       0.828
  Mondialqc       5.000       0.827      14.000       0.774
 Testarossa      12.000       0.885       4.000       0.827
      Croma      14.000       0.987       8.000       0.820
      Panda      10.000       0.960      15.000       0.929
```

```
     Regatta       1.000       0.981       2.000       0.969
    Regattad       8.000       0.844       1.000       0.839
         Uno       7.000       0.960      15.000       0.925
         X19       1.000       0.870      16.000       0.860
     Contach       5.000       0.885       4.000       0.709
       Delta       8.000       0.959       1.000       0.959
       Thema       6.000       0.987       8.000       0.811
         Y10       7.000       0.929      10.000       0.925
      Spider      11.000       0.860       3.000       0.823
```

The GROUPS parameter specifies a factor to divide the units of each similarity matrix into clusters. You may have formed the factor from a previous hierarchical cluster analysis (6.19.1). This parameter must be set if the PRINT option includes the settings typicalelement or gsimilarities.

If the PRINT option includes the setting typicalelement, Genstat prints the average similarity of each group member with the other group members. This is to help you identify typical members of each group: typical members will have relatively large average similarities compared to those of the other members. Within each group, members are printed in decreasing order of average similarity. In Example 6.19.2b, the cars are listed in order of their mean similarity with the other cars of the same make.

Example 6.19.2b

```
  40  FACTOR [LABELS=!t(Fiat,'Alfa Romeo',Lancia,Ferrari,Lamborghini,\
  41    Pinninfarina)] Maker; VALUES=!(2,2,2,4,4,1,1,1,1,1,1,5,3,3,3,6)
  42  HDISPLAY [PRINT=typical] Carsim; GROUPS=Maker

Most typical members
====================

Similarity matrix: Carsim

Fiat
Regatta         8    83.3
Uno            10    81.6
Regattad        9    77.4
Panda           7    76.4
X19            11    73.7
Croma           6    67.5

Alfa Romeo
Estate          1    89.5
Arna1.5         2    88.8
Alfa2.5         3    80.7

Lancia
Delta          13    84.3
Y10            15    74.4
Thema          14    70.4

Ferrari
Testarossa      5    82.7
Mondialqc       4    82.7

Lamborghini
Contach        12   100.0

Pinninfarina
Spider         16   100.0
```

The GSIMILARITY parameter specifies a list of symmetric matrices in which you can save the mean between-group and within-group similarities. Any structure that you have not declared already will be declared implicitly to be a symmetric matrix with number of rows equal to the

number of levels of the factor in the GROUPS parameter.

If the PRINT option includes the setting gsimilarities, Genstat prints the mean similarities between-groups and within-groups. Self-similarities are excluded. Example 6.19.2c forms the group similarity matrix based on the groups in the factor Maker, prints the matrix and saves the values in the symmetric matrix Cargsim.

Example 6.19.2c

```
  43  HDISPLAY [PRINT=gsimilarities] Carsim; GROUPS=Maker; \
  44    GSIMILARITY=Cargsim

Mean similarities between and within groups
===========================================

Similarity matrix: Carsim
Between and within groups similarity matrix: Cargsim


Fiat         1    76.6
Alfa Romeo   2    82.1  86.4
Lancia       3    79.5  84.0  76.4
Ferrari      4    43.3  53.1  48.2  82.7
Lamborghini  5    37.6  50.4  40.5  79.7  ----
Pinninfarina 6    73.7  78.7  75.5  66.5  50.6  ----
                     1     2     3     4     5     6

  45  PRINT Cargsim

                 Cargsim

        Fiat      0.7665
   Alfa Romeo     0.8209      0.8635
       Lancia     0.7952      0.8401      0.7636
      Ferrari     0.4328      0.5313      0.4817      0.8266
  Lamborghini     0.3760      0.5036      0.4054      0.7971      1.0000
 Pinninfarina     0.7369      0.7873      0.7547      0.6647      0.5059
                    Fiat  Alfa Romeo      Lancia     Ferrari Lamborghini

 Pinninfarina     1.0000

             Pinninfarina
```

The TREE parameter can specify a matrix to save the minimum spanning tree. The matrix is set up with two columns and number of rows equal to the number of units. For each unit, the value in the first column is the unit to which that unit is linked on its left; the second column is the corresponding similarity. The first unit is not linked to any unit on its left, as it is always the first unit on the tree; so the first row of the matrix contains missing values. The HFAMALGAMATIONS procedure can use the tree to form an amalgamations matrix, representing how the clusters would be formed with this similarity matrix by single-linkage cluster analysis.

Setting the PRINT option to tree prints the minimum spanning tree associated with the similarity matrix specified the SIMILARITY parameter. The minimum spanning tree (MST) is not a Genstat structure, but it can be kept in the form described above: that is, in a matrix with two columns. An MST is a tree connecting the *n* points of a multidimensional representation of the sampling units. In a tree every unit is linked to a connected network and there are no closed loops; the special feature of the MST is that, of all trees with a sampling unit at every node, it is the one whose links have minimum total length. The links include all those that join nearest neighbours; the MST is closely related to single linkage hierarchical trees (6.19.1). Minimum spanning trees are also useful if you superimpose them on ordinations (6.10) to reveal regions in which distance is badly distorted; if neighbouring points, as given by the MST, are distant in the ordination then something is badly wrong (see Gower & Ross 1969). Plots like this can be produced by procedure DMST (which uses HDISPLAY internally to form the MST); see Section

6.19.6. In Example 6.19.2d, the MST is printed and then saved in the structure `Cartree` which has been declared implicitly as a matrix.

Example 6.19.2d

```
  46  HDISPLAY [PRINT=tree] Carsim; TREE=Cartree

Minimum spanning tree
=====================

Similarity matrix: Carsim

  Estate  Arna1.5      Y10    Panda      Uno
       1......  2...... 15......  7...... 10
       (  97.6     92.4     92.9     96.0
       (
       (  Regatta    Croma    Thema Mondialq Testaros  Contach
       (......  8......  6...... 14......  4......  5...... 12
       (  98.1  (  82.0     98.7     77.4     82.7     88.5
       (        (
       (        ( Regattad
       (        (......  9
       (        (  84.4
       (        (
       (        (    Delta  Alfa2.5
       (        (...... 13......  3
       (           95.9     83.7
       (
       (      X19   Spider
       (...... 11...... 16
          87.0     86.0

Total length:   1343.4

  47  PRINT Cartree

                  Cartree
                        1           2
      Carname
       Estate           *           *
      Arna1.5       1.000       0.976
      Alfa2.5      13.000       0.837
    Mondialqc      14.000       0.774
   Testarossa       4.000       0.827
        Croma       8.000       0.820
        Panda      15.000       0.929
      Regatta       1.000       0.981
     Regattad       8.000       0.844
          Uno       7.000       0.960
          X19       1.000       0.870
      Contach       5.000       0.885
        Delta       8.000       0.959
        Thema       6.000       0.987
          Y10       2.000       0.924
       Spider      11.000       0.860
```

6.19.3 Examining the data by groups: the `HLIST` directive

`HLIST` lists the values of the data matrix in a condensed form, either in their original order or, more usefully, in the order determined by a cluster analysis (6.19.1). This representation can be very helpful for revealing patterns in the data, associated with clusters, or for an initial scan of the data to pick out interesting features of the variables.

HLIST directive

Lists the data matrix in abbreviated form.

Options

GROUPS = *factor*	Defines groupings of the units; used to split the printed table at appropriate places and to label the groups; default *
UNITS = *text* or *variate*	Names for the rows (i.e. units) of the table; default *

Parameters

DATA = *variates* or *factors*	The data variables
TEST = *string tokens*	Test type, defining how each variable is treated in the calculation of the similarity between each unit (simplematching, jaccard, russellrao, dice, antidice, sneathsokal, rogerstanimoto, cityblock, manhattan, ecological, euclidean, pythagorean, minkowski, divergence, canberra, braycurtis, soergel); default * ignores that variable
RANGE = *scalars*	Range of possible values of each variable; if omitted, the observed range is taken

The DATA parameter specifies a list of variates or factors, all of which must be of the same length. If any of them is restricted, then that restriction is applied to all of them. Any restriction on any other variate or factor must be to the same set of units.

The TEST parameter specifies a list of strings, one for each variate or factor in the DATA parameter list, to define its "type". This is similar to the TEST parameter used in FSIMILARITY (6.1.2) to determine how differences in variate or factor values for each unit contribute to the overall similarity between units. However, HLIST distinguishes only between qualitative variables (factors or variates with settings simplematching - rogerstanimoto) and quantitative variables (variates with other settings). The values of qualitative variates are printed directly. If the range of a quantitative variate is greater than 10, the printed values are scaled to lie in the range 0 to 10. This scaling is done by subtracting the minimum value, dividing by the range and then multiplying by 10. If the range is less than 10, the values are printed unscaled; so quantitative variates with values that are all less than 1 will appear as 0 in the abbreviated table. The values are printed with no decimal places, and in a field-width of 3.

The RANGE parameter contains a list of scalars, one for each variable in the DATA list. This allows you to check that the values of each variable lie within the given range. The range is also used to standardize quantitative variates, so that you can impose a standard range for example when variates are measured on commensurate scales. You can omit the RANGE parameter for all or any of the variables by giving a missing identifier or a scalar with a missing value; Genstat then uses the observed range.

The UNITS option allows you to change the labelling of the units in the table; you can specify a text or a pointer or a variate.

You can use the GROUPS option to specify a factor that will split the units into groups. The table from HLIST is then divided into sections corresponding to the groups. If the factor has labels, these are used to annotate the sections; otherwise a group number is used.

In Example 6.19.3a, you can see the effect of scaling the quantitative variables, and not scaling the qualitative variables.

Example 6.19.3a

```
  48  HLIST [UNITS=Carname] #Cd,Carb,Drive; \
  49    TEST=4(cityblock),4(Euclidean),2(cityblock),2(simplematch)

Key to condensed data matrix
============================

      Variate   Minimum     Range      Test type
  1     Engcc      965.     4202.      City block      (3)
  2      Ncyl     4.000     8.000      City block      (3)
  3     Tankl     35.00     85.00      City block      (3)
  4    Weight     720.0     786.0      City block      (3)
  5    Length     338.0     121.0      Euclidean       (5)
  6     Width    149.00     51.00      Euclidean       (5)
  7    Height    107.00     39.00      Euclidean       (5)
  8     Wbase    216.00     50.00      Euclidean       (5)
  9    Tspeed     134.0     157.0      City block      (3)
 10      Stst      4.90     14.00      City block      (3)
 11      Carb     1.000     2.000      Simple Matching (1)
 12     Drive     1.000     1.000      Simple Matching (1)


Variates listed in condensed form
=================================

          Variate    1   2   3   4   5   6   7   8   9  10  11  12
          Test       3   3   3   3   5   5   5   5   3   3   1   1
          Range     10   8  10  10  10  10  10  10  10  10   2   1


Estate         1     1   0   1   3   6   2   6   5   2   4   0   1
Arna1.5        2     1   0   1   1   5   2   8   5   2   3   0   1
Alfa2.5        3     3   2   1   5   7   2   8   7   4   2   0   0
Mondialqc      4     5   4   6   9   9   5   4   9   7   1   1   0
Testarossa     5     9   8  10  10   9   9   1   7  10   0   1   0
Croma          6     2   0   4   5   9   5   9  10   4   2   1   1
Panda          7     0   0   0   0   0   0  10   0   0   8   0   1
Regatta        8     1   0   2   3   7   3   8   5   2   3   0   1
Regattad       9     1   0   2   3   7   3   8   5   1  10   2   1
Uno           10     0   0   0   0   2   1   9   4   0   8   0   1
X19           11     1   0   1   2   4   1   2   0   2   4   0   0
Contach       12    10   8  10   9   6  10   0   5   9   0   0   0
Delta         13     1   0   1   3   4   2   7   6   3   2   0   1
Thema         14     2   0   4   5  10   5   9  10   5   1   1   1
Y10           15     0   0   1   0   0   0   9   0   2   4   0   1
Spider        16     2   0   1   4   6   2   4   2   3   2   1   0
```

The UNITS option allows you to change the labelling of the units in the table, as shown in Example 6.19.3a. You can specify a text or a pointer or a variate.

You can use the GROUPS option to specify a factor that will split the units into groups. The table from HLIST is then divided into sections corresponding to the groups. If the factor has labels, these are used to annotate the sections; otherwise a group number is used.

Example 6.19.3b

```
  50  HLIST [GROUPS=Maker; UNITS=Carname] #Cd,Carb,Drive; \
  51    TEST=4(cityblock),4(Euclidean),2(cityblock),2(simplematch)

Key to condensed data matrix
============================

      Variate   Minimum     Range      Test type
  1     Engcc      965.     4202.      City block      (3)
  2      Ncyl     4.000     8.000      City block      (3)
  3     Tankl     35.00     85.00      City block      (3)
  4    Weight     720.0     786.0      City block      (3)
```

```
  5     Length     338.0     121.0      Euclidean       (5)
  6      Width    149.00     51.00      Euclidean       (5)
  7     Height    107.00     39.00      Euclidean       (5)
  8      Wbase    216.00     50.00      Euclidean       (5)
  9     Tspeed     134.0     157.0      City block      (3)
 10       Stst      4.90     14.00      City block      (3)
 11       Carb     1.000     2.000      Simple Matching (1)
 12      Drive     1.000     1.000      Simple Matching (1)


Variates listed in condensed form, grouped by Maker
===================================================

          Variate    1   2   3   4   5   6   7   8   9  10  11  12
          Test       3   3   3   3   5   5   5   5   3   3   1   1
          Range     10   8  10  10  10  10  10  10  10  10   2   1

Fiat
Croma          6     2   0   4   5   9   5   9  10   4   2   1   1
Panda          7     0   0   0   0   0   0  10   0   0   8   0   1
Regatta        8     1   0   2   3   7   3   8   5   2   3   0   1
Regattad       9     1   0   2   3   7   3   8   5   1  10   2   1
Uno           10     0   0   0   0   2   1   9   4   0   8   0   1
X19           11     1   0   1   2   4   1   2   0   2   4   0   0

Alfa Romeo
Estate         1     1   0   1   3   6   2   6   5   2   4   0   1
Arna1.5        2     1   0   1   1   5   2   8   5   2   3   0   1
Alfa2.5        3     3   2   1   5   7   2   8   7   4   2   0   0

Lancia
Delta         13     1   0   1   3   4   2   7   6   3   2   0   1
Thema         14     2   0   4   5  10   5   9  10   5   1   1   1
Y10           15     0   0   1   0   0   0   9   0   2   4   0   1

Ferrari
Mondialqc      4     5   4   6   9   9   5   4   9   7   1   1   0
Testarossa     5     9   8  10  10   9   9   1   7  10   0   1   0

Lamborghini
Contach       12    10   8  10   9   6  10   0   5   9   0   0   0

Pinninfarina
Spider        16     2   0   1   4   6   2   4   2   3   2   1   0
```

6.19.4 Relating groups to the original data variables: the HSUMMARIZE directive

The HSUMMARIZE directive helps you to see which clusters, if any, are distinguished by each variable. It requires a factor to define the clusters, as well as the original data variables (variates or factors), together with their types and, optionally, their ranges. From this it prints a frequency table for each variable, classified by the grouping factor and the different values of the variable concerned.

For qualitative variables (variates or factors with TEST settings simplematching - rogerstanimoto) the values are integral, and for each group Genstat calculates an interaction statistic labelled chi-square. This statistic does not have a significance level attached to it, but it does draw attention to groups for which the distribution is markedly different from the overall distribution.

For quantitative variables (i.e. variates with other settings) values are rounded to the nearest point on an 11-point scale (0-10). The interaction statistic is analogous to Student's t, and it draws attention to the groups for which the mean value is markedly different from the overall mean (again with no significance level attached). Missing values are ignored in the computation of these statistics.

HSUMMARIZE directive

Forms and prints a group by levels table for each test together with appropriate summary statistics for each group.

Option

GROUPS = *factor*	Factor defining the groups; no default i.e. this option must be specified

Parameters

DATA = *variates* or *factors*	The data variable
TEST = *string tokens*	Test type, defining how each variable is treated in the calculation of the similarity between each unit (simplematching, jaccard, russellrao, dice, antidice, sneathsokal, rogerstanimoto, cityblock, manhattan, ecological, euclidean, pythagorean, minkowski, divergence, canberra, braycurtis, soergel); default * ignores that variable
RANGE = *scalars*	Range of possible values of each variable; if omitted, the observed range is taken

The parameters of the HSUMMARIZE directive are the same as those of the HLIST directive; see Section 6.19.3.

As the output from this directive can be very long, only two tables are shown in Example 6.19.4; these illustrate the difference between tables for qualitative and quantitative variables. The grouping factor is taken from the HCLUSTER example in 6.19.1. Each entry in the table gives the number of units from a particular group that have a particular value of the variable.

Example 6.19.4

```
  52  HSUMMARIZE [GROUPS=Cargrp] Weight,Carb; \
  53    TEST=cityblock,simplematch

Grouped data frequency tables for each variate
==============================================

  Variate: Weight
  Minimum: 720.0         Range: 786.0     Test type: City block
  Data scaled by factor of 0.01272

Cargrp     *   0   1   2   3   4   5   6   7   8   9  10

 1         0   3   1   1   4   1   1   0   0   0   0   0
 2         0   0   0   0   0   0   2   0   0   0   1   0
 3         0   0   0   0   0   0   0   0   0   0   1   1

   Total   0   3   1   1   4   1   3   0   0   0   2   1


Cargrp    Total    Mean       t

 1           11    2.18   -1.75
 2            3    6.33    1.33
 3            2    9.50    2.48

   Total     16    3.88


  Variate: Carb    Test type: Simple Matching
```

```
Cargrp      *    0    1    2  Total    Chi-sq

 1          0    9    1    1     11    2.53
 2          0    0    3    0      3    6.60
 3          0    1    1    0      2    0.40

   Total    0   10    5    1     16
```

6.19.5 Plotting the dendrogram: the DDENDROGRAM procedure

DDENDROGRAM procedure

Draws dendrograms with control over structure and style (P.G.N. Digby).

Options

STYLE = *string token*	Style to use for the links of the dendrogram (average, centroid, lower, full); default aver
ORDERING = *string tokens*	How to define the order of the units for the dendrogram (given, ziggurat, size, first); default zigg, size, firs
REVERSE = *string token*	Whether to reverse the order of the units in the dendrogram (no, yes); default no
ORIENTATION = *string token*	Specifies the orientation of a dendrogram produced by high-resolution graphics (north, south, east, west); default west
METHOD = *string token*	Method used to represent the scale on which the amalgamations have been made: settings other than the default are relevant only for data not generated by HCLUSTER or HDISPLAY (similarities, percentages, distances); default simi
SCREEN = *string token*	Setting to use for the SCREEN option of DGRAPH (clear, keep); default clea
CHANGE = *string token*	If a dendrogram-save structure from a previous DDENDROGRAM is used as the DATA parameter then this option specifies the area of the process where the first changes occur: see the description of the SAVE parameter (order, dendrogram, display); default orde
GRAPHICS = *string token*	Form of graphics to be used (lineprinter, highresolution); default high
DSIMILARITY = *string token*	Whether to display an axis for the similarities in high-resolution graphics (no, yes); default no
LOWSIMILARITY = *scalar*	Lower value to be used for the axis showing the similarities; default * i.e. determined from the data
NPAGES = *scalar*	Number of pages to use for a high-resolution plot; default 1
PAGEINFORMATION = *string tokens*	Controls what to include in a multi-page plot (similarity, title, pagenumber); default simi, titl, page
ENDACTION = *string token*	Action to be taken after completing the plot (continue, pause); default * uses the current setting

Parameters

DATA = *matrices* or *pointers*	Data defining each dendrogram in the form of either a matrix saved using the AMALGAMATIONS parameter of HCLUSTER (methods other than single linkage), or a matrix from the TREE parameter of HDISPLAY, or a SAVE structure from a previous use of DDENDROGRAM
PERMUTATION = *variates*	Specify or save permutations of the units for drawing each dendrogram, according to ORDERING option
LABELS = *variates* or *texts*	Supply labels to use for the units of each dendrogram; these should be in the natural order of the units, not in a permuted order
TITLE = *texts*	Titles for the dendrograms
WINDOW = *scalars*	Window to use for each dendrogram (window 1 if unset); if this is set to zero the dendrogram is not drawn, but results can still be saved using the PERMUTATION, ZIGGURAT and SAVE parameters
PENS = *scalars*, *variates*, *string* or *texts*	Scalar or string specifying the graphics pen or symbol in which to draw each (high-resolution or line-printer) dendrogram; alternatively use of a variate or text allows the structure of each dendrogram to be highlighted by drawing different links with different graphics pens or symbols
ZIGGURAT = *variates*	Save the "ziggurat-degree" of the links in each dendrogram
SAVE = *pointers*	Save the information required to plot a dendrogram, for use as input for the DATA parameter in a subsequent call to DDENDROGRAM

DDENDROGRAM draws dendrograms using line-printer or high-resolution graphics, as indicated by the GRAPHICS option. Figure 6.19.5a shows an example, which reproduces (as a high-resolution plot) the dendrogram in Example 6.19.1.

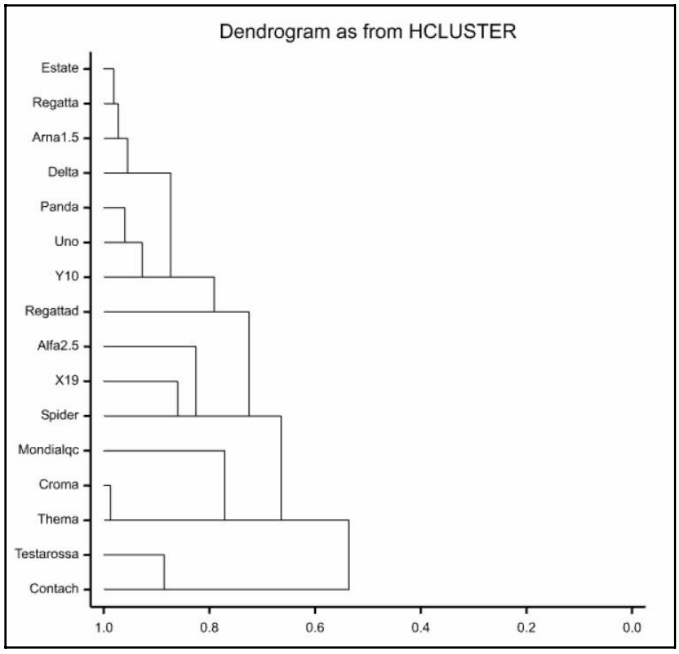


Figure 6.19.5a

Dendrograms can be drawn in many ways, often with apparently quite different results, as illustrated by Digby (1985). DDENDROGRAM provides considerable control over the way in which the dendrogram is formed; in particular allowing the order of the units and the style used for drawing the links of the dendrogram to be varied.

The information defining the dendrogram is given by the DATA parameter. This should be a matrix containing the amalgamations information from hierarchical cluster analysis (from the AMALGAMATIONS parameter of HCLUSTER; 6.19.1) or a matrix containing the minimum spanning

tree information (from the TREE parameter of HDISPLAY 6.19.2); alternatively a SAVE structure from a previous DDENDROGRAM can be used as input. However, the amalgamations matrix from HCLUSTER is unusable if the clustering has been produced by single linkage, so the minimum spanning tree information (which is equivalent) should be used as input instead.

The PERMUTATION parameter can be supplied with a variate, either to specify a permutation of the rows of the dendrogram or to save the permutation generated by DDENDROGRAM, as indicated by the ORDERING option. Setting ORDERING=given takes the ordering defined by the PERMUTATION variate. The other settings of ORDERING define partial orderings of the units, and are used in conjunction with each other to obtain the full ordering: ziggurat (Critchley 1983) is associated with ultrametric distances amongst the units; size specifies that when 2 groups merge the smaller is always placed before the larger in the order; first specifies that when 2 groups merge the group containing the lowest numbered unit is always placed before the other in the order. The orders given by settings ziggurat and size are not completely specified and recourse may be made to the other of these settings or to first. If ORDERING is not set to given, a list of settings may be specified; then the first in the list is used, the second is used to satisfy indeterminacies in the order given by the first setting in the list, and so on. The default is the list of settings: ziggurat, size, first. Option REVERSE allows the ordering thus obtained to be reversed.

The LABELS parameter can be given a variate or a text to supply labels for the rows of the dendrogram. Labelling can be suppressed altogether by using a text containing only spaces.

The STYLE option controls the style to use in forming the links of the dendrogram: its setting indicates where the line representing each new cluster should be placed. Assuming that the dendrogram has the units on the left-hand side, the settings can be described as follows: average (the default) the new line is midway between the old lines; centroid the new line is placed at the mid-point of all the units in the group it represents; lower the new line is a continuation of the lower of the two old lines (comparable with dendrograms from HCLUSTER); full the new line is a continuation of the upper or lower of the two old lines, so that each vertical line spans all the units in the group it represents.

The ORIENTATION option is relevant to high-resolution graphics, when it controls the orientation of the dendrogram: for example the setting north results in a "hanging dendrogram" with the units across the top. The default setting is west, which gives a dendrogram with the units on the left-hand side; this is also how DDENDROGRAM draws dendrograms on the line-printer.

The METHOD option indicates the scale on which the amalgamations have been made. This option need be set only if the data have been obtained from a source other than HCLUSTER or HDISPLAY.

The TITLE parameter specifies a title for each dendrogram. For high-resolution graphics, the WINDOW parameter defines the graphics window to use for each plot. With line-printer graphics, two "windows" are available: window 1 has a width of 101 characters, window 2 a width of 61 characters. If WINDOW is not set, window 1 is used. If it is set to zero, the dendrogram is not drawn but results can still be saved using the PERMUTATION, ZIGGURAT and SAVE parameters; however, if the SAVE structure is used later as input to DDENDROGRAM, the CHANGE option must not be set to display as the dendrogram stage will not have been completed.

The LOWSIMILARITY option allows the lower value of the axis showing the similarities (or percentage similarities or distances, according to the setting of the METHOD option) to be set e.g. to zero. Otherwise, this is determined automatically from the minimum value in the data. By default the axis is not plotted, but this can be changed by setting option DSIMILARITY=yes.

The NPAGES option allows the display to be split over several pages in a high-resolution plot.

The PAGEINFORMATION option then controls what information is shown on the pages:

similarity	includes the similarity axis on pages 2 onwards when DSIMILARITY=yes (otherwise it is only on page 1),
title	includes the TITLE on pages 2 onwards, and
pagenumber	includes page numbers.

As in other graphics commands, the SCREEN option controls whether to clear the high-resolution graphics screen before plotting (default clear), and the ENDACTION option controls whether Genstat pauses or continues after completing the plot.

For high-resolution graphics, the PENS parameter can be supplied with a scalar indicating the graphics pen with which to draw the dendrogram. Alternatively, if required, a variate can be specified to highlight the structure of the dendrogram by drawing different links with different pens; the links are taken in the same order as the rows of the AMALGAMATIONS matrix from HCLUSTER or in increasing order of the links of the minimum spanning tree. DDENDROGRAM will use pen 1 if the PENS parameter is not set. Any pens used by DDENDROGRAM will be set to METHOD=line, SYMBOLS=0, JOIN=given. If a scalar is supplied or PENS is not set, the pen used will also have LINESTYLE set to 1. If a variate is used, appropriate settings of COLOUR and LINESTYLE should set (using the PEN directive) prior to calling DDENDROGRAM. Similarly, with line-printer graphics, the PENS parameter can be set either to a string or to a text, according to whether the links are to be drawn with the same or different symbols; if the parameter is unset, the plus symbol (+) is used for all the links.

The ZIGGURAT parameter can be used to save the "ziggurat-degree" (Critchley 1983) of each link. This could then be used to form the setting of the PENS parameter for a later dendrogram, in order to display particular aspects of the clustering more clearly.

The SAVE parameter can be used to save the various structures that control the drawing of a dendrogram in order to save computing time when drawing a similar dendrogram. The SAVE structure should then be used as the setting of the DATA parameter, and the CHANGE option used to indicate the stage at which to start changing aspects of the previous dendrogram. The various stages (in order) involve the following options and parameters:

order	ORDERING and PERMUTATION;
dendrogram	STYLE and METHOD;
display	REVERSE, ORIENTATION, SCREEN, LABELS, TITLE, WINDOW, PENS, DSIMILARITY and LOWSIMILARITY.

Example 6.19.5 plots dendrograms to display the average linkage clustering of the cars in Section 6.19.1. First it uses the permutation variate Carperm to produce the dendrogram in the same ordering as in Example 6.19.1; see Figure 6.19.5a. Then it shows four other styles; see Figure 6.19.5b. Notice that the save structure DfrstAv is used in line 71 to avoid repeating all the calculations.

Figure 6.19.5b

Example 6.19.5

```
54  TEXT Cars; VALUES=!T(Estate,'Arna1.5','Alfa2.5',Mondialqc,\
55    Testarossa,Croma,Panda,Regatta,Regattad,Uno,\
56    X19,Contach,Delta,Thema,Y10,Spider)
57  FRAME 1; YLOWER=0; YUPPER=1; XLOWER=0; XUPPER=1
58  DDENDROGRAM [STYLE=lower; ORDERING=given; LOWSIMILARITY=0; \
59    DSIMILARITY=yes] Caramalg; PERMUTATION=Carperm; LABELS=Cars;\
60    TITLE='Dendrogram as from HCLUSTER'; SAVE=DKeep
61  " types of ordering "
62  FRAME 5...8; YLOWER=2(0.5,0.0); YUPPER=2(1.0,0.5);\
63                XLOWER=(0.0,0.5)2; XUPPER=(0.5,1.0)2
64  DDENDROGRAM [STYLE=average; ORDERING=first; REVERSE=yes; SCREEN=clear;\
65    ENDACTION=continue; CHANGE=order; DSIMILARITY=yes] DATA=DKeep;\
66    TITLE='A: STYLE=average, ORDER=first'; WINDOW=5; SAVE=DSFrstAv
67  DDENDROGRAM [STYLE=centroid; ORDERING=size,ziggurat;\
68    SCREEN=keep; ENDACTION=continue; CHANGE=order; DSIMILARITY=yes]\
69    DATA=DKeep; TITLE='B: STYLE=centroid, ORDER=size,zig'; WINDOW=6
70  DDENDROGRAM [STYLE=lower; ORDERING=first; REVERSE=yes;\
71    SCREEN=keep; ENDACTION=continue; CHANGE=dendrogram; DSIMILARITY=yes]\
72    DATA=DSFrstAv; TITLE='C: STYLE=lower, ORDER=first'; WINDOW=7
73  DDENDROGRAM [STYLE=full; ORDER=ziggurat,size; SCREEN=keep; \
74    ENDACTION=pause; CHANGE=order; DSIMILARITY=yes] DATA=DKeep;\
```

```
75    PERMUTATION=PSave; TITLE='D: STYLE=full, ORDER=zig,size'; WINDOW=8;\
76    ZIGGURAT=ZigDeg; SAVE=DSave
```

6.19.6 Plotting a minimum spanning tree: the DMST procedure

DMST procedure

Gives a high resolution plot of an ordination with minimum spanning tree (A.W.A. Murray).

Options

DIMENSIONS = *scalars*	Two numbers specifying the dimensions to display on the y- and x-axes; default 2,1
TITLE = *text*	Title for the graph
WINDOW = *scalar*	Window for the graph; default 1
KEYWINDOW = *scalar*	Window for the key; default 2
SCREEN = *string token*	Controls screen (clear, keep); default clea

Parameters

COORDINATES = *matrices* or *datamatrices*	Coordinates from ordination
TREE = *matrices*	Minimum spanning tree
SIMILARITY = *symmetric matrices*	Association matrix used to derive ordination
SYMBOLS = *factors* or *texts*	Symbols to label the coordinates
PENCOORDINATES = *scalars*	Pen to use for the coordinates
PENTREE = *scalars*	Pen to use for the minimum spanning tree

DMST plots a minimum spanning tree using coordinates saved, for example, from a PCO (6.10.1). The COORDINATES parameter specifies the coordinates for the units in the plot, using either a matrix or a pointer to a set of variates (that is, a data matrix). The minimum spanning tree can be supplied using the TREE parameter, or it can be calculated (by HDISPLAY; 6.19.2) from the original association matrix specified using the SIMILARITY parameter. If TREE supplies a matrix with no values, these will be set to the tree calculated from the SIMILARITY matrix. If the COORDINATES structure was originally declared with row labels the procedure will automatically use these to label the plots. Alternative symbols can be defined using the SYMBOLS parameter. You can also specify the pens to be used to plot the coordinates and tree, using parameters PENCOORDINATES and PENTREE respectively. The definition of these pens, outside the procedure, thus allows the colour, size, font and linestyle of links in the tree to be controlled. By default the coordinates are plotted with colour black and the tree with colour red, symbols are 0.8 of normal size, and the tree is plotted with a dotted line.

Options TITLE, WINDOW, KEYWINDOW and SCREEN function as usual for high resolution graphics. If the WINDOW is unset a default layout with appropriately labelled axes is produced in window 1. Axes will be scaled automatically unless limits have already been set outside the procedure.

Example 6.19.6 uses DMST to plot a minimum spanning tree for species recorded on the Park Grass experiment at Rothamsted (see Digby & Kempton 1987). The resulting graph is in Figure 6.19.6.

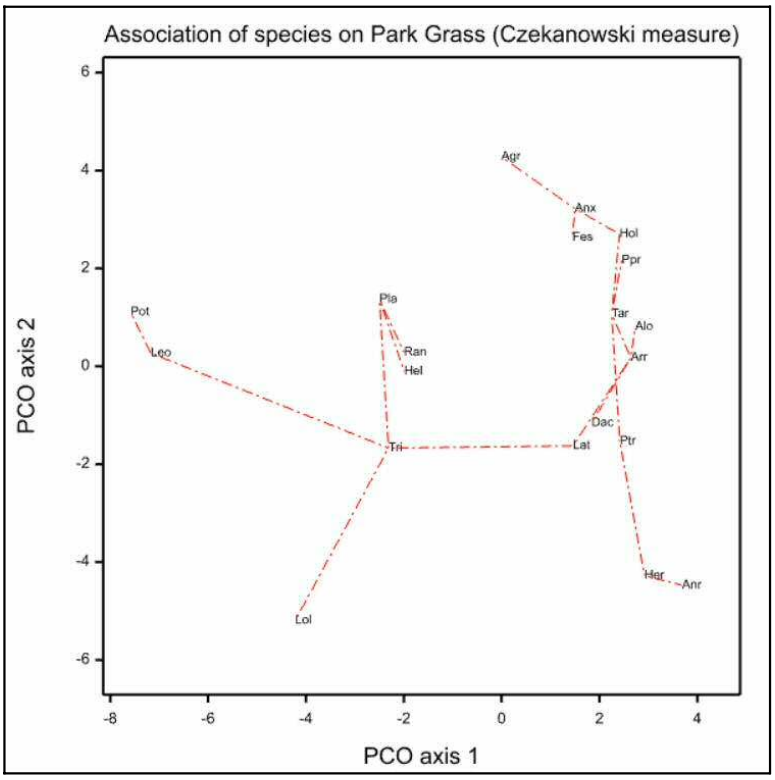


Figure 6.19.6

Example 6.19.6

```
" Data from Table 1.5 of Digby & Kempton (1987)."
TEXT Spp,title,Spp; VALUES=\
     !t(Agr,Alo,Anx,Arr,Dac,Fes,Hel,Hol,Ppr,Ptr, \
        Lol,Lat,Tri,Anr,Her,Leo,Pla,Pot,Ran,Tar),\
     'Association of species on Park Grass (Czekanowski measure)'
SYMMETRICMATRIX [ROWS=Spp] PGsim
READ [PRINT=data,errors]  PGsim

100
75 100
88 83 100
72 95 81 100
63 87 73 93 100
85 84 93 85 77 100
57 68 65 65 67 69 100
84 89 91 88 80 87 62 100
76 88 81 90 81 85 65 91 100
63 84 73 85 88 77 62 80 81 100
33 38 33 39 44 36 50 29 22 44 100
58 79 69 85 83 73 60 76 77 83 40 100
61 64 65 70 71 69 71 62 61 71 58 80 100
29 60 46 62 58 49 40 50 57 63 20 56 40 100
46 73 59 79 82 63 50 67 67 82 46 81 67 69 100
43 32 37 38 42 40 48 37 38 36 40 45 64  0 22 100
73 68 73 69 67 77 81 69 69 62 52 70 81 30 51 57 100
29 23 25 24 27 27 36 25 29 20 17 21 36  0  8 62 40 100
67 73 70 75 73 71 72 67 67 68 54 71 78 38 53 52 87 42 100
77 92 85 93 89 89 71 91 93 89 37 85 71 59 76 41 78 28 76 100 :
LRV  [ROWS=Spp; COLUMNS=3] L3
PCO  [PRINT=roots] PGsim; LRV=L3
```

```
Principal coordinates analysis
==============================

Latent Roots
------------

             1         2         3         4         5         6         7
        215.44    127.06     89.05     58.95     50.42     37.55     30.46

             8         9        10        11        12        13        14
         23.64     16.41     15.69     12.82     12.14     10.34      8.12

            15        16        17        18        19        20
          6.30      5.75      2.90      2.61      1.45      0.00


Percentage variation
--------------------

             1         2         3         4         5         6         7
         29.63     17.48     12.25      8.11      6.93      5.16      4.19

             8         9        10        11        12        13        14
          3.25      2.26      2.16      1.76      1.67      1.42      1.12

            15        16        17        18        19        20
          0.87      0.79      0.40      0.36      0.20      0.00


Trace
-----

       727.1

  31  FRAME 3; SCALING=xyequal
  32  YAXIS 3; TITLE='PCO axis 2'
  33  XAXIS 3; TITLE='PCO axis 1'
  34  DMST [WINDOW=3; KEY=0; TITLE=title] L3['Vectors']; SIM=PGsim
```

6.19.7 Comparing clusterings: the HCOMPAREGROUPINGS procedure

HCOMPAREGROUPINGS procedure

Compares groupings generated, for example, from cluster analyses (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (indexes, tests); default inde
PLOT = *string*	What to plot (histogram); default *
METHOD = *string tokens*	Which indexes to calculate (arand, jaccard, rand); default arand
NTIMES = *scalar*	Number of permutations to make for the tests; default 999

Parameters

FIRSTGROUPING = *factors*	First set of groupings
SECONDGROUPING = *factors*	Second set of groupings
ESTIMATES = *pointers*	Saves the values of the indexes calculated from the original data set
SEED = *scalars*	Seed for the random number generator used to make the permutations; default 0 continues from the previous generation or (if none) initializes the seed automatically

PERMUTATIONESTIMATES = *pointer*s
Saves the values of the indexes calculated from the permuted data sets

---

HCOMPAREGROUPINGS calculates indexes to assess the similarity between two sets of groupings, which are specified in factors using the FIRSTGROUPING and SECONDGROUPING parameters. These may, for example, have been obtained from two different cluster analyses, and must not be restricted.

The METHOD option selects the indexes, with settings:

arand	adjusted Rand index,
jaccard	Jaccard index, and
rand	Rand index.

The Rand index (Rand 1971) is defined as

$$(np_1 + np_2) / {}^{N}C_2$$

where

np_1 is the number of pairs of units that are in the same group in both factors,
np_2 is the number of pairs of units that are in different groups in both factors,
N is the total number of units, and
${}^{N}C_2$ is the total number of ways of selecting of 2 units from a sample of N units, which can be calculated as $N\times(N-1)/2$.

This ranges from zero (for no similarity) to one (for complete similarity).

The adjusted Rand index of Hubert & Arabie (1985) is defined as

$$\{ \sum_i \sum_j ({}^{mij}C_2) \} - \{ \sum_i ({}^{ai}C_2) \times \sum_j ({}^{bj}C_2) / ({}^{N}C_2) \} / \\ - \{ \sum_i ({}^{ai}C_2) + \sum_j ({}^{bj}C_2) \} - \{ \sum_i ({}^{ai}C_2) \times \sum_j ({}^{bj}C_2) / ({}^{N}C_2) \}$$

where

m_{ij} is the number of units that are in group i for the first factor, and group j for the second factor,
a_i is the number of units in group i of the first factor, and
b_j is the number of units in group j of the second factor.

The first term in the numerator measures the agreement between the groupings. The second term is the expected value of the first term, assuming a generalized hypergeometric distribution, and the first term of the denominator is its maximum value. The index has a value of zero if the groupings are independent, and one if they are in complete agreement.

The Jaccard index is defined as

$$np_1 / ({}^{N}C_2 - np_2)$$

This is similar to the Rand index, except that it excludes the pairs of units that are in different groups in both factors.

The ESTIMATES parameter can save a pointer, containing a scalar for each index, to save the calculated values. The elements of the pointer are labelled by the index names, but defined so that you can refer to them in either lower- or upper-case or a mixture.

The PRINT option controls the printed output, with settings:

indexes	prints the indexes, and
tests	prints probabilities obtained from random permutation tests.

The random permutation tests allow you to assess whether the similarity may have arisen only by chance. The NTIMES option specifies the number of permutations to take (default 999). HCOMPAREGROUPINGS checks whether NTIMES is greater than the number of possible permutations available for the data set. If so, it does an exact test instead, which uses each possible permutation once. The SEED option specifies the seed that is used to obtain the random numbers used to form the permutations.

The PERMUTATIONESTIMATES parameter can save a pointer, containing a variate for each

index, to save the values calculated in the random permutations. The elements of the pointer are labelled by the index names, but defined so that you can refer to them in either lower- or upper-case or a mixture.

You can set option `PLOT=histogram` to plot histograms showing where the calculated value of each index lies within those obtained from the permutation tests.

Example 6.19.7 compares the groupings, saved from the cluster analysis of the cars in Example 6.19.1, with those from a cluster analysis that uses the single-linkage method instead of average linkage. Unsurprisingly, the permutation test shows that the similarity bertween the groupings is unlikely to have arisen by chance!

Example 6.19.7

```
  77  HCLUSTER [PRINT=dendrogram; METHOD=singlelink] Carsim; \
  78           GTHRESHOLD=90; GROUPS=Cargrpsing

Single linkage cluster analysis
===============================


Dendrogram
----------

  ** Levels   100.0  90.0  80.0  70.0

Estate         1  ..
Regatta        8  ..)
Arna1.5        2  ..)
Delta         13  ..)..
Y10           15  .....)
Panda          7  ..   )
Uno           10  ..)..)..
X19           11  ........)
Spider        16  ........)..
Regattad       9  ...........)
Alfa2.5        3  ...........)
Thema         14  .....      )
Croma          6  .....).....)..
Mondialqc      4  ...........   )
Testarossa     5  ........   )  )
Contach       12  ........)..)..)..............

  79  PRINT    Cargrp,Cargrpsing

      Cargrp  Cargrpsing
           1           1
           1           1
           1           5
           3           7
           3           8
           2           6
           1           1
           1           1
           1           4
           1           1
           1           2
           3           9
           1           1
           2           6
           1           1
           1           3

  80  HCOMPAREGROUPINGS [PRINT=indexes,tests; METHOD=arand,jaccard,rand]\
  81           FIRSTGROUPING=Cargrp; SECONDGROUPING=Cargrpsing; SEED=93587
```

```
Rand index 0.6917, probability 0.014
Adjusted Rand index 0.3768, probability 0.014
Jaccard index 0.3729, probability 0.014
(probabilities from 999 random permutations)
```

6.19.8 Bootstrap analyses to assess the reliability of the clusters: the HBOOTSTRAP procedure

HBOOTSTRAP procedure

Performs bootstrap analyses to assess the reliability of clusters from hierarchical cluster analysis (R.W. Payne).

Options

PRINT = *string token*	Controls printed output (clusters, dendrograms; default * i.e. none
METHOD = *string token*	Criterion for forming clusters (singlelink, nearestneighbour, completelink, furthestneighbour, averagelink, mediansort, groupaverage); default sing
CLIMIT = *scalar*	Similarity value below which clusters are not recorded; default 0
UNITS = *text* or *variate*	Names to label the units of the clusters when they are printed; default *
MINKOWSKI = *scalar*	Index *t* for use with TEST=minkowski
CLUSTERS = *pointer*	Specifies or saves the clusters
REPLICATION = *variate*	Saves the replication of the clusters in the bootstrap samples
NDATASAMPLE = *scalar*	Number of DATA vectors to take in each sample; default takes the same number as supplied by the DATA parameter
NTIMES = *scalar*	Number of times to resample; default 100
SEED = *scalar*	Seed for random number generator; default continue from previous generation or use system clock

Parameters

DATA = *variates* or *factors*	The characteristics of the units to be clustered
TEST = *string tokens*	Test type, defining how each DATA variate or factor is treated in the calculation of the similarity between each unit (simplematching, jaccard, russellrao, dice, antidice, sneathsokal, rogerstanimoto, cityblock, manhattan, ecological, euclidean, pythagorean, minkowski, divergence, canberra, braycurtis, soergel); default * ignores that variate or factor
RANGE = *scalars*	Range of possible values of each DATA variate or factor; if omitted, the observed range is taken

HBOOTSTRAP uses bootstrapping to assess the reliability of clusters formed in a hierarchical cluster analysis. The characteristics of the units to be clustered are described in a list of variates and factors, specified by the DATA parameter. The TEST parameter defines how each one is to be used when calculating similarities, and the RANGE parameter can specify ranges of their

values. These operate as in the FSIMILARITY directive (6.1.2), which is used to form the similarity matrix for each cluster analysis. The MINKOWSKI option specifies the index *t* for the Minkowski tests.

For each bootstrap sample, a set of vectors is formed by sampling with replacement from the DATA vectors. The NDATASAMPLE option specifies the number of vectors to take; by default this is the same as the number of vectors supplied by DATA. The NTIMES option specifies the number of bootstrap samples; default 100. The SEED option specifies the seed to use for the random numbers used to select the sample; the default of zero continues an existing sequence of random numbers or, if none, it initializes the sequence using the system clock. HBOOTSTRAP does a cluster analysis with those vectors using the HCLUSTER directive, and obtains the clusters that it forms using the HFCLUSTERS procedure. The CLIMIT option can be used to specify a limit, below which any clusters will be excluded.

The CLUSTERS option can supply a pointer containing a list of clusters whose reliability is to be assessed. This would usually have been obtained previously, from a cluster analysis performed with all the DATA vectors, as in 6.19.1. Alternatively, if CLUSTERS is set to a pointer whose number of values has not been defined, or to an undeclared data structure, this will be defined as a pointer containing one of every cluster that has occurred during the bootstrapping. Each cluster is represented as a variate, containing the number of each unit in that cluster. (This number corresponds to the location of that unit in the DATA vectors.)

The REPLICATION option can save a variate containing the number of times each cluster has occurred during the bootstrapping. These replications can be used by the DCLUSTERLABELS procedure to label the clusters on a dendrogram.

The clusters and their replications can be printed by setting option PRINT=clusters. The UNITS option can be set to a text or a variate, to provide textual labels or other numbers to use for the units of the clusters, instead of the numbers in the CLUSTERS variates. The other PRINT setting, dendrogram, prints the dendrogram of the cluster analysis from each bootstrap sample.

The whole process is shown in Example 6.18.8.

Example 6.19.7

```
 82  " obtain the clusters from the original cluster analysis "
 83  HFCLUSTERS  Caramalg; CLUSTERS=Clusters
 84  " see often these clusters occur in 100 bootstrap samples of data variables "
 85  HBOOTSTRAP  [PRINT=clusters; METHOD=averagelink; NTIMES=100; SEED=161647;\
 86              CLUSTERS=Clusters; REPLICATION=Reps] #Cd,Carb,Drive;\
 87              TEST=4(cityblock),4(Euclidean),2(cityblock),2(simplematch)

Clusters
========

 Replication 58 76 70 83 51 65 41 54 70 68 35 36 14 37 100
     Cluster {  {  {  {  {  {  {  {  {  {  {  {  {  {   {
              1  5  6  7 11  1  3  4  7  1  1  1  1  1   1
              8 12 14 10 16  2 11  5 10  2  2  2  2  2   2
              }  }  }  }  }  8 16 12 15  8  7  7  3  3   3
                             }  }  }  } 13  8  8  7  6   4
                                      } 10  9  8  7   5
                                        13 10  9  8   6
                                        15 13 10  9   7
                                         } 15 11 10   8
                                            } 13 11   9
                                              15 13  10
                                              16 14  11
                                               } 15  12
                                                 16  13
                                                  }  14
                                                     15
                                                     16
                                                      }
```

```
88  " replot the original dendrogram "
89  DDENDROGRAM [STYLE=average; ORDERING=given; LOWSIMILARITY=0; \
90               DSIMILARITY=yes] Caramalg; PERMUTATION=Carperm;\
91               LABELS=Cars; WINDOW=1
92  " plot the numbers of occurrence on the dendrogram "
93  DCLUSTERLABELS [WINDOW=1] #Clusters; LABEL=#Reps
```

First of all, in line 83, the HFCLUSTERS procedure is used to obtain the complete set of clusters from the original cluster analysis. This requires the amalgamations matrix, which was saved in Caramalg in line 34 of Example 6.18.1. For full details of HFCLUSTERS, see the *Genstat Reference Manual, Part 3 Procedures*. For bootstrapping we just need the first two parameters: the first specifies the amalgamations matrix, and the second saves the clusters (in a pointer).

The parameters of HBOOTSTRAP, in lines 86-87 reproduce the parameters settings from the FSIMILARITY command used to form the similarity matrix for the original cluster analysis (Example 6.19.1, lines 30-31). The setting of the METHOD option is the same as in the HCLUSTER command that produced the original cluster analysis (Example 6.19.1, line 32). The NTIMES option asks for 100 bootstrap samples to be taken. (This is actually the default, and so could have been omitted.) The SEED option sets a seed for the random numbers. (We have done this here so that, if you run the example, you will obtain the same results as here.) The CLUSTERS option supplies the clusters (formed by HFCLUSTERS). Te REPLICATION option saves the number of times they occur during the bootstrapping, and the PRINT option has been set to print them.

In lines 89-91 DDENDROGRAM plots the original dendrogram, and in line 93 the DCLUSTERLABELS procedure is used to label the clusters by their replications. For full details of HFCLUSTERS, see the *Genstat Reference Manual, Part 3 Procedures*. Here we simply needed to set the WINDOW option to the number of the window containing the dendrogram, the first parameter to the clusters, and the second parameter (LABEL) to their replications. (The special symbol # replaces the pointer Clusters and the variate Reps by their individual elements.) The resulting plot, in Figure 6.19.8, suggests that most of the clusters are sensitive to the choice of vectors in the cluster analysis.

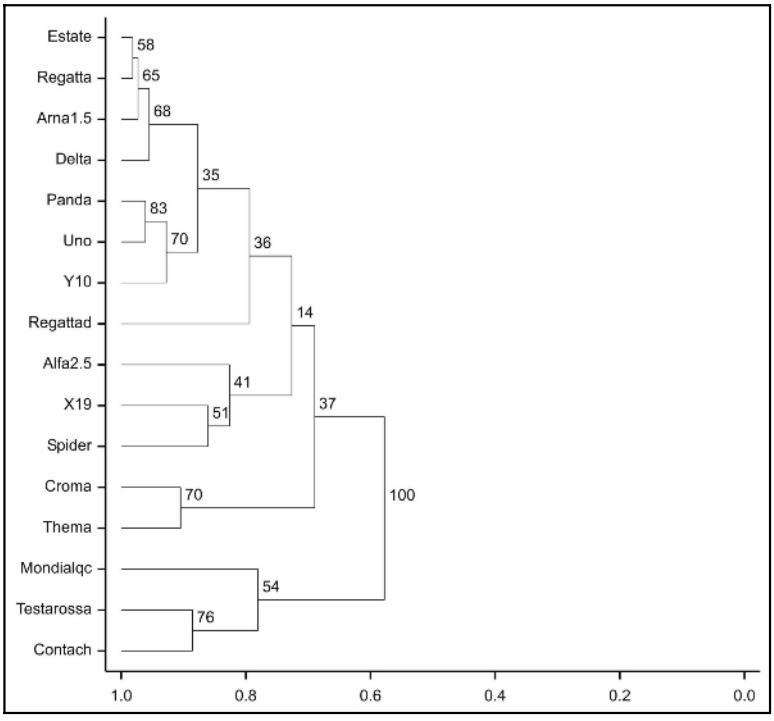


Figure 6.19.6

6.20 Non-hierarchical classification

A common statistical problem is to divide the units of a data set into some number of mutually exclusive groups, or classes. Usually you would hope that the groups will be reasonably homogeneous, and distinct from each other. When you do not know the most natural number of classes in advance, you might be interested in several classifications into different numbers of groups: you can then inspect these, and make a decision about the most acceptable number of groups. One way of achieving such groupings is to take the results of a hierarchical classification (6.19), and cut the dendrogram at appropriate levels to obtain groupings into several numbers of classes. However, the statistical properties of the resulting groups are not at all clear, and the hierarchical nature of the groupings into various numbers of classes can impose undue constraints. An alternative approach is to optimize some suitably chosen criterion directly from the data matrix, to obtain one or more non-hierarchical classifications.

Non-hierarchical classification (or *K*-means clustering) methods differ according to the criterion that they optimize and in the algorithm used to search for an optimum value of the chosen criterion. In Genstat one of four different criteria may be optimized, and the optimization algorithm uses one of two different strategies.

Which criterion to choose depends on the type of data. Suppose first that they can be considered as being a mixture of k multi-Normal distributions, with the same variance-covariance matrix. Then the maximum-likelihood estimate of this matrix is given when the grouping into k classes minimizes the determinant of the within-class variance-covariance matrix, pooled over the k groups (Friedman & Rubin 1967); in other words, the optimization criterion is to minimize this determinant.

When only two groups are to be formed, the criterion above is equivalent to maximizing the Mahalanobis distance between the two classes. However, when the number of groups to be formed is greater than two, maximizing the total Mahalanobis distance between the classes will generally give different results to minimizing the determinant of the pooled within-class dispersion matrix. Maximizing the total Mahalanobis distance is the second available criterion.

The third criterion maximizes the total Euclidean distance between the classes; this is equivalent to minimizing the total within-class sum of squares: that is, the trace of the pooled within-class dispersion matrix. This third criterion can be thought of as a simpler variant of the first, that does not rely on the assumptions of multi-Normality or equal within-class dispersion.

The fourth criterion gives maximal predictive classification (Gower 1974). It is relevant when all the data are binary: that is, when they take only two values, usually designated by zero and one. Within each class, the *class predictor* is defined to be a list with one entry for each variate: the ith entry is whichever value (zero or one) is more frequent in the class for the ith variate. The criterion, W, to be maximized is the sum over the classes of the number of agreements between units of each class and their class predictor. When several different classifications give the same maximum value for W, a subsidiary criterion B is minimized. Whereas W measures within-class homogeneity, B measures between-class heterogeneity: it is the sum of the number of correct predictions for each unit when predicted by any of the class predictors of the classes other than the one to which the unit is assigned.

The algorithm used in Genstat to search for optimal values of the chosen criterion proceeds as follows. Starting from some initial classification of the units into the required number of groups, the algorithm repeatedly transfers units from one group to another so long as such transfers improve the value of the criterion. When no further transfers can be found to improve the criterion, the algorithm switches to a second stage which examines the effect of swopping two units of different classes. The algorithm alternates between the two types of search until neither gives any improvement. Searching for swops is computationally more expensive than searching for transfers, so only one swop is performed each time before the algorithm switches to search for transfers. However, using only swops has the advantage that the group sizes remain constant: if this is what you want, you can direct Genstat to search only for swops.

There is no guarantee that the classification resulting from the above algorithm will be globally optimal: to be sure of that, you would need to try all possible classifications of the units into the required number of groups. All that is known is that no improvement can be made to the criterion by either of the types of transfer strategy. The chance that the algorithm will produce a near-optimal classification can be much improved by providing a good initial classification. You could obtain this from a hierarchical classification method, or by examining a set of principal component scores from the data. The effect of trying different initial classifications can be interesting, and provides some information on the closeness to optimality.

The methods above may not be feasible for very large data sets. Section 6.20.3 provides an alternative, which uses the distribution of the units in dimensions from principal components analysis.

These methods are all available through the cluster analysis menus in Genstat *for Windows*.

6.20.1 The CLUSTER directive

CLUSTER directive

Forms a non-hierarchical classification.

Options

PRINT = *string tokens*	Printed output required (criterion, optimum, units, typical, initial, random); default * i.e. no printing
DATA = *matrix* or *pointer*	Data from which the classification is formed, supplied as a units-by-variates matrix or as a pointer containing the variates of the data matrix
CRITERION = *string token*	Criterion for clustering (sums, predictive, within, Mahalanobis); default sums
INTERCHANGE = *string token*	Permitted moves between groups (transfer, swop); default tran (implies swop also)
START = *factor*	Initial classification; default * i.e. splits the units, in order, into NGROUPS classes of nearly equal size
NSTARTS = *scalar*	Number of starting configurations to be used; default 0
SEED = *scalar*	Seed for the random numbers used to form random starting configurations; default 0

Parameters

NGROUPS = *scalars*	Numbers of classes into which the units are to be classified: note, the values of the scalars must be in descending order
GROUPS = *factors*	Saves the classification formed for each number of classes
CRITERIONVALUE = *scalars*	Saves the criterion values (representing within-class homogeneity)
BCRITERIONVALUE = *scalars*	Saves the subsidiary criterion values (representing between-class heterogeneity for maximal predictive classification)
MEANS = *matrices*	Saves the variate means for the groups of each classification
PREDICTORS = *matrices*	Saves the group predictors from maximal predictive classification

Printed output is controlled by the PRINT option. This has the following possible settings.

criterion	prints the optimal criterion value.
optimum	prints the optimal classification.
units	prints the data with the units ordered into the optimal classes.
typical	prints a typical value for each class: for maximal predictive classification this is the class predictor; for the other methods it is the class mean.
initial	if this is set, the requested sections of output are also printed for the initial classification.
random	if this is set, the requested sections of output are also printed for the optimum configuration obtained from every random start.

The DATA option supplies the data to be classified. This specifies a single structure that must be either a matrix, with rows corresponding to the units and columns to the variables, or a pointer whose values are the identifiers of the variates in the data matrix. Internally, CLUSTER operates on a matrix, and so it will copy the variate values into a matrix if you supply a pointer as input; thus, it is more efficient to supply a matrix, especially with large data sets.

The CRITERION option specifies which criterion CLUSTER is to optimize. The four available settings are:

sums	minimize the within-group sum of squares (and thus maximize the between-group sum of squares);
predictive	maximal predictive classification;
within	minimize the determinant of the pooled within-class dispersion matrix;
mahalanobis	maximize the total Mahalanobis squared distance between the groups.

The default is sums.

The INTERCHANGE option specifies which types of interchange (transfers or swops) are to be used. The default is transfer, which is taken to imply that both transfers and swops are used, since a swop is simply two transfers. If you set INTERCHANGE=swop, only swops are used. If INTERCHANGE=* the algorithm does not attempt to improve the classification from the initial classification; you might want this, in conjunction with the PRINT=initial setting, to display the results for an existing classification which you do not wish to improve.

The START option can be used to supply a factor to define the initial classification. This might be constructed using the CLASSIFY procedure (6.20.2). If there are *k* classes, CLASSIFY finds the *k* units that are furthest apart in the multi-dimensional space defined by the data variates. These are then used as the nuclei for the classes, with each remaining unit being allocated to the class containing the nearest nucleus. The default splits the units, in order, into NGROUPS classes of nearly equal size.

As an alternative to the use of CLASSIFY, the NSTARTS option allows you to specify a number of random permutations of the initial classification to try. CLUSTER then saves the best classification that it finds. By default, NSTARTS=0, i.e. no randomization is done. The SEED option supplies the seed for the random numbers that are used to do the permutations. The default of zero continues the existing sequence of random numbers, if CLUSTER has already been used in the current Genstat job. If CLUSTER has not yet been used, Genstat picks a seed at random.

The first parameter, NGROUPS, specifies the number of groups, or classes, to be formed. Often you would want several classifications from a single data set, into different numbers of groups. In this case, the NGROUPS parameter should be a list of scalars, defining the numbers of groups in descending order. For the initial classification of the second classification, CLUSTER takes the optimal classification from the first number of groups, and does some reallocation of units to

make a smaller number of groups. This is repeated, as often as required, to provide initial classifications for all the later analyses; hence the need to specify the numbers in descending order. Random starts are done only for the first number of groups.

The GROUPS parameter can specify a list of factors to save the optimal classifications. The CRITERIONVALUE parameter can specify a list of scalars to save the criterion values for each number of groups. The subsidiary criterion values involved in maximal predictive classification can be saved (also in scalars) using the BCRITERIONVALUE parameter. The MEANS parameter can save matrices containing the means of the variates within the groups of the classifications, and the PREDICTORS parameter can save matrixes containing the group predictors from maximal predictive classifications.

Doran & Hodson (1975) give some measurements made on 28 brooches found at the archaeological site of the cemetery at Munsingen. Seven of these variables, transformed to logarithms, are used in Example 6.20.1a.

Example 6.20.1a

```
  2  UNITS [NVALUES=28]
  3  POINTER [VALUES=Foot_lth,Bow_ht,Coil_dia,Elem_dia,Bow_wdth, \
  4     Bow_thck,Length] Data
  5  READ Data[]

 Identifier   Minimum      Mean   Maximum    Values   Missing
   Foot_lth     2.398     3.278     4.554        28         0
     Bow_ht     2.079     2.842     3.296        28         0
   Coil_dia     1.792     2.166     2.833        28         0
   Elem_dia     1.099     2.026     2.708        28         0
   Bow_wdth     3.045     4.064     5.176        28         0
   Bow_thck     2.708     3.621     4.357        28         0
     Length     3.296     4.003     4.860        28         0

 34  CLUSTER [PRINT=criterion,optimum,initial; DATA=Data; SEED=-1] 5,4,3

Non-hierarchical clustering
===========================


Sums of squares criterion
-------------------------


Initial classification
----------------------

Number of classes = 5


Class contributions to criterion
--------------------------------

          1          2          3          4          5
      7.623      5.335      1.434      6.251      7.286

Criterion value = 27.93013


Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     1     1     1     1     1     1     2     2     2     2     2     2
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     3     3     3     3     3     3     4     4     4     4     4     5
 Unit    25    26    27    28
Group     5     5     5     5
```

```
Optimum classification
----------------------

Number of classes = 5


Class contributions to criterion
--------------------------------

          1           2           3           4           5
      2.205       1.715       1.965       2.361       2.633

Criterion value = 10.87899


Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     4     5     3     1     1     5     5     2     1     1     4     2
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     3     3     3     3     3     2     3     4     2     2     3     1
 Unit    25    26    27    28
Group     1     5     5     4


Initial classification
----------------------

Number of classes = 4


Class contributions to criterion
--------------------------------

          1           2           3           4
      2.205       3.839       6.580       2.361

Criterion value = 14.98493


Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     4     2     3     1     1     3     3     2     1     1     4     2
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     3     3     3     3     3     2     3     4     2     2     3     1
 Unit    25    26    27    28
Group     1     3     3     4


Optimum classification
----------------------

Number of classes = 4


Class contributions to criterion
--------------------------------

          1           2           3           4
      4.394       1.715       3.670       3.119

Criterion value = 12.89727


Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     4     3     1     1     1     3     3     2     1     4     4     2
```

```
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     1     1     1     1     1     2     3     4     2     2     1     1
 Unit    25    26    27    28
Group     1     3     3     4


Initial classification
----------------------

Number of classes = 3


Class contributions to criterion
--------------------------------

          1           2           3
     11.931       4.174       3.670

Criterion value = 19.77417


Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     1     3     1     1     1     3     3     2     1     1     1     2
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     1     1     1     1     1     2     3     1     2     2     1     1
 Unit    25    26    27    28
Group     1     3     3     2


Optimum classification
----------------------

Number of classes = 3


Class contributions to criterion
--------------------------------

          1           2           3
     15.279       1.715       2.633

Criterion value = 19.62671


Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     1     3     1     1     1     3     3     2     1     1     1     2
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     1     1     1     1     1     2     1     1     2     2     1     1
 Unit    25    26    27    28
Group     1     3     3     1
```

The seven variables, represented by the pointer `Data`, are defined on lines 3 and 4 and their values are read in line 5. The `PRINT` option of the `CLUSTER` statement (line 34) specifies that the criterion value and optimal classification are to be printed, and that the criterion value and initial classification are to be printed before the transfer and swop algorithm is used. The criterion to be optimized is the default, namely the minimum sum of squares within groups. The `DATA` option supplies the seven variables, via their pointer. The first parameter specifies that classifications are to be formed into five, then four, then three, groups.

The `SEED` option has been set to −1 and no initial classification has been supplied, so the `CLUSTER` directive assigns the units to five classes, as described above. Thus the first six units

are in class 1, and so on. This classification is printed near the beginning of the output from CLUSTER. It is preceded by the value of the minimum within-class sum of squares criterion for this classification, and a break-down of this value into the contributions from each class; each such contribution is the sum of squares within a class. At the optimal classification, Genstat prints the criterion value obtained, and its contributions from each class. You can see that the optimal classification obtained is quite different from the initial classification: in fact only 12 of the 28 units are in the same class that they started in.

To obtain an initial classification into four groups the CLUSTER directive reassigns each unit in group 5 to the nearest group: there are five such units, and four of them are closest to group 3. If you examine the initial and optimal classifications into four groups, and the optimal classification into five groups, you will see that many of the units of group 3 have transferred to group 1. This suggests that the optimal fifth group has become the third group; and that the old third and first groups have merged. The initial classification into three groups is similarly formed by reassigning the units in the fourth optimal group: of the five units involved, four are reassigned to group 1. This suggests that group 1 is becoming dominant. In fact little improvement is made to the criterion by forming the optimal classification for three groups; only two units move, both to the first group.

Example 6.20.1b illustrates the maximal predictive criterion. Remember that this method has a subsidiary criterion, *B*, as well as the main criterion *W*. The criterion *W* measures within-class consistency, and has separate contributions from each class; the criterion *B* measures between-class distinctness and has a contribution from all possible pairs of groups.

Example 6.20.1b

```
  2  POINTER [NVALUES=4] Y
  3  VARIATE [NVALUES=30] Y[]
  4  READ [PRINT=errors; SERIAL=yes] Y[]
  9  CLUSTER [PRINT=criterion,optimum,typical; DATA=Y; \
 10    CRITERION=predictive; SEED=-1] NGROUPS=5,2; GROUPS=Optimum[5,2]

Non-hierarchical clustering
===========================


Maximal predictive criterion
----------------------------


Equally optimum classifications
-------------------------------

Criterion value = 104.00000
Criterion B = 49.00000

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     3     4     2     1     1     5     3     4     3     5     1     1
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     2     4     3     5     5     1     3     4     2     5     2     5
 Unit    25    26    27    28    29    30
Group     3     5     3     1     5     1

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     3     4     2     1     1     5     3     4     3     5     1     1
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     2     3     3     5     5     4     3     4     2     5     2     5
 Unit    25    26    27    28    29    30
Group     3     5     3     1     5     1


Optimum classification
----------------------

Number of classes = 5
```

```
Class contributions to criterion
--------------------------------

          1           2           3           4           5
      25.00       14.00       28.00       12.00       25.00

Criterion value = 104.00000


Class contributions to criterion B
----------------------------------

                    1           2           3           4           5
        1       0.000      10.000       3.000      13.000      18.000
        2       6.000       0.000      10.000      10.000       2.000
        3       4.000      20.000       0.000      14.000      12.000
        4       6.000       9.000       6.000       0.000       3.000
        5      17.000       7.000      15.000      11.000       0.000


Criterion B = 49.00000


Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     3     4     2     1     1     5     3     4     3     5     1     1
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     2     3     3     5     5     1     3     4     2     5     2     5
 Unit    25    26    27    28    29    30
Group     3     5     3     1     5     1


Class predictors
----------------

           Y  Y[1]  Y[2]  Y[3]  Y[4]
           1     0     1     0     0
           2     0     0     1     1
           3     1     0     1     1
           4     0     0     0     1
           5     1     1     0     0


Optimum classification
----------------------

Number of classes = 2


Class contributions to criterion
--------------------------------

          1           2
      43.00       44.00

Criterion value = 87.00000


Class contributions to criterion B
----------------------------------

                    1           2
        1       0.000      18.000
        2      17.000       0.000


Criterion B = 35.00000
```

```
Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     2     2     2     1     1     1     2     2     2     1     1     1
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     2     2     2     1     1     1     2     2     2     1     2     1
 Unit    25    26    27    28    29    30
Group     2     1     2     1     1     1


Class predictors
----------------

            Y  Y[1]  Y[2]  Y[3]  Y[4]
            1     1     1     0     0
            2     1     0     1     1

  11  TABULATE [PRINT=counts; CLASSIFICATION=Optimum[5,2]; MARGINS=yes]

                    Count
   Optimum[2]           1           2       Count
   Optimum[5]
            1           7           0           7
            2           0           4           4
            3           0           8           8
            4           0           3           3
            5           8           0           8
        Count          15          15          30
```

Lines 2-4 define and read the data, using the pointer `Y` to specify four variates each of 30 values. The required non-hierarchical classifications are specified on lines 9 and 10. For each classification the criterion values are printed, together with the optimal classification, and the typical units for each group (that is, the class predictors). The `GROUPS` parameter has been used to specify factors to hold the optimal classifications.

When the `CLUSTER` directive has found an optimal classification, it will report all the classifications that it can find with the same optimum (provided that you have asked for the optimal classification to be printed). Several equivalent optimal classifications may often occur with maximal predictive classification, and may occur occasionally with the other criteria. When equally optimal classifications are reported, they are preceded by the criterion value together with the value of the subsidiary criterion (if relevant). If you compare the various optimal classifications printed in Example 6.20.1b, you can see that there is some ambiguity over the allocation of the 14th and 18th units.

After the details of the equally optimal classifications, Genstat prints the breakdown of the *W* and *B* criteria for the optimal classification that was found first. The (*i*,*j*)th cell of the table of class contributions to criterion *B* shows the number of correct predictions for units in group *i* when predicted by the class predictor of class *j*. For example, amongst the four units in the second group, six dichotomous values (out of 16) are correctly predicted by the first class predictor. You can check this quite easily by comparing the first class predictor (0,1,0,0) with the printed units of group 2.

The results for maximal predictive classification into two groups show a loss of within-class consistency, but improved between-class distinctness. Gower (1974) gives suggestions on how such difficulties may be resolved; for example, maximizing $W-B$ would lead to choosing the five-group classification. One preliminary to comparing two classifications is to tabulate them. This has been done on line 11, using as input the factors saved from the `CLUSTER` statement (for details of the `TABULATE` directive see 1:4.11.1). The table printed at the end of the output shows that the first group of the classification into two groups is formed from groups 1 and 5 of the five-group classification; group 2 is formed from groups 2, 3 and 4.

As mentioned already, the results of non-hierarchical classification can vary considerably

according to the initial classification. Example 6.20.1c illustrates this, using the same data as Example 6.20.1b.

Example 6.20.1c

```
  12  CLUSTER [PRINT=criterion; DATA=Y; CRITERION=predictive; SEED=-1]\
  13          NGROUPS=6,5

Non-hierarchical clustering
===========================


Maximal predictive criterion
----------------------------


Optimum classification
----------------------

Number of classes = 6


Class contributions to criterion
--------------------------------

           1           2           3           4           5           6
       18.00       24.00       19.00       22.00        4.00       22.00

Criterion value = 109.00000


Class contributions to criterion B
----------------------------------

                    1           2           3           4           5
         1       0.000       7.000      12.000       8.000      12.000
         2       7.000       0.000      17.000       9.000       9.000
         3      11.000      14.000       0.000      11.000       9.000
         4      12.000       6.000      10.000       0.000      12.000
         5       2.000       1.000       2.000       2.000       0.000
         6      10.000       8.000       2.000      10.000      14.000
                    6
         1       8.000
         2      11.000
         3       1.000
         4      14.000
         5       2.000
         6       0.000


Criterion B = 50.60000


Optimum classification
----------------------

Number of classes = 5


Class contributions to criterion
--------------------------------

           1           2           3           4           5
       18.00       24.00       19.00       22.00       24.00

Criterion value = 107.00000
```

```
Class contributions to criterion B
----------------------------------

                1          2          3          4          5
        1   0.000      7.000     12.000      8.000      8.000
        2   7.000      0.000     17.000      9.000     11.000
        3  11.000     14.000      0.000     11.000      1.000
        4  12.000      6.000     10.000      0.000     14.000
        5  12.000      9.000      4.000     12.000      0.000

Criterion B = 48.75000
```

The CLUSTER statement (lines 12 and 13) specifies that only the criterion value is to be printed, and not the detailed classifications. The number of groups to be formed is first six, then five; thus the initial classification is different from that in Example 6.6.1b. The criterion values are both only slightly better than previously ($W = 107.0$ and $B = 48.75$ compared with $W = 104.0$ and $B = 49.0$); however the contributions from the individual classes are quite different. This example illustrates the difference that the choice of initial classification can make, even with a relatively small number of units. In Example 6.20.1b the initial classification was the default partition into five groups, whereas here it is the classification into six groups, with the sixth group being dispersed.

6.20.2 Determining an initial classification: the CLASSIFY procedure

CLASSIFY procedure

Obtains a starting classification for non-hierarchical clustering (S.A. Harding).

No options

Parameters

DATA = *pointers*	Each pointer contains a set of variates giving the properties of the units to be grouped
NGROUPS = *scalars*	Indicates the number of groups required
GROUPS = *factors*	Stores the classifications formed

In non-hierarchical classification an initial classification is required, and it is advantageous to have these classes as homogeneous as possible. This reduces the risk of converging to a local optimum, and also encourages faster convergence of the iterative transfer algorithm used by the CLUSTER directive (6.20.1).

When the number of groups is greater than the number of data variates plus one, CLASSIFY forms the groups according to the positions of the units in the first dimension of a principal coordinates analysis (6.10) of the DATA variates.

Otherwise it tries to find a suitable classification into the k groups by finding the k units that are furthest apart in p-dimensional space (where p is the number of variates). These are then used as nuclei for the classes, with each of the remaining units being allocated to the class with the nearest nucleus.

The units defining the nuclei are found by first finding the two units that are furthest apart. The third unit is the unit with greatest distance from the line joining the first two units. The fourth is the unit with greatest distance from the plane containing the first three units, and so on until the kth unit is the unit furthest from the $(k-2)$ dimensional space spanned by the $(k-1)$ units already found.

The attributes of the units to be formed into groups are specified in a set of variates; these should be placed into a pointer for use as the setting for the DATA parameter. The variates must

not be restricted. The number of groups required is specified by the NGROUPS parameter; this must be less than the number of variates plus 2, and than the number of units plus one. The group allocations that are formed are stored in the factor indicated by the GROUPS parameter. This factor need not be declared in advance but will be formed by the procedure.

Example 6.20.2 uses CLASSIFY to provide an initial classification into four groups four the data in Example 6.19.1a. Notice that the same classification is then obtained by CLUSTER, but the groups are numbered in a different order.

Example 6.20.2

```
  35  CLASSIFY Data; NGROUPS=4; GROUPS=InitCl
  36  CLUSTER  [PRINT=criterion,optimum,initial; DATA=Data;\
  37           START=InitCl] 4

Non-hierarchical clustering
===========================


Sums of squares criterion
-------------------------


Initial classification
----------------------

Number of classes = 4


Class contributions to criterion
--------------------------------

           1           2           3           4
       3.119       1.715       6.150       2.073

Criterion value = 13.05619


Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     1     4     3     3     3     4     3     2     3     1     1     2
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     3     3     3     3     3     2     3     1     2     2     3     3
 Unit    25    26    27    28
Group     3     4     4     1


Optimum classification
----------------------

Number of classes = 4


Class contributions to criterion
--------------------------------

           1           2           3           4
       3.119       1.715       4.394       3.670

Criterion value = 12.89727
```

```
Classification of units
-----------------------

 Unit     1     2     3     4     5     6     7     8     9    10    11    12
Group     1     4     3     3     3     4     4     2     3     1     1     2
 Unit    13    14    15    16    17    18    19    20    21    22    23    24
Group     3     3     3     3     3     2     4     1     2     2     3     3
 Unit    25    26    27    28
Group     3     4     4     1
```

6.20.3 Large data sets: the PCPCLUSTER procedure

PCPCLUSTER procedure

Forms groups of units using the densities of their PCP scores (R.W. Payne).

Options

PRINT = *string tokens*	What to print (cellclusters, density, summary); default summ
PLOT = *string tokens*	What to plot (cellclusters, density, histogram, summary); default cell, dens, hist
NROOTS = *scalars*	Numbers of dimensions to use; default 2
NPARTITIONS = *scalars*	Numbers of partitions in each dimension; default 10
CLUSTERS = *pointer*	Saves variates defining the clusters for each minimum number of points
CELLCLUSTERS = *pointer*	Saves tables containing the clusters of cells for each minimum number of points
DENSITY = *table*	Saves the table of cell densities
SUMMARY = *pointer*	Saves the summary table
MINUNITS = *variate* or *scalar*	Minimum numbers of units within cells at which to form clusters

Parameter

SAVE = *pointer*	Save structure from the PCP analysis to use; default uses the most recent analysis

The PCPCLUSTER procedure provides a way to perform cluster analysis for a large data set. The first simplification is that it reduces the number of attributes of the units by taking scores from a PCP analysis (6.2.1). The SAVE option supplies the save structure from the PCP analysis that is to be used. The default is to use the most recent analysis. The NROOTS parameter specifies the number of dimensions of scores to use; default 2.

The second simplification addresses the space and computing problems that occur when there are large numbers of units. Instead of forming a unit-by-unit similarity matrix, the algorithm, in the PTFCLUSTERS procedure (8.5.1), divides the multi-dimensional space defined by the scores into cells, and forms a density table by tabulating the number of units in each cell. The NPARTITIONS parameter specifies the number of cells to form in each dimension; default 10. The clusters are formed by finding contiguous collections of cells in which the density (or number of units) exceeds thresholds specified by the MINUNITS option. The units in these clusters of cells will be connected to each other in a similar way to the units in a hierarchical cluster analysis. Note, though, that points in sparsely populated parts of the space will not be allocated to any cluster. These units can be thus be identified as unusual or aberrant. The default for MINUNITS is to use a list of values calculated as the maximum density multiplied by 0.8, 0,75, 0.7, 0.65, 0.6, 0.55, 0.5, 0.45, 0.4, 0,35, 0.3, 0.25 and 0.2.

PTFCLUSTERS starts with the first MINUNITS value and finds a cell containing more than that

number of units. This is the starting point for the first cluster. Additional cells are added to the cluster if they are neighbours of cells in the cluster containing more than that minimum number of units. When this cluster is complete, PTFCLUSTERS looks for a cell that is not in the cluster but which contains more than the minimum number of units. This provides the starting point for another cluster. The process continues until all the cells with more than that minimum number of units have been allocated to a cluster. PTFCLUSTERS then takes the next MINUNITS value and expands the clusters to contain neighbours with that smaller minimum number of units, merging clusters if they become neighbours. For each MINUNITS value, PTFCLUSTERS records the number of clusters, the mean number of units within the cells inside and outside the clusters, the mean number for units within the cells just inside and just outside the boundaries, the minimum number for units within cells on the boundaries, and the maximum number for units within cells just outside the boundaries. This summary information should help to assess which MINUNITS value gives the best set of clusters.

The PRINT option controls the printed output, with settings:

cellclusters	shows how the cells are clustered for each minimum number of units,
density	prints the table showing the number of units in each cell,
summary	prints the summary information recorded for each minimum number of units (default).

The PLOT option specifies how the replications are plotted, with settings:

cellclusters	this displays the clustering of the cells for each minimum number of points as a shade plot or as a 3-d graph if there are 2 or 3 dimensions respectively,
density	displays shade plots showing the numbers of units in each pair of dimensions,
histogram	plot a histogram for the numbers of units in the cells,
summary	plots the summary information against the minimum numbers of units.

The default is to plot all of these.

The CLUSTERS option can save a pointer containing details of the clusters of units formed at each MINUNITS value. The clusters have integer numbers, from one upwards. The pointer contains a variate for each MINUNITS value. These contain either cluster numbers, or missing values for units in cells that have not been allocated to any cluster.

The CELLCLUSTERS option can similarly save a pointer containing details of the clusters of cells formed at each MINUNITS value. The pointer contains a table for each MINUNITS value. These contain either a cluster number, or a missing value for cells that have not been allocated to any cluster.

The DENSITY option can save the table containing the number of units within each cell.

The SUMMARY option can save the summary table, in a pointer with elements labelled 'Min. no. points', 'No. clusters', 'Mean inside clusters', 'Mean outside clusters', 'Mean on boundary', 'Mean outside boundary', 'Min. on boundary' and 'Max. outside boundary'.

Example 6.20.3 uses the Iris data set to illustrate PCPCLUSTER. The PCP analysis shows that most of the variation is contained in the first two principal components, and so option NROOTS is set to 2 in the PCPCLUSTER statement in lines 7-9. Figure 6.20.3a shows the units, classified by their species, plotted in these dimensions.

The density table shows that the number of units in each of the 8 × 8 cells range from 12 to zero, and so the default for MINUNITS has been selected as the integers 10 down to 2. The example plots the clusterings of the cells for all of these numbers of units, with the clusters expanding as the numbers decrease. Here we have included only the final plot, in Figure 6,20.3b, and you can see that the first cluster (in black) contains plants from Setosa. The second cluster

(in red) contains a mixture of Versicolor and Virginica, and the third (in green) contains the two more isolated plants of Virginica. This is confirmed by the table at the end of the example.

This shows that PCPCLUSTER has been effective at picking out the cluster of Setosa plants, but not at finding separate clusters for Versicolor and Virginica (whose points overlap). As in all methods of cluster analysis, the successful use of PCPCLUSTER will depend on the quality of the data set to be clustered.

Example 6.20.3

```
   2  SPLOAD     '%data%/Iris.gsh'

Loading Spreadsheet File
------------------------

Catalogue of file iris.gsh

Sheet Title:
Data imported from GenStat Server
 on: 4-Mar-2003 10:08:13

Sheet Type: vector
  Index       Type    Nval   Name
      1    variate     150   Sepal_Length
      2    variate     150   Sepal_Width
      3    variate     150   Petal_Length
      4    variate     150   Petal_Width
      5     factor     150   Species

   3  POINTER    [VALUES=Sepal_Length,Sepal_Width,Petal_Length,Petal_Width] vars
   4  PCP        [PRINT=loadings,roots] vars; SCORES=Scores

4.....................................................................

Principal components analysis
=============================


Latent roots
------------

          1           2           3           4
      630.0        36.2        11.7         3.6


Percentage variation
--------------------

          1           2           3           4
      92.46        5.31        1.71        0.52


Trace
-----

      681.4


Latent vectors (loadings)
-------------------------

                       1           2           3           4
 Sepal_Length     0.36139     0.65659    -0.58203     0.31549
  Sepal_Width    -0.08452     0.73016     0.59791    -0.31972
 Petal_Length     0.85667    -0.17337     0.07624    -0.47984
  Petal_Width     0.35829    -0.07548     0.54583     0.75366

   5  PEN        1,2,3; SYMBOL='circle'; CFILL='match'
   6  DGRAPH     Scores$[*;1]; Scores$[*;2]; PEN=Species
   7  PCPCLUSTER [PRINT=cellclusters,density,summary; PLOT=cellclusters;\
   8             NROOTS=2; NPARTITIONS=8; CLUSTERS=clust]
```

```
Clustering based on densities of points in 2 dimensions
=======================================================


              density
    SCORES[2]     1     2     3     4     5     6     7     8
    SCORES[1]
            1     0     1     5    11    12     5     3     1
            2     0     0     0     2     4     4     1     1
            3     2     1     0     0     0     0     0     0
            4     1     5     6     2     0     0     0     0
            5     1     1     8    11     7     1     0     0
            6     0     2     7    10     9     4     0     0
            7     0     0     0     4     9     3     1     0
            8     0     0     0     0     1     2     0     2


Minimum number of points 10
---------------------------


    SCORES[2]   1   2   3   4   5   6   7   8
    SCORES[1]
            1   *   *   *   1   1   *   *   *
            2   *   *   *   *   *   *   *   *
            3   *   *   *   *   *   *   *   *
            4   *   *   *   *   *   *   *   *
            5   *   *   *   2   *   *   *   *
            6   *   *   *   2   *   *   *   *
            7   *   *   *   *   *   *   *   *
            8   *   *   *   *   *   *   *   *


Minimum number of points 9
---------------------------


    SCORES[2]   1   2   3   4   5   6   7   8
    SCORES[1]
            1   *   *   *   1   1   *   *   *
            2   *   *   *   *   *   *   *   *
            3   *   *   *   *   *   *   *   *
            4   *   *   *   *   *   *   *   *
            5   *   *   *   2   *   *   *   *
            6   *   *   *   2   2   *   *   *
            7   *   *   *   *   2   *   *   *
            8   *   *   *   *   *   *   *   *


Minimum number of points 8
---------------------------


    SCORES[2]   1   2   3   4   5   6   7   8
    SCORES[1]
            1   *   *   *   1   1   *   *   *
            2   *   *   *   *   *   *   *   *
            3   *   *   *   *   *   *   *   *
            4   *   *   *   *   *   *   *   *
            5   *   *   2   2   *   *   *   *
            6   *   *   *   2   2   *   *   *
            7   *   *   *   *   2   *   *   *
            8   *   *   *   *   *   *   *   *
```

```
Minimum number of points 7
--------------------------


    SCORES[2]   1   2   3   4   5   6   7   8
    SCORES[1]
            1   *   *   *   1   1   *   *   *
            2   *   *   *   *   *   *   *   *
            3   *   *   *   *   *   *   *   *
            4   *   *   *   *   *   *   *   *
            5   *   *   2   2   2   *   *   *
            6   *   *   2   2   2   *   *   *
            7   *   *   *   *   2   *   *   *
            8   *   *   *   *   *   *   *   *


Minimum number of points 6
--------------------------


    SCORES[2]   1   2   3   4   5   6   7   8
    SCORES[1]
            1   *   *   *   1   1   *   *   *
            2   *   *   *   *   *   *   *   *
            3   *   *   *   *   *   *   *   *
            4   *   *   2   *   *   *   *   *
            5   *   *   2   2   2   *   *   *
            6   *   *   2   2   2   *   *   *
            7   *   *   *   *   2   *   *   *
            8   *   *   *   *   *   *   *   *


Minimum number of points 5
--------------------------


    SCORES[2]   1   2   3   4   5   6   7   8
    SCORES[1]
            1   *   *   1   1   1   1   *   *
            2   *   *   *   *   *   *   *   *
            3   *   *   *   *   *   *   *   *
            4   *   2   2   *   *   *   *   *
            5   *   *   2   2   2   *   *   *
            6   *   *   2   2   2   *   *   *
            7   *   *   *   *   2   *   *   *
            8   *   *   *   *   *   *   *   *


Minimum number of points 4
--------------------------


    SCORES[2]   1   2   3   4   5   6   7   8
    SCORES[1]
            1   *   *   1   1   1   1   *   *
            2   *   *   *   *   1   1   *   *
            3   *   *   *   *   *   *   *   *
            4   *   2   2   *   *   *   *   *
            5   *   *   2   2   2   *   *   *
            6   *   *   2   2   2   2   *   *
            7   *   *   *   2   2   *   *   *
            8   *   *   *   *   *   *   *   *
```

Minimum number of points 3

SCORES[2]	1	2	3	4	5	6	7	8
SCORES[1]								
1	*	*	1	1	1	1	1	*
2	*	*	*	*	1	1	*	*
3	*	*	*	*	*	*	*	*
4	*	2	2	*	*	*	*	*
5	*	*	2	2	2	*	*	*
6	*	*	2	2	2	2	*	*
7	*	*	*	2	2	2	*	*
8	*	*	*	*	*	*	*	*

Minimum number of points 2

SCORES[2]	1	2	3	4	5	6	7	8
SCORES[1]								
1	*	*	1	1	1	1	1	*
2	*	*	*	1	1	1	*	*
3	2	*	*	*	*	*	*	*
4	*	2	2	2	*	*	*	*
5	*	*	2	2	2	*	*	*
6	*	2	2	2	2	2	*	*
7	*	*	*	2	2	2	*	*
8	*	*	*	*	*	2	*	3

Summary

Minimum number of points	Number of clusters	Mean inside clusters	Mean outside clusters	Mean on boundary	Mean outside boundary	Min on boundary	Max outside boundary
10	2	11.000	1.767	11.000	4.500	10	9
9	2	10.333	1.517	10.333	3.250	9	8
8	2	10.000	1.404	10.000	2.955	8	7
7	2	9.333	1.200	9.333	2.318	7	6
6	2	9.000	1.111	9.000	1.917	6	5
5	2	8.077	0.882	8.077	1.429	5	4
4	2	7.118	0.617	7.118	0.781	4	3
3	2	6.684	0.511	6.684	0.636	3	2
2	3	5.560	0.282	5.292	0.324	2	1

```
  9  CALCULATE  clust2 = MVREPLACE(clust[2]; 0)
 10  GROUPS     clust2; FACTOR=Clusters
 11  TABULATE   [PRINT=Counts; CLASSIFICATION=Species,Clusters]
```

Count				
Clusters	0	1	2	3
Species				
Setosa	4	46	0	0
Versicolor	4	0	46	0
Virginica	3	0	45	2

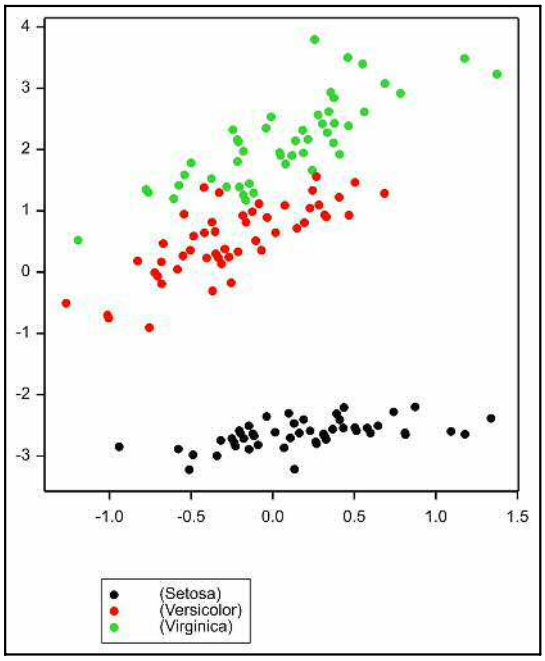


Figure 6.20-3a

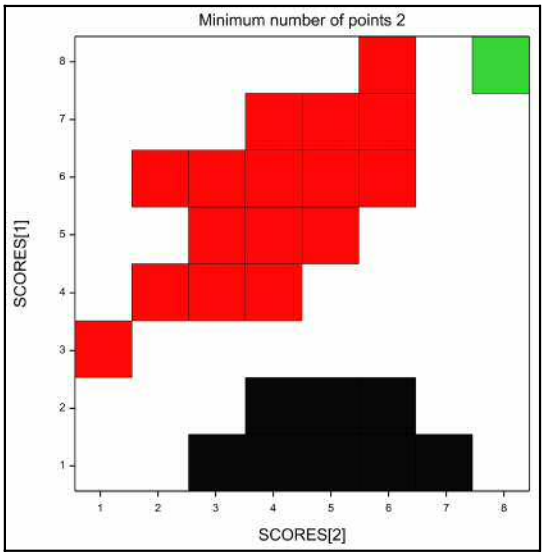


Figure 6.20.3b

6.21 Classification trees

6.21.1 Constructing a classification tree

BCLASSIFICATION procedure

Constructs a classification tree (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (summary, details, indented, bracketed, labelleddiagram, numbereddiagram, graph, monitoring); default * i.e. none
METHOD = *string token*	Selection criterion to use when constructing the tree (Gini, MPI); default Gini
GROUPS = *factor*	Groupings of the individuals in the tree
TREE = *tree*	Saves the tree that has been constructed
NSTOP = *scalar*	Number of individuals in a group at which to stop selecting tests; default 5
ANTIENDCUTFACTOR = *string token*	Adaptive anti-end-cut factor to use (classnumber, reciprocalentropy); default * i.e. none
OWNBSELECT = *string token*	Indicates whether or not your own version of the BSELECT procedure is to be used (yes, no); default no

Parameters

X = *factors* or *variates*	X-variables available for constructing the tree
ORDERED = *string tokens*	Whether factor levels are ordered (yes, no); default no

The starting point for a classification tree is a sample of individuals from several groups. The characteristics of the individuals are described in Genstat by a set of factors or variates which are specified by the X parameter of BCLASSIFICATION. The GROUPS option of BCLASSIFICATION defines the group to which each individual in the sample belongs, and the aim is to be able to identify the groups to which new individuals belong.

The tree progressively splits the individuals into subsets based on their values for the factors or variates. Construction starts at a node known as the *root*, which contains all of the individuals. A factor or variate is chosen to use there that "best" divides the individuals into two subsets. Suppose the X vectors are all factors with two levels: the first subset will then contain the individuals with level 1 of the factor, and the second will contain those with level 2. Also any individual with a missing value for the factor is put into both groups; so you can use a missing value to denote either variable or unknown observations. Factors may have either ordered or unordered levels, according to whether the corresponding value ORDERED parameter is set to yes or no. For example, a factor called Dose with levels 1, 1.5, 2 and 2.5 would usually be treated as having ordered levels, whereas levels labelled 'Morphine', 'Amidone', 'Phenadoxone' and 'Pethidine' of a factor called Drug would be regarded as unordered. For unordered factors, all possible ways of dividing the levels into two sets are tried. With variates or ordered factors with more than 2 levels, a suitable value *p* is found to partition the individuals into those with values less than or greater than *p*. The tree is then extended to contain two new nodes, one for each of the subsets, and factors or variates are selected for use at each of these nodes to subdivide the subsets further.

The effectiveness of the factor or variate to be chosen for each node depends on how the groups are split between the resulting subsets - the aim is to form subsets that is each composed of individuals from the same group. By default, this is assessed using Gini information (see Breiman *et al.*, 1984, Chapter 4) but you can set option METHOD=mpi to use the mean posterior improvement criterion devised by Taylor & Silverman (1993). The ANTIENDCUTFACTOR option allows you to request Taylor & Silverman's adaptive anti-end-cut factors (by default these are not used). The process stops when either no factor or variate provides any additional information, or the subset contains individuals all from the same group, or the subset contains fewer individuals than a limit specified by the NSTOP option (default 5). These nodes where the construction ends are known as *terminal nodes*.

The resulting tree can be saved using the TREE option. Details of the tree can be printed as selected by the PRINT option, with settings:

summary	prints a summary of the properties of the tree;
details	gives detailed information about the nodes of the tree;
bracketed	display as used to represent an identification key in "bracketed" form (printed node by node).
indented	display as used to represent an identification key in "indented" form (printed branch by branch);
labelleddiagram	diagrammatic display including the node labels;
numbereddiagram	diagrammatic display with the nodes labelled by their numbers;
graph	plots the tree using high-resolution graphics.
monitoring	prints information monitoring the construction process.

BCLASSIFICATION stores the information required for printing as part of the tree. If the X vectors are all factors with 2 levels, the labels for the labelled diagram are formed as "*identifier*==n_1", where n_1 is the first level of the factor. The lines of the indented and bracketed forms are formed similarly if the factor has no extra test and no labels. Otherwise, the form is "*xname lname*", where *xname* is the extra text if this has been defined (by the EXTRA parameter of the FACTOR command) or else the identifier of the factor, and *lname* is the label if available or the level if not. If the X vectors include variates or ordered factors with more than two levels

and there is no extra text, the labels are formed as "*identifier*<*p*" and "*identifier*>*p*", where *p* is the value chosen to partition the data for the variate concerned. If there is an extra text for a particular factor or variate, the labels are "*xname* < *p*" and "*xname* > *p*". The style is similar for unordered factors, but here the labels involve the operators `.IN.` and `.NI.` instead of < and >.

Example 6.21.1 uses `BCLASSIFICATION` to construct a classification tree for Fisher's Iris data (also see Examples 2.7.2 and 6.5) and display it in indented form. The first variable to examine in the tree is `Petal_Length`. If this is less than 2.450, the iris specimen is identified as *Setosa*. Otherwise you progress to index 2, and examine `Petal_Width`. So, a specimen of *Versicolor* might be identified by the sequence: 1 `Petal_Length` > 2.450; 2 `Petal_Width` < 1.750; 3 `Petal_Length` > 4.950; 5 `Petal_Width` > 1.550 *Versicolor*. Notice that the same variable can be used several times as the observed characteristics are refined on the way to an identification.

Example 6.21.1

```
  2  " Classification tree for Fisher's Iris Data."
  3  FACTOR   [NVALUES=150; LABELS=!t(Setosa,Versicolor,Virginica);\
  4           VALUES=50(1,2,3)] Species
  5  VARIATE  [NVALUES=150] Sepal_Length,Sepal_Width,Petal_Length,Petal_Width
  6  READ     Sepal_Length,Sepal_Width,Petal_Length,Petal_Width

   Identifier   Minimum      Mean   Maximum    Values   Missing
 Sepal_Length     4.300     5.843     7.900       150         0
  Sepal_Width     2.000     3.057     4.400       150         0
 Petal_Length     1.000     3.758     6.900       150         0
  Petal_Width    0.1000     1.199     2.500       150         0

157  " Form the classification tree."
158  BCLASSIFICATION [PRINT=indented; GROUPS=Species; TREE=Tree]\
159                  Sepal_Length,Sepal_Width,Petal_Length,Petal_Width

1 Petal_Length<2.450 Setosa
1 Petal_Length>2.450 2
 2 Petal_Width<1.750 3
  3 Petal_Length<4.950 4
   4 Petal_Width<1.650 Versicolor
   4 Petal_Width>1.650 Virginica
  3 Petal_Length>4.950 5
   5 Petal_Width<1.550 Virginica
   5 Petal_Width>1.550 Versicolor
 2 Petal_Width>1.750 6
  6 Petal_Length<4.850 Virginica
  6 Petal_Length>4.850 Virginica
```

`BCLASSIFICATION` calls procedure `BCONSTRUCT` (1:4.12.6) to form the tree. This uses a special-purpose procedure `BSELECT`, which is customized specifically to select splits for use in classification trees. You can use your own method of selection by providing your own `BSELECT` and setting option `OWNBSELECT=yes`. In the standard version of `BSELECT`, the `BASSESS` directive (1:4.12.7) is used to assess the potential splits.

6.21.2 Displaying a classification tree

`BCDISPLAY` procedure

Displays a classification tree (R.W. Payne).

Option

`PRINT` = *string tokens*	Controls printed output (`summary`, `details`, `indented`, `bracketed`, `labelleddiagram`, `numbereddiagram`, `graph`); default `*` i.e. none

Parameter	
TREE = *tree*	Tree to be displayed

Further output for a classification tree can be obtained with the BCDISPLAY procedure. The tree is specified by the TREE parameter, and the PRINT option selects the output (with settings that all operate as in the PRINT option of BCLASSIFICATION).

Example 6.21.2 uses BCDISPLAY to print detailed information about the nodes of the tree in Example 6.21.1. This displays the current prediction (i.e. the species number), the numbers of observations at the node, the distributions of the species, and then either the test to be performed or the conclusion reached (i.e. the identified species). Further examples are in Example 6.21.3.

Example 6.21.2

```
 160  BCDISPLAY [PRINT=details] Tree

Details of classification tree: Tree
===================================

 1 Current prediction: 1.000
   Number of observations: 150
       Species  Setosa Versicolor Virginica
   Proportions   0.333   0.333   0.333
   Test: Petal_Length<2.450
   Next nodes: 2 3

  2 Current prediction: 1.000
    Number of observations: 50
        Species  Setosa Versicolor Virginica
    Proportions   1.000   0.000   0.000
    Conclusion: Setosa

  3 Current prediction: 2.000
    Number of observations: 100
        Species  Setosa Versicolor Virginica
    Proportions   0.000   0.500   0.500
    Test: Petal_Width<1.750
    Next nodes: 4 5

   4 Current prediction: 2.000
     Number of observations: 54
         Species  Setosa Versicolor Virginica
     Proportions   0.000   0.907   0.093
     Test: Petal_Length<4.950
     Next nodes: 6 7

    6 Current prediction: 2.000
      Number of observations: 48
          Species  Setosa Versicolor Virginica
      Proportions   0.000   0.979   0.021
      Test: Petal_Width<1.650
      Next nodes: 8 9

     8 Current prediction:  2.000
       Number of observations: 47
           Species  Setosa Versicolor Virginica
       Proportions   0.000   1.000   0.000
       Conclusion: Versicolor

     9 Current prediction:  3.000
       Number of observations: 1
           Species  Setosa Versicolor Virginica
       Proportions   0.000   0.000   1.000
       Conclusion: Virginica

    7 Current prediction: 3.000
      Number of observations: 6
          Species  Setosa Versicolor Virginica
      Proportions   0.000   0.333   0.667
```

```
      Test: Petal_Width<1.550
      Next nodes: 10 11

    10 Current prediction:  3.000
       Number of observations: 3
           Species  Setosa Versicolor Virginica
       Proportions   0.000   0.000   1.000
       Conclusion: Virginica

    11 Current prediction:  2.000
       Number of observations: 3
           Species  Setosa Versicolor Virginica
       Proportions   0.000   0.667   0.333
       Conclusion: Versicolor

  5 Current prediction: 3.000
    Number of observations: 46
        Species  Setosa Versicolor Virginica
    Proportions   0.000   0.022   0.978
    Test: Petal_Length<4.850
    Next nodes: 12 13

  12 Current prediction: 3.000
     Number of observations: 3
         Species  Setosa Versicolor Virginica
     Proportions   0.000   0.333   0.667
     Conclusion: Virginica

  13 Current prediction: 3.000
     Number of observations: 43
         Species  Setosa Versicolor Virginica
     Proportions   0.000   0.000   1.000
     Conclusion: Virginica
```

6.21.3 Pruning a classification tree

Generally the construction of a classification tree will result in *over-fitting*. That is, it will form a tree that keeps selecting factors or variates to subdivide the individuals beyond the point that can be justified statistically. The solution is to prune the tree to remove the uninformative sub-branches. The pruning uses *accuracy* figures, which are stored for each node of the tree. The tree also stores a *prediction* for each node, which corresponds to the group with most individuals at the node. For each node of a classification tree, the accuracy is the number of misclassified individuals at the node, divided by the total number of individuals in the data set. It thus measures the impurity of the subset at that node (how far it is from it from being homogeneous i.e. having individuals all from a single group).

If possible, it is best to use "accuracy" figures that are derived from a different set or sets of data from that which was used to construct the tree. The `BCVALUES` procedure allows these to be calculated, together with new predictions for the nodes of the tree.

`BCVALUES` procedure

Forms values for nodes of a classification tree (R.W. Payne).

Options

`GROUPS` = *factor*	Groupings of the observations in the data set
`TREE` = *tree*	Tree for which predictions and accuracy values are to be formed
`REPLACE` = *string token*	Whether to replace the values stored in the tree (`yes`, `no`); default `no`
`PREDICTION` = *pointer*	New predictions for the nodes of the tree
`ACCURACY` = *pointer*	New accuracy values for the nodes of the tree
`REPLICATION` = *pointer*	New replication tables for the nodes of the tree

Parameter	
X = *factors* or *variates*	Values of the factors or variates used in the tree for the new data set

The TREE option of BCVALUES specifies the tree for which the values are to be formed. The GROUPS option specifies a factor defining the groupings of the observations in the new data set, and the X parameter defines their levels for the factors or variates as used to construct the tree. You can set option REPLACE=yes to use the new values to replace those already stored in the tree. Alternatively, you can use the PREDICTION parameter to save the predictions, in a pointer. This has an element for each node of the tree (and with the same suffix as that node) pointing to a scalar storing the prediction for the node. Similarly, the ACCURACY parameter saves the accuracies, in a pointer to a set of scalars, and the REPLICATION parameter saves the replications of the groups at each node, in a pointer to a set of tables classified by the GROUPS factor. You can use these later to replace the prediction and accuracy values in the original tree by

```
CALCULATE Tree[]['accuracy'] = ACCURACY[]
&         Tree[]['prediction'] = PREDICTION[]
&         Tree[]['replication'] = REPLICATION[]
```

Alternatively, you may want to combine them first with other estimates, for example to form bootstrapped estimates.

The pruning is performed by the BPRUNE procedure, described in 3.9.3 and 1:4.12.8. Example 6.21.3 prunes the tree from Example 6.21.1. There is no independent set of data available here, so the pruning is based on the accuracy values from the original data used to construct the tree. Examining the accuracies of the pruned trees (the column headed RT) suggests that tree 4 is the most appropriate choice. The BCUT directive (1:4.12.4) in line 164 replaces Tree with this tree, Pruned[4], renumbering its nodes at the same time. BCDISPLAY then displays the new tree.

Example 6.21.3

```
 161  " Prune the tree."
 162  BPRUNE [PRINT=table] Tree; NEWTREE=Pruned

Characteristics of the pruned trees
===================================

  Tree          RT   Number of
   no.              terminal
                       nodes

     1      0.0133          7
     2      0.0133          6
     3      0.0200          5
     4      0.0267          4
     5      0.0400          3
     6      0.3333          2
     7      0.6667          1

 163  " Use the 4th tree - renumber nodes."
 164  BCUT [RENUMBER=yes] Pruned[4]; NEWTREE=Tree
 165  " Display the tree."
 166  BCDISPLAY  [PRINT=summary,indented] Tree

Summary of classification tree: Tree
====================================

Number of nodes: 7
Number of terminal nodes: 4
Misclassification rate: 0.027
Variables in the tree: Petal_Length, Petal_Width.
```

```
1 Petal_Length<2.450 Setosa
1 Petal_Length>2.450 2
2 Petal_Width<1.750 3
 3 Petal_Length<4.950 Versicolor
 3 Petal_Length>4.950 Virginica
2 Petal_Width>1.750 Virginica
```

6.21.4 Identification using a classification tree

BCIDENTIFY procedure

Identifies specimens using a classification tree (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (identification, transcript); if PRINT is unset in an interactive run BCIDENTIFY will ask what you want to print, in a batch run the default is iden
TREE = *tree*	Specifies the tree
IDENTIFICATION = *text*	Saves the identification of each specimen
TERMINALNODES = *pointer*	Saves the numbers of the terminal nodes reached by each specimen
PROBABILITIES = *matrix*	Specimen × group matrix giving the probability that the specimens belong to each group
MVINCLUDE = *string token*	Whether to provide identifications for specimens with missing or unavailable values of the x-variables (explanatory); default expl

Parameters

X = *variates* or *factors*	Explanatory variables
VALUES = *scalars*, *variates* or *texts*	Values to use for the explanatory variables; if these are unset for any variable, its existing values are used

BCIDENTIFY identifies specimens using a classification tree. The tree can be specified using the TREE option. Alternatively, BCIDENTIFY will ask you for the identifier of the tree if you do not specify TREE when running interactively.

The characteristics of the specimens can be specified in the variates or factors listed by the X parameter. These must have identical names (and levels) to those used originally to construct the tree. You can use the VALUES parameter to supply new values, if those stored in any of the variates or factors are unsuitable.

If you do not set X when running interactively, BCIDENTIFY will ask you to supply the relevant characteristics in turn, as required by the tree. Otherwise, if an x-variable in the tree is not specified in the X parameter list, its values are assumed to be unavailable (i.e. missing).

By default, when the x-variable required at a node in the tree is unavailable or contains a missing value, BCIDENTIFY will follow all the branches from that node, and form a combined conclusion. You can set option MVINCLUDE=*, if you would prefer the identification to be missing.

The PRINT option controls printed output, with settings:

identification	prints the identifications obtained using the tree;
transcript	prints the observed characteristics when supplied in response to questions in an interactive run.

If you do not set PRINT in an interactive run, BCIDENTIFY will ask what you would like to print. In batch, the default is to print the identifications.

The IDENTIFICATION option allows you to save the identifications (in a text). The TERMINALNODES option allows you to save a pointer, with an element for each specimen, containing the numbers of the terminal nodes reached in the tree to provide its identification. This will be a scalar if the identification was derived from a single node, or a variate if it involved more than one (because several branches have been taken, as the result of a missing x-value). Finally, the PROBABILITIES option can save a specimen-by-group matrix giving the probability that the specimens belong to each group.

Example 6.21.4 identifies six Iris specimens using the pruned tree from Example 6.21.3. Notice that we can use the SETNVALUES option of the READ directive (1:3.1) to redefine the lengths of the data variates (now six values instead of the original 150).

Example 6.21.4

```
167  " Identify 6 new irises."
168  READ [SETNVALUES=yes]\
169       Sepal_Length,Sepal_Width,Petal_Length,Petal_Width

   Identifier   Minimum      Mean   Maximum    Values   Missing
 Sepal_Length     4.600     5.833     6.700         6         0
  Sepal_Width     3.000     3.350     4.000         6         0
 Petal_Length     1.200     3.733     5.700         6         0
  Petal_Width    0.2000     1.333     2.500         6         0

176  BCIDENTIFY [PRINT=*; TREE=Tree; IDENTIFICATION=Identification]\
177             Sepal_Length,Sepal_Width,Petal_Length,Petal_Width
178  PRINT      Sepal_Length,Sepal_Width,Petal_Length,Petal_Width,\
179             Identification; FIELD=4(13),15; DECIMALS=1


Sepal_Length  Sepal_Width Petal_Length  Petal_Width Identification
         4.6          3.4          1.4          0.3         Setosa
         5.8          4.0          1.2          0.2         Setosa
         5.6          3.0          4.1          1.3     Versicolor
         6.1          3.0          4.6          1.4     Versicolor
         6.7          3.3          5.7          2.5      Virginica
         6.2          3.4          5.4          2.3      Virginica
```

6.21.5 Saving information from a classification tree

BCKEEP procedure

Saves information from a classification tree (R.W. Payne).

No options

Parameters

TREE = *trees*	Tree from which the information is to be saved
SUMMARY = *variates*	Saves summary information about each tree
XVARIABLES = *pointers*	Saves the identifiers of the x-variables in each tree

BCKEEP saves information from a classification tree, constructed by the BCLASSIFICATION procedure. The tree can be saved using the TREE option of BCLASSIFICATION, and is specified for BCKEEP using its TREE parameter.

The SUMMARY parameter saves a variate containing summary information. The first element contains the number of nodes, the second contains the number of terminal nodes, and the third contains the misclassification rate.

The XVARIABLES parameter saves a pointer containing the identifiers of the x-variables in the tree.

Example 6.21.5 saves and prints information about the pruned tree from Example 6.21.3.

Example 6.21.5

```
 180  BCKEEP     Tree; SUMMARY=Summary; XVARIABLES=Xvariables
 181  PRINT      Summary & Xvariables

                               Summary

         Number of nodes         7.000
 Number of terminal nodes        4.000
   Misclassification rate        0.027


  Xvariables
Petal_Length
 Petal_Width
```

6.22 Identification

6.22.1 Constructing an identification key

BKEY procedure

Constructs an identification key (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (indented, bracketed, diagram, graph); default * i.e. none
TAXONNAMES = *text*	Names of the taxa in the key; default * uses textual versions of the numbers 1, 2 onwards
GROUPS = *factor*	Groupings of the taxa, if the key is to identify the group of a specimen rather than its taxon
CRITERION = *string token*	Criterion to use to select the character to use at each node of the key (CME, CMV, GME); default GME when GROUPS is set, otherwise CME
PARTIAL = *string token*	Controls whether or not to use partial separation; (yes, no) default no
KEY = *tree*	Saves the key

Parameters

CHARACTER = *factors*	Characters available to construct the key
COST = *scalars*	Cost of each character; default 1

Identification keys provide efficient ways of identifying objects, or *taxa*, whose properties can be described by a set of discrete-valued tests. Many applications are biological. For example, in botanical work, the taxa may be species of plant and the tests may require the observation of characters like the colours of petals or numbers of leaves. Similarly, in microbiology, the tests may involve the ability of an organism to grow in various media. Using a key involves doing a sequence of tests which continues until the unknown specimen can be identified.

The characters that are available for constructing the key are specified, as a list of factors, using the CHARACTER parameter. Each factor has a level for each possible value of the character concerned, and you can insert a missing value for a particular taxon to indicate that its value for the character is either variable or unknown. If an "extra" text has been defined for the factor (using the EXTRA parameter of the FACTOR directive), BKEY will use this when printing the

textual forms of the key instead of the identifier of the factor. (So the characters can be described in the key using any printable symbol, not just those that may be used in identifiers.) The COST parameter allows you to specify a cost for each character. This may be how much it costs to observe or may simply record your own personal preferences between the parameters. By default all the costs are 1. The names of the taxa can be specified in a text using the TAXONNAMES option. If this is omitted, they are simply numbered 1, 2 and so on. If the taxa are classified into groups, BKEY can construct a key to identify the group of a specimen rather than the taxon itself. These groupings can be supplied using the GROUPS factor.

The efficiency of a key is usually measured by its expected cost of identification. To find the optimal key using a particular set of data essentially requires the construction and comparison of all possible keys for the taxa that could be formed with the available tests. This is impracticable even for moderate numbers of tests and taxa. Thus, heuristic algorithms are used which construct the key sequentially, selecting first the test that "best" divides the taxa into sets (where set *k* for test *i* contains all the taxa that can give result *k* to test *i*), then selecting the best test to use with each set, continuing until the sets each contain only one taxon – or until no further separation is possible. The "best" test can be defined using a *selection criterion function* (Gower & Payne 1975). BKEY provides three criteria, which can be selected using the CRITERION option, with settings:

CME	is an estimate of the expected cost of completing the identification from the current point of the key, assuming that test *i* is used and that, below this point, the key is completed optimally (this is the function CMe devised by Payne 1981);
CMV	is a less optimistic estimates, which assumes that the key is completed by simple binary tests (i.e. tests for each of which one particular taxon always gives a positive response and other taxa give negative responses) which corresponds to the function CMv′ of Payne (1981);
GME	is an equivalent version of CMv for the identification of groups of taxa (see Payne, Yarrow & Barnett 1982).

CMe and CMv′ (and two other criteria) were studied by Payne & Thompson (1989), who found that each of them produced the best key for some sets of data. They thus concluded that programs for key construction should allow their users to try several so that they can choose the one that behaves best with any particular set of data.

Usually construction of the key stops when the possible taxa at that point share identical values or have missing values for all the characters. However, if the missing values represent variable rather than unknown values, it may still be worth using these tests in case a specimen of the taxon concerned is obtained that happens to give a level different from the shared level. This *partial* separation can be requested by setting option PARTIAL=yes.

The key can be printed in various formats, as requested by the PRINT option, or it can be saved using the KEY option. The settings of PRINT are:

indented	indented form – prints the key branch by branch;
bracketed	bracketed form – prints the key test by test;
diagram	diagrammatic representation;
graph	plots the key using high resolution graphics.

BKEY stores the information required for printing as part of the tree. The labels for the diagram are formed as "identifier==n_1", where n_1 is the first level of the factor. The lines of the indented and bracketed keys are formed similarly if the factor has no extra test and no labels. Otherwise, the form is "*fname lname*", where *fname* is the extra text if this has been defined (by the EXTRA parameter of the FACTOR command) or else the identifier of the factor, and *lname* is the label if available or the level if not.

Example 6.22.1 uses BKEY to construct a key to the common clincal yeasts.

Example 6.22.1

```
  2  " Construct a key to the common clinical yeasts: data from
 -3    see Payne (1992, COMPSTAT 92 Proceedings in Computational
 -4    Statistics, Volume 2, 239-244. Heidelberg: Physica-Verlag)."
  5  TEXT  [VALUES='Candida albicans','Candida glabrata',\
  6        'Candida parapsilosis','Candida tropicalis',\
  7        'Cryptococcus albidus','Cryptococcus laurentii',\
  8        'Filobasidiella neoformans',\
  9        'Issatchenkia orientalis',\
 10        'Kluyveromyces marxianus',\
 11        'Pichia guilliermondii','Rhodotorula glutinis',\
 12        'Rhodotorula mucilaginosa','Trichosporon beigelii'] Yeasts
 13  FACTOR     [NVALUES=Yeasts; LABELS=!t('-','+')]\
 14        C11; EXTRA='Maltose growth'
 15  &     C18; EXTRA='Lactose growth'
 16  &     C19; EXTRA='Raffinose growth'
 17  &     C36; EXTRA='D-Glucuronate growth'
 18  &      N1; EXTRA='Nitrate growth'
 19  &      V5; EXTRA='Growth w/o Thiamin'
 20  &      O2; EXTRA='0.1% Cycloheximide growth'
 21  &      E5; EXTRA='Splitting cells'
 22  READ [PRINT=errors] C11,C18,C19,C36,N1,V5,O2,E5; FREPRESENTATION=labels
 36  PRINT  [MISSING='V'] C11,C18,C19,C36,N1,V5,O2,E5;\
 37         FIELDWIDTH=4; DECIMALS=0

                         C11 C18 C19 C36  N1  V5  O2  E5
                Yeasts
         Candida albicans   +   -   -   -   -   +   +   -
         Candida glabrata   -   -   -   -   -   -   -   -
     Candida parapsilosis   +   -   -   -   -   +   -   -
       Candida tropicalis   +   -   -   -   -   +   +   -
     Cryptococcus albidus   +   V   V   +   +   -   -   -
   Cryptococcus laurentii   +   +   +   +   -   V   V   -
Filobasidiella neoformans   +   -   V   +   -   -   -   -
  Issatchenkia orientalis   -   -   -   -   -   +   -   -
  Kluyveromyces marxianus   -   V   +   -   -   +   +   -
    Pichia guilliermondii   +   -   +   -   -   +   +   -
     Rhodotorula glutinis   +   -   V   -   +   V   V   -
 Rhodotorula mucilaginosa   V   -   +   -   V   -   V   -
    Trichosporon beigelii   V   +   V   +   -   -   V   +

 38  FACTOR [MODIFY=yes; LABELS=!t(negative,positive)]\
 39         C11,C18,C19,C36,N1,V5,O2,E5
 40  BKEY   [PRINT=bracketed; TAXONNAMES=Yeasts; CRITERION=cme;\
 41         KEY=YeastKey] C11,C18,C19,C36,N1,V5,O2,E5

 1 D-Glucuronate growth negative 2
   D-Glucuronate growth positive 11
 2 Maltose growth negative 3
   Maltose growth positive 6
 3 Raffinose growth negative 4
   Raffinose growth positive 5
 4 Growth w/o Thiamin negative Candida glabrata
   Growth w/o Thiamin positive Issatchenkia orientalis
 5 Growth w/o Thiamin negative Rhodotorula mucilaginosa
   Growth w/o Thiamin positive Kluyveromyces marxianus
 6 Raffinose growth negative 7
   Raffinose growth positive 9
 7 Nitrate growth negative 8
   Nitrate growth positive Rhodotorula glutinis
 8 0.1% Cycloheximide growth negative Candida parapsilosis
   0.1% Cycloheximide growth positive Candida albicans, Candida tropicalis
 9 Nitrate growth negative 10
   Nitrate growth positive Rhodotorula glutinis, Rhodotorula mucilaginosa
10 Growth w/o Thiamin negative Rhodotorula mucilaginosa
   Growth w/o Thiamin positive Pichia guilliermondii
11 Nitrate growth negative 12
   Nitrate growth positive Cryptococcus albidus
12 Lactose growth negative Filobasidiella neoformans
```

```
   Lactose growth positive 13
13 Splitting cells negative Cryptococcus laurentii
   Splitting cells positive Trichosporon beigelii
```

To use the key we start at index 1 and check whether the yeast is able to grow in D-Glucuronate. If the result is negative, the next test is at index 2 (Maltose growth), while a positive result goes to index 11 (Nitrate growth). So a specimen of *Rhodotorula glutinis* would be identified by 1 D-Glucuronate growth negative, 2 Maltose growth positive, 6 Raffinose growth negative and 7 Nitrate growth positive. For more information about yeast identification, see Barnett, Payne & Yarrow (2000).

6.22.2 Displaying an identification key

BKDISPLAY procedure

Displays an identification key (R.W. Payne).

Option

PRINT = *string tokens*	Controls printed output (indented, bracketed, diagram, graph); default * i.e. none

Parameter

KEY = *tree*	Key to be displayed

Further output for a identification key can be obtained with the BKDISPLAY procedure. The tree is specified by the TREE parameter, and the PRINT option selects the output (with settings that all operate as in the PRINT option of BKEY).

Example 6.22.2 shows the indented form of display for the key to the common clinical yeasts constructed in Example 6.22.1.

Example 6.22.2

```
  42  BKDISPLAY [PRINT=indented] YeastKey

1 D-Glucuronate growth negative 2
 2 Maltose growth negative 3
  3 Raffinose growth negative 4
   4 Growth w/o Thiamin negative Candida glabrata
   4 Growth w/o Thiamin positive Issatchenkia orientalis
  3 Raffinose growth positive 5
   5 Growth w/o Thiamin negative Rhodotorula mucilaginosa
   5 Growth w/o Thiamin positive Kluyveromyces marxianus
 2 Maltose growth positive 6
  6 Raffinose growth negative 7
   7 Nitrate growth negative 8
    8 0.1% Cycloheximide growth negative Candida parapsilosis
    8 0.1% Cycloheximide growth positive Candida albicans, Candida tropicalis
   7 Nitrate growth positive Rhodotorula glutinis
  6 Raffinose growth positive 9
   9 Nitrate growth negative 10
   10 Growth w/o Thiamin negative Rhodotorula mucilaginosa
   10 Growth w/o Thiamin positive Pichia guilliermondii
   9 Nitrate growth positive Rhodotorula glutinis, Rhodotorula mucilaginosa
1 D-Glucuronate growth positive 11
11 Nitrate growth negative 12
 12 Lactose growth negative Filobasidiella neoformans
 12 Lactose growth positive 13
  13 Splitting cells negative Cryptococcus laurentii
  13 Splitting cells positive Trichosporon beigelii
```

```
11 Nitrate growth positive Cryptococcus albidus
```

6.22.3 Identification using a key

BKIDENTIFY procedure

Identifies specimens using a key (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (identification, transcript); if PRINT is unset in an interactive BKIDENTIFY will ask what you want to print, in a batch run the default is iden
KEY = *tree*	Specifies the key
IDENTIFICATION = *variate*	Saves the identification of each specimen
TERMINALNODE = *variate*	Saves numbers of the terminal nodes reached by the specimens

Parameter

CHARACTER = *factors*	Character values of the specimens

BKIDENTIFY identifies specimens using an identification key. The key can be supplied using the KEY option. Alternatively, BKIDENTIFY will ask you for the identifier of the key if you do not specify KEY when running interactively.

The characteristics of the specimens can be specified by using the CHARACTER parameter. This must be set to a list of factors with names (and levels) identical to those used originally to construct the key. If you do not set CHARACTER when running interactively, BKIDENTIFY will ask you to examine the characters in turn, as required by the key.

The PRINT option controls printed output, with settings:

identification	prints the identifications obtained using the key;
transcript	prints the observed characteristics when supplied in response to questions in an interactive run.

If you do not set PRINT in an interactive run, BKIDENTIFY will ask what you would like to print. In batch, the default is to print the identifications.

The IDENTIFICATION option allows you to save the identifications (in a text), and the TERMINALNODE option allows you to save a variate containing the numbers of the terminal nodes that the specimens reached in the key.

Example 6.22.3 uses BKIDENTIFY to see how well the key constructed in Example 6.22.1 identifies the common clinical yeasts. Notice that the characters available for constructing the key do not enable *Candida albicans* to be distinguished from *Candid tropicalis*. No identification can be made if a specimen has a missing entry recorded for one of tests in the key. This is the situation with *Rhodotorula glutinis* for Raffinose growth at index 6, and with *Rhodotorula mucilaginosa* for Maltose growth at index 2. (When constructing the key, a missing value is used to record a variable entry, but during identification it is taken to mean that the test result is unavailable.)

For more information the identification of these yeasts, see Barnett, Payne & Yarrow (2000).

Example 6.22.3

```
43  BKIDENTIFY [PRINT=*; KEY=YeastKey; IDENTIFICATION=Identification]\
44             C11,C18,C19,C36,N1,V5,O2,E5
45  PRINT      Yeasts,Identification; JUST=left
```

```
Yeasts                    Identification
Candida albicans          Candida albicans,Candida tropicalis
Candida glabrata          Candida glabrata
Candida parapsilosis      Candida parapsilosis
Candida tropicalis        Candida albicans,Candida tropicalis
Cryptococcus albidus      Cryptococcus albidus
Cryptococcus laurentii    Cryptococcus laurentii
Filobasidiella neoformans Filobasidiella neoformans
Issatchenkia orientalis   Issatchenkia orientalis
Kluyveromyces marxianus   Kluyveromyces marxianus
Pichia guilliermondii     Pichia guilliermondii
Rhodotorula glutinis      *
Rhodotorula mucilaginosa  *
Trichosporon beigelii     Trichosporon beigelii
```

6.22.4 Saving information from a key

BKKEEP procedure

Saves information from an identification key (R.W. Payne).

No options

Parameters

KEY = *trees*	Identification key from which the information is to be saved
SUMMARY = *variates*	Saves summary information about each key
CHARACTERS = *pointers*	Saves the identifiers of the characters in each key

BKKEEP saves information from an identification key, constructed by the BKEY procedure. The key can be saved using the KEY option of BKEY, and is specified for BKKEEP using its KEY parameter. The SUMMARY parameter saves a variate containing summary information. The first element contains the number of nodes, and the second contains the number of terminal nodes. The CHARACTERS parameter saves a pointer containing the identifiers of the characters in the key.

Example 6.22.3 uses BKKEEP to save and print information about the key constructed in Example 6.22.1.

Example 6.22.4

```
 46  BKKEEP     YeastKey; SUMMARY=Summary; CHARACTERS=Characters
 47  PRINT      Summary & Characters

                              Summary

         Number of nodes       27.00
Number of terminal nodes       14.00


 Characters
        C36
        C11
         N1
        C19
        C18
         V5
         E5
         O2
```

6.22.5 Interactive identification

`IDENTIFY` procedure

Identifies an unknown specimen from a defined set of objects (R.W. Payne).

Options

`PRINT` = *string tokens*	Controls printed output (`identification`, `transcript`); default `iden`, `tran`
`METHOD` = *string token*	Type of run (`batch`, `interactive`); if this is not set `IDENTIFY` checks whether the run of Genstat itself is batch or interactive
`TAXA` = *text* or *factor*	Names for the taxa (i.e. the objects); default uses the positive integers 1, 2...
`NMISTAKE` = *scalar*	Number of mistakes to allow for; default 0
`IDENTIFICATION` = *text*	Saves the names of the taxa that are identified; default `*` i.e. not saved
`DIFFERENCES` = *variate*	Saves the number of differences between the observed character states and those that can be displayed by each taxon; default `*` i.e. not saved

Parameters

`CHARACTER` = *factors* or *tables*	Define the characteristics of the taxa; must be set
`OBSERVATION` = *scalars* or *texts*	Can define an observation for each character; default `*` i.e. none
`COST` = *scalars*	Costs of observing each character; default 1

As an alternative to constructing and using an identification key, you can use the `IDENTIFY` procedure to identify an unknown specimens interactively. The specimen is identified by comparing observations that you specify for the specimen against the characteristics that you have defined for the full set of taxa that may occur. Each character is assumed to have a set of distinct possible *states*, which are represented by the levels of a factor.

So, like `BKEY` (6.22.1), `IDENTIFY` assumes that the values of the characters are discrete. Often the characters will be binary, representing the presence or absence of some attribute. Alternatively, they may involve counts, for example of numbers of leaves or petals. If you want to use continuous variables, you will need to classify the values into ranges (for example using the `GROUPS` directive).

Generally, the properties of the taxa with respect to each character can be defined by a factor, whose levels represent the range of values that can occur for the character. If a taxon only ever displays one state of the character (i.e. if it has a *fixed* response), the unit of the factor corresponding to that taxon should be set to the relevant level. Conversely, if different specimens of the taxon can display different states of the character (i.e. it has a *variable* response), the unit should contain a missing value.

Representing the properties for a character by a factor assumes that, if a taxon is variable, any of the states of the character may occur. Information will thus be lost for taxa that can show several, but not all, of the states of a character. Thus `IDENTIFY` allows an alternative representation, which uses a table classified by two factors: one representing the states of the factor, and another representing the taxa. So, there is a the table has a row for each taxon. This contains a zero value for the states that the taxon cannot display, and a non-zero value (usually one) for those that it can display. The same convention is used with the `IRREDUNDANT` directive; see 6.11.6 for an example.

The factors and/or tables defining the properties of the taxa must be listed using the

CHARACTER parameter. If any of these is a table, the TAXA option must be set to the factor used to represent the taxa there. The levels of the factor (or its labels if present) then supply names for the taxa that are used in the output. If there are no CHARACTER tables, TAXA can be set to a text containing the taxon names instead. If TAXA is not set, IDENTIFY uses the integers 1, 2... The COST parameter can be used to supply a list of scalars indicating the cost of observing each character; if this is not set, the costs are all assumed to be equal to one.

The METHOD option defines whether IDENTIFY operates interactively, or in batch mode. If this is not set, IDENTIFY checks whether Genstat itself is running interactively or in batch. In an interactive run, IDENTIFY displays menus to guide you through to achieving an identification. The main menu allows you to select any one of the following actions.

1) list potential identifications – IDENTIFY compares the observations that you specify for the specimen against the characteristics that you have defined for the taxa. It then lists the taxa (if any) that can display all of the character states that you have observed, then those that can display all except one, all except two, and so on. The list is displayed in sections, and you can terminate it at any time.
2) select and observe a character – IDENTIFY assesses the characters using the selection criterion function *CMV'* of Payne (1981), and lists them in order of their effectiveness. Alongside each one it prints an estimate of the number (of cost if the COST parameter has been set) of the characters that must be observed to complete the identification, assuming that this one is observed next. After you have chosen a character, it displays another menu for you to specify the state that you have observed.
3) specify an observed character (find in list) – IDENTIFY lists the characters so that you can indicate which one you wish to observe next. After you have chosen a character, it displays another menu for you to specify the state that you have observed.
4) specify an observed character (type name) – IDENTIFY asks you to type the name of the character that you wish to observe next. If you type just the initial part of the name, IDENTIFY will give you a list of all the characters whose names begin like that. After you have chosen a character, it displays another menu for you to specify the state that you have observed.
5) modify an observation – IDENTIFY lists the characters that have already been observed to allow you to choose which you want to modify. After you have chosen a character, it displays another menu for you to specify the revised value.
6) display observations – IDENTIFY displays the characters that have already been observed.
7) list the characteristics of a taxon – IDENTIFY lists the taxa so that you can indicate the one whose characteristics you wish to display.
8) show differences between 2 taxa – IDENTIFY lists the taxa so that you can indicate the two that you want to compare. IDENTIFY then lists the characters that differ between them.
9) set configuration options – IDENTIFY generates a menu allowing you to set various configuration options. Firstly, you can ask IDENTIFY to take account of a specified number of mistakes in your observations. It will then up to this number of differences between your observations and the characteristics of each taxon when suggesting which character to observe next, or when making an identification. The initial setting for the number of mistakes is set by the NMISTAKE option, with a default of zero (i.e. none). You can also control whether or not to produce a transcription of your activities and whether or not to print the identification obtained at the end of your run. The initial settings for these two aspects are set by the PRINT option; by default both are printed.
10) start a new identification (clearing observed characters) – IDENTIFY clears the current observations so that you can start again.
11) save/print identification and then exit – IDENTIFY prints and saves the identification, as requested, and then stops.

The identification is saved by setting the IDENTIFICATION option to a text to contain the names of all the taxa that can display the observed character states, allowing for any requested number of mistakes. You can also set the DIFFERENCES option to a variate to contain the number of differences between the observed character states and those that can be displayed by each taxon.

For a batch run, you should use the OBSERVATION parameter to supply values for all the characters that you have observed. These can be either scalars (referring to levels of the factor) or one-line texts (referring to its labels), or a missing value to denote characters that have not been observed. This parameter can be also used in an interactive run, as an alternative to supplying the observations through the menus.

6.22.6 Irredundant test sets

IRREDUNDANT directive

Forms irredundant test sets for the efficient identification of a set of objects.

Options

PRINT = *string tokens*	Controls printed output (numbers, diagram, notdistinguished, messages); default numb, diag, notd, mess
BESTSET = *pointer*	Saves the best set
SETS = *matrix*	Saves details of the available sets
NOTDISTINGUISHED = *matrix*	Saves details of the objects that cannot be distinguished
METHOD = *string token*	Algorithm to use (exact, sequential); default exac
TAXONNAMES = *text* or *variate* or *factor*	Defines labels for the objects (or *taxa*) to be identified; default uses the unit labels vector of the CHARACTER factors
GROUPS = *factor*	Defines groupings of the objects so that the sets are constructed to distinguish only between the objects that belong to different groups; default constructs sets to distinguish between individual objects
OBJECT = *scalar* or *text*	If this is specified, sets are constructed just to distinguish the specified object (or *taxon*) from the other objects
NDISTINCTIONS = *scalar*	Number of factors required in each set to distinguish between each pair of objects; default 1
MAXPREFERENCE = *scalar*	Maximum preference of the factors to be included in the sets
MAXSIZE = *scalar*	Limit on number of factors in a set (sets containing more than this are discarded); default * i.e. none
NPRINT = *scalar*	Number of sets to print (a positive number specifies the number to print, a negative number sets a tolerance on the difference between the sizes of the sets printed and the size of the best set); default * prints them all
NSAVE = *scalar*	Number of sets to save in the SETS matrix; default * saves them all
LIMSETS = *variate*	Variate containing two numbers n_1 and n_2, if this is specified then every time that there are more than n_1 sets under construction using the exact method, the sets are arranged in order of increasing size and all sets

	containing more factors than set n_2 are deleted
DISTINCTIONS = *string token*	Whether or not to store the distinctions or recalculate them at every stage in the exact algorithm (`store`, `calculate`); default `stor`
CRITERION = *string token*	Function to be use to select factors by the sequential method (`ndistinctions`, `weightedndistinctions`); default `ndis`
MAXCYCLE = *scalar*	Maximum number of improvement cycles to perform during the sequential method; default 20
EQUIVALENCE = *scalar*	Value for determining equivalence of the selection criteria of tests selected during the sequential method

Parameters

CHARACTER = *identifiers*	Factors, and/or tables classified by a single factor, defining the properties of the objects to to be identified
COST = *scalars*	Cost associated with each factor; default 1
PREFERENCE = *scalar*	Preference rating for each factor (1 representing most preferred etc.); default 1
VARIABLE = *scalar* or *text*	Factor level used to represent variable information; default is to use a missing value
INAPPLICABLE = *scalar* or *text*	Factor level used to indicate that the information provided by that factor is inapplicable for a particular object

Like the other commands described in this Section, the `IRREDUNDANT` directive is relevant when you have a set of objects (or *taxa*) whose properties can be described by a set of discrete-valued tests. `IRREDUNDANT` helps you to select an efficient set of tests that can be applied, in a batch, to identify any unknown specimen of any of the objects. (The batch of tests is then often printed as a diagnostic table; see Payne & Preece 1980.) As all the tests in the set are to be used for every identification, it is best for the set to contain as few tests as possible. So there should thus be no *redundant* tests: these are tests that can be deleted from the set without causing any object (or *taxon*) to be no longer identifiable. Sets of tests that contain no redundant tests are known as *irredundant*.

Consider taxa A, B, C and D, whose responses to tests 1-5 are shown in the table below. The symbol "+", for example in the entry for taxon A and test 1, indicates that all specimens of taxon A will always give a positive result to test 1, the symbol "–" for taxon D with test 1 indicates a negative result, and the symbol "v" for taxon B with test 3 indicates that some specimens of D will give a positive result to test 3 but others will give a negative result.

	Test				
Taxon	1	2	3	4	5
A	+	+	+	–	+
B	+	–	v	–	–
C	+	–	–	+	+
D	–	+	+	+	–

As Example 6.22.6 shows, the table contains several irredundant sets, one of which contains the tests 1, 3 and 5. (If, for example, test 3 is deleted from this set, taxa A and C can no longer be distinguished). Another set contains tests 2 and 4. So, the irredundant sets can be of different sizes. The optimum set will often be defined to be one containing a minimum number of tests. Alternatively, if the test cost different amounts to apply, the optimum set may be one with

minimum total cost. However, whichever of these situations applies, the optimum set will be irredundant, as otherwise a better set could be obtained by deleting a redundant test.

The characteristics of the taxa and tests are specified using the CHARACTER parameter. In the simplest situation, this provides a list of factors, one for each test (or character), as with the BKEY procedure (6.22.1). The factors contain a unit for each taxon, and the level stored in that unit indicates how the taxon can respond to the test.

Example 6.22.6

```
   2  TEXT   [VALUES=A,B,C,D] Taxa
   3  FACTOR [NVALUES=4; LEVELS=2] T1,T2,T3,T4,T5
   4  READ   [PRINT=data,errors] T1,T2,T3,T4,T5

   5  2  2  2  1  2
   6  2  1  *  1  1
   7  2  1  1  2  2
   8  1  2  2  2  1  :
   9  IRREDUNDANT [TAXONNAMES=Taxa] T1,T2,T3,T4,T5

Irredundant test sets
=====================


Pairs of objects that cannot be distinguished
---------------------------------------------

There are no pairs of objects that cannot be distinguished


Factors in the sets
-------------------

 1)  T1
 2)  T2
 3)  T3
 4)  T4
 5)  T5

 1) 2 tests: 2 4
 2) 2 tests: 2 5
 3) 2 tests: 4 5
 4) 3 tests: 1 3 5

Best irredundant set is number 1.


Diagram of the composition of the sets
--------------------------------------

              1 2 3 4

           T1 - - - 1
           T2 1 1 - -
           T3 - - - 1
           T4 1 - 1 -
           T5 - 1 1 1
```

Level 1 of the factors T1 - T5 represents a negative response, and level 2 represents a positive response (see lines 5-8). The variable response of taxon B with test 3 is represented by a missing value, but you can use the VARIABLE parameter to use a particular level of the factor instead. There may be tests that are not applicable to some of the taxa. For example, when identifying insects, tests concerning colours of wings are not applicable to those that do not fly! The level to be used to indicate these responses is specified by the INAPPLICABLE parameter. Costs for the test can be specified by the COST parameter; by default, these are all taken to be one. Names for the taxa can be supplied, in either a text or a variate or a factor, using the TAXONNAMES

parameter. If this is not set, IRREDUNDANT uses the unit labels of the CHARACTER factors if any have been defined (see the FACTOR directive), or otherwise the integers 1, 2 upwards.

The use of the VARIABLE option works well with responses that are completely variable i.e. where the specimens of the taxon may give any of the available results to the test. However, when the tests have more than two possible results, there may be taxa that can give some but not all of the available results to a test. As with the IDENTIFY procedure (6.22.5), The responses to a test like this should be specified by a two-way table classified by one factor with a level for each possible result, and another with a level for each taxon. The table should then contain a positive (e.g. one) whererever the taxon concerned can deliver the result, and zero elsewhere. For example suppose that, with test T6, taxon A, C and D always give result 1, 2 and 3 respectively, but taxon B can give either or results 2 or 3. The relevant table could then be constructed and used as follows:

```
FACTOR [LABELS=Taxa] Taxfact
FACTOR [LEVELS=3] T6fact
TABLE  [CLASSIFICATION=T6fact,Taxfact; VALUES=\
       " level 1:"  1, 0, 0, 0, \
       " level 2:"  0, 1, 1, 0, \
       " level 3:"  0, 1, 0, 1 ] T6tab
IRREDUNDANT [TAXONNAMES=Taxfact] T1,T2,T3,T4,T5,T6tab
```

The standard irredundant sets contain at least one test to distinguish each pair of taxa. However, to guard against mistakes in either the original data on during the subsequent use of the set, you can set the NDISTINCTIONS option to ask for the set to include a larger number of tests able to distinguish each pair. Another refinement is that you can set the GROUPS option to a factor defining groupings of the taxa. The sets are then formed to distinguish only pairs of taxa that belong to different groups. Alternatively, you may want a set of tests to either confirm whether or not the specimen belongs to one particular taxon. The taxon of interest should then be indicated by setting the OBJECT option to the number of the taxon or, if textual taxon names have been defined, to the text identifying the taxon. Finally, if you set both GROUPS and OBJECT, the sets will be constructed to confirm whether or not a specimen belongs to a particular group.

IRREDUNDANT takes account of restrictions on any of the CHARACTER factors or on TAXONNAMES or GROUPS.

Two methods are provided for constructing the irredundant sets. The default is to use an exact method (Payne 1991) which constructs all possible sets for the dataset concerned. However, with some datasets, there may be too many sets to construct them all. If you run out of workspace (or time), you can use the LIMSETS to specify a variate containing two integers n_1 and n_2. Then whenever there are more than n_1 sets under construction, the sets are arranged in order of increasing size and all sets containing more factors than set n_2 are deleted. The method then no longer guarantees to find all the irredundant sets containing the fewest number of tests or with the minimum total costs, but in the situations where this modification is needed, it is very unlikely that it will fail to find any of them.

Alternatively, you can set option METHOD=sequential to use a sequential algorithm (Payne & Preece 1980, Section 6.6). This does not guarantee to find a set with minimum size or cost, but it takes much less computing time and should always should produce a satisfactory set. The sequential method starts with an initial set containing all the essential tests, and then adds additional tests, one at a time, until each pair of taxa can be distinguished. (A test is *essential* if it is the only test which can distinguish between a particular pair of taxa.) The criterion for selecting the test to add to the set at each stage is usually the number of pairs of taxa that the test distinguishes, of those pairs not distinguished by tests already in the set. If costs have been defined, this number of pairs is divided by the cost of the test concerned.

Setting option CRITERION=weighted uses a refinement, suggested by Barnett *et al*. (1983), which weights each pair of taxa by the reciprocal of the number of tests that can distinguish between them. The criterion is then the maximum *weighted* number of pairs of taxa (divided by

the cost of the test, if defined). This causes tests that distinguish "difficult" pairs of taxa (those with nearly identical characteristics) to be selected earlier during the construction of the set, and thus tends to generate smaller sets. You can set a preference rating for each test using the PREFERENCE parameter; the most-preferred tests should have ratings of one, and less-preferred tests should have ratings of two and upwards. Then, if at any stage there is then more than one test with the best criterion value, the most-preferred test is selected. If these preferances are especially important, you may also also want to set the EQUIVALENCE option to a scalar, *e* say. Then all tests whose criterion values are within *e* of the current maximum are regarded as equivalent, and the best test is selected from within these tests according to the preferences.

The main disadvantage of most sequential methods is that they produce only a single set of tests. In order to allow a choice of sets and as a way of improving the original set, IRREDUNDANT can run through a sequence of cycles. In each of these, the tests in the best set are deleted in turn, further tests are selected to separate the pairs of taxa distinguished only by the deleted test, and any redundant tests are deleted. If no improvement is achieved, all the non-essential tests are deleted, and the set is reformed without using those tests. The process can be then repeated until no improvements are being achieved of until the number of cycles exceeds the setting of the MAXCYCLE option (default 20).

Printed output is controlled by the PRINT option, with settings:

numbers	numbers of the tests in the sets,
diagram	table showing the contents of the sets,
notdistinguished	lists of pairs of taxa that cannot be distinguished,
messages	messages for example when the number of sets has been reduced as requested by the LIMSETS option, or concerning pairs of taxa than cannot be distinguished.

The default is PRINT=numb,diag,notd,mess.

The best set can be saved using the BESTSET option, as a pointer containing the relevant factors. The SETS option can save a matrix, with a row for each set and a column for each test, representing all the sets that have been formed. In each row the matrix generally stores the number one in the columns corresponding to the tests in that set, and zero elsewhere. However, if the sets have been constructed to confirm the identification of a single taxon, the matrix contains more informative numbers than one. So, down each column wherever one would be stored, it instead stores the level given by the taxon for the factor corresponding to the test concerned. The NOTDISTINGUISHED option can save information about the pairs of taxa that cannot be distinguished, or that are distinguished by less than NDISTINCTIONS tests. The matrix has a row for each such pair of taxa, and three columns. Columns 1 and 2 contain the numbers of the taxa in the pair, and column 3 contains the number of tests that can distinguish them.

7 Analysis of time series

A *time series* in Genstat is a sequence of observations at equally spaced points in time. Each time series is stored in a variate for which the unit number indexes the time points. Genstat cannot deal explicitly with unequal spacing in time. So if you have such a sequence, you will need to do some form of adjustment or interpolation before using the methods described here. Alternatively, you could try the facilities for modelling repeated measurements by REML (5.4) or those for regression and nonlinear models with correlated errors (8.1.6). Genstat will handle missing values in time series, but these should not represent more than a small fraction of the data. Usually you will want to describe or model the structure of a series. You can do this without reference to any other variable than the series itself, by examining the relationship between successive measurements. You can also treat a time series as a response variable, which is related to present and past values of explanatory variables that are also time series. *Forecasts* of future values of time series can be derived from these relationships. You can use *filters* to modify time series, for example to smooth them, or to remove trends.

Most of this chapter describes how to analyse time series by the methods advocated by Box & Jenkins (1970). They recommend a modelling procedure involving three stages: model selection (a term used here in preference to that used by Box and Jenkins, which is "identification"), model estimation and model checking (used here in preference to "verification"). The facilities described in this chapter also provide the basic techniques for spectral analysis, as described by Bloomfield (1976).

Section 7.1 describes how to derive sample statistics from time series, such as *autocorrelations*: these help you select time-series models. Section 7.2 shows how to calculate the *Fourier transform*, which can be useful for revealing cyclical behaviour; it also describes how to construct the *periodogram*, often called the sample spectrum. Section 7.3 describes *autoregressive integrated moving-average* (ARIMA) models, using the notation of Box and Jenkins. It also describes how these are used as *univariate models*: that is, models to describe the behaviour of a single series. There are directives to let you save the results of estimation, so that you can check models. Once a model has been fitted, you can make forecasts of the future values of the series. Section 7.4 shows how to fit regression models between time series, using an ARIMA model to represent correlated errors. Section 7.5 shows how to extend this to general *transfer-functions* between series: again you can estimate, check and forecast. Section 7.6 covers the *filtering* of time series by transfer-function models, as used for example in exponential smoothing or seasonal adjustment. Filtering can also be done by ARIMA models, as used in *pre-whitening*. Section 7.7 presents some ways of displaying the properties of the fitted models, such as the theoretical autocorrelations of ARIMA models.

The index for a time-series variate goes from *1* to *N*, *N* being the number of observations. However for defining Fourier transformations, the conventional index is *t=0...(N−1)*, and we adhere to this too.

The information in this chapter is grouped mainly by type of analysis, rather than by command. So first we summarize the commands, giving references to the sections where they are described. Details of those not covered here can be found in the Genstat Reference Manual. The directive `CORRELATE` provides sample correlation functions:

`CORRELATE`	forms correlations between variates, autocorrelations of variates, and lagged cross-correlations between variates (7.1.1)

The analysis of Box-Jenkins models is specified by several directives:

`TSM`	defines Box-Jenkins models (7.3.2, 7.5.1)
`FTSM`	forms preliminary estimates of parameters in time-series models (7.7.1)

TRANSFERFUNCTION	specifies input series and transfer-function models for subsequent estimation of a model for an output series (7.4.1, 7.5.2)
TFIT	estimates parameters in Box-Jenkins models for time series (7.3.3, 7.4.2, 7.5.3)

Information can be saved in Genstat data structures, or further output can be produced:

TDISPLAY	displays further output after an analysis by TFIT (7.3.5)
TKEEP	saves results after TFIT (7.3.6, 7.5.4)
TFORECAST	forecasts future values (7.3.7, 7.4.3, 7.5.5)
TSUMMARIZE	displays time series model characteristics (7.7.3)

You can filter a time series or perform spectral analysis, using the TFILTER and FOURIER directives, or perform Kalman filtering with the KALMAN procedure.

TFILTER	filters time series by time-series models (7.6.1)
FOURIER	calculates cosine or Fourier transforms of a real or complex series (7.2.1)
KALMAN	calculates estimates from the Kalman filter
DKALMAN	plots results from an analysis by KALMAN

The Genstat procedure library contains procedures which use the directives described in this chapter, together with graphical presentation of the results, to extend the facilities and to enable standard analyses to be carried out more conveniently.

BJESTIMATE	fits an ARIMA model, with forecasts and residual checks (7.3.1)
BJFORECAST	plots forecasts of a time series using a previously fitted ARIMA (7.3.8)
BJIDENTIFY	displays time series statistics useful for ARIMA model selection (7.1.3)
DFOURIER	performs a harmonic analysis of a univariate time series (7.2.7)
MCROSSPECTRUM	performs a spectral analysis of a multiple time series (7.2.8)
PERIODTEST	gives periodogram-based tests for white noise in time series
PREWHITEN	filters a time series before spectral analysis
REPPERIODOGRAM	gives periodogram-based analyses for replicated time series
SMOOTHSPECTRUM	forms smoothed spectrum estimates for univariate time series (7.2.6)
TVARMA	fits a vector autoregressive moving average (VARMA) model
TVFORECAST	forecasts future values from a vector autoregressive moving average (VARMA) model
TVGRAPH	plots a vector autoregressive moving average (VARMA) model

In Genstat *for Windows*, ARIMA modelling can be done using the ARIMA Model Fitting menu, while the Time Series - Data Exploration menu produces useful summaries and plots using CORRELATE and BJIDENTIFY.

7.1 Correlation

7.1.1 The CORRELATE directive

CORRELATE directive

Forms correlations between variates, autocorrelations of variates, and lagged cross-correlations between variates.

Options

PRINT = *string tokens*	What to print (correlations, autocorrelations, partialcorrelations, crosscorrelations); default *
GRAPH = *string tokens*	What to display with graphs (autocorrelations, partialcorrelations, crosscorrelations); default *
MAXLAG = *scalar*	Maximum lag for results; default * i.e. value inferred from variates to save results
CORRELATIONS = *symmetric matrix*	Stores the correlations between the variates specified by the SERIES parameter

Parameters

SERIES = *variates*	Variates from which to form correlations
LAGGEDSERIES = *variates*	Series to be lagged to form crosscorrelations with first series
AUTOCORRELATIONS = *variates*	To save autocorrelations, or to provide them to form partial autocorrelations if SERIES=*
PARTIALCORRELATIONS = *variates*	To save partial autocorrelations
CROSSCORRELATIONS = *variates*	To save crosscorrelations
TESTSTATISTIC = *scalars*	To save test statistics
VARIANCES = *variates*	To save prediction error variances
COEFFICIENTS = *variates* or *matrices*	To save prediction coefficients: in a variate to keep only those for the maximum lag, or in a matrix to keep the coefficients for all lags up to the maximum

The most straightforward use of the CORRELATE directive is to calculate correlation coefficients between a set of variates. In Example 7.1.1, the PRINT option is set to correlations to display the correlations as a lower-triangular matrix.

Example 7.1.1

```
  2  " Display correlations of 5 time series of United Kingdom Pig Production
 -3    from 'Data. A Collection of Problems from Many Fields for the Student
 -4    & Research Worker', D.F.Andrews & A.M.Herzberg, Springer-Verlag 1985."
  5  OPEN 'UKpig.dat';CHANNEL=3
  6  READ [CHANNEL=3] Year,Quarter,Gilts,Profit,Slaughter,Cleanpig,Herdsize

  Identifier   Minimum      Mean   Maximum    Values   Missing
        Year      1967      1973      1978        48         0
     Quarter     1.000     2.500     4.000        48         0
       Gilts     77.00     111.2     140.0        48         0
      Profit     5.049     7.064     8.639        48         0
   Slaughter     7.870     10.58     14.00        48         0
```

```
      Cleanpig      2540      3085      3501        48         0
      Herdsize     703.0     803.1     922.0        48         0

   7  CLOSE 3
   8  CORRELATE [PRINT=correlations] Gilts,Profit,Slaughter,Cleanpig,Herdsize

Correlation matrix
------------------
        Gilts     1.000
       Profit     0.409     1.000
    Slaughter    -0.522    -0.611     1.000
     Cleanpig    -0.252    -0.396     0.428     1.000
     Herdsize     0.558     0.002    -0.127     0.592     1.000
                  Gilts    Profit Slaughter  Cleanpig  Herdsize
```

Example 7.1.1 prints the correlations between five time series of quarterly indicators of the pig market. The correlations can be saved in a symmetric matrix using the `CORRELATIONS` option. Note that, if there are missing values, `CORRELATE` uses only those units where none of the variates is missing.

These correlations measure only the simultaneous relationship between the series. More useful are the autocorrelations of the series, that is the correlations between values in the series lagged by particular time intervals. The set of autocorrelations for all possible lags is the *autocorrelation function*. You can derive the *partial autocorrelation function* from these. To look at the relationship between two series, you should use the *cross-correlation function* between one series and the other lagged by the various intervals.

The ways of interpreting the correlation functions are described by many standard books about time series. The books by Anderson (1976) and Nelson (1973) are introductory texts, but do not cover the whole range of models covered in this chapter. The book by Box & Jenkins (1970) gives a full description.

7.1.2 Autocorrelation

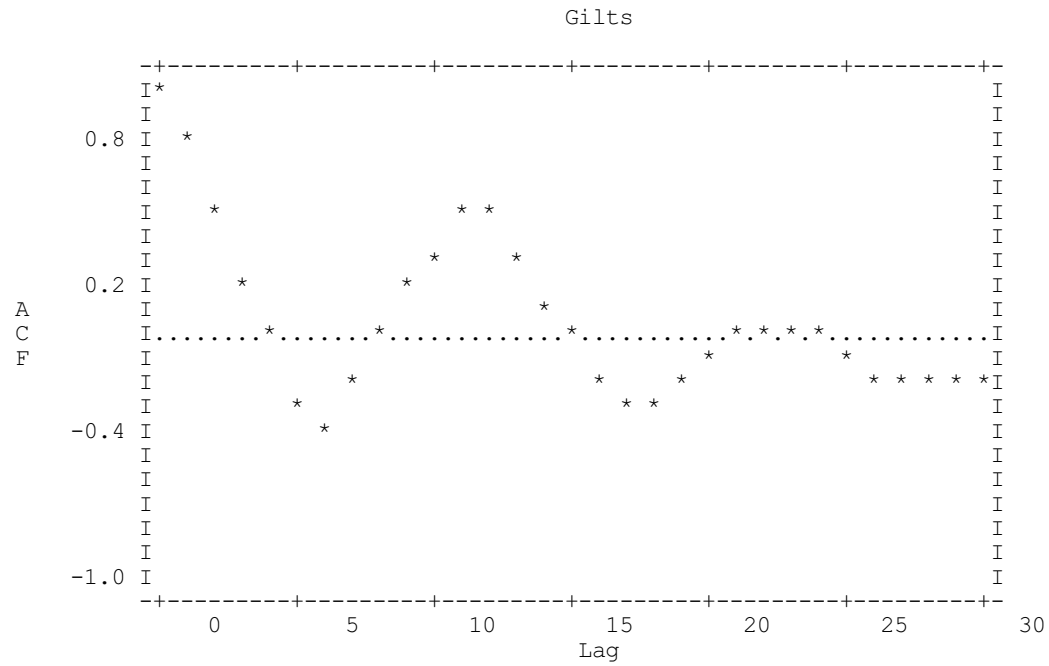
You can use the `CORRELATE` directive to display the sample autocorrelation function of a series, either as a table of numbers, or as a graph – called a *correlogram*. In either case, you must specify the maximum lag for which the autocorrelation is to be calculated, *m* say. You can do this either by setting the `MAXLAG` option to *m*, or by specifying a variate with a pre-defined length of *m*+1 to store the calculated values using the `AUTOCORRELATIONS` parameter. If you do not specify the maximum lag, a default is determined from the length *N* of the time series as follows:

N	default `MAXLAG` setting
< 21	*N*–1
21-40	20
41-120	int(*N*/2)
> 120	60 + int[(*N*–120)/10]

Hence the value of `MAXLAG` increases as the length of the time series increases. Example 7.1.2 plots, saves and prints the autocorrelations up to lag 30 of the time series of `Gilts` used in Example 7.1.1.

Example 7.1.2

```
   9  " Show the autocorrelation function of the time series of Gilts from
 -10    Example 7.1.1. The values are saved in a variate then printed."
  11  CORRELATE [MAXLAG=30; GRAPH=autocorrelations] Gilts; \
  12    AUTOCORRELATIONS=Giltsacf
```




```
  13  PRINT [ORIENTATION=across; SQUASH=yes] Giltsacf

     Giltsacf      1.0000      0.7870      0.4772      0.2384     -0.0118
     Giltsacf     -0.2620     -0.3557     -0.2099     -0.0006      0.1668
     Giltsacf      0.3155      0.4775      0.5060      0.3458      0.1484
     Giltsacf     -0.0234     -0.1529     -0.2819     -0.3262     -0.2356
     Giltsacf     -0.1215     -0.0409     -0.0136      0.0397      0.0448
     Giltsacf     -0.0599     -0.1757     -0.1977     -0.1886     -0.2104
     Giltsacf     -0.1734
```

Genstat includes the autocorrelation at lag 0 in the autocorrelation function; this is always unity. The formula used for the sample autocorrelation at lag k is

$$r_k = (1 - k/n) \times C_k / C_0$$

where

$$C_k = \frac{1}{n_k} \sum_{t=1}^{n-k} (y_t - \bar{y})(y_{t+k} - \bar{y})$$

The number n_k is the number of terms included in the sum. The series can contain missing values, but the summation excludes any product that involves any missing values at all. The value $\bar{y}$ is the ordinary sample mean of the whole series, and n is the number of non-missing values in the series. You can restrict a series, but the restricted set must consist of a contiguous set of units. Thus, you can look at the autocorrelation function derived from just the first section of a series, or from just the last section, or from a section in the middle; but you cannot use restriction to exclude a section from the middle of the series, or to exclude just individual observations.

The AUTOCORRELATIONS parameter allows you to save the calculated autocorrelations. If you want to display a correlogram in a different form from the standard one produced by the GRAPH option, you must save the autocorrelations and plot them explicitly using either the GRAPH or DGRAPH directives. You will then need to define the variate of lags from *0* to *m*.

The TESTSTATISTIC parameter of CORRELATE allows you to save a statistic that can be used to test the hypothesis that the true autocorrelation is zero for positive lags. It is defined as

$$S = n \sum_{k=1}^{m} r_k^2$$

Provided n (the number of data values) is large and m (the maximum lag) is much smaller than n, then under the null hypothesis, S has a chi-square distribution with m degrees of freedom. Thus, a large value of S provides evidence of autocorrelation in a time series.

You can calculate autocorrelation functions for several series in one statement by specifying several variates with the `SERIES` parameter.

7.1.3 The `BJIDENTIFY` procedure

Procedure `BJIDENTIFY` provides a convenient way of calculating and plotting autocorrelations, together with partial correlations and the sample spectrum of a time series.

`BJIDENTIFY` procedure

Displays time series statistics useful for ARIMA model selection (G. Tunnicliffe Wilson & S.J. Welham).

Options

`PRINT` = *string token*	Controls printed output (`description`); default `desc`
`GRAPHICS` = *string token*	What type of graphics to use (`lineprinter`, `highresolution`); default `high`
`WINDOWS` = *scalar* or *variate*	Windows to be used for the plots: a scalar `N` indicates that plots are to be produced on separate pages in window `N` (as currently defined), whereas a variate specifies four separate windows to be redefined (within the procedure) for plotting four graphs on one page; default 1
`PENS` = *variate*	The three pens to be used (after being defined appropriately) for drawing the plots; default `!(1,2,3)`

Parameters

`SERIES` = *variates*	Variates holding the time series for which the statistics are to be produced
`LENGTH` = *scalars* or *variates*	Specifies the units to be used from each series: a scalar `N` indicates that the first `N` units of the series are to be used, a variate of length 2 gives the index of the first and last units of the subseries to be used; by default the whole series is used

---

`BJIDENTIFY` displays time series statistics useful for ARIMA model selection. For a time series, specified (in a variate) using the `SERIES` parameter, four graphs are produced. These are of the series itself, its sample autocorrelation function and partial autocorrelation function, and its sample spectrum (or periodogram). The `LENGTH` parameter can specify that only part of the series is to be used: setting `LENGTH` to a scalar `N` indicates that the first `N` values are to be used; alternatively, a variate of length 2 can be specified holding the positions of the first and last units of the subseries. The maximum lag of the autocorrelations and the frequency grid for the periodogram are determined automatically by the procedure.

Printed output can be suppressed by setting the option `PRINT=*`; by default, `PRINT=description`, which gives a description of the series.

Graphical output is controlled by the options GRAPHICS, WINDOWS and PENS. Option GRAPHICS controls whether plots are produced for line-printer output or on the current high-resolution graphics device; by default high-resolution plots are given. Option WINDOWS controls the way in which the high-resolution plots are arranged. If WINDOWS is set to a scalar N, all the graphs are produced in window N on separate pages; the FRAME directive can then be used to set the attributes of window N before calling the procedure. Alternatively, WINDOWS can be set to a variate of length four; the attributes of the four windows specified are then redefined within the procedure so that four graphs are produced on the same page. By default WINDOWS=1. The PENS option controls which pens are to be used for the plots; the attributes of these pens are modified within the procedure. By default pens 1-3 are used, but these can be changed by setting option PENS to a variate of length 3 containing the numbers of the three different pens required.

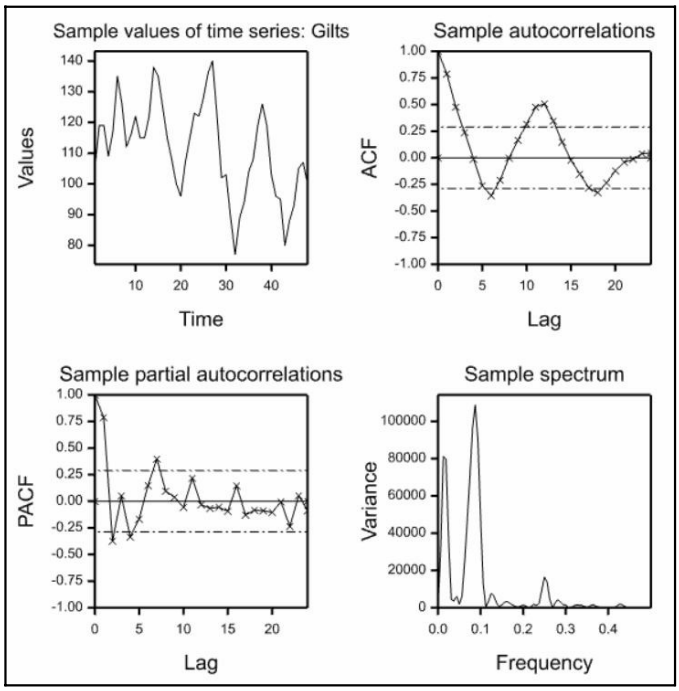


Figure 7.1.3

Example 7.1.3 shows the use of BJIDENTIFY to calculate and plot the autocorrelations of the series from Examples 7.1.1 and 7.1.2 above. In addition, the original series is plotted, together with the partial autocorrelations and the sample spectrum described in Sections 7.1.4 and 7.2.2. The graphs produced by BJIDENTIFY are shown in Figure 7.1.3.

Example 7.1.3

```
 14  " Use procedure BJIDENTIFY to display the time series and its sample
-15    autocorelation function of the time series of Gilts, together with the
-16    sample partial autocorrelations and sample spectrum or periodogram."
 17  BJIDENTIFY [WINDOWS=!(1,2,3,4)] Gilts

Analysis of whole of series Gilts, length 48
showing sample acf and pacf up to lag 24
and sample spectrum with frequency range divided into 80 intervals
```

7.1.4 Partial autocorrelation

Genstat forms partial autocorrelations from an autocorrelation function. The value at lag k is defined as

$$\text{corr}(y_t, y_{t-k} \mid y_{t-1}, y_{t-2} \ldots y_{t-k+1})$$

representing the excess correlation between values separated by k timepoints that is not accounted for by the intermediate points; it is denoted by $\varphi_{k,k}$ because it is also the value of the last in the set of coefficients in the autoregressive prediction equation:

$$y_t = c + \varphi_{k,1} y_{t-1} + \ldots + \varphi_{k,k} y_{t-k} + e_{k,t}$$

Genstat calculates these coefficients recursively for k=1...m by

$$\varphi_{k,k} = (r_k - \varphi_{k-1,1} r_{k-1} - \ldots - \varphi_{k-1,k-1} r_1) / v_{k-1}$$

$$\varphi_{k,j} = \varphi_{k-1,j} - \varphi_{k,k} \varphi_{k-1,k-j} , \; j=1 \ldots k-1$$

$$v_k = v_{k-1}(1 - \varphi_{k,k}^2)$$

It starts with v_0=1, the quantity v_k being the kth order prediction error variance ratio

variance($e_{k,t}$) / variance(y_t).

Partial correlations provide a valuable alternative way of displaying the autocorrelation structure of a series. You can display the partial autocorrelation function either as a table of numbers, or as a graph as shown in Example 7.1.3. Two methods are available for doing this. You can supply the series using the SERIES parameter, in which case the autocorrelations are formed first, automatically, and the partial autocorrelations are then derived from them. Alternatively, you can set SERIES=*, and provide the autocorrelations using the AUTOCORRELATIONS parameter.

You can save the partial autocorrelation function using the PARTIALCORRELATIONS parameter. You can set the VARIANCES and COEFFICIENTS parameters to variates to save the *prediction-error variances* $v_0 \ldots v_m$, and the *prediction coefficients 1,* $\varphi_{m,1} \ldots \varphi_{m,m}$ for the maximum lag m. Genstat sets the first coefficient to 1, and also the first element of the partial autocorrelation sequence to 1: you should find this to be a useful convention for the lag 0 values. Alternatively, if the COEFFICIENTS parameter is set to a matrix structure, the rows of this matrix will be used to save the prediction coefficients for *all* the orders up to the maximum lag. Example 7.1.4 uses some of the previously calculated autocorrelations to produce partial autocorrelations and the matrix of prediction coefficients. Note that the partial autocorrelations also appear down the diagonal of the matrix. The graph in Figure 7.1.3 suggests that an order of 7 would be appropriate for a predictor, the coefficients being in the row labelled 7 of the matrix.

Example 7.1.4

```
" The first 10 autocorrelations formed in Example 7.1.3 for the
  time series of Gilts are used to calculate the prediction coefficients
  up to a maximum lag of 10. These are saved in a matrix and printed."
VARIATE [NVALUES=11] Shortacf
CALCULATE Shortacf = Giltsacf$[!(1...11)]
TEXT [VALUES='00','01','02','03','04','05','06','07','08','09','10'] \
  Laglabels
MATRIX [ROWS=Laglabels; COLUMNS=Laglabels] Predcoef
CORRELATE SERIES=*; AUTOCORRELATIONS=Shortacf; COEFFICIENTS=Predcoef
PRINT [RLWIDTH=10] Predcoef; FIELDWIDTH=6; DECIMALS=2
```

```
          Predcoef
Laglabels    00    01    02    03    04    05    06    07    08    09    10
Laglabels
       00  1.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
       01  1.00  0.79  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
       02  1.00  1.08 -0.37  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
       03  1.00  1.10 -0.43  0.05  0.00  0.00  0.00  0.00  0.00  0.00  0.00
       04  1.00  1.12 -0.57  0.42 -0.34  0.00  0.00  0.00  0.00  0.00  0.00
       05  1.00  1.06 -0.50  0.32 -0.15 -0.17  0.00  0.00  0.00  0.00  0.00
       06  1.00  1.09 -0.48  0.28 -0.07 -0.33  0.15  0.00  0.00  0.00  0.00
       07  1.00  1.03 -0.35  0.30 -0.18 -0.14 -0.28  0.40  0.00  0.00  0.00
       08  1.00  0.99 -0.32  0.32 -0.17 -0.16 -0.25  0.30  0.10  0.00  0.00
       09  1.00  0.98 -0.33  0.33 -0.16 -0.16 -0.26  0.31  0.06  0.04  0.00
       10  1.00  0.99 -0.33  0.34 -0.17 -0.17 -0.27  0.33  0.04  0.09 -0.06
```

CORRELATE will print a warning if you include missing values in an autocorrelation function that you have supplied, or if for some other reason the autocorrelations are invalid. In particular, if a partial autocorrelation value is obtained outside the range (−1, 1), Genstat will truncate the sequence at the previous lag.

7.1.5 Cross-correlation

You can calculate cross-correlations between two series by specifying one series with the SERIES parameter and the other with the LAGGEDSERIES parameter. You must define the

maximum lag, as for autocorrelations (7.1.2). You can plot or tabulate the resulting function. Example 7.1.5 shows the correlation between one series and the later values of a second series, along with the correlation of the second series with later values of the first. This second set of correlations may be considered as correlations between the first series and the second series at *negative* lags. The two sets of correlations are displayed in the same graph to emphasize this interpretation.

Example 7.1.5

```
" Save and plot the crosscorrelations between the series
  Profit and Gilts in Example 7.1.1."
CORRELATE [MAXLAG=20] SERIES=Profit,Gilts; LAGGEDSERIES=Gilts,Profit;\
          CROSSCORRELATIONS= P_G_ccf , G_P_ccf
VARIATE [VALUES=0...20] Lag
CALCULATE Neglag=-Lag
FRAME [GRID=xy,yx] 1; XLOWER=0.05; XUPPER=0.95; YLOWER=0.45; YUPPER=0.95
XAXIS 1; TITLE='LAG'; LOWER=-21; UPPER=21
YAXIS 1; TITLE='CCF'; LOWER=-1.0; UPPER=1.0
PEN   1; LINESTYLE=1; METHOD=line; SYMBOL=2
DGRAPH [TITLE='Cross correlations between Profit and Gilts'; \
  WINDOW=1; KEYWINDOW=0] Y=P_G_ccf,G_P_ccf; X=Lag,Neglag; PEN=1
```

The graph produced by Example 7.1.5 is displayed in Figure 7.1.5.

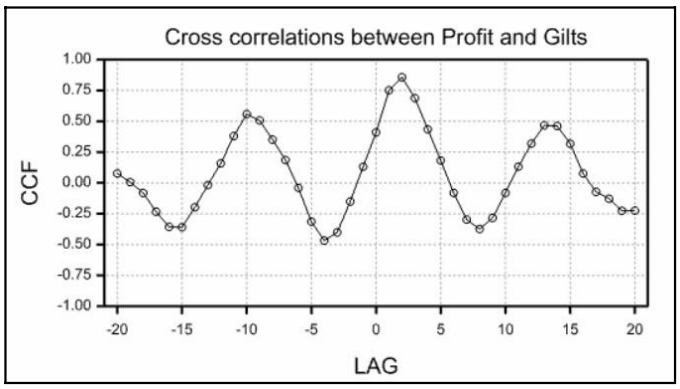


Figure 7.1.5

Missing values are allowed, as for autocorrelations. Genstat calculates the sample cross-correlation between the first series x_t and the lagged series y_t at lag k using:

$$r_k = (1 - k/n)\, C_k / (s_x\, s_y)$$

where

$$C_k = \frac{1}{n_k} \sum_{t=1}^{n-k} (x_t - \bar{x})(y_{t+k} - \bar{y})$$

The series x_t and y_t may be of different lengths. The summation includes all possible terms, but excludes any product containing missing values; the number n_k is the number of terms included in the sum. The values $\bar{x}$ and $\bar{y}$ are the sample means, and s_x, s_y are the sample standard deviations. The number n is the minimum of the number of values of x and of y, excluding missing values. You can restrict either series to a set of contiguous units: if both are restricted, their restrictions must match.

You can save the cross-correlation function using the `CROSSCORRELATIONS` parameter. You can also save a test statistic using the `TESTSTATISTIC` parameter; this is used similarly to the statistic described in Section 7.1.2 to test for lack of lagged cross-correlation in one direction of the relationship between two series. However the test is valid only if each of the series has a zero autocorrelation function. Cross-correlations take precedence in the storage. Thus if you request both autocorrelations and cross-correlations in a single `CORRELATE` statement, the stored test statistic will relate to the cross-correlations: that for the autocorrelations will not be stored.

7.2 Fourier transformation

This section describes various types of Fourier transformation. These allow you to do most types of spectral analysis with a few Genstat statements. You may want to put these into procedures (1:5.3) for repeated use. The Genstat procedure library contains four procedures that use Fourier transformations. BJIDENTIFY, which plots the sample spectrum, is described in 7.1.3. The other three are described at the end of this section. SMOOTHSPECTRUM (7.2.6) can be used to calculate and plot smoothed spectrum estimates, DFOURIER (7.2.7) performs a harmonic analysis of a univariate time series, and MCROSSPECTRUM (7.2.8) performs a spectral analysis of a multiple time series.

The Fourier or spectral analysis of time series is described comprehensively by Bloomfield (1976) and Jenkins & Watts (1968). The Fourier transformation of a series calculates the coefficients of the sinusoidal components into which the series can be analysed. There are four types of transformation described below, which are appropriate for different types of symmetry in the series. You may often want the length of the variate holding the supplied series to determine implicitly a natural grid of frequencies at which values of the transform are calculated. Genstat will do this if you have not previously declared the identifier supplied for the transform. Alternatively you may want to determine the transform at a finer grid of frequencies, and you can achieve this by declaring a transform variate that is as long as you require. You can do this only for the two types of Fourier transform that apply to real series.

You can also recover the series corresponding to a particular transform; that is, you can invert a transformation.

The conventional index for the series that is being transformed is 0...(N−1) in the defining formulae, so that the first element corresponds to the origin for the sinusoidal components in the analysis.

7.2.1 The FOURIER directive

FOURIER directive

Calculates cosine or Fourier transforms of real or complex series.

Option

PRINT = *string tokens*	What to print (transforms); default *

Parameters

SERIES = *variates*	Real part of each input series
ISERIES = *variates*	Imaginary part of each input series
TRANSFORM = *variates*	To save real part of each output series
ITRANSFORM = *variates*	To save imaginary part of each output series
PERIODOGRAM = *variates*	To save periodogram of each transform

Series of real numbers are stored in single variates, and series of complex numbers in pairs of variates. You can use the FOURIER directive to calculate the cosine transform of the real series { a_t, t=0...N−1 } stored in a variate A by

```
FOURIER [PRINT=transform] A
```

You calculate the Fourier transform of the complex series { a_t+ib_t, t=0...N−1 } by storing the values a_t in one variate, A say, the corresponding values b_t in another, B say, and giving the statement:

```
FOURIER [PRINT=transform] A; ISERIES=B
```

You can restrict the series specified by either the SERIES or ISERIES parameter to a contiguous

set of units – as for the CORRELATE directive (7.1). Genstat applies the transformation only to the restricted series of values. Similarly, you may supply restricted variates with the TRANSFORM and ITRANSFORM parameters to save the transform: Genstat will then carry out the transformation so as to supply the required number of values (if that is possible according to the rules at the end of Section 7.2.2). There must be no missing values in the variates in the SERIES or ISERIES parameters, unless you exclude them by a restriction.

Genstat carries out the Fourier transformation using a fast algorithm which relies on the order of the transformation being highly composite (de Boor 1980). In practice, an appropriate order is a round number such as 300 or 6000, consisting of a digit followed by zeroes. If, however, the order has a large prime factor, the transformation may take much longer. For example, a transformation of order 499 is about 25 times slower than one of order 500. In the description below, therefore, we clearly state the order of each form of the transformation, to illustrate a sensible choice of size.

7.2.2 Cosine transformation of a real series

This can be used to calculate the spectrum from a set of autocorrelations. Suppose the variate R contains the values $r_0 \ldots r_n$, and the variate F is to hold the calculated values $f_0 \ldots f_m$ of the spectrum. These values correspond to angular frequencies of $\pi j/m$; that is, periods of $2m/j$, for j=0...m. You apply the transformation by putting

```
FOURIER R; TRANSFORM=F
```

If F has not been declared previously, this statement defines it automatically as a variate with n+1 values (so m=n). If F has been declared to have m+1 values, then m must be greater than or equal to n; otherwise Genstat will redeclare F to have n+1 values.

The transform is defined when m>n by

$$f_j = r_0 + \sum_{k=1}^{n} \left\{ 2 r_k \cos\left(k \frac{\pi j}{m} \right) \right\}$$

When m=n the final term in this sum is

$$r_n \cos(\pi j) = r_n (-1)^j$$

and it appears without the multiplier 2. The order of the transformation is $2m$.

If R contains sample autocorrelations, you must multiply it by a variate holding a lag window in order to obtain a smooth spectrum estimate (see Bloomfield 1976, page 166, or Jenkins & Watts 1968, page 243).

7.2.3 Fourier transformation of a real series

This can be used to calculate the periodogram of a time series. Suppose the variate X of length N contains the supplied series values $x_0 \ldots x_{N-1}$. The result of the transformation is a set of coefficients $a_0 \ldots a_m$ of the cosine components and $b_0 \ldots b_m$ of the sine components of the series, held in variates A and B, say. Normally the number of such components is related to the length of the series by taking m=N/2 if N is even or m=(N–1)/2 if N is odd. Then the coefficients correspond to angular frequencies of $2\pi j/N$, which is the same as saying that they correspond to periods N/j for j=0...m. Since by definition b_0=0, and b_m=0 if N is even, there are N "free" coefficients in A and B (which you can think of as the real and imaginary parts of a complex transform with values a_j+ib_j). You can save the periodogram values $p_0 \ldots p_m$ in a variate P, say: these are the squared amplitudes of the sinusoidal components, and are calculated by Genstat as $p_j = a_j^2 + b_j^2$.

You obtain the transform by putting

```
FOURIER X; TRANSFORM=A; ITRANSFORM=B; PERIODOGRAM=P
```

If you want only the periodogram, you can put

```
FOURIER X; PERIODOGRAM=P
```

If you have not declared A previously Genstat defines it automatically, here as a variate of length m+1 where m has the default value defined above. If you have previously declared A, it should have length greater than or equal to m+1; otherwise Genstat declares it to have this length. In any case, B and P should have the same length as A, and will be declared (or redeclared) if required.

In the usual case when A, B or P has the default length m+1, the transform is defined by:

$$a_j = \sum_{t=0}^{N-1} \left\{ x_t \cos\left(t \frac{2\pi j}{N}\right) \right\}; \qquad j=0...m$$

$$b_j = \sum_{t=0}^{N-1} \left\{ x_t \sin\left(t \frac{2\pi j}{N}\right) \right\}; \qquad j=0...m$$

In this case, the order of the transformation is N. If A, B and P have length m'+1 with $m'>m$, Genstat computes the results at a finer grid of frequencies $2\pi j/N'$, j=0...m' where $N'=2m'$. These replace $2\pi j/N$ in the above defining sums. The upper limit on the sums remains as $N-1$, although internally Genstat treats it as $N'-1$ with the extra values of $x_N...x_{N'-1}$ being taken as zero. The order of the transformation is then N'. There are various conventions used for scaling the periodogram with factors $2/m$, $1/m$ or $1/\pi m$. You can apply these by using a CALCULATE statement (1:4.1.1) after the transformation. You may also want to apply mean correction to the series before calculating the periodogram. Figure 7.1.3 showed the sample spectrum of the time series Gilts. This is just the scaled periodogram calculated using FOURIER as described above. The graph shows a strong peak at frequency 0.08 corresponding to the obvious cycle of period approximately 12 quarters. It also reveals a peak at frequency 0.25 which reflects an annual pattern of period 4 quarters. This is difficult to detect simply by looking at the graph of the series.

7.2.4 Fourier transformation of a complex series

This is the most general form of the Fourier transformation; the other three types are essentially special cases in which some coefficients are zero or have a symmetric structure. Suppose variates X and Y contain values x_0 ... x_{N-1} and y_0 ... y_{N-1}, which may be viewed as the real and imaginary parts of the series $\{ x_t+iy_t, t=0 ... N-1 \}$. The results of the transformation are coefficients a_0 ... a_{N-1} and b_0 ... b_{N-1} which can be held in variates A and B, say: these may similarly be considered as parts of complex coefficients a_t+ib_t, t=0 ... $N-1$.

You can do the transformation by putting

```
FOURIER SERIES=X; ISERIES=Y; TRANSFORM=A; ITRANSFORM=B
```

Both X and Y must be variates with the same length N. Similarly A and B must have length N, and if they do not Genstat will declare (or redeclare) them as variates of length N. The order of the transformation is N.

The results are defined by

$$a_j = \sum_{t=0}^{N-1} \left\{ x_t \cos\left(t \frac{2\pi j}{N}\right) - y_t \sin\left(t \frac{2\pi j}{N}\right) \right\}; \qquad j=0...m$$

$$b_j = \sum_{t=0}^{N-1} \left\{ x_t \sin\left(t \frac{2\pi j}{N}\right) + y_t \cos\left(t \frac{2\pi j}{N}\right) \right\}; \qquad j=0...m$$

or equivalently in complex form by

$$(a_j + ib_j) = \sum_{t=0}^{N-1} (x_t + iy_t)\, e^{(it\frac{2\pi j}{N})}$$

The complex transform can be used in cross-spectral analysis.

You can view a Fourier transformation as an orthogonal matrix transformation. Hence its inverse is another Fourier transformation (apart from some simple scaling). You can use this to calculate convolutions. In particular, the correlations of a time series can be obtained by applying the inverse cosine transformation to the periodogram. Example 7.2.4 shows that a repeated Fourier transformation returns the original series – with appropriate scaling.

Example 7.2.4

```
" Repeat a Fourier transformation on random numbers."
SCALAR Nvalues; VALUE=25
CALCULATE Rstart,Istart = URAND(6672,0; Nvalues)
FOURIER Rstart; ISERIES=Istart; TRANSFORM=Rmiddle; ITRANSFORM=Imiddle
CALCULATE Rmiddle,Imiddle = Rmiddle,Imiddle * 1,-1 / SQRT(Nvalues)
FOURIER Rmiddle; ISERIES=Imiddle; TRANSFORM=Rfinish; ITRANSFORM=Ifinish
CALCULATE Rfinish,Ifinish = Rfinish,Ifinish * 1,-1 / SQRT(Nvalues)
PRINT Rstart,Istart,Rmiddle,Imiddle,Rfinish,Ifinish; DECIMALS=4
```

Rstart	Istart	Rmiddle	Imiddle	Rfinish	Ifinish
0.4236	0.6865	2.5847	-2.6468	0.4236	0.6865
0.4458	0.7316	0.0363	-0.5219	0.4458	0.7316
0.3443	0.5548	-0.1036	-0.2434	0.3443	0.5548
0.0174	0.7045	0.4952	0.2670	0.0174	0.7045
0.0388	0.7507	-0.0092	-0.3748	0.0388	0.7507
0.7562	0.9707	0.0938	-0.2235	0.7562	0.9707
0.3171	0.7538	-0.0380	0.0790	0.3171	0.7538
0.5931	0.6838	-0.0152	0.4113	0.5931	0.6838
0.9229	0.0015	-0.0863	-0.4419	0.9229	0.0015
0.9485	0.5462	0.1806	-0.2726	0.9485	0.5462
0.3938	0.1294	-0.2906	0.0565	0.3938	0.1294
0.6251	0.4935	0.1896	-0.0188	0.6251	0.4935
0.4973	0.7353	0.1773	0.2825	0.4973	0.7353
0.1379	0.2087	-0.1829	-0.3463	0.1379	0.2087
0.2643	0.6310	-0.4662	-0.2856	0.2643	0.6310
0.9029	0.1571	0.2154	0.3256	0.9029	0.1571
0.3597	0.1690	-0.2685	-0.0011	0.3597	0.1690
0.6736	0.4674	-0.1093	0.3781	0.6736	0.4674
0.7469	0.2263	-0.1546	-0.3584	0.7469	0.2263
0.9657	0.8123	-0.2118	-0.0051	0.9657	0.8123
0.0724	0.4666	-0.0544	0.0780	0.0724	0.4666
0.4650	0.6966	0.1744	-0.0849	0.4650	0.6966
0.5000	0.5380	0.5275	0.4453	0.5000	0.5380
0.6257	0.7017	-0.2402	0.3814	0.6257	0.7017
0.8854	0.4171	-0.3260	-0.3117	0.8854	0.4171

7.2.5 Fourier transformation of a conjugate sequence

It is easiest to think of the Fourier transform of a conjugate sequence as the reverse of the transformation of a real series (7.2.2), with the roles of the series and the transform interchanged. For the true inverse transformation some simple scaling is also required.

Thus if variates A and B of length m+1 are supplied containing values $a_0 \ldots a_m$ and $b_0 \ldots b_m$, which may be viewed as parts of complex coefficients a_j+ib_j, the result of the transformation is a single real series $x_0 \ldots x_{N-1}$ held in a variate X of length N.

X can be declared to have length N=2m or N=2m+1 (corresponding to the case N even or odd in Section 7.2.2). The value of b_0 must be zero; also if N=2m, the value of b_m must be zero. If either of these conditions is not satisfied, Genstat sets the values of these elements to zero and gives a warning. If X has not been declared previously (or has been declared with a length equal to neither 2m nor 2m+1), then it is declared (or redeclared) with a length governed by whether

b_m is 0: N=2m if b_m=0, and N=2m+1 if b_m≠0. The value of b_0 is checked to be zero as before.

You can obtain the transform using the statement

```
FOURIER SERIES=A; ISERIES=B; TRANSFORM=X
```

The definition of the transform is, in the case N=2m+1,

$$x_t = a_0 + \sum_{j=1}^{m} 2\left\{ a_j \cos\left(t\frac{2\pi j}{N}\right) + b_j \sin\left(t\frac{2\pi j}{N}\right)\right\}$$

In the case N=2m, the final term in the sum is simply

$a_m \cos(t\pi) = a_m (-1)^t$

and it appears without the multiplier 2. The order of this transformation is N.

7.2.6 The SMOOTHSPECTRUM procedure

SMOOTHSPECTRUM procedure

Forms smoothed spectrum estimates for univariate time series (G. Tunnicliffe Wilson & S.J. Welham)

Options

PRINT = *string token*	Controls printed output (description); default desc
METHOD = *string token*	Method to be used for smoothing (lagwindow, direct, YuleWalker, exactautoregressive); default lagw
BANDWIDTH = *scalar*	Frequency domain bandwidth for the smoothing window; must be set if METHOD=dire
MAXLAG = *scalar*	Specifies the cut-off lag (i.e. the maximum lag of autocovariance used in the spectrum calculation) for METHOD=lagw, or the order of the autoregression for METHOD=Yule or exac; if this option is not set then BANDWIDTH must be set, and will be used to determine an appropriate value of MAXLAG
DIVISIONS = *scalar*	Determines the number of frequency divisions into which the range [0.0, 0.5] is divided for calculating the spectrum; the default is chosen so that the bandwidth covers about four intervals
PROBABILITY = *scalar*	Probability value used for confidence limits; default 0.9
TAPER = *scalar*	The proportion of data to be tapered (applied for all settings of METHOD except exac); default 0.0
SHAPE = *scalar*	The shape of the trapezium window (a value of 1.0 specifies a rectangular, and 0.0 a triangular window); default 0.5
YLOG = *string token*	Whether to plot with a log-transformed Y-axis (yes, no); default no
XLOG = *string token*	Whether to plot with a log-transformed X-axis (yes, no); default no
GRAPHICS = *string token*	What sort of graphics to use (lineprinter, highresolution); default high
WINDOW = *scalar*	Window to be used for plotting; default 1
PENS = *variate*	The two pens to be used (after being defined appropriately) for drawing the plots; default !(1,2)

Parameters

SERIES = *variate*	The series for which the spectrum is to be calculated
LENGTH = *scalar* or *variate*	Scalar specifying that the first *N* units of the series are to be used, or a variate specifying the first and last units of the series to be used
SPECTRUM = *variate*	Saves the smoothed spectrum; need not be declared in advance, but will be set up as a variate of the appropriate length within the procedure
LOWER = *scalar* or *variate*	Scalar to save the multiplier of the spectrum used to calculate the lower limit, or a variate to save the values of the lower limit
UPPER = *scalar* or *variate*	Scalar to save the multiplier of the spectrum used to calculate the upper limit, or a variate to save the values of the upper limit
FREQUENCY = *variate*	Saves the frequency values at which the spectrum is calculated

SMOOTHSPECTRUM calculates smoothed spectrum estimates for a univariate time series. The series is specified in a variate by the SERIES parameter. The parameter LENGTH can be used to specify that only part of the series is to be used: if LENGTH is set to a scalar N, then only units 1...N are used; alternatively, it can define a sub-series by being set to a variate of length 2 holding the numbers of the first and last units to be used. The spectrum can be saved by the SPECTRUM parameter. The method to be used for the smoothing is controlled by the METHOD option, with settings lagwindow for Parzen lag window smoothing, direct for frequency domain smoothing using a trapezium window, YuleWalker for autoregressive spectrum estimation based on Yule-Walker coefficients, and exactautoregressive for autoregressive estimation based on exact likelihood estimation of the coefficients.

For frequency domain smoothing (METHOD=direct), option BANDWIDTH specifies the bandwidth of the smoothing window and option SHAPE the shape of the trapezium window. The BANDWIDTH option is also used to determine an appropriate default for the MAXLAG option if this is not specified with other METHOD settings: for METHOD=lagwindow, MAXLAG specifies the cut-off lag (i.e. the maximum lag of autocovariance used in the spectrum calculation), while for METHOD=YuleWalker or exactautoregressive, it specifies the order of the autoregression.

The DIVISIONS option can define the number of frequency divisions into which the range [0.0, 0.5] is divided for calculating the spectrum; if this is omitted a default is chosen so that the bandwidth covers about four intervals. The frequency values at which the spectrum is calculated can be saved, in a variate, by the FREQUENCY parameter. The proportion of data to be tapered (relevant to all settings of METHOD except exactautoregressive) is controlled by the TAPER option; by default there is no tapering.

The LOWER and UPPER parameters can be set to scalars to save the scaling factor used to calculate the upper and lower bounds, or to variates to save the upper and lower bounds for the SPECTRUM variate.

None of the input or output structures must be restricted (but restriction of the input series to a contiguous set of units can be achieved by use of the LENGTH parameter, as described above).

Printed output can be suppressed by setting the option PRINT=*; by default, PRINT=description. The PROBABILITY option indicates the probability value used for confidence limits; 0.9 is used as the default.

The procedure will also plot the spectrum: option GRAPHICS controls whether this is for line printer or on a high-resolution device. With high-resolution graphics, the plot will be produced using the current settings of the window specified by the WINDOW option; by default WINDOW=1. The FRAME directive can be used to set the attributes of the window prior to calling the

procedure. The PENS option controls which pens are to be used for the plots; the attributes of these pens are modified within the procedure. By default pens 1 and 2 are used, but these can be changed by setting option PENS to a variate of length 2 containing the numbers of the two pens required. Options YLOG and XLOG allow the X- and Y-axes to be represented on a logarithmic scale.

Example 7.2.6 uses SMOOTHSPECTRUM to calculate and plot an estimate of the spectrum of a time series of annual temperature measurements. The graph produced by SMOOTHSPECTRUM is shown in Figure 7.2.6. The lag window method of smoothing is specified as an option. Error limits for the estimate are included in the graph. The frequency scale is given in cycles per unit time. There is evidence for cycles of periods just over 3 years and 2 years.

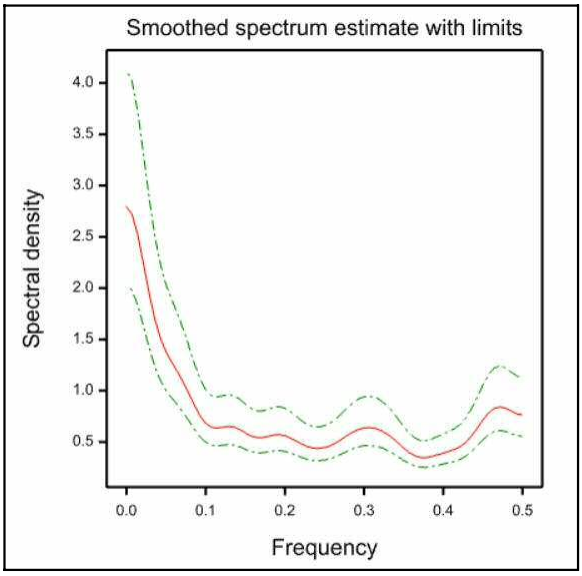


Figure 7.2.6

Example 7.2.6

```
  2  " Smooth spectrum estimation for a series of annual
 -3    measurements of  Central England Average Temperature:
 -4    data from Manley, G. (1974), Central England temperatures:
 -5    monthly means 1659-1973, Quart.J.Met.Soc., 100, 378-405."
  6  VARIATE [NVALUES=315] Cetave
  7  OPEN    '%GENDIR%/Examples/GuidePart2/Cetave.dat'; 3
  8  READ    [CHANNEL=3] Cetave

   Identifier   Minimum      Mean   Maximum    Values   Missing
       Cetave     6.800     9.140     10.60       315         0

  9  CLOSE   3
 10  SMOOTHSPECTRUM [METHOD=lagwindow; BANDWIDTH=0.07; GRAPHICS=high] Cetave

Analysis of whole of series Cetave, length 315
Bandwidth used for estimate is  0.07132
Degrees of freedom of estimate are 44
Frequency division of estimates is 70
Probability value used for limits is 0.900
Upper and lower multipliers for limits are 1.477   0.7275
Lag window smoothing used with cut-off lag 26
```

7.2.7 The DFOURIER procedure

DFOURIER procedure

Performs a harmonic analysis of a univariate time series (G. Tunnicliffe Wilson & R.P. Littlejohn).

Options

PRINT = *string tokens*	Controls printed output (accumulated, means, tsm); default *
PLOT = *string tokens*	What to plot (periodogram, harmonics, means, residuals, cumulative, range); default peri, harm, mean, resid
MODELTYPE = *string token*	What harmonic regression model to fit (none, best, full); default none
GROUPS = *factor*	Groups for plot of means
ORDER = *variate*	Order for time series model; default !(1,0,0)
COLOURS = *text* or *variate*	Colour for each level of GROUPS
FACSHORTCYCLE = *factor*	Factor giving levels of the short cycle
NCOMPONENTS = *scalar*	Number of nested cycles, must be 0, 1, or 2; default 0
SHORTCYCLE = *scalar*	Length of the short cycle; default 24
LONGCYCLE = *scalar*	Length of the long cycle; default 365.225
LABSHORTCYCLE = *text*	Label for the short cycle; default 'daily'
LABLONGCYCLE = *text*	Label for the long cycle; default 'annual'
NHSHORTCYCLE = *scalar*	Number of harmonics for the short cycle; default 5
NHLONGCYCLE = *scalar*	Number of harmonics for the long cycle; default 3
RANGE = *variate*	Variate with two values, defining the frequency range within [0,0.5] to draw a portion of the periodogram

Parameters

DATA = *variates*	Time series
PERIODOGRAM = *variates*	Saves the periodogram of DATA
FREQUENCY = *variates*	Saves the frequencies at which the periodogram is calculated
MEANS = *tables*	Saves the table of means of the fitted model for each value of FACSHORTCYCLE by each level of GROUPS
RESIDUALS = *variates*	Saves the residuals from the fitted model
FITTEDVALUES = *variates*	Saves the fitted values from the model

DFOURIER performs a harmonic analysis for a univariate time series which is supplied, in a variate, by the DATA parameter. In its basic form, it can produce 3 pages of graphs to study the series. These graphs are all controlled by the PLOT option. Setting PLOT=periodogram produces a page of graphs showing the time series, its periodogram and its log periodogram. The frequencies for the periodogram are calculated internally, and noted in the output. These can be saved, in a variate, by the FREQUENCY parameters, and the PERIODOGRAM parameter can save the periodogram.

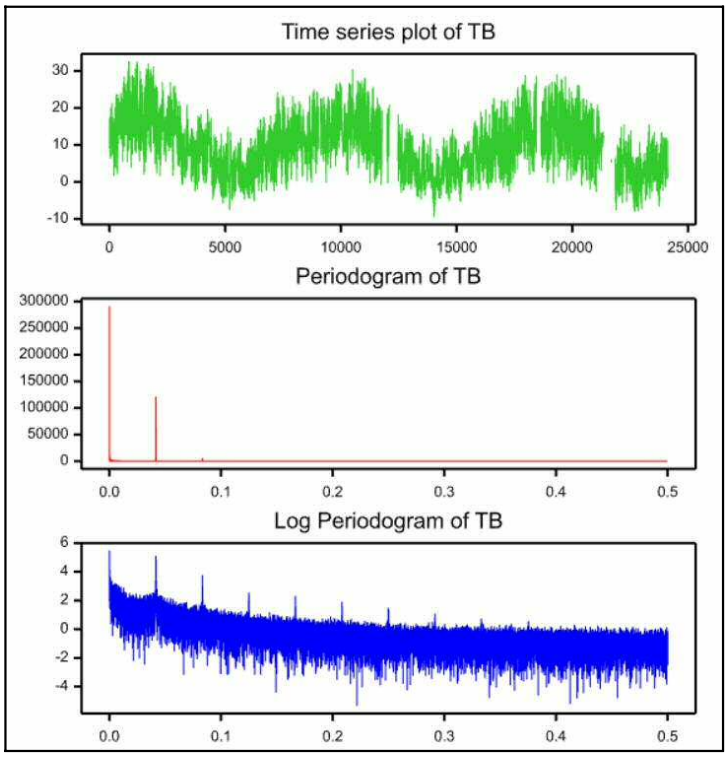


Figure 7.2.7a

Figure 7.2.7a shows this combination of plots for the data in Example 7.2.7; these are hourly temperatures from December 1998 to August 2001 at the Tara Base.

The cumulative setting of PLOT plots the cumulative periodogram (on a separate page), and the range setting plots the periodogram over the range specified by the RANGE option (this must be a value within [0,0.5]). See Figures 7.2.7b and 7.2.7c.

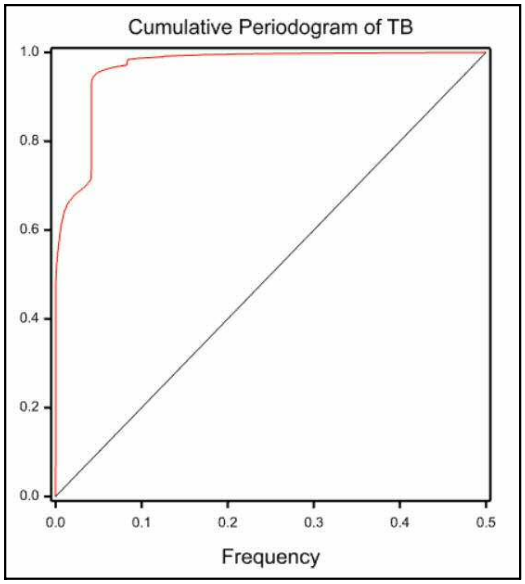


Figure 7.2.7b

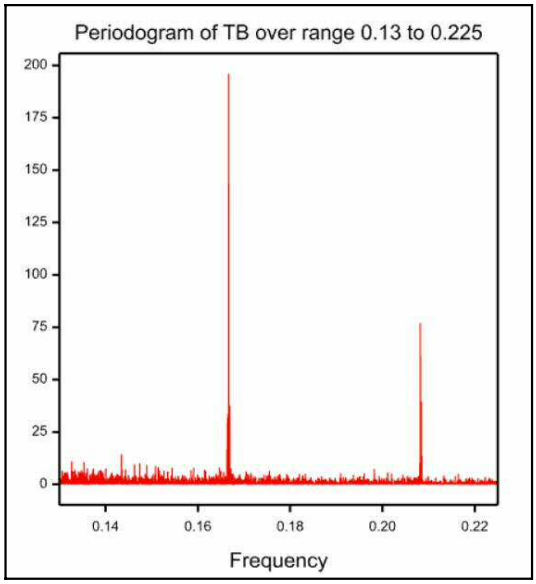


Figure 7.2.7c

Other graphs are useful if you anticipate that the series will show some specific components. The number of these components is specified by the NCOMPONENTS option, and may be either 0 (no components, the default), 1 (a "short" cycle) or 2 (a "short" and a "long" cycle). The lengths of the long and short cycles are specified by the LONGCYCLE and SHORTCYCLE options,

respectively. The defaults 365.225 and 24, correspond to hourly measurement of annual and daily cycles. The LABLONGCYCLE and LABSHORTCYCLE options supply labels for these cycles for the plots, with defaults of 'annual' and 'daily' respectively.

The components are particularly useful for analysing meterological time series (such as air temperatures) measured hourly over several years, where you want to describe how the diurnal pattern varies throughout the year. A single (non-sinusoidal) periodic component with period p (e.g. $p = 24$ for hourly observations) produces a main spike in the periodogram at the frequency $f = 1/p$, followed by a series of diminishing spikes at integer multiples of f known as *harmonics*.

When there are two periodic components with interacting rhythms, signals are observed in the periodogram not only at harmonics of each frequency, but at integer differences of the lower frequency from the higher. Thus, if hourly and annual frequencies are denoted by f_d and f_a, spikes may be observed in the periodogram at

$$f_{da} = n \times f_d + m \times f_a,$$

where n is a non-negative integer, and m is an integer, which must be positive when n is zero.

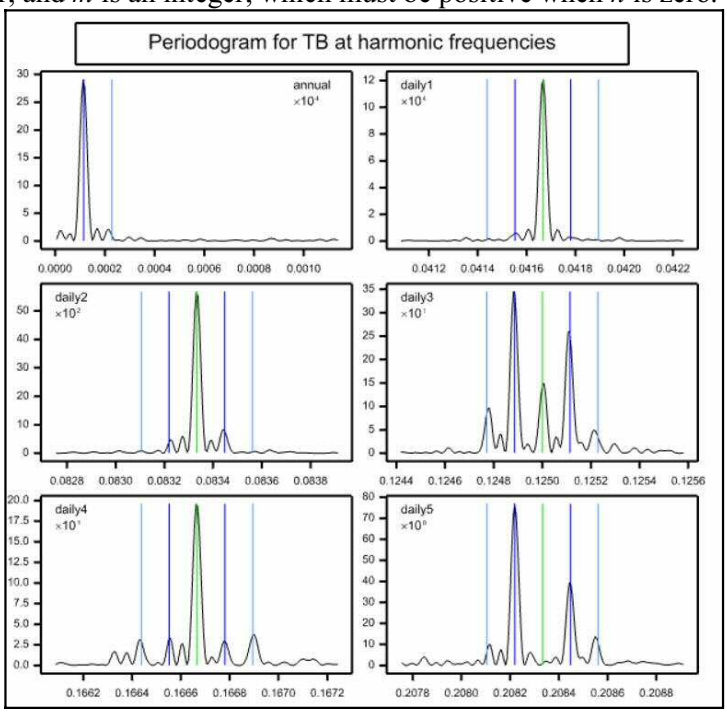


Figure 7.2.7d

These spikes generated by the interaction are generally hard to discern in an ordinary graph of the periodogram. The harmonic setting of PLOT, shown in Figure 7.2.7d, produces a trellis plot that zooms in on a narrow range of about $n \times f_d$, for integer values of n ranging from 1 up to a value defined by the NHSHORTCYCLE option. This can be set to either 5 (default), 7 or 8, producing respectively a 3 × 2, 4 × 2 or 3 × 3 array of graphs. The NHLONGCYCLE option specifies the number of vertical lines to be drawn, within each graph, at positions corresponding to differences due to the long cycle. This can be set to 0, 1, 2 or 3 (default). It should be set to 0 if there is only one periodicity in the sampling protocol. The y-axes of the plots are scaled individually to a suitable order of magnitude, which is denoted in each graph. The frequency range for each panel is

$$n \times f_d +/- 5.1 \times f_a.$$

The MODELTYPE option allows a a harmonic regression analysis to be conducted on DATA. The setting full fits sine and cosine terms for each frequency indicated in the harmonics graph. Alternatively, the setting best fits the full model and then drops terms that are non-significant at the 5% level. This does not guarantee that all terms remaining in the model are necessarily significant at the 5% level. In practice, however, dropping these additional terms will usually make little difference to the fitted model or residual variance. The accumulated setting of the PRINT option prints the accumulated analysis of deviance table from the fit.

With the tsm setting of the PRINT option, the model fitted as above is then used as the TRANSFERFUNCTION in a time series analysis of DATA. The TSM is defined by the ORDER option; by default this is set to a first-order autoregression (i.e. ORDER=!(1,0,0)). Note that

this may take a long time to fit if there are many missing values in the data.

The fitted values and residuals from the final model (`tsm` is fitted after `best`, which is fitted after `full`) can be saved by the `FITTEDVALUES` and `RESIDUALS` parameters. The `residuals` setting of `PLOT`, shown in Figure 7.2.7d, produces time-series plots of the residuals, from the `BJIDENTIFY` procedure.

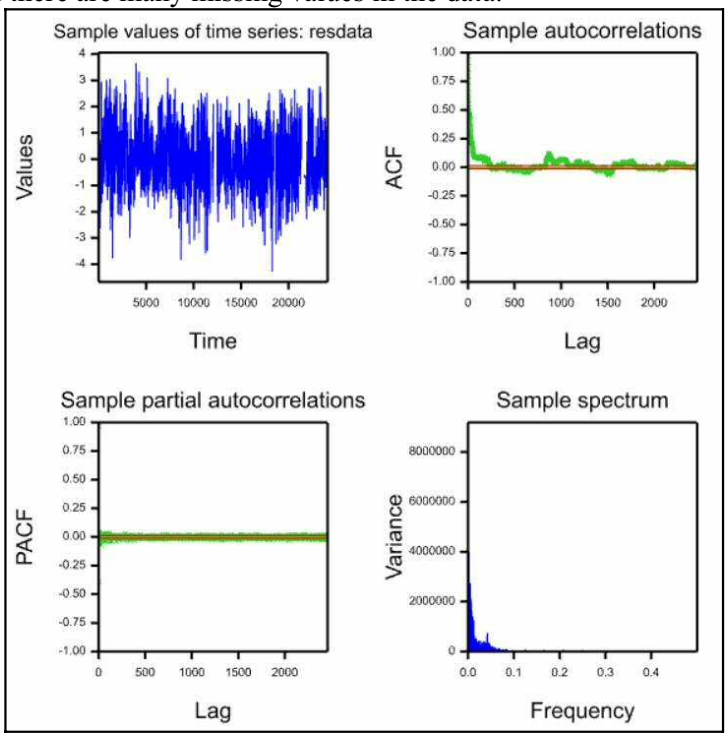


Figure 7.2.7e

`DFOURIER` forms tables of means of the fitted values classified by the the short cycle component and another factor, specified by the `GROUPS` option. You can supply the short cycle factor using the `FACSHORTCYCLE` option; this must have `SHORTCYCLE` levels or a fault will be generated. If `FACSHORTCYCLE` is unset, the required factor will be internally generated with levels 1...`SHORTCYCLE`. The factor `GROUPS` may, for example, be month or season. The `SHORTCYCLE` factor should be nested within `GROUPS` to provide meaningful output, but no checks are carried out on this.

You can plot the means using the `means` setting of the `PLOT` option. The points in each group are plotted in different colours, and you can supply these using the `COLOURS` option. If `COLOURS` is unset, the colours are set by default. If `GROUPS` has 4 levels, it is assumed they correspond to season, and pens 1 to 4 are defined to be red, gold, blue and green, corresponding to summer, autumn, winter and spring. If `GROUPS` has 12 levels, it is assumed that they correspond to months, and pens 1 to 12 are given decreasing intensities within the seasonal shades in clusters of three. Thus pens 1 to 3 are given crimson, red and salmon for the summer months. Note that this is tuned to a southern hemisphere calendar.

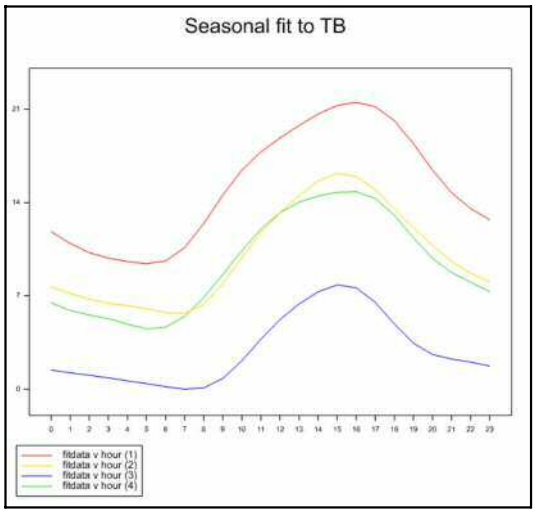


Figure 7.2.7f

Example 7.2.7

```
" Hourly temperatures at Tara Base, courtesy of Alison Rutherford."
IMPORT   [PRINT=*] '%gendir%/examples/DFOU-1.gsh'
DFOURIER [PRINT=accumulated,means; MODELTYPE=best;\
          PLOT=periodogram,harmonics,means,residuals,cumulative,range;\
          GROUPS=season; FACSHORT=hour; NCOMPONENTS=2; NHSHORTCYCLE=5;\
```

```
   7          NHLONGCYCLE=2; RANGE=!(0.13,0.225)] TB

Analysis of series TB, length 24120, showing sample spectrum with frequency
range divided into 80000 intervals.

Harmonic regression analysis: MODELTYPE = best
==============================================

Term               d.f.         s.s.         m.s.      v.r.   F pr.
       fd[1]          2    259276.75    129638.37   8859.64   0.000
       fd[2]          2     12099.20      6049.60    413.44   0.000
       fd[3]          2       302.21       151.11     10.33   0.000
       fd[4]          2       404.23       202.11     13.81   0.000
       fa[1]          2    572515.81    286257.91  19563.21   0.000
       fa[2]          2     10841.43      5420.71    370.46   0.000
   fad[1][1]          2     10620.24      5310.12    362.90   0.000
   fad[1][2]          2       332.34       166.17     11.36   0.000
   fad[2][1]          2       787.88       393.94     26.92   0.000
   fad[3][1]          2       753.38       376.69     25.74   0.000
   fad[3][2]          2       135.85        67.93      4.64   0.010
   fad[5][1]          2       184.83        92.42      6.32   0.002
   fda[1][1]          2      4274.13      2137.06    146.05   0.000
   fda[1][2]          2      1146.43       573.22     39.17   0.000
   fda[2][1]          2      1466.25       733.13     50.10   0.000
   fda[2][2]          2       149.85        74.92      5.12   0.006
   fda[3][1]          2       445.89       222.94     15.24   0.000
Residual          22581    330415.52        14.63
Total             22615   1206152.21        53.33

Table of means for Short Time Cycle by Group
============================================

       season          1          2          3          4
         hour
            0     11.821      7.675      1.441      6.479
            1     10.939      7.196      1.230      5.904
            2     10.244      6.744      1.056      5.572
            3      9.836      6.443      0.850      5.287
            4      9.579      6.273      0.629      4.883
            5      9.414      6.048      0.419      4.533
            6      9.623      5.733      0.195      4.642
            7     10.616      5.681      0.002      5.457
            8     12.425      6.376      0.097      6.878
            9     14.562      7.923      0.798      8.613
           10     16.429      9.891      2.134     10.394
           11     17.798     11.730      3.751     12.006
           12     18.843     13.241      5.223     13.250
           13     19.786     14.532      6.396     14.039
           14     20.646     15.610      7.305     14.492
           15     21.283     16.183      7.824     14.777
           16     21.528     15.954      7.612     14.820
           17     21.198     14.963      6.511     14.302
           18     20.141     13.567      4.893     13.047
           19     18.418     12.115      3.449     11.359
           20     16.439     10.769      2.608      9.819
           21     14.741      9.598      2.259      8.758
           22     13.562      8.686      2.033      8.022
           23     12.702      8.052      1.741      7.301
```

7.2.8 The MCROSSPECTRUM procedure

MCROSSPECTRUM procedure

Performs a spectral analysis of a multiple time series (G. Tunnicliffe Wilson & R.P. Littlejohn).

Options

`PRINT` = *string token* Controls printed output (`description`); default `desc`

PLOT = *string tokens*	Variables for which to plot the analysis (explanatory, response); default expl, resp
CORRECT = *string token*	Whether to mean or trend correct the series (mean, linear, quadratic, none); default mean
BANDWIDTH = *scalar*	Bandwidth for smoothing, must be between 0 and 0.5; if unset, a default is calculated automatically
MAXLAG = *scalar*	Maximum lag for the time domain outputs; if unset, a default is calculated automatically
PROBABILITY = *scalar*	Probability value for confidence limits; default 0.95
TAPER = *scalar*	The proportion of data to be tapered using a cosine bell window; default 0
YLOG = *string token*	Whether to plot the univariate spectra with a $\log_{10}$-transformed y-axis (yes, no); default no

Parameters

Y = *variates*	Response time series
X = *variates or pointers*	Explanatory time series
SPECTRUM = *pointers*	Saves autospectra, co-spectra and quad-spectra
FREQUENCY = *variate*	Saves the frequency values at which the spectra are calculated
VARSPECTRUM = *pointers*	Saves information about the variation of the spectrum: coefficient of variation, degrees of freedom, and lower and upper multiplicative limits for the univariate spectra
MULTICOHERENCYSQUARED = *pointers*	Saves estimates, significance limits, lower and upper confidence limits for the squared multiple coherency between the response and explanatory series
PARTIALCOHERENCYSQUARED = *pointers*	Saves estimates, significance limits, lower and upper confidence limits for the squared partial coherency of the response series with each explanatory series
GAIN = *pointers*	Saves estimates, lower and upper limits for the estimated gain of response series from each of the explanatory series
PHASE = *pointers*	Saves estimates, lower and upper limits for the estimated phase of response series from each of the explanatory series
NOISESPECTRUM = *variates*	Saves the estimated spectrum of the noise process
IMPULSERESPONSE = *pointers*	Saves the impulse response from −maxlag to +maxlag: estimates and significance limit
LAGS = *variates*	Saves the lags for the impulse response
ACFNOISE = *variates*	Saves the ACF of the noise process

MCROSSPECTRUM performs a spectral analysis of a multiple time series. The response series is specified by the Y parameter. The explanatory series are specified by the X parameter; the setting can be a single variate if there is only one explanatory series, or a pointer of variates if there are several. All the series should be the same length, n say, and this must be greater than 10. There must also be no missing values and no restrictions. The ALIGN parameter can supply a variate, with a value for each explanatory variate, which specifies a shift s so that $X(t-s)$ is more closely aligned with $Y(t)$. These are used to improve the accuracy of the analysis but the results still relate to the original (unshifted) series.

The band-width of the smooth is specified by the BANDWIDTH option. If this is unset, a default is calculated automatically. If BANDWIDTH is less than 1/*n*, only the sample spectra are returned with no smoothing. The MAXLAG option defines the maximum lag for the time domain outputs. If this is not set, a default is calculated automatically. Also, if the supplied value of MAXLAG is too great in relation to the series length or the bandwidth used, then it is adjusted as necessary. The TAPER option specifies the tapering proportion (default 0), and the PROBABILITY option defines the size of confidence limits and acceptance region for coherencies (default 0.95).

The CORRECT option has settings mean, linear, quadratic and none to control whether a mean, linear or quadratic trend correction is applied to all the series. The default is mean correction.

Printed output can be suppressed by setting the option PRINT=*; by default, PRINT=description, which summarizes the variables used and the option settings. The plots that are produced are controlled by the PLOT option, with settings:

explanatory	produces a graphics page for each explanatory variable containing the spectrum, its partial coherency squared with the response variable, phase, gain and impulse response function,
response	produces a graphics page with the response and noise spectra, the multiple coherency squared, and the autocorrelation function for the noise process. Where given, green lines denote null significance limits.

By default, both pages are produced.

The YLOG option specified the transformation to be made to the y-axes of the autospectra plots. By default, the plot is on the natural, untransformed scale. Alternatively, you can set YLOG=yes, to plot on the scale of logarithm, base 10.

The SPECTRUM parameter saves a pointer, with 2 suffixes, storing variates of spectra: "diagonals" (e.g. [1][1], [2][2] etc.) store autospectra, "super-diagonals" ([1][2] etc.) store co-spectra, and "sub-diagonals" ([2][1] etc.) store quad-spectra. The frequency values at which the spectra are calculated can be saved, in a variate, by the FREQUENCY parameter. The frequency range is from 0 to 0.5 cycles per sampling interval of the series. This range is divided into a round number of intervals with approximately 10 divisions covering one bandwidth.

The VARSPECTRUM parameter saves a pointer with information about the variation of the spectrum. The first element of the pointer is a variate storing the coefficient of variation of the spectrum. Similarly the second element stores the corresponding degrees of freedom, and the third and fourth elements store lower and upper multiplicative limits for the univariate spectra.

The MULTICOHERENCYSQUARED parameter saves a pointer containing the squared multiple coherency between the response and explanatory series. The first element of the pointer is a variate storing the estimates, the second element stores the significance limits, and the third and fourth elements store the lower and upper confidence limits.

The PARTIALCOHERENCYSQUARED, GAIN, PHASE and IMPULSERESPONSE parameters each save their results in variates within a pointer with two suffixes. The first suffix changes according to the type of result, while the second suffix has an element 1...*m* for each of the *m* explanatory variates. The PARTIALCOHERENCYSQUARED parameter saves results for the squared partial coherency of response series with the explanatory series; its first suffix has elements 1-4 to store the estimates, the significance limits, and the lower and upper confidence limits. The GAIN and PHASE parameters save the estimated gain and phase of response series from each of the explanatory series; their first suffixes have elements 1...3, storing the estimates, the lower and the upper limits. The IMPULSERESPONSE parameter saves the impulse response, from −maxlag to +maxlag; its first suffix has elements 1 and 2, storing the estimates and the significance limits.

The NOISESPECTRUM and ACFNOISE parameters store the estimated spectrum and ACF of

the noise process, in a variate. Finally, the LAGS parameter stores the lags for the impulse response, again in a variate.

7.3 ARIMA modelling

An ARIMA model is an equation relating the present value y_t of an observed time series to past values. The equation includes lagged values not only of the series itself, but also of an unobserved series of *innovations*, a_t; you can interpret the innovations as the error in predicting y_t from past values y_{t-1}, y_{t-2} .... The usual statistical model assumes that the innovations are a series of independent Normal deviates with mean zero and constant variance. The residuals obtained from fitting the model can be used to estimate the innovations.

A time-series model is specified by three things: the orders, which are the numbers of lagged values that appear in the equation; the parameters, which are the associated coefficients; and, optionally, the actual values of the lags, if these differ from the progression 1...*m*, where *m* is the number of lags. For example, consider the model

$$\nabla y_t - c = \varphi_1(\nabla y_{t-1} - c) + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2}$$

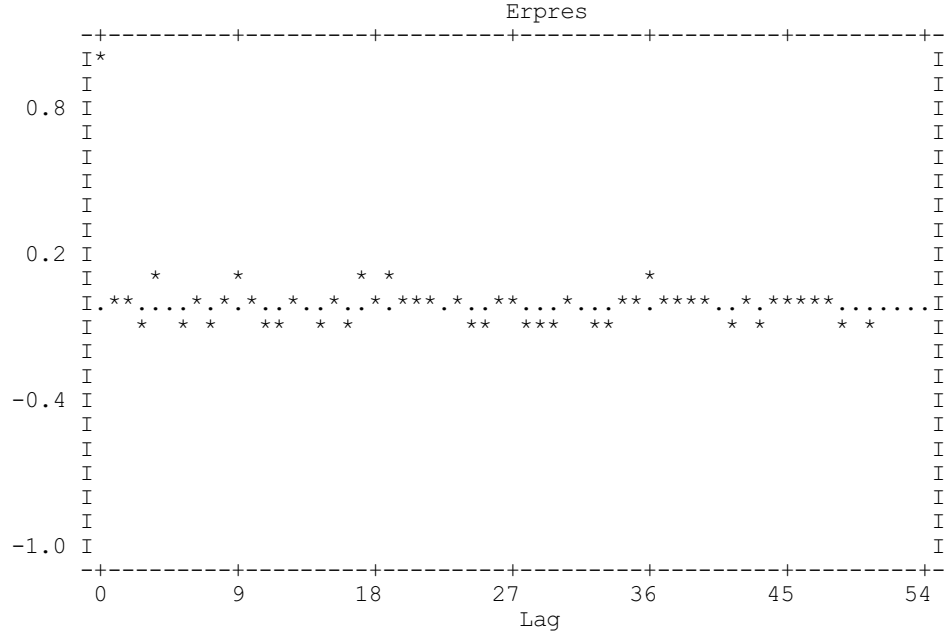
This equation is for the first differences, ∇y_t, of the data, and so has *differencing order d*=1. The *constant term c* represents the mean of ∇y_t. The model has *autoregressive order p*=1 with one parameter φ_1, and *moving-average order q*=2 with parameters θ_1 and θ_2.

Example 7.3 fits this model to a series of length 150, and produces forecasts of the next 10 points.

Example 7.3

```
  2  " Fit an ARIMA(1,1,2) model to the series of daylengths, 1821-1970.
 -3    Display the correlations, check the residuals, and forecast till 1980.
 -4    Data from Shi-fang et al. (1977)."
  5  OPEN 'Daylength.dat'; CHANNEL=3
  6  READ [CHANNEL=3; SETNVALUES=yes] Daylength

   Identifier   Minimum      Mean   Maximum    Values   Missing
    Daylength    -347.0     63.88     421.0       150         0

  7  CLOSE 3
  8  TSM Erp; ORDERS=!(1,1,2)
  9  TFIT Daylength; TSM=Erp

Time-series analysis
====================

Output series: Daylength    Noise model: Erp

Residual deviance           = 36959.
Innovation variance         = 251.9

Number of units present     = 150
Residual degrees of freedom = 145


Summary of models
-----------------

          Orders:  Delay     AR   Diff     MA   Seas
Model     Type          B      P      D      Q      S

Erp       ARIMA         -      1      1      2      1
```

```
Parameter estimates
-------------------

Model  Seas.   Diff.  Delay  Parameter  Lag  Ref  Estimate     s.e.      t
       Period  Order

Noise       1      0      -   Constant   -    1      3.98     4.52   0.88
            1      1      -   Phi (AR)   1    2     0.380    0.104   3.64
                            Theta (MA)   1    3   -0.5565   0.0897  -6.20
                                         2    4   -0.6194   0.0794  -7.80

  10  TKEEP RESIDUALS=Erpres
  11  CORRELATE [MAXLAG=50; GRAPH=autocorrelations] Erpres
```




```
  12  TFORECAST [MAXLEAD=10]

Forecasts
=========

Maximum lead time: 10


Forecasts for future values
---------------------------

    Lead time     forecast  lower limit  upper limit
            1        297.0        270.9        323.1
            2        305.8        248.9        362.7
            3        311.6        216.6        406.5
            4        316.2        188.3        444.2
            5        320.5        164.4        476.5
            6         325.         144.         505.
            7         329.         126.         531.
            8         333.         111.         555.
            9         337.          96.         577.
           10         341.          83.         598.
```

The `TSM` statement specifies the orders (p,d,q) of the model as (1,1,2), and names the model `Erp` (for Earth rotation period). The parameters of the model could also have been specified here; but they have been omitted because they have yet to be estimated. The initial values for c, φ_1, θ_1 and θ_2 are therefore set by Genstat to zero (the default).

The `TFIT` statement fits the model to the series by an iterative process, and, in this example,

the maximum number of iterations and the convergence criterion are determined by default. The results display the estimated *innovation variance* (or residual variance) and estimates of the other model parameters together with their standard errors. Note that the model also allows for a transformation parameter, which by default is not estimated and has the fixed value of 1.0 indicating no transformation.

The `TKEEP` statement accesses the variate of residuals a_t; these can also be thought of as the estimated innovations. `CORRELATE` is used to plot their autocorrelations as a way of checking that the fitted model accounts for all the correlation in the data.

Finally the `TFORECAST` statement prints the forecasts of the next 10 values of the series together with their 90% probability limits.

You can use the `RESTRICT` directive (1:4.4.1) to fit models to unbroken sub-series of the data. Genstat automatically estimates missing values in a time series together with the model parameters: all these estimates are allowed for in the number of degrees of freedom.

Further examples of all these directives are shown in Section 7.3.7. There is also a procedure `BJESTIMATE` which allows most of the analyses in Example 7.3 to be carried out by issuing a one-line command.

7.3.1 The `BJESTIMATE` procedure

`BJESTIMATE` procedure

Fits an ARIMA model, with forecast and residual checks (G. Tunnicliffe Wilson & S.J. Welham).

Options

`PRINT` = *string tokens*	Controls printed output (`description`, `monitoring`, `model`); default `desc`, `moni`, `mode`
`GRAPHICS` = *string token*	What type of graphics to use (`lineprinter`, `highresolution`); default `high`
`WINDOWS` = *scalar* or *variate*	Windows to be used for residual plots: a scalar `N` indicates that plots are to be produced on separate pages in window `N` (as currently defined), whereas a variate specifies four separate windows to be redefined (within the procedure) for plotting four graphs on one page; default 1
`PENS` = *variate*	The three pens to be used (after being defined appropriately) for drawing the plots; default `!(1,2,3)`

Parameters

`SERIES` = *variates*	Holds the time series to which the model is to be fitted
`LENGTH` = *scalars* or *variates*	Specifies the units to be used from each series: a scalar `N` indicates that the first `N` units of the series are to be used, a variate of length 2 gives the index of the first and last units of the subseries to be used; by default the whole series is used
`ORDERS` = *variates*	Variate holding the orders for the ARIMA model to be fitted to each series
`PARAMETERS` = *variates*	Variate specifying the initial values for the parameters (to be used by the `TFIT` directive)
`TSM` = *TSMs*	TSM to store each fitted model, also to supply values for orders and parameters if `ORDERS` and `PARAMETERS` are unset

RESIDUALS = *variates*	Variate to save the residuals from fitting the model to each series

BJESTIMATE fits an ARIMA model of specified orders to a time series given by the SERIES parameter. If only part of the series is to be used, this should be specified by the parameter LENGTH, using either a scalar N to indicate that the first N values should be used, or a variate of length 2 holding the positions of the first and last units of the subseries to be included. If only a subseries is used in the estimation, forecasts of any later series values are plotted to act as a check on the fitted model. The fit of the model is examined using the procedure BJIDENTIFY on the residual series; this residual series is plotted, together with its sample autocorrelations, partial autocorrelations and periodogram. The residuals from the fitted model can be saved using the RESIDUALS parameter.

The orders of the ARIMA model can be specified by the ORDERS parameter; alternatively, if parameter TSM has been set to the identifier of a TSM structure to save the results, ORDERS can be omitted and the orders will be taken from those of the TSM. Likewise, the PARAMETERS parameter can be set to a variate of initial values for the TFIT directive, used by the procedure to fit the model; if PARAMETERS is unset these will again be taken from the setting of the TSM parameter, if available. Any unset initial values are determined automatically by TFIT.

Printed output is controlled by the option PRINT; by default, a description of the series, monitoring of the estimation process and the fitted model are printed.

Graphical output is controlled by the options GRAPHICS, WINDOWS and PENS. Option GRAPHICS controls whether plots are produced for line-printer output or on the current high-resolution graphics device; by default high-resolution plots are given. Option WINDOWS controls the way in which the high-resolution plots are arranged. First of all there may be a graph of forecasts; this is plotted on a new page (i.e. a cleared screen), using the first window specified. Then procedure BJIDENTIFY is called to produce four different plots of residuals. If WINDOWS is set to a scalar N, the graphs are all produced in window N on separate pages; the FRAME directive can be used to set the attributes of window N before calling the procedure. Alternatively, WINDOWS can be set to a variate of length four; the attributes of the four windows specified are then redefined within the procedure so that four graphs are produced on the same page. By default WINDOWS=1. The PENS option controls which pens are used for the plots; the attributes of these pens are modified appropriately within the procedure. By default pens 1-3 are used, but these can be changed by setting option PENS to a variate of length 3 containing the numbers of the three different pens required.

Example 7.3.1 illustrates the use of BJESTIMATE by fitting an ARIMA model to the first 40 points of the series of Gilts from Example 7.1.1. If a subset of the series is used in procedure BJESTIMATE, graphs of forecasts are produced for any later timepoints. In this example, the last 8 points are forecast and plotted with the actual values as displayed in Figure 7.3.1a. A comparison of the forecasts to the actual data provides a simple validation of the fitted model.

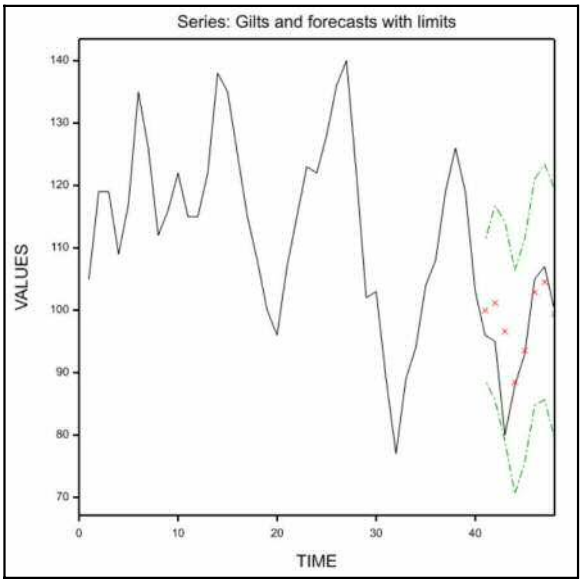


Figure 7.3.1a

The residuals are also analysed using procedure BJIDENTIFY within BJESTIMATE, producing the graphs shown in Figure 7.3.1b. Here only the series and the model orders are specified. The model contains a seasonal part; this is described in Section 7.3.2.

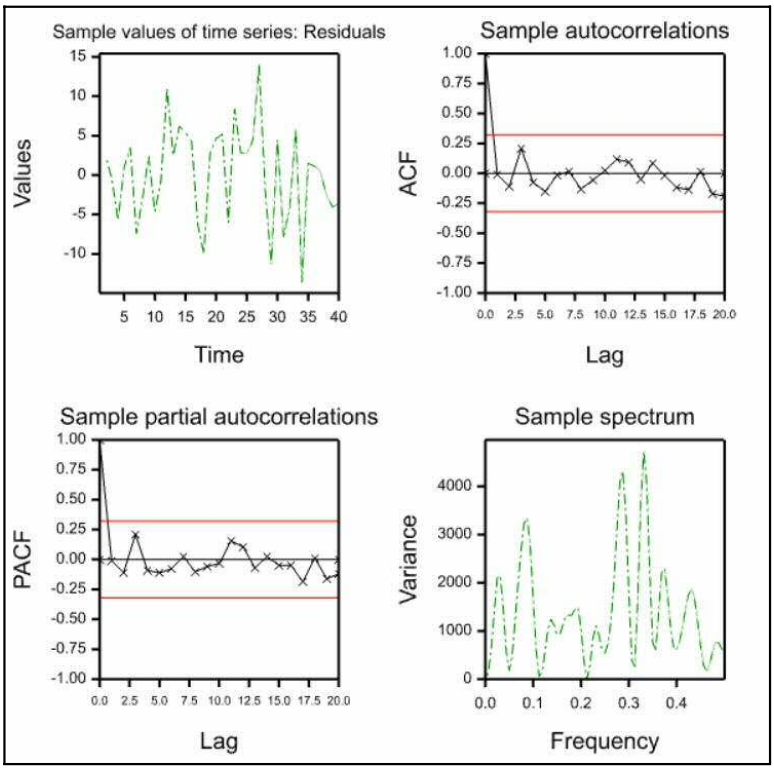


Figure 7.3.1b

Example 7.3.1

```
  2  " Fit a seasonal ARIMA model to the first 10 years of the quarterly
 -3    time series of Gilts used in Example 7.1.1 using procedure BJESTIMATE.
 -4    The final two years of data are forecast as a form of cross-validation
 -5    of the model, and the residuals are analysed."
  6  OPEN  'UKpig.dat';CHANNEL=3
  7  READ  [PRINT=errors; CHANNEL=3] \
  8        Year,Quarter,Gilts,Profit,Slaughter,Cleanpig,Herdsize
  9  CLOSE 3
 10  BJESTIMATE [GRAPHICS=high; WINDOWS=!(1,2,3,4)] SERIES=Gilts; LENGTH=40;\
 11        ORDERS=!(2,0,1,0,1,1,4)

Analysis of series x: first 40     values of series Gilts, length 48


Time-series analysis
====================

Output series: x            Noise model: amod

Residual deviance           = 1768.
Innovation variance         = 46.07

Number of units present     = 40
Residual degrees of freedom = 31


Summary of models
-----------------

          Orders:  Delay     AR   Diff     MA   Seas
Model     Type          B      P      D      Q      S

amod      ARIMA         -      2      0      1      1
                        -      0      1      1      4
```

```
Parameter estimates
-------------------

Model  Seas.   Diff.  Delay  Parameter  Lag  Ref  Estimate    s.e.      t
       Period  Order

Noise       1      0      -   Constant   -    1     -1.893   0.741  -2.55
                              Phi (AR)   1    2      1.625   0.104  15.62
                                         2    3    -0.8691  0.0906  -9.60
                            Theta (MA)   1    4      0.649   0.197   3.29
            4      1      - Theta (MA)   4    5      0.777   0.172   4.51
```

7.3.2 Defining ARIMA models for time series with the TSM directive

TSM directive

Declares one or more TSM data structures.

Option

MODELTYPE = *string token*	Type of model (arima, transfer); default arim

Parameters

IDENTIFIER = *identifiers*	Identifiers of the TSMs
ORDERS = *variates*	Orders of the autoregressive, integrated and moving-average parts of each TSM
PARAMETERS = *variates*	Parameters of each TSM
LAGS = *variates*	Lags, if not default

Here we describe how to use the TSM directive for ARIMA models, which correspond to the default setting of its MODELTYPE option (MODELTYPE=arima). The definition of transfer-function models is described in Section 7.5.1.

In many applications you will need only a simple form of the directive, such as:

```
TSM Erp; ORDERS=!(1,1,2)
```

Notice that TSM simply sets up a named Genstat structure which you can then use in directives such as TFIT. It can also, for example, be saved in a backing-store file (3.5) for further use. In that sense it is analogous to a TERMS statement (3.2.3), which sets up a maximal model for regression analysis, or a TREATMENTSTRUCTURE statement (4.1.1), which sets up a treatment model for analysis of variance.

If a TSM identifier, say Erp, has been declared, you can print the whole model in a descriptive format with the statement:

```
PRINT Erp
```

You can refer to the variates corresponding to the ORDERS, PARAMETERS and LAGS of the TSM by Erp[1], Erp[2] and Erp[3], or for example by Erp['Orders']. Thus the autoregressive order can be assigned to a scalar P by:

```
CALCULATE P = Erp[1]$[1]
```

since Erp[1] holds the orders of the TSM and its first element is the number of autoregressive parameters.

You can change the values of a TSM at any time, for example by CALCULATE statements. Genstat checks that the TSM values specify a valid model whenever they are used in a time-series directive such as TFIT. However, you must be careful if you change the values of a TSM that you are currently using to fit a model. For example, you could get strange results if you changed the parameter values of the model between the TFIT and TFORECAST statements in

Example 7.3.

Using the notation of Box & Jenkins (1970), the simple non-seasonal ARIMA model for the time series y_t is

$$\varphi(B)\ \{\nabla^d y_t^{(\lambda)} - c\} = \theta(B)a_t$$

where B is the backward shift operator $B^p y_t = y_{t-p}$,
∇ is the differencing operator $\nabla y_t = y_t - y_{t-1}$, $\nabla^d y_t = \nabla^{d-1}(y_t - y_{t-1})$, and

$$\varphi(B) = 1 - \varphi_1 B - \ldots - \varphi_p B^p$$
$$\theta(B) = 1 - \theta_1 B - \ldots - \theta_q B^q$$

The parameter λ specifies a Box-Cox power transformation defined by

$$y_t^{(\lambda)} = (y_t^{\lambda} - 1) / \lambda, \quad \lambda \neq 0$$
$$y_t^{(0)} = \log(y_t)$$

However, in the default case when λ is fixed and not estimated, the value λ=1 implies no transformation and then $y_t^{(1)} = y_t$ rather than $y_t - 1$. If $\lambda \neq 1$ or if λ is to be estimated, then Genstat will not let you have values of $y_t \leq 0$. The usual case however is that λ=1 and is not to be estimated, so that y_t may take any values.

The ORDERS parameter is a list of variates, one for each of the models. For each simple ARIMA model, the variate contains the three values p, d and q.

The PARAMETERS parameter is a list of variates, one for each of the models. For each simple ARIMA model, the variate contains $(3+p+q)$ values: λ, c, σ_a^2, $\varphi_1 \ldots \varphi_p$, $\theta_1 \ldots \theta_q$. You must always include the first three parameters. The parameter σ_a^2 is the innovation variance.

Whenever a TSM is used, Genstat checks its values. The orders must all be non-negative. The parameters λ and c can take any values, but σ_a^2 must be non-negative. The next $p+q$ values specify the autoregressive and moving-average parameters: they must satisfy the stationarity and invertibility conditions for ARIMA models (see Box & Jenkins 1970). An exception is that before estimation the model parameters may be unset, in which case Genstat sets them to default values. You can omit the PARAMETERS parameter, in which case an unnamed structure is defined to contain the default values. However, you should usually specify the variate of parameters, and if possible assign good preliminary values before estimation (see 7.7.1) as this will speed up the model fitting process.

For convenience when setting the values of parameters, you may wish first to declare scalars or variates containing the separate components:

```
SCALAR Lam,C,Ivar; VALUES=1,4,200
VARIATE [VALUES=0.4] Phi
& [VALUES=-0.5,-0.6] Theta
```

Then to pack these into the parameter variate, you can put

```
VARIATE [VALUES=Lam,C,Ivar,#Phi,#Theta] Erpar
```

Similarly, in order to extract the components after estimation, you can use the EQUATE directive (1:4.3):

```
EQUATE Erpar; NEWSTRUCTURES=!P(Lam,C,Ivar,Phi,Theta)
```

The LAGS parameter is a list of variates, one for each of the models. For each simple ARIMA model, this variate contains $p+q$ values, one corresponding to each of the autoregressive and moving-average parameters. Genstat then modifies the ARIMA model by defining

$$\varphi(B) = 1 - \varphi_1 B^{l_1} - \ldots - \varphi_p B^{l_p}$$

$$\theta(B) = 1 - \theta_1 B^{m_1} - \ldots - \theta_q B^{m_q}$$

The LAGS parameter for this model contains $l_1 \ldots l_p$, $m_1 \ldots m_q$. The sequences of lags $l_1 \ldots l_p$ must be positive integers that are strictly increasing; the default values are $1 \ldots p$ if LAGS is not set. The same rule applies to $m_1 \ldots m_q$.

The seasonal ARIMA model for the time series y_t is an extension of the simple model, to the

form

$$\varphi(B)\,\Phi(B^s)\,\{\,\nabla^d\nabla_s^D y_t^{(\lambda)} - c\,\} = \theta(B)\,\Theta(B^s)\,a_t$$

where the extra, seasonal, operators associated with seasonal period s are of three types:

$$\Phi(B^s) = 1 - \Phi_1 B^s - \ldots - \Phi_P B^{Ps}$$

which is seasonal autoregression of order P;

$$\nabla_s^D$$

which is seasonal differencing of order D; and

$$\Theta(B^s) = 1 - \Theta_1 B^s - \ldots - \Theta_Q B^{Qs}$$

which is seasonal moving average of order Q.

When seasonal terms are to be included, you must extend the ORDERS parameter so that it contains p, d, q, P, D, Q and s. Even if the non-seasonal part of the model has p=d=q=0, these parameters must still be included at the beginning of the list. The seasonal orders must satisfy $P \geq 0$, $D \geq 0$, $Q \geq 0$ and $s \geq 1$.

You must also extend the PARAMETERS parameter to contain:

λ, c, σ_a^2, $\varphi_1 \ldots \varphi_p$, $\theta_1 \ldots \theta_q$, $\Phi_1 \ldots \Phi_P$, $\Theta_1 \ldots \Theta_Q$

You can modify the seasonal model to allow other lags:

$$\Phi(B^s) = 1 - \Phi_1 B^{L_1} - \ldots - \Phi_P B^{L_P}$$

$$\Theta(B^s) = 1 - \Theta_1 B^{M_1} - \ldots - \Theta_Q B^{M_Q}$$

The sequence of lags $L_1 \ldots L_P$ must be strictly increasing and must be positive-integer multiples of the period s; the default values are s, $2s$ ... Ps. The same rules apply to $M_1 \ldots M_Q$. For any seasonal model, you must extend the LAGS parameter, if supplied, so that it contains

$l_1 \ldots l_p, m_1 \ldots m_q, L_1 \ldots L_P, M_1 \ldots M_Q$.

You can use multiple seasonal periods, by extending the variate of ORDERS with further seasonal orders P', D', Q' and s'. You must correspondingly extend the variates of PARAMETERS and LAGS. It is also possible to set the seasonal periods to 1, which means you can estimate non-seasonal models with factored operators.

You can declare an ORDERS variate to have more values than is necessary, provided that the extra values are filled with zeroes, and that the number of values is $3+4k$, k being the number of seasonal periods. The same applies to PARAMETERS and LAGS variates, except that Genstat ignores the extra values whatever they may be. Thus you can extend a simple model to a seasonal model, simply by resetting the extra values.

Finally note that you can use the same ORDERS, PARAMETERS and LAGS variates in more than one TSM.

7.3.3 The TFIT directive

TFIT directive

Estimates parameters in Box-Jenkins models for time series.

Options

PRINT = *string tokens*	What to print (model, summary, estimates, correlations, monitoring); default mode,summ,esti
LIKELIHOOD = *string token*	Method of likelihood calculation (exact,

	leastsquares, marginal); default exac
CONSTANT = *string token*	How to treat the constant (estimate, fix); default esti
RECYCLE = *string token*	Whether to continue from previous estimation (yes, no); default no
WEIGHTS = *variate*	Weights; default *
MVREPLACE = *string token*	Whether to replace missing values by their estimates (yes, no); default no
FIX = *variate*	Defines constraints on parameters (ordered as in each model, tf models first): zeros fix parameters, parameters with equal numbers are constrained to be equal; default *
METHOD = *string token*	Whether to carry out full iterative estimation, to carry out just one iterative step, to perform no steps but still give parameter standard deviations, or only to initialize for forecasting by regenerating residuals (full, onestep, zerostep, initialize); default full
MAXCYCLE = *scalar*	Maximum number of iterations; default 15
TOLERANCE = *scalar*	Criterion for convergence; default 0.0004
SAVE = *identifier*	To name save structure, or supply save structure with transfer-functions; default * i.e. transfer-functions taken from the latest model

Parameters

SERIES = *variate*	Time series to be modelled (output series)
TSM = *TSM*	Model for output series
BOXCOXMETHOD = *string token*	How to treat transformation parameter in output series (fix, estimate); default fix
RESIDUALS = *variate*	To save residual series

The main use of TFIT is to fit parameters to time-series models, although you can also use it to initialize for the TFORECAST directive, even when the model parameters are already known. In many applications of estimating a univariate ARIMA model, you will need only a simple form of the directive, such as:

```
TFIT Daylength; TSM=Erp
```

Examples of TFIT are given at the beginning of Section 7.3 and in Section 7.3.7.

The SERIES parameter specifies the variate holding the time series data to which the model is to be fitted.

The TSM parameter specifies the ARIMA model that is to be fitted to the time-series data. This TSM must already have been declared and its ORDERS must have been set. If the LAGS parameter of the TSM has been set, the lags must have been given values. However, if the PARAMETERS of the TSM model have been set, these need not have been declared previously nor given values. When the parameter values are not set, default values are used: these are all zero, except for the transformation parameter, which is set to 1.0 if it is not to be estimated (see BOXCOXMETHOD and FIX below). Any parameter values that you do specify will be used as initial values for the parameters in the model; Genstat replaces any missing values by the default values. If any group of autoregressive or moving-average parameters do not satisfy the required conditions for stationarity or invertibility, all the parameters to be estimated are reset by Genstat to the default values. After TFIT, the parameters of the TSM contain the estimated parameter values.

The BOXCOXMETHOD parameter allows you to estimate the transformation parameter λ.

The RESIDUALS parameter saves the estimated innovations (or residuals). As explained in the

description of the LIKELIHOOD option in the next section, the residuals are calculated for $t=t_0...N$, where $t_0=1+p+d-q$ for a simple ARIMA model. If $t_0>1$, missing values will be inserted for $t=1...t_0-1$.

The PRINT option controls printed output. If you specify monitoring, then at each cycle of the iterative process of estimation, Genstat prints the *deviance* (7.3.4) for the current fitted model, together with the current estimates of model parameters. The format is simple with the minimum of description, to let you judge easily how quickly the process is converging; see Example 7.4a. The other settings of PRINT control output at the end of the iterative process. If you specify model, the model is briefly described, giving the identifier of the series and the time-series model, together with the orders of the model. If you specify summary, the deviance of the final model is printed, along with the residual number of degrees of freedom. If you specify estimates, the estimates of the model parameter are printed in a descriptive format, together with their estimated standard errors and reference numbers. If you specify correlations, the correlations between estimates of parameters are printed, with reference numbers to identify the parameters; see Example 7.3.5.

The LIKELIHOOD option specifies the criterion that Genstat minimizes to obtain the estimates of the parameters: this is described in the next section. The default setting exact is recommended for most applications.

You can use the CONSTANT option to specify whether Genstat is to estimate the constant term c in the model. If CONSTANT=fix, the constant is held at the value given in the initial parameter values; this need not be zero.

The RECYCLE option allows a previous TFIT statement to continue; this can save computing time. If RECYCLE=yes, the most recent TFIT statement is continued, unless the SAVE option has been set to the save structure from some other TFIT statement. The SERIES and TSM settings are then taken from this previous TFIT statement: Genstat ignores any specified in the current statement. Most of the settings of other parameters and options are carried over from the previous statement, and new values are ignored. However, there are some exceptions. You can change the RESIDUALS variate, you can reset MAXCYCLE to the number of further iterations you require, and you can change the settings of TOLERANCE and PRINT. You can also change the values of the variate in the WEIGHTS option; you can thus get reweighted estimation. You can change the values of the SERIES itself, although you cannot change missing values; if the MVREPLACE option was previously set to yes, you must put the original missing values back into the SERIES variate before the new TFIT statement.

The WEIGHTS option includes in the likelihood a weighted sum-of-squares term

$$\sum_{t=t_0}^{N} w_t \, a_t^2$$

where w_t, $t=1...N$ are provided by the WEIGHTS variate. The values of w_t must be strictly positive. If $t_0<1$, where $t_0=1+d+p-q$, then w_t is taken as 1 for $t<1$.

The MVREPLACE option allows you to request any missing values in the time-series to be replaced by their estimates after estimation. Genstat will always estimate the missing values, irrespective of the setting of MVREPLACE; so you can also obtain these estimates later from TKEEP (7.3.6).

The FIX option allows you to place simple constraints on parameter values throughout the estimation. The units of the FIX variate correspond to the parameters of the TSM, excluding the innovation variance. The values of the FIX variate are used to define the parameter constraints and must be integers. If an element of the FIX variate is set to 0, the corresponding parameter is constrained to remain at its initial setting. If an element is not 0, and the value is unique in the FIX variate, the parameter is estimated without any special constraint. If two or more values are equal, the corresponding parameters are constrained to be equal throughout the estimation. The

number that you give to a parameter by FIX will appear as the reference number of the parameter in the printed model and correlation matrix. This option overrides any setting of CONSTANT and BOXCOXMETHOD. Example 7.3.3a uses the FIX option to constrain some of the parameters in the model as fitted in Example 7.3.

Example 7.3.3a

```
   2  " Fix parameters in ARIMA(1,1,2) model for daylength:
  -3    transformation fixed at 1, Constant unconstrained, AR parameter
  -4    fixed at previous estimate, MA parameters constrained to be equal."
   5  OPEN  'Daylength.dat'; CHANNEL=3
   6  READ  [PRINT=errors; CHANNEL=3; SETNVALUES=yes] Daylength
   7  CLOSE 3
   8  TSM   Erp; ORDERS=!(1,1,2)
   9  TFIT  [PRINT=*] Daylength; TSM=Erp
  10  TFIT  [FIX=!(0,1,0,2,2)] Daylength; TSM=Erp

Time-series analysis
====================

Output series: Daylength    Noise model: Erp

Residual deviance           = 37102.
Innovation variance         = 249.5

Number of units present     = 150
Residual degrees of freedom = 147


Summary of models
-----------------

          Orders:  Delay     AR   Diff     MA   Seas
Model     Type          B      P      D      Q      S

Erp       ARIMA         -      1      1      2      1


Parameter estimates
-------------------

Model  Seas.   Diff.  Delay  Parameter  Lag  Ref  Estimate    s.e.      t
       Period  Order

Noise       1      0      -   Constant   -    1       3.97    4.51   0.88
            1      1      -   Phi (AR)   1    0    0.38013   Fixed      -
                             Theta (MA)   1    2    -0.5906  0.0596  -9.90
                                          2    2    -0.5906  0.0596  -9.90
```

The MAXCYCLE option specifies the maximum number of iterations to be performed.

The TOLERANCE option specifies the convergence criterion. Genstat decides that convergence has occurred if the fractional reduction in the deviance in successive iterations is less than the specified value, provided also that the search is not encountering numerical difficulties that force the step length in the parameter space to be severely limited. You can use monitoring to judge whether, for all practical purposes, the iterations have converged. Genstat gives warnings if the specified number of iterations is completed without convergence, or if the search procedure fails to find a reduced value of the deviance despite a very short step length. Such an outcome may be due to complexities in the likelihood function that make the search difficult, but can be due to your specifying too small a value for TOLERANCE.

The SAVE option allows you to save the *time-series save structure* produced by TFIT. You can use this in further TFIT statements with RECYCLE=yes, or in TFORECAST statements. It can also be used by the TDISPLAY and TKEEP directives. Genstat automatically saves the structure from the most recent TFIT statement, but this is over-written when the next TFIT statement is

executed, unless you have used SAVE to give it an identifier of its own. You can access the current time-series save structure by the SPECIAL option of the GET directive (1:5.6.2), and reset it by the TSAVE option of the SET directive (1:5.6.1).

The METHOD option has four possible settings. The default setting is full which gives the usual estimation to convergence or until the maximum number of iterations has been reached.

With the initialize setting of METHOD, TFIT carries out only the residual regeneration steps (that is, calculation of a_t for $t=t_0...N$) which are needed before TFORECAST (7.3.7) can be used. If the model has just been estimated using the default full setting, this is unnecessary. The setting initialize is useful when the time series is supplied with a known model and a minimal amount of calculation is wanted to prepare or initialize for forecasting. None of the model parameters are changed, and no standard errors of parameter estimates are available. Missing values in the series *are* estimated so this setting provides an efficient way of getting their values when the time series model is known; they can then be obtained using TKEEP (7.3.6). The deviance value is also available from TKEEP (7.3.6). This setting is therefore useful for efficient calculation of deviance values when you want to plot the shape of the deviance as a function of parameter values. Example 7.3.3b below illustrates this by producing the contour plot (shown in Figure 7.3.3) of the log deviance for the daylength model fitted in Example 7.3. All parameters have their estimated values except the two moving-average parameters. These vary over a grid of 800 points. Values corresponding to non-invertible models are skipped and the contours plotted inside the triangular region of invertible model parameters.

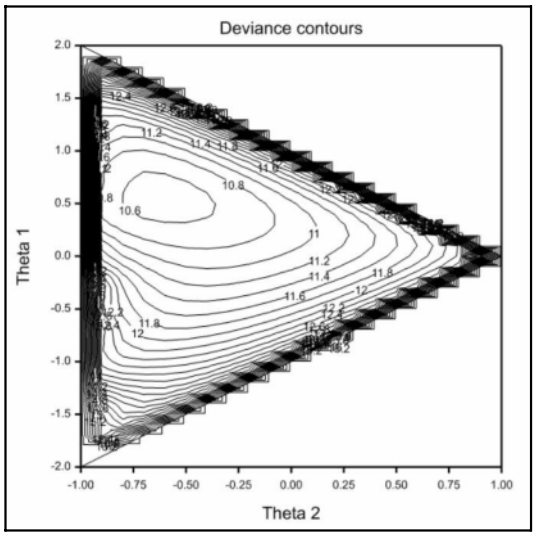


Figure 7.3.3

Example 7.3.3b

```
  2  " The deviance function for the model fitted to the series of
 -3    daylengths in Example 7.3 is plotted as the moving average
 -4    parameters are varied."
  5  OPEN 'Daylength.dat'; CHANNEL=3
  6  READ [CHANNEL=3; SETNVALUES=yes] Daylength

   Identifier   Minimum      Mean   Maximum    Values   Missing
    Daylength    -347.0     63.88     421.0       150         0

  7  CLOSE 3
  8  " Set the model parameters to their previously estimated values "
  9  VARIATE [VALUES=1,3.98,251.9,0.380,-0.5565,-0.6194] Modpar
 10  TSM Moderp; ORDERS=!(1,1,2); PARAMETERS=Modpar
 11  SCALAR R,Large,Mdev; VALUE=0.999,12000000,0
 12  " Set up a grid of parameter values over which to evaluate
-13    the deviance."
 14  CALCULATE Vth1,Vth2 = !(-20...20),!(-10...10)*0.099999
 15  " Define the matrix to hold the deviance values "
 16  MATRIX [ROWS=41; COLUMNS=21] Devgrid
 17  FOR Drow=1...41; Dth1=#Vth1
 18    FOR Dcol=1...21; Dth2=#Vth2
 19    " Check that the parameters lie within the invertibility region."
 20    IF ((ABS(Dth2)<R).AND.((ABS(Dth1)/ABS(1-Dth2))<R))
 21      CALCULATE Modpar$[5,6] = Dth1,Dth2
 22      TFIT      [PRINT=*; METHOD=initialize] Daylength; Moderp
 23      TKEEP     DEVIANCE=Mdev
 24    ELSE
 25      " Set the deviance to a large value if the parameters are not
```

```
-26         invertible."
 27       CALCULATE Mdev = Large
 28     ENDIF
 29     CALCULATE ELEMENT(Devgrid; Drow; Dcol) = Mdev
 30     ENDFOR
 31   ENDFOR
 32   " Use log deviances so as to reveal the lower contours."
 33   CALCULATE Devgrid = LOG(Devgrid)
 34   FRAME 1; YLOWER=0.05; YUPPER=0.95; XLOWER=0.05; XUPPER=0.95; BOX=include
 35   XAXIS 1; ACTION=hide
 36   YAXIS 1; ACTION=hide
 37   DCONTOUR [WINDOW=1; KEYWINDOW=0] Devgrid; PENCONTOUR=1; PENFILL=0;\
 38            INTERVAL=0.2
 39   PEN    2,4; LINESTYLE=1; METHOD=closed,line; SYMBOLS=0; COLOUR='black'
 40   XAXIS  1; TITLE='Theta 2'; LOWER=-1; UPPER=1; ACTION=display
 41   YAXIS  1; TITLE='Theta 1'; LOWER=-2; UPPER=2; ACTION=display
 42   DGRAPH [WINDOW=1; KEYWINDOW=0; SCREEN=keep; TITLE='Deviance contours']\
 43          !(2,0),!(-2,0);!(-1,1); PEN=4
```

With the setting METHOD=zerostep the effect is the same as for initialize except that TFIT also calculates the standard errors of the parameters as if they had just been estimated. These can be used together with other quantities available from TKEEP (7.3.6) to construct confidence intervals and carry out tests on the parameter values, which remain unchanged except that the innovation variance in the ARIMA model is replaced by its estimate conditional on all other parameters.

The setting METHOD=onestep gives the same results as specifying the option MAXCYCLE=1 in TFIT. It is convenient for carrying out quick tests of model parameters as illustrated in Example 7.3.3c. The model fitted in Example 7.3 is extended to have three autoregressive parameters, with the new parameters set to zero and the old parameters kept at their estimated values. Then after one step of TFIT the estimates of the new autoregressive coefficients at lags 2 and 3 can be compared with their standard errors to see if there is evidence that they should be retained in the model. In this case the evidence is insufficient. Although iteration to convergence would be very quick for this example, the onestep setting can save time when checking a complicated model for a variety of possible extensions.

Example 7.3.3c

```
  2  " The model previously fitted to the series of daylengths in Example
 -3    7.3 is extended to include two more autoregressive parameters, the
 -4    old parameters being kept at their estimated values. The option
 -5    METHOD=onestep of ESTIMATE is used to assess whether the new
 -6    parameters should be retained in the model."
  7  OPEN  '%GENDIR%/Examples/GuidePart2/Daylength.dat'; CHANNEL=3
  8  READ  [PRINT=errors; CHANNEL=3; SETNVALUES=yes] Daylength
  9  CLOSE 3
 10  TSM   Erp; ORDERS=!(1,1,2)
 11  TFIT  [PRINT=*] Daylength; TSM=Erp
 12  " Save the previous model parameters and redefine the model with
-13    higher autoregressive orders and extended parameter variate."
 14  CALCULATE Modpar = Erp['Parameters']
 15  &         Modparx = !(Modpar$[1,2,3,4],0,0,Modpar$[5,6])
 16  " Save the parameter values."
 17  VARIATE   Oldparx; VALUES=Modparx
 18  TSM       Erp; ORDERS=!(3,1,2); PARAMETERS=Modparx
 19  TFIT      [METHOD=onestep] Daylength; TSM=Erp

******** Warning 32, code TS 21, statement 1 on line 19

Command: TFIT [METHOD=onestep] Daylength; TSM=Erp
The iterative estimation process has not converged.
The maximum number of cycles is 1
```

```
Time-series analysis
====================

Output series: Daylength    Noise model: Erp

Residual deviance           = 36553.
Innovation variance         = 252.5

Number of units present     = 150
Residual degrees of freedom = 143


Summary of models
-----------------

          Orders:  Delay     AR   Diff     MA   Seas
Model     Type          B      P      D      Q      S

Erp       ARIMA         -      3      1      2      1


Parameter estimates
-------------------

Model   Seas.   Diff.  Delay  Parameter  Lag  Ref   Estimate        s.e.      t
        Period  Order

Noise        1      0      -   Constant   -    1       4.05        4.51   0.90
             1      1      -   Phi (AR)   1    2      0.319       0.155   2.05
                                          2    3      0.166       0.161   1.03
                                          3    4     -0.102       0.139  -0.73
                                Theta (MA)   1    5     -0.608       0.136  -4.46
                                          2    6     -0.544       0.138  -3.95
```

```
  20  " Calculate and print the changes in the parameter values excluding
 -21    the transformation and innovation variance parameters. "
  22  CALCULATE Delpar = Modparx-Oldparx
  23  &         Del = Delpar$[!(2,4,5...8)]
  24  PRINT     Del

         Del
     0.07414
    -0.06133
     0.16587
    -0.10153
    -0.05192
     0.07503
```

7.3.4 Technical information about how Genstat fits ARIMA models

This section describes the estimation of ARIMA models in more detail. You may want to skip this if you are doing fairly routine work.

The first step in deriving the likelihood for a simple model is to calculate

$$w_t = \nabla^d y_t - c\,, \qquad t = 1+d \ldots N$$

This has a multivariate Normal distribution with dispersion matrix $V\sigma_a^2$, where V depends only on the autoregressive and moving-average parameters. The likelihood is then proportional to

$$\{ \sigma_a^{2m} |V| \}^{-½} \exp\{ -w'V^{-1}w/2\sigma_a^2 \}$$

where $m=N-d$. In practice Genstat evaluates this by using the formula

$$w'V^{-1}w = W + \sum_{t=t_0}^{N} a_t^2 = S$$

where $t_0=1+d+p-q$. The term W is a quadratic form in the p values $w_{1+d-q} \ldots w_{p+d-q}$. It takes account of the starting-value problem for regenerating the innovations a_t, and avoids losing information as would happen if the process used only a conditional sum-of-squares function. If

q>0, Genstat introduces unobserved values of $w_{1+d-q} \ldots w_d$ in order to calculate the sum S. Genstat uses linear least-squares to calculate these q starting values for w, thus minimizing S. We shall call them *back-forecasts*, though if p>0 they are actually computationally convenient linear functions of the proper back-forecasts. We shall call S the sum-of-squares function: it is the sum of the quadratic form and the sum-of-squares term, and is identical to the value expressed by Box and Jenkins as

$$\sum_{t=-\infty}^{N} a_t^2$$

using infinite back-forecasting; that is, using:

$$W = \sum_{t=-\infty}^{t_0-1} a_t^2$$

The values a_t for t=t_0...N agree precisely with those of Box and Jenkins.

To clarify all this, consider examples with no differencing; that is, d=0. If p=0 and q=1, then W=0 and t_0=0, and one back-forecast w_0 is introduced. If p=1 and q=0, then $W=(1-\varphi_1^2)w_1^2$ and t_0=2, and no back-forecasts are needed. If p=q=1, then $W=(1-\varphi_1^2)w_0^2$ and t_0=1, and so one back-forecast w_0 is needed. In this case the proper back-forecast is in fact $w_0/(1-\theta_1\varphi_1)$.

The value of $|V|$ is a by-product of calculating W and the back-forecast. For example, if p=0 and q=1, then

$$|V| = (1 + \theta_1^2 + \ldots + \theta_1^{2N})$$

If p=1 and q=0,

$$|V| = 1 / (1 - \varphi_1^2)$$

and if p=q=1,

$$|V| = 1 + (\varphi_1 - \theta_1)^2 (1 + \theta_1^2 + \ldots + \theta_1^{2N-2}) / (1 - \varphi_1^2)$$

Concentrating the likelihood over σ_a^2 by setting σ_a^2=S/m yields a value proportional to $\{ |V|^{1/m} S \}^{-m/2}$.

The default setting of the `LIKELIHOOD` option is `exact`. In this case the concentrated likelihood is maximized, by minimizing the quantity

$$D = |V|^{1/m} S$$

which is called the deviance.

The setting `leastsquares` specifies that Genstat is to minimize only the sum-of-squares term S. This criterion corresponds to the back-forecasting sum-of-squares used by Box and Jenkins, and will in many cases give estimates close to those of the exact likelihood. However, some discrepancy arises if the series is short or the model is close to the invertibility boundary. This is because of limitations on the back-forecasting procedure, as described in the algorithms of Box and Jenkins. The deviance value D that Genstat prints is, with this setting, simply S.

The setting `marginal` is described in Section 7.4.

When you use exact likelihood, the factor $|V|^{1/m}$ reduces bias in the estimates of the parameter; you would get bias if you used `leastsquares` instead. However, $|V|^{1/m}$ is generally close to one, unless the series is short or the model is either seasonal or close to the boundaries of invertibility or stationarity. The `leastsquares` setting is therefore adequate for most long, non-seasonal sets of data; using it may reduce the computation time by up to 50%. When you specify that Genstat is to estimate the parameter λ of the Box-Cox transformation, Genstat also includes the Jacobian of the transformation in the likelihood function. The result is an extra factor $G^{-2(\lambda-1)}$ in the definition of the deviance, G being the geometric mean of the data,

$$G = \left(\prod_{t=1}^{N} y_t \right)^{\frac{1}{N}}$$

Note that this is not included unless λ is being estimated, even if $\lambda \neq 1$.

You can treat differences in $N\log(D)$ as a chi-square variable in order to test nested models: this is supported by asymptotic theory, and by experience with models that have moderately large sample sizes. Similarly, you can select between different models by using $N\log(D)+2k$ as an information criterion, k being the number of estimated parameters. But both of these test procedures are questionable if the estimated models are close to the boundaries of invertibility or stationarity. Provided all the models that are being compared have the same orders of differencing, with the differenced series being of length m, it is recommended that $m\log(D)$ be used rather than $N\log(D)$ in these tests since $m\log(D)$ is precisely minus two multiplied by the log-likelihood as defined above.

7.3.5 The TDISPLAY directive

TDISPLAY directive

Displays further output after an analysis by TFIT.

Options

PRINT = *string tokens*	What to print (model, summary, estimates, correlations); default mode,summ,esti
CHANNEL = *scalar*	Channel number for output; default * i.e. current output channel
SAVE = *identifier*	Save structure to supply fitted model; default * i.e. that from the last model fitted

No parameters

You can use TDISPLAY to print further output from an TFIT statement. However, if the TFIT statement used the setting METHOD=initialize you will not be able to print the standard errors or correlations between the parameter estimates (see 7.3.3).

The PRINT option has the same interpretation as in TFIT, except that information is not available to monitor convergence. Example 7.3.5 illustrates TDISPLAY in a continuation of Example 7.3.3a.

Example 7.3.5

```
  11  TFIT Daylength; TSM=Erp

Time-series analysis
====================

Output series: Daylength    Noise model: Erp

Residual deviance           = 36960.
Innovation variance         = 251.9

Number of units present     = 150
Residual degrees of freedom = 145


Summary of models
-----------------

          Orders:  Delay     AR   Diff     MA   Seas
Model     Type          B      P      D      Q      S

Erp       ARIMA         -      1      1      2      1
```

```
Parameter estimates
-------------------

Model  Seas.   Diff.  Delay  Parameter  Lag  Ref  Estimate    s.e.      t
       Period  Order

Noise       1      0      -   Constant   -    1      3.98    4.52   0.88
            1      1      -   Phi (AR)   1    2     0.380   0.105   3.63
                            Theta (MA)   1    3    -0.5581  0.0901  -6.19
                                         2    4    -0.6181  0.0797  -7.75

  12  TDISPLAY [PRINT=correlations]

Time-series analysis
====================


Correlations
------------

   1  1.000
   2  0.007  1.000
   3  0.004  0.662  1.000
   4 -0.008  0.497  0.559  1.000
         1      2      3      4
```

The CHANNEL option allows you to send the output to another output channel.

You can use the SAVE option to specify the time-series save structure (from TFIT) from which the output is to be taken. By default TDISPLAY uses the structure from the most recent TFIT statement.

7.3.6 The TKEEP directive

TKEEP directive

Saves results after an analysis by TFIT.

Option

SAVE = *identifier*	Save structure to supply fitted model; default * i.e. that from last model fitted

Parameters

OUTPUTSERIES = *variate*	Output series to which model was fitted
RESIDUALS = *variate*	Residual series
ESTIMATES = *variate*	Estimates of parameters
SE = *variate*	Standard errors of estimates
INVERSE = *symmetric matrix*	Inverse matrix
VCOVARIANCE = *symmetric matrix*	Variance-covariance matrix of parameters
DEVIANCE = *scalar*	Residual deviance
DF = *scalar*	Residual degrees of freedom
MVESTIMATES = *variate*	Estimates of missing values in series
SEMV = *variate*	Standard errors of estimates of missing values
COMPONENTS = *pointer*	Variates to save components of output series
SCORES = *variate*	To save scores (derivatives of the log-likelihood with respect to the parameters)

An TFIT statement produces many quantities that you may want to use to assess, interpret, and apply the fitted model. The TKEEP directive allows you to copy these quantities into Genstat data

structures. If the METHOD option of the TFIT statement was set to initialize, then the results saved by the options SE, INVERSE, VCOVARIANCE and SCORE are unavailable. However, you can save the estimates of the missing values and their standard errors. The residual degrees of freedom in this case does not make allowance for the number of parameters in the model, but does allow for the missing values that have been estimated.

The OUTPUTSERIES parameter specifies the variate that was supplied by the SERIES parameter of the TFIT statement; this can be omitted.

You can use the RESIDUALS parameter to save the residuals in a variate, exactly as in the TFIT directive.

The ESTIMATES parameter can supply a variate to store the estimated parameters of the TSM. Each estimated parameter is represented once, but the innovation variance is omitted entirely. Genstat includes only the first of any set of parameters constrained to be equal using the FIX option of TFIT. The order of the parameters otherwise corresponds to their order in the variate of parameters in TSM, and is unaffected by any numbering used in the FIX option.

The SE parameter allows you to specify a variate to save the standard errors of the estimated parameters of the TSM. The values correspond exactly to those in the ESTIMATES variate. Parameters in a time series model may be aliased. This is detected when the equations for the estimates are being solved, and the message ALIASED is printed instead of the standard error when the PRINT option of TFIT or TDISPLAY includes the setting estimates. The corresponding units of the SE variate are set to missing values.

The INVERSE parameter can provide a symmetric matrix to save the product $(X'X)^{-1}$, where X is the most recent design matrix derived from the linearized least-squares regressions that were used to minimize the deviance. The ordering of the rows and columns corresponds exactly to that used for the ESTIMATES variate. The row of this matrix corresponding to any aliased parameter is set to zero except that the diagonal element is set to the missing value.

The VCOVARIANCE parameter allows you to supply a symmetric matrix for the estimated variance-covariance matrix, $\hat{\sigma}_a^2(X'X)^{-1}$, of the TSM parameters. The ordering of the rows and columns and the treatment of aliased parameters corresponds exactly to that used for the ESTIMATES variate.

The DEVIANCE parameter specifies a scalar to hold the final value of the deviance criterion defined by the LIKELIHOOD option of TFIT.

The DF parameter saves the residual number of degrees of freedom, defined for a simple ARIMA model by $N-d-$(number of estimated parameters). If a seasonal model is used, this number is further reduced by Ds.

The MVESTIMATES parameter specifies a variate to hold estimates of the missing values of the series, in the order they appear in the series. You can thereby obtain forecasts of the series, by extending the SERIES in TFIT with a set of missing values. This is less efficient than using the TFORECAST directive, but it does have the advantage that the standard errors of the estimates take into account the finite extent of the data, and also the fact that the model parameters are estimated.

The SEMV parameter can supply a variate to hold the estimated standard errors of the missing values of the series, in the order they appear in the series.

The COMPONENTS parameter is used when there are explanatory variables, and is described in Section 7.5.4.

The SCORE parameter can specify a variate to hold the model scores. The scores are usually defined as the first derivatives of the log likelihood with respect to the model parameters. To get these, the scores supplied by TKEEP should be scaled by dividing by the estimated residual variance and reversing its sign. The elements of the SCORE variate correspond exactly to the parameters as they appear in the ESTIMATES variate. After using TFIT to fit a time series model, the scores should in theory be zero provided the model parameters do not lie on the boundary of their allowed range. The scores are used within TFIT to calculate the parameter changes at

each iteration.

Example 7.3.6 is very similar to Example 7.3.3c which printed the parameter changes when using TFIT with METHOD=onestep. Here METHOD is set to zerostep. The matrix obtained from INVERSE and the variate from SCORE are multiplied to give values very close to the parameter changes. This is not always the case because TFIT shortens the step if the new parameters would have been outside their allowed range. A test statistic is calculated, as a quadratic form in the scaled score and the matrix obtained from VCOVARIANCE. Under the null hypothesis that the two new parameters have been set to their true values, the distribution of this statistic is chi-square on two degrees of freedom. The value obtained is consistent with this.

Example 7.3.6

```
  2  " The model previously fitted to the series of daylengths in Example
 -3    7.3 is extended to include two more autoregressive parameters,
 -4    the old parameters being kept at their estimated values. The score
 -5    is saved after using ESTIMATE with the option METHOD=zerostep. The
 -6    Inverse matrix is also saved and used to calculate a variate of
 -7    parameter corrections. The Variance-Covariance matrix is saved and
 -8    used with the scaled score to form a test statistic to assess whether
 -9    the new parameters should be retained in the model."
 10  OPEN     '%GENDIR%/Examples/GuidePart2/Daylength.dat'; CHANNEL=3
 11  READ      [PRINT=errors; CHANNEL=3; SETNVALUES=yes] Daylength
 12  CLOSE     3
 13  TSM       Erp; ORDERS=!(1,1,2)
 14  TFIT      [PRINT=*] Daylength; TSM=Erp
 15  " Save the previous model parameters and redefine the model with
-16    higher autoregressive orders and extended parameter variate."
 17  CALCULATE Modpar = Erp['Parameters']
 18  &         Modparx = !(Modpar$[1,2,3,4],0,0,Modpar$[5,6])
 19  TSM       Erp; ORDERS=!(3,1,2); PARAMETERS=Modparx
 20  TFIT      [METHOD=zerostep] Daylength; TSM=Erp

Time-series analysis
====================

Output series: Daylength    Noise model: Erp

Residual deviance           = 36959.
Innovation variance         = 255.4

Number of units present     = 150
Residual degrees of freedom = 143


Summary of models
-----------------

          Orders:  Delay     AR   Diff     MA   Seas
Model     Type         B      P      D      Q      S

Erp       ARIMA        -      3      1      2      1


Parameter estimates
-------------------

Model   Seas.   Diff.  Delay  Parameter  Lag  Ref   Estimate        s.e.      t
        Period  Order

Noise        1      0      -   Constant   -    1       3.98        4.55   0.87
             1      1      -   Phi (AR)   1    2      0.380       0.158   2.41
                                           2    3      0.000       0.180   0.00
                                           3    4      0.000       0.141   0.00
                                Theta (MA) 1    5     -0.557       0.132  -4.21
                                           2    6     -0.619       0.130  -4.77

 21  TKEEP     SCORE=Sc; INVERSE=W; VCOVARIANCE=V
 22  PRINT     Sc
```

```
         Sc
        0.0
       -1.9
     1127.0
     -533.2
       19.3
     -127.1

 23  " Calculate and print the parameter correction variate."
 24  CALCULATE Del = PRODUCT(W; Sc)
 25  PRINT     Del

                    Del
                      1
           1     0.07477
           2    -0.06208
           3     0.16752
           4    -0.10249
           5    -0.05255
           6     0.07579

 26  " Form the scaled score and test statistic."
 27  CALCULATE Scsc = Sc/Modpar$[3]
 28  SCALAR    Tstat
 29  CALCULATE Tstat = QPRODUCT(T(Scsc); V)
 30  PRINT     Tstat

      Tstat
     0.9377
```

As in TDISPLAY, You can use the SAVE option to specify the time-series save structure from which the output is to be taken. By default TKEEP uses the structure from the most recent TFIT statement.

7.3.7 The TFORECAST directive

TFORECAST directive

Forecasts future values of a time series.

Options

PRINT = *string tokens*	What to print (forecasts, limits, setransform, sfe); default fore,limi
CHANNEL = *scalar*	Channel number for output; default * i.e. current output channel
ORIGIN = *scalar*	Number of known values to be incorporated; default 0
UPDATE = *string token*	Whether to update the forecast origin to the end of the new observations (yes, no); default no
NEWOBSERVATIONS = *variate*	Variate of length ≥ ORIGIN providing new values of the time series to be incorporated (must be set if ORIGIN > 0)
SFE = *variate*	Saves standardized forecast errors; default *
MAXLEAD = *scalar*	Maximum lead time i.e number of forecasts to be made; default * defines the number as the length of FORECAST variate
FORECAST = *variate*	Variate of length MAXLEAD to save forecasts of output series; default *
SETRANSFORM = *variate*	Saves standard errors of the forecasts (on transformed scale, if defined); default *
LOWER = *variate*	Saves lower confidence limits; default *

UPPER = *variate*	Saves upper confidence limits; default *
PROBABILITY = *scalar*	Probability level for confidence limits; default 0.9
COMPONENTS = *pointer*	Contains variates (of length ORIGIN + MAXLEAD) to save components of the forecast
SAVE = *identifier*	Save structure to supply fitted model; default * i.e. that from last model fitted

Parameters

FUTURE = *variates*	Variates (of length ORIGIN + MAXLEAD) containing future values of input series
METHOD = *string tokens*	How to treat future values of input series (observations, forecasts); default obse

In many applications of forecasting with univariate ARIMA models, you will need only a simple form of the directive. For example

```
TFORECAST [MAXLEAD=10]
```

will cause Genstat to print forecasts for 10 lead times, that is, the next 10 time points after the end of your data. However, you must already have used TFIT to specify the time series to be forecast, and the model to be used for forecasting. This information is supplied by the SAVE option; if SAVE is not specified, TFORECAST uses the information from the most recent TFIT statement. Once you have used TFIT, you can give successive TFORECAST statements to incorporate new observations of the time series, and to produce forecasts from the end of the new data.

If the time series is supplied with a known model (that is, one with all its orders and parameters specified) you can use TFIT with option setting METHOD=initialize before you use TFORECAST. This will carry out just sufficient calculations, in particular the regeneration of the model residuals, for TFORECAST to be used. The model parameters will not be changed – not even the innovation variance. This setting of METHOD restricts the structures, such as parameter standard errors, that can be accessed using TDISPLAY and TKEEP after TFIT. The SAVE structure created by using TFIT with METHOD=initialize thus requires less space than that produced by the other settings.

The formal parameters of TFORECAST are relevant only when the time-series model incorporates explanatory variables, and are described in Section 7.4.3.

The best way to understand the options of TFORECAST is by example. Example 7.3.7a illustrates how to use TFIT to initialize for TFORECAST, with a series of 132 points and using a previously estimated model.

Example 7.3.7a

```
 2  " Forecast number of airline passengers in 1960, using
-3    a seasonal ARIMA model whose parameters have already
-4    been estimated, and based on numbers observed 1949-59.
-5    Data from Box and Jenkins (1970) page 304."
 6  OPEN      '%GENDIR%/Examples/GuidePart2/Airline.dat'; CHANNEL=3
 7  UNITS     [NVALUES=132]
 8  READ      [CHANNEL=3] Apt

  Identifier   Minimum      Mean   Maximum    Values   Missing
         Apt     104.0     262.5     559.0       132         0

 9  CLOSE     3
10  VARIATE   [VALUES=0,1,1, 0,1,1,12] Ord
11  &         [VALUES=0,0,0.00143, 0.34, 0.54] Par
12  TSM       Airpass; ORDERS=Ord; PARAMETERS=Par
13  TFIT      [PRINT=model; METHOD=initialize] Apt; TSM=Airpass
```

```
Time-series analysis
====================

Output series: Apt          Noise model: Airpass


Summary of models
-----------------

          Orders:  Delay     AR   Diff     MA   Seas
Model     Type          B      P      D      Q      S

Airpass   ARIMA         -      0      1      1      1
                        -      0      1      1     12

  14  TFORECAST [MAXLEAD=12; FORECAST=Fcst12]

Forecasts
=========

Maximum lead time: 12


Forecasts for future values
---------------------------

    Lead time     forecast  lower limit  upper limit
            1        419.6        394.3        446.5
            2        398.9        370.2        429.7
            3        466.7        428.6        508.1
            4        454.4        413.5        499.5
            5        473.9        427.5        525.3
            6        547.6        490.1        611.8
            7        623.3        553.8        701.5
            8        631.7        557.4        716.0
            9        527.2        462.1        601.4
           10        462.8        403.1        531.2
           11        407.1        352.6        470.2
           12        452.7        389.7        525.8
```

The FORECAST option specifies that the forecast values are to be stored in the variate Fcst12: you could then, for example, display them graphically.

Now suppose that a further set of observations of the time series has become available, for example a variate New6 containing the next six values of the series. In order to revise the forecasts, you can incorporate this new information as follows.

Example 7.3.7b

```
  15  " Read observed numbers for January to June 1960, and give revised
 -16    forecasts for these months with standardized forecast errors."
  17  READ      [PRINT=data; SETNVALUES=yes] New6

  18  417.0   391.0   419.0   461.0   472.0   535.0:
  19  TFORECAST [PRINT=sfe; ORIGIN=6; MAXLEAD=0; NEWOBSERVATIONS=New6]

Forecasts
=========

Forecast origin:   6
Maximum lead time: 0
```

```
Incorporated observations
-------------------------

    Lead time    New value       s.f.e.
           -5          417.        -0.16
           -4          391.        -0.42
           -3          419.        -2.46
           -2          461.         2.39
           -1          472.         0.33
            0          535.        -0.40
```

The setting PRINT=sfe now causes Genstat to print the standardized errors of the forecast. These are the innovation values that are generated as each successive new observation is incorporated, divided by the square root of the TSM innovation variance. They provide a useful check on the continuing adequacy of the model. For example, excessively large values (compared to the standard Normal distribution) may indicate that you should revise the model. The ORIGIN option specifies the number of new values to be incorporated, and the UPDATE option specifies whether these new observations are to be incorporated internally onto the end of the time series and the internal pointer moved to the end of the new observations. If UPDATE=yes is used, then ORIGIN=0 in future calls to TFORECAST will point to the end of the *n* new observations. If the default, UPDATE=no is used, the internal pointer remains at the end of the original series. The number of future values to be forecast is set by option MAXLEAD. These new values can be saved in a variate of length MAXLEAD using the FORECAST option.

Revised forecasts of the next six values of the series can then be produced by a further statement, as shown in Example 7.3.7c.

Example 7.3.7c

```
  20  " Forecast for July to December 1960."
  21  TFORECAST [MAXLEAD=6; UPDATE=yes; FORECAST=Fcst6]

Forecasts
=========

Maximum lead time: 6


Forecasts for future values
---------------------------

    Lead time     forecast  lower limit  upper limit
            1        612.1        575.2        651.4
            2        620.4        575.8        668.4
            3        517.7        475.5        563.7
            4        454.5        413.5        499.5
            5        399.8        360.7        443.2
            6        444.6        397.9        496.7
```

The PROBABILITY option determines the width of the error limits on the forecast. It defines the probability that the actual value will be contained within the limits at any particular lead time. Note that the limits do not apply simultaneously over all lead times.

The SETRANSFORM option specifies a variate to store the standard errors that Genstat used in calculating the error limits of the forecasts, starting at lead time 1. These are the standard errors of the transformed series, according to the value of the Box-Cox transformation parameter; they are functions of the model only, not of the data.

The LOWER option specifies a variate to store the lower limits of the forecasts. This must be the same length as the FORECAST variate. The TFORECAST directive puts the values of the lower limit into the variate, matching the forecasts in the FORECAST variate. The UPPER option similarly allows the upper limits to be saved. Note that the limits are constructed as symmetric

percentiles, assuming Normality of the transformed time series. Similarly, the forecast is a median value – not necessarily the mode or the mean, unless the transformation parameter is 1.0.

The SFE option specifies a variate to save the standardized errors of the forecasts: see above. The variate must be the same length as the FORECAST variate. The TFORECAST directive places values of the errors in the variate, matching the new observations in the FORECAST variate.

The COMPONENTS option is relevant only when the time-series model incorporates explanatory variables, and is described in Section 7.5.5.

7.3.8 The BJFORECAST procedure

BJFORECAST provides a convenient single command for calculating and plotting forecasts. Internally it uses the TFORECAST directive, described in Section 7.3.7.

BJFORECAST procedure

Plots forecasts of a time series using a previously fitted ARIMA (G. Tunnicliffe Wilson & S.J. Welham).

Options

PROBABILITY = *scalar*	Probability value used for forecast limits; default 0.9
GRAPHICS = *string token*	What type of graphics to use (lineprinter, highresolution); default high
WINDOW = *scalar*	Window to be used for plotting; default 1
PENS = *variate*	The three pens to be used (after being defined appropriately) for drawing the plots; default !(1,2,3)

Parameters

SERIES = *variates*	Variates holding the time series to be used for producing forecasts
LENGTH = *scalars* or *variates*	Specifies the units to be used from each series: a scalar N specifies that the first N units of the series are to be used, a variate of length 2 gives the time index of the first and last units of the subseries to be used; by default the whole series is used
TSM = *TSMs*	ARIMA model to be used for forecasting
TIMERANGE = *variates*	The first and second elements of each variate specify respectively the first and last time index, relative to the whole series, of the range to be forecast
ORIGIN = *scalars*	The time of the latest observation to be used to construct forecasts with increasing leadtimes for each series; if ORIGIN is unset, the default is to take the latest time point in the series prior to the range given by TIMERANGE, unless parameter LEADTIME is set, in which case fixed leadtime forecasts are constructed
LEADTIME = *scalars*	The fixed leadtime to be used to construct forecasts if ORIGIN is unset
FORECAST = *variates*	Save the values of the constructed forecasts
LOWER = *variates*	Save the lower limits of the forecasts
UPPER = *variates*	Save the upper limits of the forecasts
SFE = *variates*	Save the standardized forecast errors, available only for LEADTIME=1

For a time series variate, given by the SERIES parameter, BJFORECAST plots forecasts calculated from a previously fitted ARIMA model, specified by the TSM parameter. The set of time points for which forecasts are produced is defined by setting the TIMERANGE parameter to a variate of length 2 holding the first and last time index. If only part of the series is to be used to initialize for forecasting, this is specified by setting parameter LENGTH, either to a scalar N to indicate that the first N values are to be used, or to a variate of length 2 holding the positions of the first and last units to be included. The procedure also prints a description of the series, and details of the model involved in the initialization for forecasting.

There are two options to control the type of forecasting. Setting the ORIGIN parameter to a scalar indicates that forecasts are calculated from this time point (at increasing leadtimes) for the range of future times specified by the TIMERANGE parameter. Alternatively, if ORIGIN is unset, it is possible to produce forecasts with a fixed leadtime, by setting the parameter LEADTIME to the required value. If neither ORIGIN nor LEADTIME are set, a default origin is taken, namely the last element before the time range to be forecast. Where possible, the values of the supplied series are also plotted for comparison. If one-step-ahead forecasts are requested (fixed leadtime set to 1), the standardized forecast errors are plotted as a tracking signal for use in checking the continuing adequacy of the model.

The FORECAST parameter can be used to save the calculated forecasts in a variate and parameters LOWER and UPPER can save the lower and upper confidence limits for these forecasts. If the forecasts are from a fixed leadtime of 1, the standardized forecast errors can be saved in a variate given by parameter SFE; because of the way in which the standard errors are calculated, the last value of this variate is always missing. The PROBABILITY option indicates the probability value to be used for the confidence limits, with 0.9 as the default value.

Option GRAPHICS controls whether plots are produced for line printer or for the current high-resolution graphics device; by default high-resolution plots are produced. The window to be used for high-resolution plots is specified by the WINDOW option; by default WINDOW=1. The FRAME directive can be used to set the attributes of this window before calling the procedure, and these will be unchanged on leaving the procedure. The PENS option controls which pens are to be used for the plots; the attributes of these pens are modified within the procedure. By default pens 1-3 are used, but these can be changed by setting option PENS to a variate of length 3 containing the numbers of the three different pens required.

Example 7.3.8 and Figure 7.3.8 show the use of the procedure BJFORECAST to construct and plot these same forecasts as in Example 7.3.7a, together with their error limits.

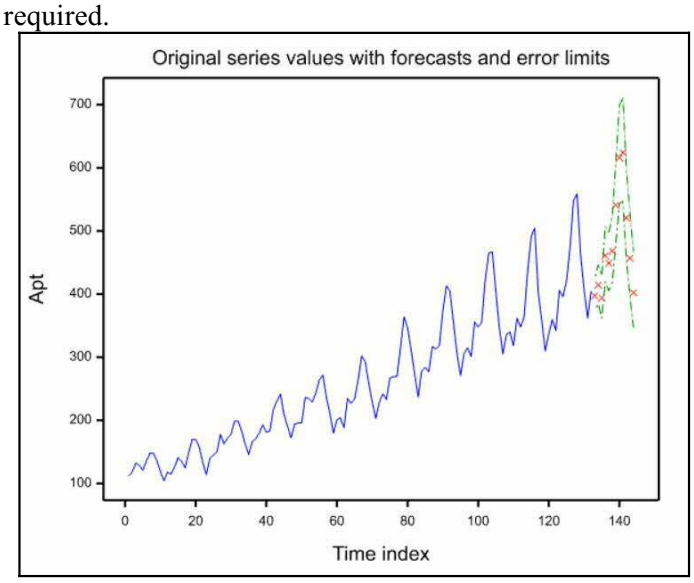


Figure 7.3.8

Example 7.3.8

```
  2  " Use procedure BJFORECAST to calculate and display forecasts
 -3    of the last 12 values based upon the previous 132."
  4  OPEN    '%GENDIR%/Examples/GuidePart2/Airline.dat'; CHANNEL=3
  5  READ    [CHANNEL=3] Apt

   Identifier   Minimum      Mean   Maximum    Values   Missing
          Apt     104.0     262.5     559.0       132         0

  6  CLOSE   3
  7  VARIATE [VALUES=0,1,1, 0,1,1,12] Ord
  8  VARIATE [VALUES=0,0.0,0.00143,0.34,0.54] Par
  9  TSM     Airpass; ORDERS=Ord; PARAMETERS=Par
 10  FRAME   1; YLOWER=0.1; YUPPER=0.9; XLOWER=0; XUPPER=1
 11  BJFORECAST Apt; TSM=Airpass; ORIGIN=132; TIMERANGE=!(133,144)

Forecasts from fixed origin 132 over time range 133 to 144 with probability
limits of size 0.900 using whole of series.
```

7.4 Regression with autocorrelated (ARIMA) errors

At the beginning of Chapter 3, we noted that regression analysis is not valid if the residuals cannot be assumed to be independent. When modelling observations of a variable that are taken at successive points in time, it is likely that there will be some dependence. A simple check for this is to fit a regression model as in Chapter 3, and then calculate the sample autocorrelation function (7.1.2) of the residuals from the regression. If you think that there might be appreciable autocorrelation, you should try fitting the regression model using an ARIMA model for the errors, as described in this section.

We shall use as an example a time series y_t of daily gas demand (corrected for the effects of days of the week), and a corresponding indicator x_t of the coldness of the days, compiled from temperature, windspeed, and so on. Example 7.4a fits a regression between the variates `Demand` and `Coldness` which hold 104 consecutive values of the two series. A first-order autoregressive model, AR(1), is specified for the errors: that is, the model is

$$y_t = c + b\,x_t + e_t$$
$$e_t = \varphi_1\, e_{t-1} + a_t$$

where a_t is the series of independent innovations of the errors e_t. We have set `PRINT=monitoring` in the `TFIT` statement to show the course of the convergence.

Example 7.4a

```
  2  " Regress daily gas demand on coldness, using an AR(1) model for errors."
  3  OPEN  '%GENDIR%/Examples/GuidePart2/Demand.dat',\
  4        '%GENDIR%/Examples/GuidePart2/Cold.dat'; CHANNEL=2,3
  5  READ  [CHANNEL=2; SETNVALUES=yes] Demand

   Identifier   Minimum      Mean   Maximum    Values   Missing
       Demand     239.3     348.7     471.8       104         0

  6  &     [CHANNEL=3] Coldness

   Identifier   Minimum      Mean   Maximum    Values   Missing
     Coldness    -117.3    -49.87     42.60       104         0

  7  TSM   Erm; ORDERS=!(1,0,0)
  8  TRANSFERFUNCTION Coldness
  9  " Monitor convergence."
 10  TFIT [PRINT=monitoring,estimates] Demand; TSM=Erm; BOXCOX=estimate
```

```
Convergence monitoring
----------------------

Cycle          Deviance      Current parameters
    1         12803380.           0.      1.00000           0.           0.
    2          8684909.        1.7447       1.2880       499.04     -0.45365
    3          209142.8        3.0618       1.2890       748.64      0.75300
    4          38869.89        5.3578       1.3048       1517.3      0.83100
    5          28399.29        5.3906       1.3060       1925.6      0.79741
    6          27741.34        5.6954       1.3124       1921.7      0.73305
    7          27618.36        5.6691       1.3059       1883.2      0.72642
    8          27601.48        5.6040       1.3032       1858.0      0.71415
    9          27571.63        5.2138       1.2901       1734.7      0.71193
   10          27538.84        4.7622       1.2747       1599.7      0.71021
   11          27506.65        4.4128       1.2613       1493.8      0.70996
   12          27475.55        4.0239       1.2457       1376.4      0.70890
   13          27445.39        3.7473       1.2332       1291.5      0.70889
   14          27416.51        3.4103       1.2173       1188.4      0.70777
   15          27388.23        3.1955       1.2057       1121.4      0.70800

******** Warning 46, code TS 21, statement 1 on line 10

Command: TFIT [PRINT=monitoring,estimates] Demand; TSM=Erm; BOXCOX=estimate
The iterative estimation process has not converged.
The maximum number of cycles is 15


Time-series analysis
====================

Parameter estimates
-------------------

Model    Seas.   Diff.  Delay  Parameter  Lag  Ref   Estimate        s.e.      t
         Period  Order

Input 1      1      0      0      Omega    0    1      2.898       0.896   3.24
Noise        1      0      -    Box-Cox    -    2     1.1894      0.0514  23.14
                               Constant    -    3      1029.        270.   3.80
                               Phi (AR)    1    4     0.7067      0.0714   9.89
```

The `TSM` statement specifies the AR(1) model for the errors. The `TRANSFERFUNCTION` statement here merely specifies the explanatory variate. You could use this directive to specify a response model that includes lagged effects of the explanatory variate (7.5.2), but in Example 7.4a, the response model is a simple linear regression: this is the default.

The warning shows that the convergence criterion has not been reached within 15 iterations. To satisfy the criterion, we could either increase the limit on the number of iterations by setting the option `MAXCYCLE=25`, say, or initialize the parameters to rough estimates of the parameters in the model, perhaps using the `FTSM` directive (7.7.1 and 7.7.2). The statements that follow `TFIT` in this program use the best parameter values found by `TFIT`, without further comment.

The `TFIT` statement simultaneously estimates the regression coefficients c and b and the AR parameter φ_1. Also in this case, a Box-Cox transformation is estimated for the response variate, `Demand`. Note in the printed results that the estimate of b appears under "`Transfer-function model 1`", as a moving-average parameter at lag 0. By default, Genstat fixes the transformation and constant parameters associated with the explanatory variables to be 1.0 and 0.0. Alternatively, you could estimate these parameters, as described in Section 7.5.

The constant term c in the regression is included in the results for the autoregressive moving-average model, as is the transformation parameter of the `Demand` variable, and the estimate of φ_1.

You can obtain forecasts of the demand series, by specifying future values of the explanatory variable. In Example 7.4b, the variate `Newcold` contains the next seven values of coldness.

Example 7.4b

```
  11  " Forecast gas demand for the next week, given values for coldness."
  12  READ      [CHANNEL=3; SETNVALUES=yes] Newcold

    Identifier   Minimum      Mean   Maximum    Values   Missing
       Newcold    -138.3    -102.3    -75.60         7         0

  13  CLOSE     2,3
  14  TFORECAST [MAXLEAD=7] Newcold

Forecasts
=========

Maximum lead time: 7


Forecasts for future values
---------------------------

    Lead time     forecast  lower limit  upper limit
            1        318.6        290.9        346.0
            2        294.3        259.6        328.1
            3        313.9        277.0        350.0
            4        324.5        286.5        361.7
            5        278.4        238.5        317.2
            6        261.7        221.0        301.2
            7        299.2        259.5        338.0
```

Genstat constructs the forecasts by calculating the predicted linear response at the `Newcold` values, and adding it to the forecast values of the autocorrelated errors. The forecast limits take this into account.

In practice you would be unlikely to know the future values of explanatory variables. Exceptions are where the variable has a fixed deterministic form such as in a trend, or a cycle, or an intervention variable; or when the variable is under the control of the experimenter, as when sales are related to prices; or when the analysis is retrospective, as in this example. You can predict the explanatory variables in various ways. For example, ordinary weather forecasts are used in practice to forecast gas demand. You cannot usually include the uncertainties in predicting the explanatory variables in the error limits of the forecast. These uncertainties would usually be assessed by trying out different future values of the explanatory variables. Thus the `TFORECAST` statement in the example could be repeated with a variety of future values. But there is one case where you can allow for the uncertainty of predicting the explanatory variables. This is when the future values of the explanatory variables are predictions obtained using univariate ARIMA models. Then you can allow for the errors by setting the `ARIMA` parameter of the `TRANSFERFUNCTION` directive, and the `METHOD` parameter of the `TFORECAST` directive.

7.4.1 The `TRANSFERFUNCTION` directive

`TRANSFERFUNCTION` directive

Specifies input series and transfer-function models for subsequent estimation of a model for an output series.

Option

`SAVE` = *identifier*	To name time-series save structure; default *

Parameters

`SERIES` = *variates*	Input time series

TRANSFERFUNCTION = *TSMs*	Transfer-function models; if omitted, model with 1 moving-average parameter, lag 0
BOXCOXMETHOD = *string tokens*	How to treat transformation parameters (fix, estimate); default fix
PRIORMETHOD = *string tokens*	How to treat prior values (fix, estimate); default fix
ARIMA = *TSMs*	ARIMA models for input series

For regression with autocorrelated errors, you should use TRANSFERFUNCTION to specify the variates that are to be the explanatory variables in a subsequent TFIT statement. Thus in many applications you will need only a simple form of the directive, such as

```
TRANSFERFUNCTION Coldness
```

The first parameter, SERIES, specifies a list of variates holding the time series of explanatory variables.

The BOXCOXMETHOD parameter allows you to estimate separate power transformations for the explanatory variables: the variable x_t is transformed to

$$x_t^{(\lambda)} = (x_t^{\lambda} - 1) / \lambda , \qquad \lambda \neq 0$$
$$x_t^{(0)} = \log(x_t)$$

The default is no transformation, corresponding to $x_t^{(\lambda)} = x_t$. You can choose whether the transformations are to be fixed or estimated, by specifying one string for each explanatory variable.

The ARIMA parameter allows you to associate with each explanatory variable a univariate ARIMA model for the time-series structure of that variable. If you think such a model is inappropriate, then you should give a missing value in place of the TSM identifier, or leave this parameter unset. You can use these models in any subsequent TFORECAST statement to incorporate, into the error limits of the forecasts, an allowance for uncertainties in the predicted explanatory variables; the allowance assumes that the future values of the explanatory variables are forecasts obtained using these ARIMA models (7.4.3).

The TRANSFERFUNCTION and PRIORMETHOD parameters are not relevant in this context, and are described in Section 7.5.2.

The SAVE option allows you to name the time-series save structure created by TRANSFERFUNCTION. You can use this identifier in a later TFIT statement, and eventually in a TFORECAST statement. If you do not name the save structure Genstat will use the most recent save structure, which will be overwritten each time a new TRANSFERFUNCTION statement is given.

7.4.2 Extensions to the TFIT directive for regression with ARIMA errors

The SERIES parameter of TFIT now specifies the response variate, and the TSM parameter specifies the ARIMA model for the errors. Note however, that the transformation parameter of this ARIMA model is used to define a transformation for the response variable, not the errors, and the BOXCOXMETHOD parameter controls its estimation.

The constant term in the ARIMA model corresponds to the usual regression constant term only if there is no differencing specified by the ARIMA model; otherwise it is equivalent to a constant term in a regression between the differenced series.

The PRINT option is the same as described in Section 7.3.3. But note that the regression estimates for the explanatory variables are printed in a sequence of simple transfer-function models, followed by the ARIMA error model, as shown in Example 7.4a.

The LIKELIHOOD option settings exact and leastsquares are essentially the same as for univariate ARIMA modelling in Section 7.3. The likelihood for the model is defined as that of the univariate error series e_t which is defined in general by

$$e_t = y_t - b_1 x_{1,t} - \ldots - b_m x_{m,t}$$

(the x_i being m explanatory variables). The constant term therefore appears in the model after any differencing of e_t; for example

$$\nabla e_t = c + (1 - \theta_1 B)a_t$$

You can get bias in the estimates of the parameters of an ARIMA model because the regression is estimated at the same time. You can guard against this by specifying `LIKELIHOOD=marginal`. This can be particularly important if the series are short or if you use many explanatory variables (Tunnicliffe Wilson 1989). The deviance is now defined as

$$D = S\left(\left|X'V^{-1}X\right| \left|V\right|\right)^{1/m}$$

where m is reduced by the number of regressors (including the constant term) and the columns of X are the differenced explanatory series: the other terms are as in the exact likelihood described in Section 7.3.4.

You can use this setting also for univariate ARIMA modelling, when the constant term is the only explanatory term. Furthermore, Genstat deals with missing values in the response variate by doing a regression on indicator variates; these too are included in the X matrix. However, you cannot use marginal likelihood and estimate a transformation parameter in either the transfer-function model or an ARIMA model. Neither can you use it if you set the `FIX` option in `TFIT`. In these cases Genstat automatically resets the `LIKELIHOOD` option to `exact`.

At every iteration with the setting `LIKELIHOOD=marginal`, the regression coefficients are the maximum-likelihood estimates conditional upon the estimated values of the parameters of the ARIMA model: these are also the generalized least-squares estimates, conditioned in the same way. This is so even if `MAXCYCLE=0`; that is, the coefficients of the regression are re-estimated even at iteration 0. Therefore you must not use the `marginal` setting with the option `METHOD=initialize` to initialize for `TFORECAST`. You can compare deviance values that were obtained using marginal likelihood only for models with the same explanatory variables and the same differencing structure in the error model.

You can use the setting `CONSTANT=fix` with marginal likelihood. You can use the `FIX` option to impose constraints across any or all of the parameters of the regression and the ARIMA model. In order to do this, you may find it easiest to use `TFIT` without the `FIX` option first, so that you can ascertain the ordering of the parameters; then give a second statement with the option set. The variate specified in the `FIX` option must have one element for each parameter that is printed with a reference number. These are, in order, three parameters for each explanatory variate, followed by the ARIMA model parameters. Genstat uses the variate to provide a parameter numbering as described for the `FIX` option in Section 7.4.2. Note that this numbering overrides the `BOXCOXMETHOD` parameter and the `CONSTANT` option. Thus you can constrain the transformation parameters to be equal for all or some of the variables. You can also estimate a constant term for an input series. For details of this see 7.5.3.

The results of `TFIT`, accessible by `TDISPLAY` and `TKEEP`, are essentially the same as in univariate models. The variate of parameter estimates and associated structures now refers to the whole set of parameters in the order in which they are printed. The variate of missing-value estimates holds first the values from the response variate, and then those from the explanatory variate, in the order in which they were listed in the `SERIES` parameter of `TRANSFERFUNCTION`.

7.4.3 Extensions to the `TFORECAST` directive for regression with ARIMA errors

A `TFORECAST` statement for regression with ARIMA errors must be preceded by a `TRANSFERFUNCTION` statement and an `TFIT` statement: these initialize the save structure of the time series that is to be used by `TFORECAST`. You use option `METHOD=initialize` of `TFIT` to do this as described in Section 7.3.7.

You use the `FUTURE` parameter to specify a list of variates, corresponding to the list of variates specified by the `SERIES` parameter of `TRANSFERFUNCTION`. These variates must all have the same length. They hold future values of the explanatory variables to be used either for constructing forecasts of the response variable, or for incorporating new observations in order

to revise the forecasts. The use of these future values is similar to the use of the FORECAST variate as described in Section 7.3.7. For example, let Fcdem be a variate of length seven in Examples 7.4a or 7.4b. The statement

```
TFORECAST [MAXLEAD=7; FORECAST=Fcdem] FUTURE=Newcold
```

would cause forecasts of the next week's demand figures to be placed in Fcdem. Suppose that in a week's time, the actual demand had been recorded and was held in the variate Newdem. Then in order to revise the forecasts, you must first incorporate this new information by

```
TFORECAST [ORIGIN=7; MAXLEAD=0; FORECAST=Newdem] \
  FUTURE=Newcold
```

Note that if Newcold had previously contained forecasts from an ARIMA model, say, you would have to alter it to contain the recorded values before this statement. You can get revised forecasts of the next week's demand by once more amending Newcold, to hold the values for the coming week, and then using

```
TFORECAST [UPDATE=yes; MAXLEAD=7; FORECAST=Fcdem] \
  FUTURE=Newcold
```

An alternative to the previous two statements would be to use variates of length 14, with Newcold holding the seven values just recorded followed by the seven values for the coming week. Similarly Newdem should hold the last seven days' demand, followed by seven missing values. The statement

```
TFORECAST [ORIGIN=7; MAXLEAD=7; FORECAST=Newdem] \
  FUTURE=Newcold
```

would then incorporate the first seven values (up to the ORIGIN setting) of each variate, and use the last seven values (specified by MAXLEAD) of Newcold to place revised forecasts into the last seven values of Newdem.

You can use the METHOD parameter when some or all of the future values of the explanatory variables are forecasts obtained using univariate ARIMA models. You can amend the error limits of the forecasts for the response variable to allow for the uncertainty in these future values, but you need to assume that there is no cross-correlation between the errors in these predictions. The list of strings specified by the METHOD parameter indicates for each explanatory variable whether such an allowance should be made. The future values of a series are by default treated as known values if no corresponding ARIMA model is present, or if the transformation parameter of the ARIMA model is not equal to the value used in the regression model for that series. You can change the settings of the METHOD parameter in successive TFORECAST statements.

7.5 Multi-input transfer-function models

A transfer-function model allows for lagged effects of an explanatory variable on the response variable, as well as for autocorrelated errors. Using the notation of Box & Jenkins (1970), including a transfer-function model with an ARIMA model for a response variable gives the equation

$$y_t = \nu(B)x_t + \psi(B)a_t$$

where we shall now call y_t the output series and x_t the input series. You can have several input series, so we shall call the full model for y_t a multi-input model, corresponding to the term "multiple regression" used in Chapter 8. Writing $y_t=z_t+n_t$ where $z_t = \nu(B)x_t$ and $n_t = \psi(B)a_t$, we shall call z_t the component due to input x_t, and n_t the noise component. An ARIMA TSM is used to represent the structure of n_t, and a transfer-function TSM to represent the structure of z_t as a function of x_t. For example, consider the lagged response, with $|\delta|<1$:

$$y_t = \omega(x_{t-1} + \delta x_{t-2} + \delta^2 x_{t-3} + \dots) + n_t.$$

Then $\nu(B) = \omega B / (1 - \delta B)$.

Example 7.5 fits this model to a series of length 40, and produces forecasts of the next eight points; see Figure 7.5.

Example 7.5

```
  2  " One-input transfer-function model relating level of gilts to profits."
  3  VARIATE [VALUES=1...40] Time
  4  UNITS   Time
  5  " Read data on gilts and profits from separate files."
  6  OPEN    '%GENDIR%/Examples/GuidePart2/Gilts.dat',\
  7          '%GENDIR%/Examples/GuidePart2/Profits.dat'; CHANNEL=2,3
  8  READ    [CHANNEL=2] Gilts

   Identifier   Minimum      Mean   Maximum    Values   Missing
        Gilts    -26.25     1.037     27.97        40         0

  9  &       [CHANNEL=3] Profits

   Identifier   Minimum      Mean   Maximum    Values   Missing
      Profits    -1.807   0.02747     1.487        40         0

 10  " Set up transfer-function model with delay time 1 and one AR-type
-11    parameter."
 12  TSM     [MODELTYPE=transfer]  Tf; ORDERS=!(1,1,0,0); PARAMETERS=!(1,0,0,0.1)
 13  TRANSFERFUNCTION Profits; TRANSFER=Tf
 14  " Set up ARIMA model for the noise, with one AR parameter."
 15  TSM     Ar; ORDERS=!(1,0,0); PARAMETERS=!(1,0,0,0)
 16  TFIT    Gilts; TSM=Ar

Time-series analysis
====================

Input series 1: Profits     Transfer fn: Tf
Output series: Gilts        Noise model: Ar

Residual deviance           = 900.6
Innovation variance         = 24.52

Number of units present     = 40
Residual degrees of freedom = 36


Summary of models
-----------------

          Orders:  Delay     AR   Diff     MA   Seas
Model     Type          B      P      D      Q      S

Tf        TF            1      1      0      0      1
Ar        ARIMA         -      1      0      0      1


Parameter estimates
-------------------

Model   Seas.   Diff.  Delay  Parameter  Lag  Ref   Estimate        s.e.      t
        Period  Order

Input 1      1      0      1      Delta   1    1      0.6273      0.0805   7.79
                                 Omega   0    2        8.74        1.16   7.51
Noise        1      0      -   Constant   -    3       -1.06        2.87  -0.37
                               Phi (AR)   1    4       0.740       0.118   6.26

 17  " Save the components of the series in variates."
 18  TKEEP   COMPONENTS=!P(Fprofits,Noise)
 19  PEN     1,2; COLOUR='black'; METHOD=line,point; SYMBOLS=0,1; LINE=1
 20  DGRAPH  [TITLE='Fitted series with original data'; WINDOW=3; KEY=0]\
 21          Fprofits,Gilts; Time; PEN=1,2
 22  " Read future values of profits, and forecast corresponding gilts."
 23  READ    [CHANNEL=3; SETNVALUES=yes] Nprofits

   Identifier   Minimum      Mean   Maximum    Values   Missing
     Nprofits    -1.165   -0.1374    0.4904         8         0
```

```
  24  TFORECAST [MAXLEAD=8] Nprofits

Forecasts
=========

Maximum lead time: 8


Forecasts for future values
---------------------------

    Lead time     forecast  lower limit  upper limit
            1        -6.50       -14.64         1.65
            2       -10.37       -20.51        -0.24
            3       -17.20       -28.27        -6.12
            4       -16.11       -27.67        -4.55
            5       -12.25       -24.06        -0.44
            6        -4.39       -16.34         7.56
            7         1.10       -10.93        13.13
            8         4.14        -7.93        16.21

  25  CLOSE   2,3
```

In this example, the first TSM statement defines the orders of the transfer-function model, the initial values of parameters δ and ω being given as 0.0 and 0.1 respectively. The second TSM statement defines the autoregressive error structure. The TRANSFERFUNCTION statement then specifies the input series to be Profits, and gives the associated transfer-function model. The TFIT statement specifies the output series and the noise model.

After the model has been estimated, the TKEEP statement accesses the two components of Gilts. The first of these, Fprofits, is plotted together with Gilts, to reveal how well the output series has been modelled by the input series.

Finally, new values of the input series are used to construct forecasts of the output series, using the TFORECAST directive.

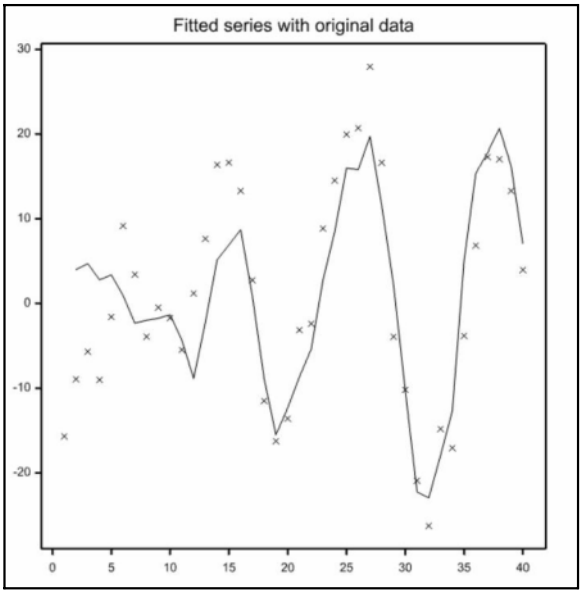


Figure 7.5

7.5.1 Declaring transfer-function models with the TSM directive

The basic structure of the TSM directive, and of the models that it defines, is given in Section 7.3.2. Here we describe the ORDERS, PARAMETERS and LAGS variates for the option setting MODELTYPE=transferfunction.

The simple non-seasonal transfer-function model relates a component z_t of the output series to the corresponding input series x_t, by the equation

$$\delta(B)\,\nabla^d z_t = \omega(B)\,B^b\,\{x_t^{(\lambda)} - c\}$$

where

$$\delta(B) = 1 - \delta_1 B - \ldots - \delta_p B^P$$

$$\omega(\mathrm{B}) = \omega_0 - \omega_1 B - \ldots - \omega_q B^q\,.$$

The integer $b>0$ defines a pure *delay*, and the integer $d>0$ defines the order of differencing in the transfer function.

The parameter λ specifies a Box-Cox power transformation for the input series, and the parameter c specifies a reference level for the transformed input. There is no mean correction of the input series when transfer-function models are estimated, and you should use a value of c close to the series mean so as to improve the numerical conditioning of the estimation procedure. However, if the input series x_t is trend-like rather than stationary, you could alternatively use a value for c close to the early series values, because this reduces the transient errors that arise when the transfer function is applied. The PRIORMETHOD parameter of TRANSFERFUNCTION, described below, provides further means of handling these transients.

The parameters λ and c are not estimated unless you specify otherwise by the BOXCOXMETHOD parameter of TRANSFERFUNCTION or the FIX option of TFIT. Often c in the transfer-function model is aliased with the constant term in the ARIMA errors, and so they should not both be estimated. In some circumstances, however, they both could be estimated, for example in a differenced transfer-function model with stationary noise.

The ORDERS parameter for the simple transfer-function model described above specifies a variate containing the four values b, p, d and q.

The PARAMETERS parameter specifies a variate containing $3+p+q$ values: λ, c, δ_1, ... δ_p, ω_0, ω_1 ... ω_q. You must always include the parameters λ, c and ω_0. When you use a transfer-function model, Genstat will check its parameter values. In particular the operator $\delta(B)$ must satisfy the stability or stationarity condition.

The LAGS parameter is optional, and may be used to change the lags associated with the parameters, from the default values of 1 ... p, 1 ... q. The variate of lags contains values corresponding to the parameters δ_1 ... δ_p, ω_1 ... ω_q. They have the same interpretation as the lags in ARIMA models, and must satisfy the same conditions as specified in Section 7.3.1. Note that there is no lag associated with ω_0, because the delay b provides the necessary flexibility for this.

You can also have seasonal extensions of transfer-function models:

$$\delta(B)\Delta(B^s)\nabla^d\nabla_s^D z_t = \omega(B)\Omega(B^s)B^b\{x_t^{(\lambda)} - c\}$$
$$\Delta(B^s) = 1 - \Delta_1 B^s - ... - \Delta_P B^{Ps}$$
$$\Omega(B^s) = 1 - \Omega_1 B^s - ... - \Omega_Q B^{Qs}$$

Note that there is no Ω_0 coefficient, because ω_0 is always present in the model and provides sufficient flexibility.

The ORDERS parameter here contains b, p, d, q, P, D, Q and s, and the PARAMETERS parameter contains λ, c, δ_1 ... δ_p, ω_0 ... ω_q, Δ_1 ... Δ_P, Ω_1 ... Ω_Q. You can analogously extend the LAGS parameter. You can have extensions to multiple seasonal periods, as for ARIMA models.

7.5.2 Extensions to the TRANSFERFUNCTION directive for multi-input models

This directive specifies several input series and the associated transfer-function model to be used in a subsequent TFIT statement to fits a multi-input model to an output series.

The SERIES and BOXCOXMETHOD parameters are as described in Section 7.4.1.

The TRANSFERFUNCTION parameter specifies the transfer-function TSMs that are to be associated with the input series. A missing value in place of a TSM identifier causes Genstat to treat the corresponding input series as a simple explanatory variable, equivalent to a transfer-function model with orders (0,0,0,0).

The PRIORMETHOD parameter specifies, for each input series, how Genstat is to treat the transients associated with the early values of the transfer-function response. In calculating the input component z_t from the input x_t, Genstat has to make assumptions about the unknown values of x_t which came before the observation period. The default is that x_t (or generally $x_t^{(\lambda)}$) is assumed to be equal to the reference constant c of the transfer-function model. The pattern of the transient can be controlled by introducing a number $\max(p+d,b+q)$ of nuisance parameters to represent the combined effects of all earlier input values on the observed output. Setting PRIORMETHOD=estimate specifies that these nuisance parameters are estimated so as to minimize the transients. You should, however, be careful in using this. Often all you will have

to do is make a sensible choice of the reference constant c. Estimating the transients is best done as a final stage in refining the model; earlier, this may give poor numerical conditioning.

7.5.3 Extensions to the TFIT directive for multi-input models

TFIT fits a multi-input model to output series that have a specified model for the output noise. The input series and transfer-function models must have been specified in an earlier TRANSFERFUNCTION statement.

The PRINT option is the same as before, but note that the transfer-function models are printed in a descriptive format similar to the ARIMA model, with parameter reference numbers used throughout.

The LIKELIHOOD option settings exact and leastsquares are similar to the settings described in Section 7.4.2 for regression with ARIMA errors. For example, with a single input, the likelihood is defined as that for the univariate noise series n_t, calculated as $n_t=y_t-z_t$.

The marginal likelihood is permitted only when all the transfer-function models are equivalent to simple regression.

You can use the FIX option as described in Sections 7.3.2 and 7.4.2, to impose constraints among the parameters while the model is being estimated. These constraints operate here across the whole set (in order) of the parameters of the transfer-function models and of the ARIMA model, excluding the innovation variance. You can thus use this option to estimate the constant term in a transfer-function model (but bear in mind the remarks in Section 7.5.1 about possible aliasing).

7.5.4 Extensions to the TKEEP directive for multi-input models

After a multi-input model has been fitted using TFIT, you can use the COMPONENTS parameter to access the components of the output series that are due to the various input series; you can also access the output noise. In simple regression, the input components are proportional to the input series. But the component resulting from a transfer-function model may be quite different from this. You can examine these components separately, or sum them to show the total fit to the output series that is explained by the input series. Note that the fitted values may appear to be offset from that output series, because the constant term is part of the noise component, and so is not included. Example 7.5 includes a graph of the output component due to the single input. You may want to examine the output noise component. For example, if you thought that the ARIMA model for the output noise was inadequate, you could investigate the noise component with the univariate ARIMA modelling methods described earlier in this chapter.

7.5.5 Extensions to the TFORECAST directive for multi-input models

TFORECAST for multi-input models is the same as for regression models with ARIMA errors (7.4.3). But it does have one further useful option.

The COMPONENTS option specifies a pointer to variates in which you can save components of future values of the output series. There is a variate for each input component and for the output noise component. These variates correspond exactly to the variates that were specified by the FUTURE parameter for the input series, and by the FORECAST variate for the output series; corresponding lengths must match. The values that the variates hold can therefore be components of the forecasts of the output series, or can be new observations. The can be used to investigate the structure of forecasts.

If the input series ARIMA model and the transfer-function model have differing transformation parameters, then the METHOD option reverts to its default action of treating the values of any future input series as known quantities rather than forecasts.

7.6 Filtering time series

Filtering is a means of processing a time series so as to produce a new series. The purpose is usually to reveal some features and remove other features of the original series. Filters in Genstat are one-sided: that is, each value in the new series depends only on present and past values of the original series. However, you can do two-sided filtering by using the SHIFT and REVERSE functions of CALCULATE (1:4.2.1).

A filter is defined by a time-series model. For example, consider the exponentially weighted moving average (EWMA) filter

$$y_t = \lambda\, y_{t-1} + (1 - \lambda)\, x_t$$

which smoothes x_t to produce y_t. Then

$$y_t = \{(1 - \lambda) / (1 - \lambda B)\}\, x_t.$$

You can represent this by a transfer function applied to x_t. Example 7.6 applies this filter to smooth a time series of annual temperatures in Central England, taking λ=0.8: the mean of the series is subtracted from the series before smoothing and restored afterwards. The smoothed series, Smtemp, is shown with the original data in Figure 7.6. This is one way to reduce transient errors at the start of the smoothed series.

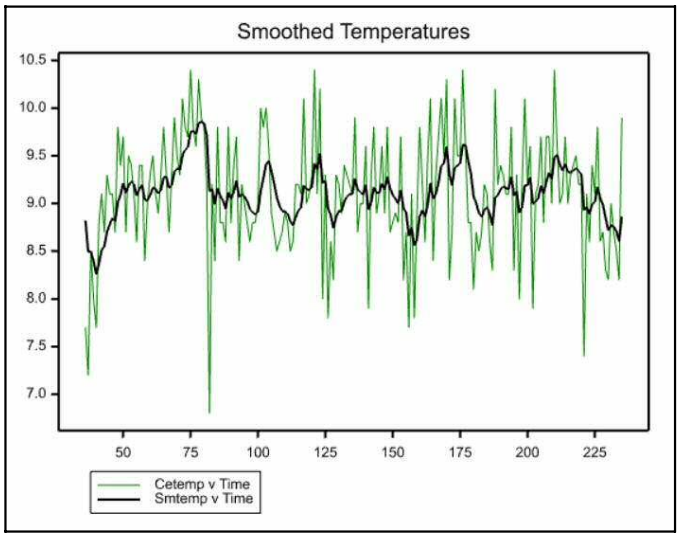


Figure 7.6

Example 7.6

```
 2  " Smoothing a series of Central England Temperatures using an
-3    exponentially weighted filter: data from Manley, G. (1974),
-4    Central England temperatures: monthly means 1659-1973,
-5    Quart.J.Met.Soc., 100, 378-405. To illustrate the end-effect
-6    problems of filtering a subset of the data is used."
 7  VARIATE   [NVALUES=315] Cetave
 8  OPEN      'Cetave.dat'; 3
 9  READ      [CHANNEL=3] Cetave

  Identifier   Minimum      Mean   Maximum    Values   Missing
      Cetave     6.800     9.140     10.60       315         0

10  CLOSE     3
11  VARIATE   [VALUES=36...235] Time
12  CALCULATE Cetemp = Cetave$[Time]
13  &         Tmean = MEAN(Cetemp)
14  &         Mcetemp = Cetemp-Tmean
15  TSM       [MODELTYPE=transfer] Ewma; ORDERS=!(0,1,0,0);\
16            PARAMETERS=!(1,0,0.8,0.2)
17  TFILTER   Mcetemp; NEWSERIES=Smtemp; FILTER=Ewma
18  CALCULATE Smtemp = Smtemp+Tmean
19  FRAME     1; YLOWER=0.2; YUPPER=0.9; XLOWER=0; XUPPER=1
20  PEN       1,2; METHOD=line; LINE=0,1; SYMBOL=0; COLOUR='green','black';\
21            THICKNESS=0.5,2.0
22  DGRAPH    [TITLE='Smoothed Temperatures'] Cetemp,Smtemp; Time; PEN=1,2
```

In this example the filter is defined by a transfer-function model. Alternatively, you can use an ARIMA model to define a filter, in which case the model pre-whitens the series. Suppose, for example, an AR(1) model is specified, with parameter φ_1; the result of applying this to a series x_t is to generate a series a_t:

$$a_t = x_t - \varphi_1 x_{t-1}$$

Such an operation is usefully applied to whiten a series before calculating its spectrum, or to whiten a pair of series before calculating their cross-correlation.

7.6.1 The TFILTER directive

TFILTER directive

Filters time series by time-series models.

Option

PRINT = *string tokens*	What to print (series); default *

Parameters

OLDSERIES = *variates*	Time series to be filtered
NEWSERIES = *variates*	To save filtered series
FILTER = *TSMs*	Models to filter with respect to
ARIMA = *TSMs*	ARIMA models for time series

The OLDSERIES and NEWSERIES parameters of TFILTER specify respectively the time series to be filtered, and the series that result from filtering. A new series must not have the same identifier as the series from which it was calculated. Genstat interprets any missing values in the old series as zero. But if you use the ARIMA parameter (see below), Genstat replaces them by interpolated values when it calculates the filtered series; the missing values remain in the old series.

The FILTER parameter specifies the TSMs to be used for filtering. If the TSM is a transfer-function model (7.5.1), the new series y_t is calculated from the old series x_t by

$$y_t = \{ \omega(B)B^b / \delta(B)\nabla^d \} x_t.$$

The filter does not use the power transformation nor the reference constant. This lets you apply a single filter conveniently to a set of time series, for which different transformations and different constants might be appropriate. You can always use the CALCULATE directive to apply a transformation to a series before using TFILTER.

If the TSM is an ARIMA model (7.3.1), then the new series a_t is calculated from the old series y_t by

$$a_t = \{ \varphi(B)\nabla^d / \theta(B)\} y_t.$$

Note that the TSM does not have to be the model appropriate for y_t. Again, Genstat ignores the parameters λ, c and σ_a^2; you can set them to 1,0,0, for example.

The ARIMA parameter specifies a time-series model for the old series. The purpose is to reduce transient errors that arise in the early part of the new series: these arise because Genstat does not know the values of the old series that came before those that have been supplied. If you do not use this parameter, then Genstat takes these earlier values to be zero. This can cause unacceptable transients which can only be partially removed by procedures such as mean-correcting the old series. If you do use the ARIMA parameter, then Genstat uses the specified model to estimate (or back-forecast) the values of the old series earlier than those that have been supplied.

You do not have to have a good ARIMA model for the old series in order to achieve worthwhile reductions in the transients. Thus a model with orders (0,1,1) and parameters (1,0,0,0.7) would estimate the prior values to be constant, at a level that is a backward EWMA of the early values of the series. Example 7.6.1a is a continuation of Example 7.6, in which the ARIMA parameter is used. The results are shown in Figure 7.6.1a: the smoothed series, `TCSmtemp`, fits the series much more closely at the start; the old version of the smoothed series, `Smtemp`, is also shown on the graph (using a dashed line), to reveal the difference at the start of the series.

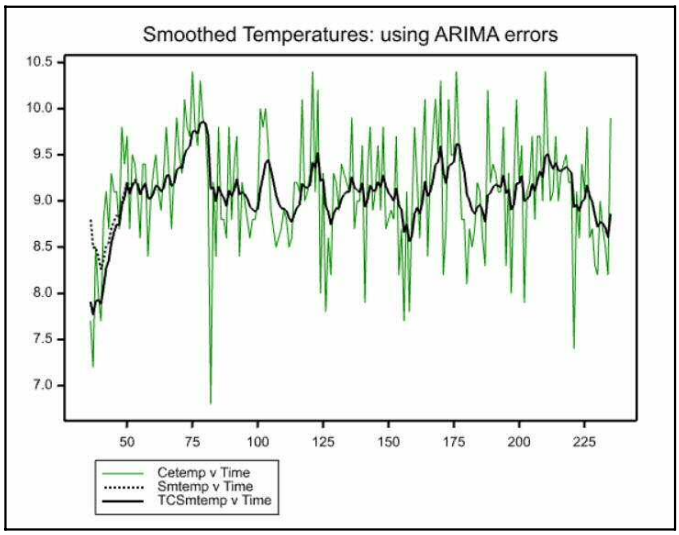


Figure 7.6.1a

Example 7.6.1a

```
" Filter the temperatures using an ARIMA model to reduce the transients"
TSM       Back; ORDERS=!(0,1,1); PARAMETERS=!(1,0,0,0.7)
TFILTER   Cetemp; NEWSERIES=TCSmtemp; FILTER=Ewma; ARIMA=Back
PEN       3; METHOD=line; LINE=2; SYMB=0; COLOUR='black'; THICKNESS=2
DGRAPH    [TITLE='Smoothed Temperatures: using ARIMA errors'] \
          Cetemp,Smtemp,TCSmtemp; Time; PEN=1,3,2
```

For a seasonal monthly time series, an appropriate ARIMA model could have orders (0,1,1,0,1,1,12) and parameters (1,0,0,0.7,0.7). However you must give the supplied model a transformation parameter λ=1. Any other value for λ breaks the assumption of linearity that underlies the calculations for correcting the transients. The constant term in the ARIMA model can be non-zero, and should be if that is appropriate for the old series. Note that the ARIMA model does not define the filter.

If you specify the `ARIMA` parameter, Genstat uses this model to interpolate any missing values in the old series before it calculates the new series. Suppose for example that the filter is the identity, defined by a transfer-function model with orders (0,0,0,0) and parameters (1,0,0); then the new series will be the old series with any missing values replaced.

Example 7.6.1b shows how a two-sided filter arises by smoothing the smoothed series a second time *after* it has been reversed. The ARIMA model has its moving average parameter set to zero because this is appropriate for the series to which the filter is now applied. The result is reversed again and displayed using DGRAPH, see Figure 7.6.1b.

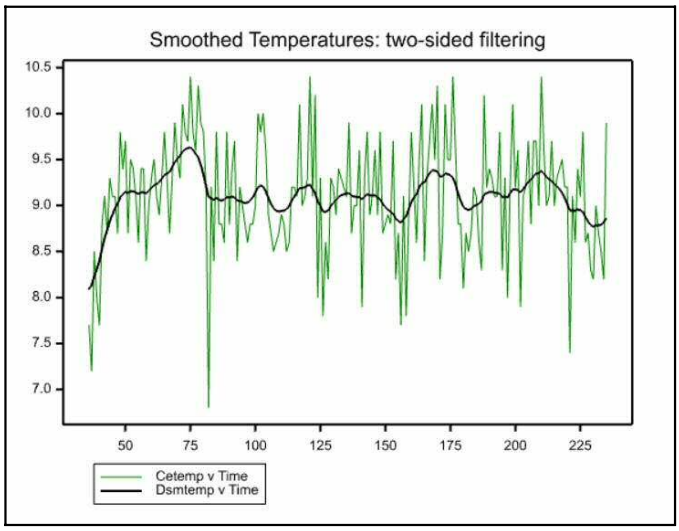


Figure 7.6.1b

Example 7.6.1b

```
 29  " Carry out two-sided filtering by applying the filter to the
-30    smoothed series in reverse."
 31  CALCULATE Rsmtemp = REVERSE(TCSmtemp)
 32  &         Back[2]$[4] = 0
 33  TFILTER   Rsmtemp; NEWSERIES=Dsmtemp; FILTER=Ewma; ARIMA=Back
 34  CALCULATE Dsmtemp = REVERSE(Dsmtemp)
 35  DGRAPH    [TITLE='Smoothed Temperatures: two-sided filtering'] \
 36            Cetemp,Dsmtemp; Time; PEN=1,2
```

7.7 Forming preliminary estimates and displaying models

The TFIT directive (7.3.3) carries out a lot of computation to find the best estimates of the parameters of a time-series model. The amount of computation can be reduced if you provide rough initial values for the parameters, especially when there are many of them. You can get Genstat to do this by using the FTSM directive. FTSM obtains moment estimators of a simple kind, by solving equations between the unknown parameters of the ARIMA or transfer-function model and the autocorrelations or cross-correlations calculated from the observed time series. Sometimes these equations have no solution, or their solution provides values inconsistent with the constraints demanded of the parameters. If so, Genstat sets the corresponding parameters to missing values. The form of the directive is the same for ARIMA and transfer-function models, but the interpretation is slightly different. So we describe the two cases separately.

The TSUMMARIZE directive helps you investigate time-series models by displaying various characteristics. These are the theoretical autocorrelation function of an ARIMA model, and the pi-weights and psi-weights; also the impulse-response function of a transfer-function model. TSUMMARIZE can derive the expanded form of a model, in which all seasonal terms are combined with the non-seasonal term.

7.7.1 Preliminary estimation of ARIMA model parameters

FTSM directive

Forms preliminary estimates of parameters in time-series models.

Option

PRINT = *string tokens*	What to print (models); default *

Parameters

TSM = *TSMs*	Models whose parameters are to be estimated
CORRELATIONS = *variates*	Auto- or cross-correlations on which to base estimates for each model
BOXCOXTRANSFORM = *scalars*	Box-Cox transformation parameter
CONSTANTTERM = *scalars*	Constant term
VARIANCE = *scalars*	Variance of ARIMA model, or ratio of input variance to output variance for transfer model

A typical FTSM statement might be

```
FTSM [PRINT=model] Yatsm; CORRELATIONS=Yacf;\
  BOXCOX=Ytran; CONSTANTTERM=Ymean; VARIANCE=Yvar
```

You must previously have declared the time-series model Yatsm to be of type ARIMA with appropriate orders, and lags if you need to specify them. Genstat takes this model to be associated with observations of a time series y_t. The aim of the directive is to set the values of the variate of model parameters equal to preliminary estimates derived from the variate Yacf and scalars Ytran, Ymn and Yvar.

The variate Yacf should contain sample autocorrelations r_0 ... r_m. You should obtain these from the original time series, stored in variate Y say, by first using the CALCULATE directive to transform Y according to the Box-Cox equations with transformation parameter Ytran (if you do indeed want a transformation). You should then form the differences of the transformed series, according to the degrees of differencing already set in the model; you can use the DIFFERENCE function with the CALCULATE directive for this (1:4.2.1). Finally, you should use the AUTOCORRELATIONS parameter of the CORRELATE directive (7.1.2) to store the autocorrelations of the resulting series in Yacf. Often you will have done these operations already in order to produce Yacf for selecting a model.

At the same time, you can supply the scalars Ytran, Ymean and Yvar to set the first three elements of the parameters variate of Yatsm; these cannot be set using Yacf alone. The scalar Ytran should be the parameter used to transform Y, and Genstat will copy it into the first element of the variate of parameters. Genstat will copy the scalar Ymean into the second element, which is the constant term of the model; the recommended value for this is the sample mean of the series from which Yacf is calculated, but you may prefer the value 0. The scalar Yvar is used to set the innovation variance, which is the third element of the variate of parameters. The recommended value is the sample variance of the series from which Yacf is calculated. If you set Yvar to 1.0, then Genstat will set the innovation variance to the variance ratio Variance(e)/Variance(y), as estimated from Yacf according to the model.

If any of the BOXCOX, CONSTANTTERM or VARIANCE parameters is not set, Genstat will leave unchanged the corresponding value in the variate of parameters of the model. The only exception to this rule is if a parameter is missing. Then Genstat initially sets the transformation parameter to 1.0 (corresponding to no transformation), and the constant to 0.0; the innovation variance is left missing.

7.7.2 Preliminary estimation of transfer-function model parameters

A typical FTSM statement for a transfer-function model might be

```
FTSM [PRINT=model] Xytsm; CORRELATIONS=Xyccf; \
  BOXCOX=Xtran; CONSTANTTERM=Xmean; VARIANCE=Xyvratio
```

You must previously have declared the time-series model Xytsm to be of type transferfunction with appropriate orders, and lags if you need to specify them. Genstat assumes that this model represents the dependence of an output series y_t on an input series x_t in a multi-input model. The directive sets the values of the parameters of the model equal to preliminary estimates derived from Xyccf, Xtran, Xmean and Xyvratio.

You should put into the variate Xyccf an estimate of the impulse-response function of the model, from which Genstat will derive the parameters. This estimate is usually a sample cross-correlation sequence $r_0 \ldots r_m$ obtained from variates Y and X1 containing observations of y_t and x_t according to one of the following four rules:

(a) In the simple case, the differencing orders of Xytsm are all zero, and you do not want to use any Box-Cox transformation of either y_t or x_t. Then the cross-correlations should be those between variates Alpha and Beta, say, derived from X and Y by filtering (or pre-whitening), as described in Section 7.6.2. The ARIMA model that you used for the filter should be the same for X and Y, and you should choose it so that the values of Alpha represent white noise.

(b) If the differencing orders of Xytsm are not zero, then before you calculate the cross-correlations you should further difference the series Beta as specified by these orders.

(c) If a Box-Cox transformation is associated with y_t, you should apply it to Y before the filtering. However this transformation parameter must not be associated with Xytsm: you should assign it to the univariate ARIMA model that you have specified for the error term (7.3.2).

(d) If a Box-Cox transformation is associated with x_t, it must be the same as the one you used in the ARIMA model for x_t from which the series Alpha was derived. The scalar Xtran must contain this transformation parameter. Genstat copies it into the first element of the parameter variate of Xytsm. If the Box-Cox parameter is unset, Genstat leaves the transformation parameter of Xytsm unchanged; it is set to 1.0 if it was originally missing.

Genstat copies the scalar Xmean into the second element of the variate of parameters. The recommended value is the sample mean of X after any transformation has been applied. If you do not set the CONSTANTTERM parameter, Genstat leaves the constant parameter of Xytsm unchanged; it is set to 0.0 if it was originally missing.

You use the scalar Xyvratio to obtain the correct scaling of non-seasonal moving-average parameters in Xytsm. All the other autoregressive parameters and moving-average parameters are invariant under scale changes in y_t and x_t. You should set the scalar to the ratio of the sample variances of the variates from which the cross-correlations were calculated; that is, Variance(Beta)/Variance(Alpha). If you do not set this, Genstat uses the value 1.0.

You can use FTSM to go backwards from autocorrelations to the original time-series model. If you apply it to the autocorrelations that were constructed from a time-series model by means of TSUMMARIZE (7.7.3), it will recover the parameters of the model exactly, provided the model is non-seasonal. If the model contains seasonal parameters, with seasonal period s, the parameters will not be recovered exactly, except in one special circumstance: that is, when the non-seasonal part of the model, considered in isolation from the seasonal part, has a theoretical autocorrelation function that is zero beyond lag $s/2$. Otherwise, the non-seasonal and seasonal parts of the model interact, and so Genstat loses accuracy in the recovered parameters. When you use sample autocorrelations, this loss of accuracy tends to be small in comparison with the sampling fluctuations of the estimates. But if s is small, say s=4 for quarterly data, the loss could be serious. Exactly the same considerations apply to transfer-function models.

7.7.3 The TSUMMARIZE directive

TSUMMARIZE directive

Displays characteristics of time series models.

Options

PRINT = *string tokens*	What to print (autocorrelations, expansion, impulse, piweight, psiweight); default *
GRAPH = *string tokens*	What to display with graphs (autocorrelations, impulse, piweight, psiweight); default *
MAXLAG = *scalar*	Maximum lag for results; default 30

Parameters

TSM = *TSMs*	Models to be displayed
AUTOCORRELATIONS = *variates*	To save theoretical autocorrelations
IMPULSERESPONSE = *variates*	To save impulse-response function
STEPFUNCTION = *variates*	To save step function from impulse
PIWEIGHTS = *variates*	To save pi-weights
PSIWEIGHTS = *variates*	To save psi-weights
EXPANSION = *TSMs*	To save expanded models
VARIANCE = *scalars*	To save variance of each TSM

For an ARIMA model in the TSM parameter, you can set only the AUTOCORRELATIONS, PSIWEIGHTS and PIWEIGHTS parameters. Also, you can set the IMPULSERESPONSE parameter only for a transfer-function model. You can set the EXPAND parameter for either type of model. The TSMs in any TSUMMARIZE statement must be completely defined; that is, you must have set the orders and parameters, and the lags if you are using them. The only exceptions are that Genstat takes the transformation parameter to be 1.0 if it is missing, and that the innovation variance of an ARIMA model need not be set.

The MAXLAG option specifies the maximum lag to which Genstat is to do calculations: this applies to autocorrelations, psi-weights, pi-weights and impulse responses.

You can set the PRINT and GRAPH options independently of the parameters: these store results, and display the various characteristics of models.

The AUTOCORRELATIONS parameter allows you to store the theoretical autocorrelation function of an ARIMA model. Such a model uniquely defines an autocorrelation function whose values $r_0 \ldots r_m$ are assigned by Genstat to the variate R, where m is the maximum lag. If the model has differencing parameters d=D=0, then the autocorrelation function is that of a series y_t that follows this model.

If either d>0 or D>0, then the theoretical autocorrelations are calculated as if d=D=0, and so they correspond to those of the differenced y_t series. This is because the autocorrelations of y_t are undefined for non-stationary models.

Example 7.7.3

```
2  " Display the autocorrelations of an AR[2] model."
3  TSM        AR[2]; ORDERS=!(2,0,0); PARAMETERS=!(1,15,2.5,0.5,-0.5)
4  TSUMMARIZE [MAXLAG=12; PRINT=autocorrelations] AR[2]
```

```
Summary of model AR[2]
======================

          Lag          ACF
            0        1.000
            1        0.333
            2       -0.333
            3       -0.333
            4        0.000
            5        0.167
            6        0.083
            7       -0.042
            8       -0.062
            9       -0.010
           10        0.026
           11        0.018
           12       -0.004
```

The PSIWEIGHTS parameter allows you to store the theoretical psi-weights $\psi_0 \ldots \psi_m$ of an ARIMA model. These are used internally by Genstat when error limits are calculated for forecasts obtained using the model. You will need them for example if you want to calculate the variance of the total of the forecast values up to some specified maximum lead time. They are defined for a non-seasonal model by

$$1 + \psi_1 B + \psi_2 B^2 + \ldots = \theta(B) \,/\, \{\, \varphi(B)\nabla^d \,\}$$

The PIWEIGHTS parameter allows you to store the theoretical pi-weights $\pi_0 \ldots \pi_m$ of an ARIMA model: these show explicitly how past values contribute to a forecast. The weights are defined by:

$$1 - \pi_1 B - \pi_2 B^2 - \ldots = \{\, \varphi(B)\nabla^d \,\} \,/\, \theta(B)$$

The IMPULSERESPONSE parameter allows you to store the theoretical impulse-response function, $v_0 \ldots v_m$, of a transfer-function model. This function can help you interpret the model. The sequence is defined for a non-seasonal transfer-function model by:

$$\nu_0 + \nu_1 B + \nu_2 B^2 + \ldots = \omega(B)B^b \,/\, \{\, \delta(B)\nabla^d \,\}$$

7.7.4 Deriving the generalized form of a time-series model

For an ARIMA model you can combine into one generalized autoregressive operator all the differencing operators, the non-seasonal autoregressive operators, and the seasonal autoregressive operators. The non-seasonal and seasonal moving-average operators may similarly be combined.

Normally you would want this expanded model to help you understand a series. But you might also want to re-estimate the parameters in the expanded model, to test whether the differencing operators or seasonal factors unnecessarily constrain the structure of the original model.

Example 7.7.4

```
   5  " Expand the seasonal ARIMA model used for modelling the number of
  -6    airline passengers in Section 7.3.7."
   7  VARIATE    [VALUES=0,1,1, 0,1,1,12] Ord
   8  &          [VALUES=0,0,0.00143, 0.34, 0.54] Par
   9  TSM        Airpass; ORDERS=Ord; PARAMETERS=Par
  10  PRINT      Airpass

Airpass

Innovation variance 0.001430

                        parameter
Transformation                  0.
Constant                        0.

Non-seasonal; differencing order 1
```

```
                lag    parameter
Moving-average    1     0.340000

Seasonal; period 12; differencing order 1

                lag    parameter
Moving-average   12     0.540000


  11  TSUMMARIZE [PRINT=expansion] Airpass

Expansion of model Airpass
==========================

Autoregressive moving-average model
-----------------------------------

Innovation variance 0.001430

                        parameter
Transformation                 0.
Constant                       0.

Non-seasonal; no differencing

                lag    parameter
Autoregressive    1      1.00000
                 12      1.00000
                 13     -1.00000
Moving-average    1     0.340000
                 12     0.540000
                 13    -0.183600
```

If you have not previously defined one of the identifiers supplied by the EXPANSION parameter, Genstat will automatically define it to be a TSM, and its component variates will be set up to have the length defined by the corresponding model in the TSM parameter.

The expansion does not change the transformation parameter of the model, nor the constant term, nor the innovation variance. If the model that you have supplied contains non-zero differencing orders, then the generalized model does not satisfy the stationarity constraint on the parameters; neither does the constant term have the same interpretation as it had in the supplied model.

The expansion of transfer-function models exactly parallels that of ARIMA models.

8 Spatial and temporal modelling

This chapter describes the specialist facilities in Genstat for the analysis of data whose distribution in space or time is the main interest. These are in addition to the covariance modelling facilities provided by the REML directive (see Section 5.4).

Section 8.1 describes several of the procedures in the Genstat Library for the analysis of repeated measurements. Others are covered elsewhere in this book, or in Part 3 of the *Genstat Reference Manual*.

DREPMEASURES	plots profiles and differences of profiles for repeated measures data (8.1.1)
VORTHPOLYNOMIAL	calculates orthogonal polynomial time-contrasts for repeated measures (8.1.2)
AREPMEASURES	produces an analysis of variance for repeated measurements (8.1.3)
MANOVA	performs multivariate analysis of variance and covariance (6.6.1, 8.1.4)
RMULTIVARIATE	provides multivariate linear regression with accumulated testing of terms (6.6.2)
ANTORDER	assesses order of ante-dependence for repeated measures data (8.1.5)
ANTTEST	calculates overall tests based on a specified order of ante-dependence (8.1.5)
ANTMVESTIMATE	estimates missing values in repeated measurements using ante-dependence structure
RAR1	fits regressions with an AR1 or a power-distance correlation model (8.1.6)
NLAR1	fits curves with an AR1 or a power-distance correlation model (8.1.6)
CUMDISTRIBUTION	fits frequency distributions to accumulated counts
DTIMEPLOT	produces horizontal bars displaying a continuous time record
GEE	fits models to longitudinal data by generalized estimating equations (3.5.10)
VHOMOGENEITY	tests homogeneity of variances
AFCARRYOVER	forms factors to represent carry-over effects in cross-over trials
AGCROSSOVERLATIN	generates Latin squares balanced for carry-over effects (4.9.4)

Profile plots, antedependence analysis, analysis of variance of repeated measurements and multivariate analysis of variance are all accessible through repeated measurements menus in Genstat *for Windows* (click on Stats on the menu bar, select Repeated Measurements and then the analysis required).

Section 8.2 covers the specialist procedures for survival analysis.

KAPLANMEIER	calculates the Kaplan-Meier estimate of the survivor function (8.2.1)
RLIFETABLE	calculates the life-table estimate of the survivor function (8.2.3)
RPHFIT	fits the proportional hazards model to survival data as a generalized linear model (8.2.5)

RPHCHANGE	modifies a proportional hazards model fitted by RPHFIT (8.2.5)
RPHDISPLAY	prints output for a proportional hazards model fitted by RPHFIT (8.2.5)
RPHKEEP	saves information from a proportional hazards model fitted by RPHFIT (8.2.5)
RPHVECTORS	forms vectors for fitting proportional hazards data as a generalized linear model
RSURVIVAL	models survival times of exponential, Weibull or extreme-value distributions (8.2.4)
RSTEST	compares groups of right-censored survival data by nonparametric tests (8.2.2)

These are all accessible through the survival analysis menus in Genstat *for Windows* (click on Stats on the menu bar, select Survival Analysis and then the analysis required).

Section 8.3 describes the facilities for spatial analysis by "kriging", a method originating in geostatistics for analysing data distributed in two dimensions. The kriging model specifies how successive measurements of a variable in space are correlated with each other, in terms of a "variogram". This is analogous to the "correlogram" used in the analysis of time series, but for two-dimensional (spatial) data rather than one-dimensional (temporal) data. There are also commands for "cokriging", which models the spatial behaviour of several variables at once (8.3.4). This is useful if a variable, that is difficult or expensive to observe, is correlated with other variables that are easier or cheaper.

FVARIOGRAM	forms auto-variograms for individual variates or cross-variograms for pairs of variates (8.3.1)
MVARIOGRAM	fits models to an experimental variogram (8.3.2)
DVARIOGRAM	plots fitted models to an experimental variogram (8.3.3)
KRIGE	calculates kriged estimates using a model fitted to a sample variogram (8.3.4)
KCROSSVALIDATION	computes cross validation statistics for punctual kriging
FCOVARIOGRAM	forms a covariogram structure containing auto-variograms of individual variates and cross-variograms for pairs from a list of variates (8.3.6)
MCOVARIOGRAM	fits models to sets of variograms and cross-variograms (8.3.7)
DCOVARIOGRAM	plots 2-dimensional auto- and cross-variograms (8.3.8)
COKRIGE	calculates kriged estimates using a model fitted to the sample variograms and cross-variograms of a set of variates (8.3.9)

These are also accessible through menus in Genstat *for Windows*, this time in the geostatistics section (click on Stats on the menu bar, select Geostatistics and then the analysis required).

Section 8.4 introduces the procedures for plotting, manipulating and analysing spatial or spatial and temporal point patterns.

DKSTPLOT	produces diagnostic plots for space-time clustering
DPOLYGON	draws polygons using high-resolution graphics
DPTMAP	draws maps for spatial point patterns using high-resolution graphics
DPTREAD	adds points interactively to a spatial point pattern
DRPOLYGON	reads a polygon interactively from the current graphics device
DPSPECTRALPLOT	calculates an estimate of the spectrum of a spatial point

	pattern
FHAT	calculates an estimate of the F nearest-neighbour distribution function
FZERO	gives the F function expectation under complete spatial randomness
GHAT	calculates an estimate of the G nearest-neighbour distribution function
GRLABEL	randomly labels two or more spatial point patterns
GRTHIN	randomly thins a spatial point pattern
GRTORSHIFT	performs a random toroidal shift on a spatial point pattern
GRCSR	generates completely spatially random points in a polygon
KCSRENVELOPES	simulates K function bounds under complete spatial randomness
KHAT	calculates an estimate of the K function
KLABENVELOPES	gives bounds for K function differences under random labelling
KSED	calculates s.e. for K function differences under random labelling
KSTHAT	calculates an estimate of the K function in space, time and space-time
KSTMCTEST	performs a Monte-Carlo test for space-time interaction
KSTSE	calculates the standard error for the space-time K function
KTORENVELOPES	gives bounds for the bivariate K function under independence
K12HAT	calculates an estimate of the bivariate K function
MSEKERNEL2D	estimates the mean square error for a kernel smoothing
PTAREAPOLYGON	calculates the area of a polygon
PTBOX	generates a box bounding or surrounding a spatial point pattern
PTCLOSEPOLYGON	closes open polygons
PTDESCRIBE	gives summary and second order statistics for a point process
PTGRID	generates a grid of points in a polygon
PTINTENSITY	calculates the overall density for a spatial point pattern
PTKERNEL2D	performs kernel smoothing of a spatial point pattern
PTK3D	performs kernel smoothing of space-time data
PTREMOVE	removes points interactively from a spatial point pattern
PTROTATE	rotates a point pattern
PTSINPOLYGON	returns points inside or outside a polygon

Finally, Section 8.5 describes facilities forming cluster of points in multi-dimensional space. These are particularly useful for large and very large data sets.

ADJACENTCELLS	finds cells adjacent to other cells in a multi-dimensional array
NEIGHBOURS	finds the neighbours of cells in a multi-dimensional array
PTFCLUSTERS	forms clusters of points from their densities in multi-dimensional space (8.5.1)
PTFILLCLUSTERS	fills holes in clusters of points within multi-dimensional space (8.5.2)

8.1 Repeated measurements

A repeated-measurements study is one in which subjects (animals, people, plots, etc) are observed on several occasions. Each subject usually receives some randomly allocated treatment, either at the outset or repeatedly through the investigation, and is then observed at successive occasions to see how the treatment effects develop. Genstat has a comprehensive collection of procedures for the analysis of such data (see the list at the start of this chapter). Most of them assume that the repeated measurements are *balanced*, that is that the subjects are all observed at the same times relative to the start of the study. The data are stored in separate variates, each containing the measurements at one of the times.

8.1.1 Plotting repeated measurements

DREPMEASURES procedure

Plots profiles and differences of profiles for repeated measures data (J.T.N.M. Thissen).

Options

TITLE = *text*	Title for the plots; default *
GROUPS = *factors*	List of one or two factors; one factor gives one plot while a list with two factors gives as many plots as the number of levels of the first factor in the list; must be set
TIMEPOINTS = *variate* or *factor*	When the DATA parameter is set to a pointer containing a separate variate of observations for each time this can specify the actual time points (otherwise the suffixes of the DATA pointer are used), when there is a single DATA variate this must supply a factor to indicate the time of each observation
DIFFERENCES = *string token*	Can suppress plotting of the differences (no, yes); default no

Parameters

DATA = *pointers* or *variates*	Data observations either in a pointer to a list of variates (one for each time), or a single variate (with TIMEPOINTS set to a factor indicating the time of each observation)
GROUPMEANS = *tables*	To save the calculated treatment means at each timepoint

It is usually helpful first to plot the data. Example 8.1.1 uses DREPMEASURES to plot data from a study of the effects of the drugs morphine and trimethaphan on histamine release and hypotension in dogs; see Figures 8.1.1a and 8.1.1b The treatments had a 2 × 2 factorial structure. Half the dogs received intravenous morphine sulphate and the remainder received intravenous trimethaphan as indicated by the Drug factor. The other aspect, factor Hist, was that some of the dogs were treated so that their supplies of available histamine were deleted when the treatment drugs were innoculated. Measurements were made of blood histamine immediately before treatment and at one, three and five minutes afterwards.

The data can be specified in one of two ways. The first is to set the DATA parameter to a pointer containing a list of variates, each one containing the measurements made on the subjects at one of the successive occasions on which they were observed. The TIMEPOINTS option can then supply a variate to define the time point corresponding to each DATA variate; if TIMEPOINTS is unset, the suffixes of the DATA pointer are used. The second possibility is to

supply set DATA to a variate containing the data from all the times. The TIMEPOINTS option must then be set to a factor indicating the time of each observation.

The grouping of the subjects can be specified by either one or two factors, input using the GROUPS option. If one factor is specified, the means of the observations at each level of the factor are plotted in one graph. If, as in Example 8.1.1, two factors are specified several graphs are produced: each graph is a plot of the means of the observations at the various levels of the second factor for a particular level of the first. The means are calculated with the directive TABULATE. If the data variates contain missing values a warning is printed indicating that the results may be misleading; missing values in repeated measurements can be estimated using ANTMVESTIMATE (8.1.5).

If DATA is set to a pointer, you can arrange to plot only a subset of the measurements by restricting any of the DATA variates or GROUPS factors. The variate specified by TIMEPOINTS for a DATA pointer must not be restricted. Similarly if DATA is set to a variate, you can restrict either the DATA variate or the GROUPS or TIMEPOINTS factors. If more than one variate or factor is restricted, they must all be restricted to the same set of units.

Setting the DIFFERENCES option to yes produces two plots: one of the profiles and the other of differences with the first level, and the TITLE option can provide a title for the plots.

Example 8.1.1

```
  2  " Blood histamine levels in dogs:
 -3    data from Morris & Zeppa, 1963, J. Surg. Res. 3, 313-317;
 -4    also see Cole & Grizzle, 1966, Biometrics 22, 810-828. "
  5  FACTOR [LABELS=!t(morphine,trimethaphan)] Drug
  6  & [LABELS=!t(intact,depleted)] Histlev
  7  READ Drug,Histlev,Hist[0,1,3,5]; FREPRESENTATION=labels

   Identifier   Minimum      Mean   Maximum    Values   Missing
      Hist[0]   0.02000   0.07687    0.1700        16         0
      Hist[1]   0.05000    0.5331     3.130        16         0    Skew
      Hist[3]   0.02000    0.3644     2.060        16         0    Skew
      Hist[5]   0.02000    0.2707     1.230        16         1    Skew

   Identifier    Values   Missing    Levels
         Drug        16         0         2
      Histlev        16         0         2

 24  DREPMEASURES [GROUPS=Drug,Histlev] Hist

******** Warning, code UF 2, statement 63 in procedure DREPMEASURES

There are missing values in the DATA pointer; plots of means can be misleading.
```

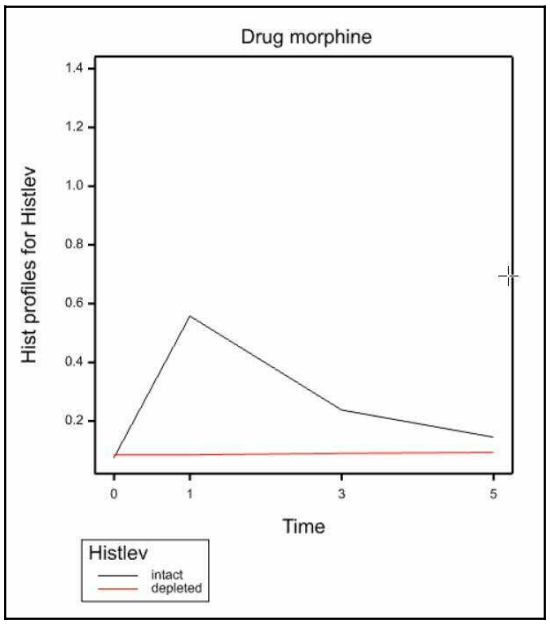


Figure 8.1.1a

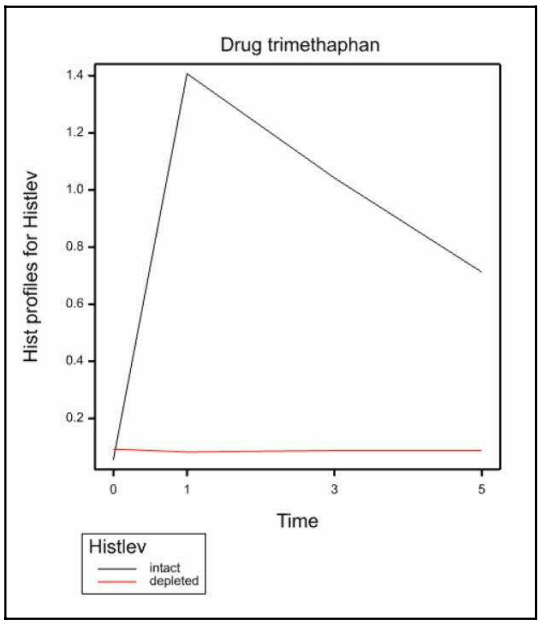


Figure 8.1.1b

8.1.2 Analysis of polynomial contrasts

VORTHPOLYNOMIAL procedure

Forms orthogonal polynomials over time for repeated measures (J.T.N.M. Thissen).

Options

TIMEPOINTS = *variate*	Variate of timepoints; default uses the suffixes of the DATA pointer
MAXDEGREE = *scalar*	The number of contrasts (excluding the mean); default is the number of identifiers in the CONTRAST pointer minus 1

Parameters

DATA = *pointers*	Each pointer contains the data variates (observed at successive times); must be set
CONTRAST = *pointers*	To save the calculated contrasts: the first variate contains the means, the second the linear polynomial contrasts, the third the quadratic polynomial contrasts etc; must be set

With measurements like milk yields or amounts of industrial production, for example, the main interest may be in the total of the measurements for each subject. Alternatively, with measurements of growth, you might want to analyse the change over the period. Polynomials can also be popular and, with balanced repeated measurements, the coefficients of orthogonal polynomials can be calculated automatically using procedure VORTHPOLYNOMIAL. The observed data and timepoints are specified in the same way as for the DREPMEASURES procedure (8.1.1). The calculated polynomial contrasts are saved in a pointer whose identifier must be specified by the CONTRAST parameter. This contains a list of variates: the first variate saves the means over the DATA variates, the second variate saves the linear polynomial contrast, the third the quadratic polynomial, and so on. Provided the MAXDEGREE option is used to specify the required number of contrasts, the pointer need not be declared in advance, and its suffixes will be defined to be

0, 1, 2 ... If MAXDEGREE is not set, the number of contrasts is taken from the length of the CONTRAST pointer. If a subject has a missing value at any time, the contrasts for the subject will also be missing. Example 8.1.2 forms the contrasts up to order 3 for the data plotted in Example 8.1.1, and then analyses them by analysis of variance, using the ANOVA directive (4.1.2). The analysis indicates that there are indeed differences between the treatments but confirms the impression from the graphs that, for this set of data, there is no particularly straightforward polynomial representation.

Example 8.1.2

```
 25  CALCULATE Hist[] = LOG(Hist[])
 26  VORTHPOLYNOMIAL [MAXDEGREE=3] Hist; Pol
 27  TREATMENT Drug*Histlev
 28  ANOVA [PRINT=aov; FPROB=yes] Pol[]

Analysis of variance
====================

Variate: Pol[0]

Source of variation     d.f.(m.v.)      s.s.       m.s.    v.r.  F pr.
Drug                         1          1.5906     1.5906    2.88  0.118
Histlev                      1          4.1316     4.1316    7.48  0.019
Drug.Histlev                 1          1.2996     1.2996    2.35  0.153
Residual                  11(1)         6.0765     0.5524
Total                     14(1)        12.7890


Analysis of variance
====================

Variate: Pol[1]

Source of variation     d.f.(m.v.)      s.s.       m.s.    v.r.  F pr.
Drug                         1        0.085185   0.085185   22.25  <.001
Histlev                      1        0.098368   0.098368   25.69  <.001
Drug.Histlev                 1        0.152116   0.152116   39.73  <.001
Residual                  11(1)       0.042116   0.003829
Total                     14(1)       0.374690


Analysis of variance
====================

Variate: Pol[2]

Source of variation     d.f.(m.v.)      s.s.       m.s.    v.r.  F pr.
Drug                         1        0.021733   0.021733    6.26  0.029
Histlev                      1        0.232346   0.232346   66.95  <.001
Drug.Histlev                 1        0.029160   0.029160    8.40  0.014
Residual                  11(1)       0.038174   0.003470
Total                     14(1)       0.307387


Analysis of variance
====================

Variate: Pol[3]

Source of variation     d.f.(m.v.)      s.s.       m.s.    v.r.  F pr.
Drug                         1        0.002568   0.002568    1.26  0.285
Histlev                      1        0.146339   0.146339   72.06  <.001
Drug.Histlev                 1        0.005825   0.005825    2.87  0.118
Residual                  11(1)       0.022338   0.002031
Total                     14(1)       0.168578
```

8.1.3 Repeated-measures analysis of variance

The data in Examples 8.1.1 and 8.1.2 may seem to come from a split-plot design, with subjects (dogs) corresponding to whole plots, and the occasions of observation to the sub-plots. There are, however, some important differences between the two situations. With repeated measurements, there is likely to be a greater correlation between observations that are made at adjacent time points than between those that are more greatly spaced. Furthermore, the `Time` factor cannot, by its very nature, be allocated at random to the occasions within subjects. In the customary split-plot situation we can usually assume that there is an equal correlation between the sub-plots of each whole plot and, even if this were not so, the sub-plot treatment should have been allocated at random to the sub-plots within each whole plot.

Before discussing the formal conditions for the validity of the split-plot analysis, it is worth pointing out, though, that this problem affects only the `Subject.Time` stratum. The `Subject` stratum contains an analysis of variance of the measurements totalled over the subjects, and this part of the analysis will be valid whatever the within-subject correlation structure. A further point is that, when measurements are taken on only two occasions, the analysis in the `Subject.Time` stratum will also be valid; there can then be only one within-subject correlation, and the analysis in the `Subject.Time` stratum is of the difference between the observations at time 2 and time 1 on each subject.

Another potential problem arising from the systematic nature of the `Time` factor is that effects arising from the "length of treatment time" will be confounded with any effects arising from the duration of the experiment, such as age of subject (which may be important with short-lived material such as aphids), season of year, time of day, and so on. This does not affect the validity of the analysis, and some of the confusion may be capable of being unravelled by running the experiment during more than one period. Nevertheless, care needs to be taken in drawing conclusions about time-effects.

The `Subject.Time` information, describing the way in which the treatment effects change differentially with time, is generally the aspect of most interest in the study. The formal requirement for the validity of the analysis in the sub-plot stratum of a split-plot design is that all the normalised contrasts in that stratum have an equal variance. The only practical arrangement of covariances between times that satisfies this condition would have a single variance down the diagonal and a single covariance off-diagonal. This pattern is known as a *uniform covariance structure* or, equivalently, the matrix is said to show *compound symmetry*; Box (1950) describes how this can be tested. In the usual split-plot analysis, the `Subject.Time` sum of squares is assumed to be distributed as $\sigma^2 \times \chi^2_r$ where σ^2 is a constant and χ^2_r has a chi-square distribution on r degrees of freedom. Similarly, under the assumption that there is no `Treatment.Time` interaction, the `Treatment.Time` sum of squares is assumed to be distributed as $\sigma^2 \times \chi^2_t$ where χ^2_t has a chi-square distribution on t degrees of freedom. If the variance-covariance structure does not exhibit compound symmetry, it is possible to show that the distributions can still be approximated by chi-square distributions, but the degrees of freedom are instead $\varepsilon \times r$ and $\varepsilon \times t$. The correction factor ε lies between one, which would give the ordinary split-plot analysis, and 1/(number of times minus one), which would leave just one degree of freedom within each subject (remember that when there are only two observation on each subject, and thus just one within-subject degree of freedom, the analysis is valid); ε can be estimated be maximum likelihood, as described by Greenhouse & Geisser (1959).

AREPMEASURES procedure

Produces an analysis of variance for repeated measurements (R.W. Payne).

Options

`PRINT` = *string tokens*	Controls output about the covariance structure (`vcovariance`, `correlation`, `epsilon`, `test`);

	default `epsi, test`
`APRINT` = *string tokens*	Printed output from the analysis of variance (as for the `ANOVA PRINT` option); default `*`
`TREATMENTSTRUCTURE` = *formula*	Defines the treatments given to the subjects; if this is not set, the default is taken from any existing setting defined by the `TREATMENTSTRUCTURE` directive
`BLOCKSTRUCTURE` = *formula*	Defines any block structure over the subjects if this is not set, the default is taken from any existing setting defined by the `BLOCKSTRUCTURE` directive
`COVARIATE` = *variates*	Specifies any covariates on the subjects if this is not set, the default is taken from any existing setting defined by the `COVARIATE` directive
`FACTORIAL` = *scalar*	Limit in the number of factors in the terms generated from the `TREATMENTSTRUCTURE` formula
`TIMEPOINTS` = *variate*, *text* or *factor*	When the `DATA` parameter supplies a separate variate of observations for each time this can specify numbers or labels for the time points, when there is a single `DATA` variate this must supply a factor to indicate the time of each observation
`CONTRASTS` = *scalar*	Limit on the order of a contrast of a treatment term; default 4
`DEVIATIONS` = *scalar*	Limit on the number of factors in a treatment term for the deviations from its fitted contrasts to be retained in the model; default 9
`FPROBABILITY` = *string token*	Printing of probabilities for variance ratios in the aov table (`no`, `yes`); default `no`
`PSE` = *string tokens*	Standard errors to be printed with tables of means (`differences`, `lsd`, `means`); default `diff`
`MAXCYCLE` = *scalar*	Maximum number of iterations for estimating missing values; default 20
`LSDLEVEL` = *scalar*	Significance level (%) to use in the calculation of least significant differences; default 5
`EPSILON` = *scalar*	Saves the correction factor epsilon
`SAVEFACTORS` = *pointer*	Saves the factors used in the analysis of variance
`ASAVE` = *identifier*	Saves the `ANOVA` save structure from the analysis of variance

Parameter

`DATA` = *variates*	Data observations either in a list of variates (one for each time), or a single variate (with `TIMEPOINTS` set to a factor indicating the time of each observation)

Procedure `AREPMEASURES` can be used to generate an analysis of variance for repeated measurements, estimating and applying the adjustment factor, ε, for the degrees of freedom. The estimated value of the adjustment factor, ε, can be saved by the `EPSILON` option.

Information about the patterns of the covariances is controlled by the strings listed for the `PRINT` option:

`vcovariance`	variance-covariance matrix,
`correlation`	correlation matrix,
`epsilon`	Greenhouse-Geisser ε,

`test` test for compound symmetry.

The output from the analysis of variance is controlled by the `APRINT` option, with settings identical to those in the `PRINT` option of the `ANOVA` directive (4.1.2). The `FPROBABILITY`, `PSE` and `LSDLEVEL` options also operate exactly as in `ANOVA`.

The treatments applied to the subjects can be specified (as a model formula) using the `TREATMENTSTRUCTURE` option, the block structure (if any) on the subjects can be specified by the `BLOCKSTRUCTURE` option, and the `COVARIATE` option can be used to list any covariates on the subjects (i.e. these must be constant across the times on each subject). If any of these options is unset, the default is taken from any existing setting defined by the directives `TREATMENTSTRUCTURE`, `BLOCKSTRUCTURE` or `COVARIATE`, respectively. The `FACTORIAL` `CONTRASTS`, `DEVIATIONS` and `MAXCYCLE` options operate as in the `ANOVA` directive (4.1.2).

In Example 8.1.3, we use `AREPMEASURES` to continue the analysis of the data above. Notice that the data are specified (using the `DATA` parameter) as a list of variates, rather than in a pointer. The `TIMEPOINTS` option can supply a variate or text to define numbers or labels to use in output to identify the time point corresponding to each `DATA` variate. If this is unset, the labels are formed automatically from the identifiers of the `DATA` variates themselves. In line 30 we include only the observations that took place after the treatments were applied (`Hist[0]` could of course be included as a covariate). `AREPMEASURES` uses `ANOVA` to produce the analysis of variance, but sets a private parameter inside the `ANOVA` save structure so that the degree of freedom adjustment is applied when calculating probabilities for variance ratios or least significant differences.

Example 8.1.3

```
 29  AREPMEASURES [PRINT=epsilon,correlation,vcovariance,test; APRINT=aov;\
 30    FPROBABILITY=yes] Hist[1,3,5]

Variance-covariance matrix
--------------------------

      Hist[1]   1      0.6702
      Hist[3]   2      0.6967      0.8064
      Hist[5]   3      0.5722      0.6841      0.5932
                            1           3           5

Common variance:   0.6899
Common covariance: 0.6510


Correlation matrix
------------------

      Hist[1]   1      1.0000
      Hist[3]   2      0.9478      1.0000
      Hist[5]   3      0.9075      0.9891      1.0000
                            1           3           5

Common correlation: 0.9436


Box's tests for compound symmetry of the covariance matrix
----------------------------------------------------------

Chi-square 18.82 on 4 degrees of freedom: probability 0.001

F-test 4.70 on 4 and 2904 degrees of freedom: probability 0.001


Greenhouse-Geisser epsilon
--------------------------

epsilon 0.7005
```

```
Analysis of variance
====================

Variate: Hist[1,3,5]

Source of variation     d.f.(m.v.)      s.s.       m.s.    v.r.  F pr.

Subject stratum
Drug                         1         7.45319    7.45319    4.05  0.067
Histlev                      1        25.54624   25.54624   13.88  0.003
Drug.Histlev                 1         8.79034    8.79034    4.77  0.049
Residual                    12        22.09389    1.84116   48.88

Subject.Time stratum
d.f. correction factor 0.7005
Time                         2         1.37618    0.68809   18.27  <.001
Time.Drug                    2         0.16759    0.08380    2.22  0.150
Time.Histlev                 2         1.97925    0.98963   26.27  <.001
Time.Drug.Histlev            2         0.21265    0.10633    2.82  0.102
Residual                    23(1)      0.86641    0.03767

Total                       46(1)     68.31623

(d.f. are multiplied by the correction factors before calculating
F probabilities)
```

The DATA variates are appended into a single variate for the analysis, and the block and treatment factors are expanded to match. You can specify a pointer using the SAVEFACTORS option to save the expanded factors. The elements of the pointer are labelled by the factor names, and the time factor is also included, with the label 'Time factor'. You would need to use these, for example, if you wanted to plot the means using AGRAPH. So, for example, we could

```
AREPMEASURES [PRINT=epsilon,correlation,vcovariance,test;\
  APRINT=aov; FPROBABILITY=yes; SAVEFACTORS=f] Hist[1,3,5]
AGRAPH [METHOD=lines] f['Time factor']; GROUPS=f['Histlev']
```

to make a line plot of the time by Histlev means.

An alternative way of arranging the data is to put the observations from all the times into a a single DATA variate. The TIMEPOINTS option must then be set to a factor indicating the time of each observation. The block and treatment factors must be have been defined to match the DATA variate (i.e. with a unit for every time × subject combination, all in the same order as in the DATA variate itself), and each subject should be represented by a unique combination of the block factors. If not, Genstat prints a warning and assumes that the subjects occur in the same order within each time. To simplify the use of AREPMEASURES in general programs, the SAVEFACTORS pointer is also formed when the data are in a single variate. (However, it then contains the original factors.)

The ASAVE option allows you to save the save structure from the ANOVA analysis.

8.1.4 Multivariate analysis of variance

Multivariate analysis of variance provides an alternative way of producing a combined analysis of all the repeated measurements, generating statistics that make no assumptions about the covariance structure of the measurements. With balanced data (analysable using the ANOVA directive), this can be done using the MANOVA procedure described in Section 6.6.1. Alternatively, you can use the RMULTIVARIATE procedure described in Section 6.6.2. Example 8.1.4 uses MANOVA to continue the analysis of the data in Examples 8.1.1 - 8.1.3. Notice that we do not need to specify the model to be analysed. There is only one error term, and so the BLOCKSTRUCTURE option can be omitted. The TREATMENTSTRUCTURE option (which defines the treatment terms for the analysis) can also be omitted as Genstat will then take the treatment formula specified by the TREATMENTSTRUCTURE directive in line 27 of Example 8.1.2.

Example 8.1.4

```
  31  MANOVA Hist[]

Multivariate analysis of variance
=================================

Y-variates: Hist[0], Hist[1], Hist[3], Hist[5].

Test statistics
===============

        Term  d.f.  Wilks' lambda  Rao F  n.d.f.  d.d.f. F prob.
        Drug     1         0.2945   4.79       4       8   0.029
     Histlev     1         0.1127  15.75       4       8   0.001
Drug.Histlev     1         0.1806   9.07       4       8   0.005

        Term  d.f.  Pillai-Bartlett trace  Roy's maximum root test  Lawley-Hotelling trace
        Drug     1                 0.7055                   0.7055                   2.396
     Histlev     1                 0.8873                   0.8873                   7.873
Drug.Histlev     1                 0.8194                   0.8194                   4.537
```

8.1.5 Ante-dependence structure

The lack of structure assumed for the covariances in multivariate analysis of variance means that it can be inefficient with moderate or small data sets. In particular, it cannot be used at all if the number of time points is greater than the number of residual degrees of freedom.

Ante-dependence analysis can be regarded as a generalization of multivariate analysis of variance that allows for the patterns of covariances that typify repeated measurements. The variates observed at the successive times are said to have an ante-dependence structure of order r if each ith variate ($i>r$), given the preceding r, is independent of all further preceding variates (Gabriel 1961, 1962). An ante-dependence structure of maximum order (number of times minus one) is equivalent to the assumption of an unstructured variance-covariance matrix made in multivariate analysis of variance. Procedure ANTORDER calculates statistics to assist in the selection of an appropriate order of ante-dependence structure for sets of repeated measures data, using the method of Kenward (1987). Once the order of ante-dependence structure has been established, the individual variates can be analysed individually by analysis of covariance, adjusting for the r previous variates, to assess the times at which treatment effects occurred. Alteratively, procedure ANTTEST can be used to perform overall tests of treatment effects. Knowledge of the ante-dependence structure may also be used by procedure ANTMVESTIMATE to estimate missing values.

ANTORDER procedure

Assesses order of ante-dependence for repeated measures data (M.S. Ridout & R.W. Payne).

Options

TREATMENTSTRUCTURE = *formula*	Treatment formula for the model at each time; if this is not set, the default is taken from the setting (which must already have been defined) of the TREATMENTSTRUCTURE directive
BLOCKSTRUCTURE = *formula*	Block formula for the model at each time; if this is not set, the default is taken from any existing setting specified by the BLOCKSTRUCTURE directive and if neither has been set the design is assumed to be unstratified (i.e. to have a single error term)

MAXORDER = *scalar*	Maximum order against which to test; default is maximum possible order
FACTORIAL = *scalar*	Limit on the number of factors in a treatment term
TIME = *factor*	Indicates the time of each observation when there is a single DATA variate

Parameter

DATA = *variates*	Data observations either in a list of variates (one for each time), or a single variate (with TIME set to a factor indicating the time of each observation)

The model for the analysis is specified by options of the procedure. TREATMENTSTRUCTURE specifies a model formula to define the treatment terms in the analysis; if this is unset, ANTORDER will use the model already defined by the TREATMENTSTRUCTURE directive, or will fail if that too has not been set. BLOCKSTRUCTURE defines the underlying structure of the design, and ANTORDER will use the model (if any) previously defined by the BLOCKSTRUCTURE directive if this is not set; these can both be omitted if there is only one error term (i.e. if the design is unstratified). Option MAXORDER specifies the maximum order of ante-dependence structure to be tested; by default, this is taken as the maximum possible order (see Kenward 1987). So in Example 8.1.5a we can again use the default values, taking the treatment formula specified by TREATMENTSTRUCTURE in line 27 of Example 8.1.2.

The data are specified by the DATA parameter in one of two ways. The first is to supply a list of variates, each one containing the measurements made on the subjects at one of the successive occasions on which they were observed.

The second possibility is to supply a single DATA variate containing the data from all the times. The TIME option must then be set to a factor indicating the time of each observation. The block and treatment factors must be defined to match the DATA variate, and each subject should be represented by a unique combination of the block factors. If not, Genstat prints a warning and assumes that the subjects occur in the same order within each time.

The data may contain missing values but these should represent "dropouts": that is, once subjects start to record missing values, their observations should continue to be missing at all subsequent times.

Example 8.1.5a

```
  32  ANTORDER Hist[]

Sequential comparison of ante-dependence structures
===================================================

                        Unadjusted              Adjusted
                        chi-square  Adjustment  chi-square
                         statistic      factor   statistic   d.f.      Prob
  Order  0  v. order  1      110.68       0.632       70.00      3   <0.001
  Order  1  v. order  2       20.46       0.559       11.44      2    0.003
  Order  2  v. order  3        0.87       0.467        0.41      1    0.524

Comparison of ante-dependence structures with max order
=======================================================
```

				Unadjusted chi-square statistic	Adjustment factor	Adjusted chi-square statistic	d.f.	Prob
Order	0	v. order	3	132.01	0.576	75.97	6	<0.001
Order	1	v. order	3	21.33	0.527	11.25	3	0.010
Order	2	v. order	3	0.87	0.467	0.41	1	0.524

The tables of Chi-square values show that an ante-dependence structure of order 2 represents the structure of the data better than an ante-dependence structure of order 1, but that it is not necessary to move to a structure of order 3. Assuming order 2, we can now use procedure ANTTEST to calculates overall tests for the treatment terms.

ANTTEST procedure

Calculates overall tests based on a specified order of ante-dependence (R.W. Payne & M.S. Ridout).

Options

TREATMENTSTRUCTURE = *formula*	Treatment formula for the model at each time; if this is not set, the default is taken from the setting (which must already have been defined) of the TREATMENTSTRUCTURE directive
BLOCKSTRUCTURE = *formula*	Block formula for the model at each time; if this is not set, the default is taken from any existing setting specified by the BLOCKSTRUCTURE directive and if neither has been set the design is assumed to be unstratified (i.e. to have a single error term)
ORDER = *scalar*	Number of past times for which to adjust; default is maximum possible order
FACTORIAL = *scalar*	Limit on the number of factors in a treatment term
TIME = *factor*	Indicates the time of each observation when there is a single DATA variate

Parameter

DATA = *variates*	Data observations either in a list of variates (one for each time), or a single variate (with TIME set to a factor indicating the time of each observation)

The DATA parameter and the TREATMENTSTRUCTURE and BLOCKSTRUCTURE options of ANTTEST are as in ANTORDER. Option ORDER specifies the order of ante-dependence structure to be assumed for the tests; by default, this is taken as the maximum possible order.

Example 8.1.5b

```
  33  ANTTEST [ORDER=2] Hist[]

Tests of Drug assuming ante-dependence structure of order 2
-----------------------------------------------------------

        Test for change at each time   Overall  test up to each time
 time  Statistic   d.f.  Probability  Statistic    d.f.  Probability
    1      0.003      1        0.960      0.003       1        0.960
    2      4.063      1        0.044      4.251       2        0.119
    3      7.601      1        0.006     12.394       3        0.006
    4      0.087      1        0.768     11.823       4        0.019
```

```
Overall test using data from all the times

statistic 11.823, d.f. 4, probability 0.019


Tests of Histlev assuming ante-dependence structure of order 2
--------------------------------------------------------------

        Test for change at each time   Overall  test up to each time
 time  Statistic   d.f.  Probability  Statistic    d.f.  Probability
    1      2.318      1        0.128      2.318       1        0.128
    2     20.232      1       <0.001     23.368       2       <0.001
    3      3.583      1        0.058     26.149       3       <0.001
    4      0.513      1        0.474     25.329       4       <0.001


Overall test using data from all the times

statistic 25.329, d.f. 4, probability <0.001


Tests of Drug.Histlev assuming ante-dependence structure of order 2
-------------------------------------------------------------------

        Test for change at each time   Overall  test up to each time
 time  Statistic   d.f.  Probability  Statistic    d.f.  Probability
    1      0.141      1        0.707      0.141       1        0.707
    2      5.783      1        0.016      6.182       2        0.045
    3      7.636      1        0.006     14.268       3        0.003
    4      3.512      1        0.061     17.578       4        0.001


Overall test using data from all the times

statistic 17.578, d.f. 4, probability 0.001
```

The overall test produced by ANTTEST confirms that there are differences in all the treatment terms. The right-hand columns of each table contain overall tests using the data up to each successive time, indicating how the weight of evidence builds up as the time progresses. The left-hand columns assess the information contributed by each time that is additional to that provided by earlier times (Kenward 1987, page 303); provided (as here) there is a reasonably large correlation between measurements, they can be construed as testing for a treatment effect at each time point.

Knowledge of the ante-dependence structure can be used to estimate missing values in simple designs, as shown in Example 8.1.5c. The procedure, ANTMVESTIMATE, allows for a single treatment factor, and assumes that there are replicate observations within each of its levels.

ANTMVESTIMATE procedure

Estimates missing values in repeated measurements (M.G. Kenward & R.W. Payne).

Options

PRINT = *string tokens*	Controls output from the procedure (meanprofiles); default * i.e. none
GROUPS = *factor*	Factor indicating the plot on which each sequence of observations was made
ORDER = *scalar*	Order of ante-dependence structure (i.e. number of past times for which to adjust)

Parameters

DATA = *variates*	Observations at each time
NEWDATA = *variates*	Data variates with missing observations replaced by

	their estimates
MEANPROFILE = *tables*	Estimated mean profiles at each time

The treatment factor is specified using the GROUPS option. In Example 8.1.5c we first need to use procedure FACPRODUCT first, to construct a single factor from the combinations of Drug and Histlev. The ORDER option specifies the order of ante-dependence structure; if this is not set, ANTMVESTIMATE takes the maximum possible order (number of times minus one). Using this assumption, ANTMVESTIMATE predicts the missing values and calculates the mean profiles at each time. These can be saved, in tables indexed by the GROUPS factor, using the MEANPROFILES parameter, or printed by setting the PRINT option to meanprofiles. The NEWDATA parameter allows new variates to be saved with the missing values replaced by their estimates.

Example 8.1.5c

```
 34  FACPRODUCT !p(Drug,Histlev); Treat
 35  ANTMVESTIMATE [GROUPS=Treat; ORDER=2] Hist[]; NEWDATA=RepmvH[0,1,3,5]
 36  PRINT Hist[0],RepmvH[0],Hist[1],RepmvH[1],\
 37    Hist[3],RepmvH[3],Hist[5],RepmvH[5]; FIELD=9,10

Hist[0] RepmvH[0]  Hist[1] RepmvH[1]  Hist[3] RepmvH[3]  Hist[5] RepmvH[5]
 -3.219    -3.219   -1.609    -1.609   -2.303    -2.303   -2.526    -2.526
 -3.912    -3.912   -2.813    -2.813   -3.912    -3.912   -3.912    -3.912
 -2.659    -2.659    0.336     0.336   -0.734    -0.734   -1.427    -1.427
 -1.772    -1.772   -0.562    -0.562   -1.050    -1.050   -1.427    -1.427
 -2.303    -2.303   -2.408    -2.408   -2.040    -2.040   -1.966    -1.966
 -2.120    -2.120   -2.207    -2.207   -2.303    -2.303        *    -2.399
 -2.659    -2.659   -2.659    -2.659   -2.659    -2.659   -2.659    -2.659
 -2.996    -2.996   -2.659    -2.659   -2.813    -2.813   -2.659    -2.659
 -3.507    -3.507   -0.478    -0.478   -1.171    -1.171   -1.514    -1.514
 -3.507    -3.507    0.049     0.049   -0.315    -0.315   -0.511    -0.511
 -2.659    -2.659   -0.186    -0.186    0.068     0.068   -0.223    -0.223
 -2.408    -2.408    1.141     1.141    0.723     0.723    0.207     0.207
 -2.303    -2.303   -2.408    -2.408   -2.408    -2.408   -2.526    -2.526
 -2.526    -2.526   -2.408    -2.408   -2.408    -2.408   -2.303    -2.303
 -2.040    -2.040   -2.303    -2.303   -2.120    -2.120   -2.120    -2.120
 -2.813    -2.813   -2.996    -2.996   -2.996    -2.996   -2.996    -2.996
```

8.1.6 Regression with correlated errors

RAR1 procedure

Fits regressions with an AR1 or a power-distance correlation model (R.W. Payne).

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, cparameter, cmonitoring, cplot); default mode, summ, esti, cpar
CALCULATION = *expression structures*	Calculation of explanatory variates involving nonlinear parameters
CONSTANT = *string token*	How to treat the constant (estimate, omit); default esti
FACTORIAL = *scalars*	Limit for expansion of model terms; default 3
POOL = *string token*	Whether to pool ss in accumulated summary between all terms fitted in a linear model (yes, no); default no

DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality, vertical, df, inflation); default *
FPROBABILITY = *string token*	Printing of probabilities for variance and deviance ratios (yes, no); default no
TPROBABILITY = *string token*	Printing of probabilities for t-statistics (yes, no); default no
SELECTION = *string tokens*	Statistics to be displayed in the summary of analysis produced by PRINT=summary, seobservations is relevant only for a Normally distributed response, and %cv only for a gamma-distributed response (%variance, %ss, adjustedr2, r2, seobservations, dispersion, %cv, %meandeviance, %deviance, aic, bic, sic); default %var, seob if DIST=normal, %cv if DIST=gamma, and disp for other distributions
SELINEAR = *string token*	Whether to calculate s.e.s for linear parameters when nonlinear parameters are also estimated (yes, no); default no
WEIGHTS = *variate*	Prior weights for the units
CMETHOD = *string token*	Estimation method (maximumlikelihood, reml); default maxi
CPARAMETER = *scalars*	Correlation parameter
CPOSITIONS = *variate*	Correlation positions
CGROUPS = *factor*	Groupings of correlation positions
MAXCYCLE = *scalars*	Maximum number of iterations; default 100
TOLERANCE = *scalars*	Convergence criterion; default 10^{-5}

Parameter

TERMS = *formula*	Terms to be fitted

RAR1 allows you to fit regression and nonlinear models to data, such as repeated measurements, where the residuals may follow an AR1 or a power-distance correlation model. The CPOSITIONS option specifies the coordinates of the observations in the direction (e.g. time) along which the correlation model operates. You can also use the CGROUPS option to specify a factor to define groups of observations for the model – the correlation model is then defined only over the observations that belong to the same groups. The parameter phi of the AR1 or power-distance model is estimated within RAR1, and is assumed to be the same for every group. (Note that the model will be AR1 if the observations are each one unit apart within each group – the power-distance model is the natural extension of the AR1 model to unequally-spaced data.) You can save the estimated value of phi, in a scalar, using the CPARAMETER option.

Otherwise, RAR1 is used much like FIT (3.1.2). It must be preceded by a MODEL statement (3.1.1). You can also give an RCYCLE statement (3.5.4) first if you want to estimate nonlinear parameters. The MODEL statement must have the WEIGHT option set to a symmetrix matrix, which need not have any values defined. RAR1 will set the values according to the distances (CPOSITIONS), groups (CGROUPS) and estimated parameter phi. These values remain set after RAR1. So you can display or save further output using RDISPLAY (3.1.3), RGRAPH (3.1.6), RCHECK (3.1.7) or RKEEP (3.1.4), in the usual way. You could also, for example, use RAR1 to fit

a full set of regression terms, and then use DROP (3.2.4) to investigate smaller models while still using the phi estimate from the full model. RAR1 has a TERMS parameter to specify the terms to be fitted, like the parameter of FIT. It also has options CALCULATION, CONSTANT, FACTORIAL, POOL, DENOMINATOR, NOMESSAGE, FPROBABILITY, TPROBABILITY, SELECTION and SELINEAR which operate like those of FIT. Note, however, that restrictions are not allowed.

The PRINT option is also similar, except that it has three additional settings:

cparameter	prints the estimated value of the correlation phi, together with a test for phi=0,
cmonitoring	provides monitoring information for the estimation of phi,
cplot	plots the likelihood (or REML likelihood) for phi.

Note, the likelihood values omit some constant terms that depend only on the regression terms. The default is PRINT=model,summary,estimates,cparameter.

The other options control the estimation. The CMETHOD option controls whether phi is estimated for regression models by REML or by maximum likelihood (default maxi); with nonlinear models only maximum likelihood is available. The MAXCYCLE option defines the maximum number of iterations (default 100) used to estimate phi, and the TOLERANCE option specifies the convergence criterion i.e. the accurary to which phi is to be estimated (default 10^{-5}).

Example 8.1.6a analyses the data from Example 7.4, and fits a regression of gas demand on coldness, taking account of the correlations between the observations. Note that the analysis differs from the time-series analysis given earlier, as there is now no Box-Cox transformation, and the AR1 parameter phi is estimated by REML.

Example 8.1.6a

```
 15  CALCULATE nunits = NVALUES(Demand)
 16  VARIATE   [VALUES=1...nunits] Time
 17  SYMMETRIC [ROWS=nunits] wmat
 18  MODEL     [WEIGHTS=wmat] Demand
 19  RAR1      [PRINT=model,summary,estimates,cparameter; CMETHOD=reml;\
 20            CPOSITIONS=Time] Coldness

Regression analysis
===================

 Response variate: Demand
    Weight matrix: wmat based on power-distance correlation model
     Fitted terms: Constant, Coldness


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression       1       85303.      85303.1    147.61
Residual       101       58366.        577.9
Total          102      143669.       1408.5

Percentage variance accounted for 59.0
Standard error of observations is estimated to be 24.0.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
            2        333.3       -3.13
            3        346.0       -2.74
           28        463.4        3.08
           30        443.8        3.23
           35        449.9        3.52
           36        471.8        4.84
           37        458.7        4.02
           38        408.5        3.47
           56        239.3       -3.61
           86        404.0        3.47
           88        391.1        3.09
```

```
          89        411.1        2.96
          90        405.5        3.08
          92        371.7        2.94


Estimates of parameters
-----------------------

Parameter      estimate         s.e.    t(101)
Constant         395.12         7.04     56.11
Coldness         0.9487       0.0781     12.15

* MESSAGE: one has been subtracted from the total degrees of freedom to take
account of the estimation of Phi.

Correlation parameter estimate
------------------------------

Phi: 0.7344
Test for phi non-zero: chi-square 49.100 on 1 d.f., probability <0.001
```

Procedure NLAR1 operates similarly to RAR1 but provides the ability to fit standard curves as well as nonlinear models.

NLAR1 procedure

Fits curves with an AR1 or a power-distance correlation model (R.W. Payne).

Options

PRINT = *string tokens*	What to print (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, cparameter, cmonitoring, cplot); default mode, summ, esti, cpar
CURVE = *string token*	Which standard curve to fit (exponential, dexponential, cexponential, lexponential, logistic, glogistic, gompertz, ldl, qdl, qdq, fourier, dfourier, gaussian, dgaussian); default expo
SENSE = *string token*	Sense of a standard curve (right, left); default righ
ORIGIN = *scalars*	Constrained origin for a standard curve; default * i.e. not constrained
NONLINEAR = *string token*	How to treat nonlinear parameters between groups in standard curves (common, separate); default comm
CALCULATION = *expression structures*	Define a nonlinear model involving explanatory variates and nonlinear parameters; default * implies that a standard curve is fitted
CONSTANT = *string token*	How to treat the constant (estimate, omit); default esti
FACTORIAL = *scalars*	Limit for expansion of model terms; default 3
POOL = *string token*	Whether to pool ss in accumulated summary between all terms fitted in a linear model (yes, no); default no
DENOMINATOR = *string token*	Whether to base ratios in accumulated summary on rms from model with smallest residual ss or smallest residual ms (ss, ms); default ss
NOMESSAGE = *string tokens*	Which warning messages to suppress (dispersion, leverage, residual, aliasing, marginality,

	vertical, df, inflation); default *
FPROBABILITY = *string token*	Printing of probabilities for variance and deviance ratios (yes, no); default no
SELECTION = *string tokens*	Statistics to be displayed in the summary of analysis produced by PRINT=summary (%variance, %ss, adjustedr2, r2, seobservations, dispersion, %cv, %meandeviance, %deviance, aic, bic, sic); default %var, seob
SELINEAR = *string token*	Whether to calculate s.e.s for linear parameters when nonlinear parameters are also estimated (yes, no); default no
WEIGHTS = *variate*	Prior weights for the units
CPARAMETER = *scalars*	Correlation parameter
CPOSITIONS = *variate*	Correlation positions
CGROUPS = *factor*	Groupings of correlation positions
MAXCYCLE = *scalars*	Maximum number of iterations; default 100
TOLERANCE = *scalars*	Convergence criterion; default 10^{-5}

Parameter

TERMS = *formula*	Terms to be fitted

User-defined nonlinear models are defined using the CALCULATION option, as in RAR1. However, NLAR1 also has extra options CURVE, SENSE, ORIGIN and NONLINEAR that are used to specify a standard curve, when CALCULATION is not set. These options operate exactly like the identically-named options of FITCURVE (see 3.7.1), which is used inside NLAR1 to fit the curves. Otherwise, the options and parameter of NLAR1 operate exactly like those of RAR1, except that TERMS must contain no more than one variate and/or factor for a standard curve, and that CPOSITIONS and CGROUPS will use that variate and factor, respectively, as their default if they are unset.

Example 8.1.6b uses NLAR1 to fit an exponential curve. The message that the residuals do not appear to be random is not surprising given their correlation structure (FITCURVE knows only that there is a weight matrix, not how it was derived by NLAR1).

Example 8.1.6b

```
  2  VARIATE   [VALUES=5...30] x
  3  &         [VALUES=1.30,3.55,5.13,6.48,7.85,8.96,9.84,10.91,11.29,11.76,\
  4            12.12,12.55,12.70,13.14,13.47,13.78,14.01,14.11,14.55,14.71,\
  5            14.57,14.30,14.67,14.68,15.03,15.00] y
  6  SYMMETRIC [ROWS=26] wt
  7  MODEL     [WEIGHTS=wt] y
  8  NLAR1     [CURVE=exponential] x

Nonlinear regression analysis
=============================

 Response variate: y
    Weight matrix: wt based on power-distance correlation model
      Explanatory: x
     Fitted Curve: A + B*(R**X)
      Constraints: R < 1
```

```
Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression       2     196.4892     98.24459   2996.62
Residual        22       0.7213      0.03279
Total           24     197.2105      8.21710

Percentage variance accounted for 99.6
Standard error of observations is estimated to be 0.181.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
            8       10.910        2.40
           13       12.700       -2.32
           22       14.300       -2.17

* MESSAGE: the residuals do not appear to be random;
           for example, fitted values in the range 11.793 to 14.177
           are consistently larger than observed values
           and fitted values in the range 7.808 to 11.226
           are consistently smaller than observed values.

* MESSAGE: the following units have high leverage.
          Unit     Response    Leverage
             1        1.300        0.50
             2        3.550        0.28


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
R                  0.85432      0.00288
B                  -30.166        0.594
A                  15.1216       0.0749

* MESSAGE: one has been subtracted from the total degrees of freedom to take
account of the estimation of Phi.

Correlation parameter estimate
------------------------------

Phi: 0.4008
Test for phi non-zero: chi-square 4.313 on 1 d.f., probability 0.038
```

8.2 Survival analysis

Survival data are data in which the response variate is, for example, the lifetime of a component or the survival time of a patient. Typically these are censored, i.e. some individuals survive beyond the end of the study, and so their survival time is unknown. The survivor function F(t) is defined as the probability that an individual is still surviving at time t.

The Kaplan-Meier estimate of the survivor function is simply the proportion surviving out of the number at risk in each time interval. This can be calculated using the `KAPLANMEIER` procedure.

8.2.1 Kaplan-Meier estimation

`KAPLANMEIER` procedure

Calculates the Kaplan-Meier estimate of the survivor function (J.T.N.M. Thissen).

Options

`PRINT` = *string tokens* — What output to print and whether to display the Kaplan-Meier estimate in a graph (`estimate`, `mean`,

	quantiles, summary, graph); default esti, grap
GRAPHICS = *string token*	Type of graphics to use (lineprinter, highresolution); default high
TITLE = *text*	General title for the graph; default *
WINDOW = *scalar*	Window number for the high-resolution graph; default 1
KEYWINDOW = *scalar*	Window number for the key (zero for no key); default 2
SCREEN = *string token*	Whether to clear the screen before plotting or to continue plotting on the old screen (clear, keep); default clea
PROBABILITY = *scalar*	Probability level of the confidence interval for the Kaplan-Meier estimates; default 0.95
XLOWER = *scalar*	Lower bound for x-axis; default 0
XUPPER = *scalar*	Upper bound for x-axis; default * i.e. a value slightly larger than the maximum of the TIME parameter (or EVENT parameter if TIME is not set) is used
PLOT = *string tokens*	What additional plotting features to include (referenceline, censored); default * i.e. none
PERCENTILES = *variate* or *scalar*	Percentiles at which to estimate quantiles of survival times; default 25,50,75

Parameters

TIME = *variates*	Observed timepoints
CENSORED = *variates*	Variate specifying whether the corresponding element of TIME is censored (1) or not (0); default is to assume no censoring
GROUPS = *factors*	Factor specifying the different groups for which the survivor function is estimated
EVENT = *variates*	Saves the distinct TIME values when TIME is set; otherwise supplies an input variate specifying the endpoint of each interval
NDEATH = *variates*	Saves the number of deaths at each EVENT when TIME is set; otherwise supplies an input variate specifying the number of deaths in each interval
NATRISK = *variates*	Saves the number of units at risk at each EVENT when TIME is set; otherwise supplies an input variate with the number at risk in each interval
ESTIMATE = *variates*	Saves the Kaplan-Meier estimates of the survivor function
NEWGROUPS = *factors*	Saves the grouping of the EVENT, NDEATH, NATRISK and ESTIMATE variates when TIME is set

KAPLANMEIER allows for two different types of data. In the first type, illustrated at the start of Example 8.2.1 and Figure 8.2.1a, the timepoints are all accurately observed. The observed timepoints or the timepoints at which censoring took place are then specified using the TIME parameter. The CENSORED parameter allows you to specify a variate containing the values 0 and 1 to indicate whether the corresponding element of TIME is censored (1) or not (0); if there was no censoring, this can be omitted. The GROUPS parameter can be used to specify a factor to indicate different groups whose survivor functions are to be estimated separately. The distinct TIME values can be saved using the EVENT parameter, and the number of deaths and the number of units at risk at each individual EVENT can be saved using parameters NDEATH and NATRISK respectively. The Kaplan-Meier estimate can be saved with the ESTIMATE parameter. The

NEWGROUPS parameter can be used to save a factor indicating the group structure of the output variates.

The second type of data, shown in the second half of Example 8.2.1 and Figure 8.2.1b, is relevant when the units are observed at the end of time-intervals. The exact times are then unknown and the data are defined using parameters EVENT, NDEATH and NATRISK to specify respectively the timepoints, the number of deaths and the number at risk at the end of each interval. The GROUPS parameter can again be used to request separate group estimates.

The PRINT option selects the output to be displayed with settings:

estimate	the events, number of deaths, number of units at risk and the Kaplan-Meier estimate with a confidence interval,
summary	summary of censored and uncensored observations,
quantiles	estimates quantiles of the distribution of survival times (observed timepoints only),
mean	mean and standard error (observed timepoints only),
graph	plots the Kaplan-Meier estimate against the time points.

The default is PRINT=estimates,graph.

The probability level for the Kaplan-Meier estimate confidence interval can be set using the PROBABILITY option; by default this is 0.95. Percentiles for estimating survival times can be set using the PERCENTILES option; by default this is 25,50,75. If PRINT=graph is set, then the PLOT option can be used to include censored observations and a reference line at $S(t)$=0.5 to indicate the median survival time. If GRAPHICS=highresolution different lines are drawn for different groups, whereas GRAPHICS=lineprinter produces separate graphs for the different groups. Lower and upper bounds for the x-axis can be set by options XLOWER and XUPPER, the TITLE option can specify a title for the plots. Options WINDOW and KEYWINDOW control the windows used for high-resolution graphs.

Example 8.2.1

```
  2  " First type of data."
  3  FACTOR [LEVELS=2; VALUES=19(1), 21(2)] Sample
  4  VARIATE [NVALUES=40] Day, Censored
  5  READ Day, Censored

   Identifier   Minimum      Mean   Maximum    Values   Missing
          Day     142.0     227.9     344.0        40         0
     Censored    0.0000    0.1000     1.000        40         0    Skew

 10  XAXIS 1; TITLE='Days'
 11  YAXIS 1; TITLE='Survivor function S'
 12  KAPLANMEIER [TITLE='Data from Table 1.1 in Kalbfleisch and Prentice']\

Kaplan-Meier estimation
=======================

Group 1
-------

Lower and Upper are the boundaries of the 95% confidence interval
```

Time of death	Number of deaths	Number at risk	Kaplan-Meier Lower	Kaplan-Meier estimate	Kaplan-Meier Upper
143.0	1	19	0.681	0.947	0.992
164.0	1	18	0.641	0.895	0.973
188.0	2	17	0.532	0.789	0.915
190.0	1	15	0.479	0.737	0.881
192.0	1	14	0.428	0.684	0.844
206.0	1	13	0.379	0.632	0.804
209.0	1	12	0.332	0.579	0.763
213.0	1	11	0.287	0.526	0.719
216.0	1	10	0.244	0.474	0.673
220.0	1	8	0.196	0.414	0.621
227.0	1	7	0.152	0.355	0.566
230.0	1	6	0.112	0.296	0.509
234.0	1	5	0.076	0.237	0.447
246.0	1	3	0.031	0.158	0.374
265.0	1	2	0.006	0.079	0.288
304.0	1	1	0.000	0.000	0.000

Group 2

Lower and Upper are the boundaries of the 95% confidence interval

Time of death	Number of deaths	Number at risk	Kaplan-Meier Lower	Kaplan-Meier estimate	Kaplan-Meier Upper
142.0	1	21	0.707	0.952	0.993
156.0	1	20	0.670	0.905	0.975
163.0	1	19	0.620	0.857	0.952
198.0	1	18	0.569	0.810	0.924
205.0	1	16	0.514	0.759	0.892
232.0	2	15	0.412	0.658	0.820
233.0	4	13	0.235	0.455	0.652
239.0	1	9	0.196	0.405	0.605
240.0	1	8	0.159	0.354	0.556
261.0	1	7	0.124	0.304	0.506
280.0	2	6	0.063	0.202	0.397
296.0	2	4	0.017	0.101	0.275
323.0	1	2	0.003	0.051	0.207

```
 14  " Second type of data."
 15  VARIATE [VALUES=  1,   2,   3,   4,   5,  6,  7, 8] Year
 16  VARIATE [VALUES=358, 269, 181, 136, 112, 68, 26, 6] Natrisk
 17  VARIATE [VALUES= 89,  88 , 45,  24,   8, 12,  0, 0] Ndeath
 18  KAPLANMEIER EVENT=Year; NDEATH=Ndeath; NATRISK=Natrisk
```

Kaplan-Meier estimation
=======================

Lower and Upper are the boundaries of the 95% confidence interval

Time of death	Number of deaths	Number at risk	Kaplan-Meier Lower	Kaplan-Meier estimate	Kaplan-Meier Upper
1.000	89	358	0.703	0.751	0.793
2.000	88	269	0.453	0.506	0.556
3.000	45	181	0.330	0.380	0.430
4.000	24	136	0.265	0.313	0.361
5.000	8	112	0.244	0.291	0.338
6.000	12	68	0.194	0.239	0.287
7.000	0	26	0.194	0.239	0.287
8.000	0	6	0.194	0.239	0.287

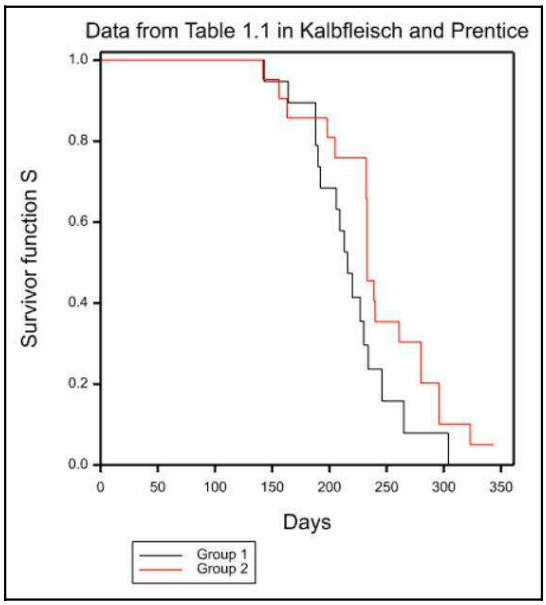


Figure 8.2.1a

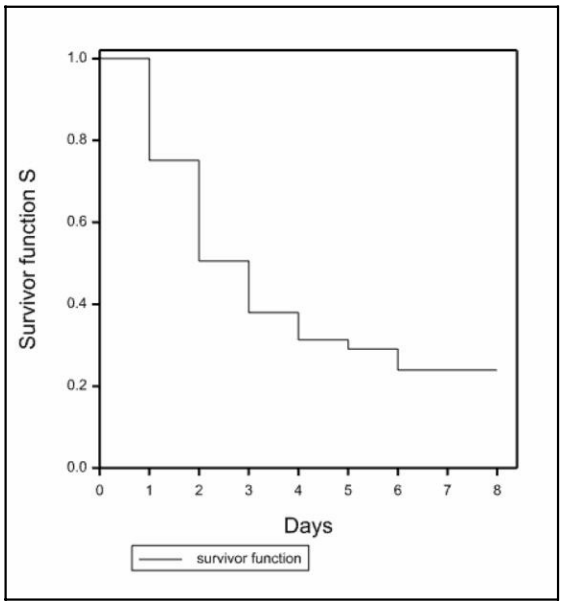


Figure 8.2.1b

8.2.2 Nonparametric tests

RSTEST procedure

Compares groups of right-censored survival data by nonparametric tests (D.A. Murray).

Options

PRINT = *string token*	Controls printed output (test); default test
METHOD = *string tokens*	Types of test required (logrank, breslow, petoprentice, taroneware); default logr, bres, peto, taro
BLOCKS = *factor*	Factor specifying groupings for a stratified test; default * i.e. none

Parameters

TIMES = *variates*	Observed timepoints
CENSORED = *variates*	Variate specifying whether the corresponding element of TIMES is censored (1) or not (0)
GROUPS = *factors*	Factor specifying the different groups
TESTS = *pointers*	Pointer to variates (length 3) to save test statistic, d.f. and probability value for each chosen method

RSTEST calculates nonparametric tests to compare the survival distributions of two or more groups of right-censored survival data. The type of test to be performed is specified by the METHOD option, with settings:

logrank	log-rank test (see Collett 1994, Section 2.5.2);
breslow	Wilcoxon (Breslow) test (see Collett 1994, Section 2.5.3);
petoprentice	Wilcoxon (Peto-Prentice) test (see Collett 1994, Section 11.1.2);
taroneware	Tarone-Ware test (see Collett 1994, Section 11.1.2).

The observed timepoints or the timepoints at which censoring took place are specified using the TIMES parameter. The CENSORED parameter specifies a variate containing the value one if

the corresponding element of TIMES is censored or zero if it was not. CENSORED can be omitted if there was no censoring. The groups to be compared are indicated using the GROUPS parameter. The BLOCKS option can be used to specify a factor to indicate different groupings for a stratified test, for example these might represent different centres or laboratories. If the input variates or factors are restricted, the tests will be based only on the units not excluded by the restriction.

The TESTS parameter allows the statistics to be saved in a pointer to a set of variates (length 3) for each of the chosen methods containing the statistic, its degrees of freedom and probability level. If you are saving the tests you may want to set option PRINT=* to stop them being printed.

Example 8.2.2 illustrates the use of RSTEST, using data from Collett (1994).

Example 8.2.2

```
  2  " Survival times of melanoma patients in two treatment
 -3    groups stratified by age-group (Collett 1994, page 48)."
  4  VARIATE  [NVALUES=30] Day, Censor
  5  READ     Day,Censor

   Identifier   Minimum      Mean   Maximum    Values   Missing
          Day     4.000     15.30     34.00        30         0
       Censor    0.0000    0.6667     1.000        30         0

  9  FACTOR   [LEVELS=3;LABELS=!t('21_40','41_60','61_')] AgeGroup;\
 10           VALUES=!(6(1),3(2),2(3),9(1),7(2),3(3))
 11  FACTOR   [LEVELS=2;LABELS=!t('BCG','c.parvum')] Treat;\
 12           VALUES=!(11(1),19(2))
 13  RSTEST   [BLOCKS=AgeGroup] Day; CENSORED=Censor; GROUPS=Treat

Test Statistics for Equality of Survival Curves for Treat
=========================================================

     Stratified by AgeGroup

                          Statistic  d.f.  probability
               Log-rank      0.688     1        0.407
      Wilcoxon (Breslow)     0.179     1        0.673
             Tarone-Ware     0.405     1        0.524
 Wilcoxon (Peto-Prentice)    0.666     1        0.414
```

8.2.3 Life-table estimates

RLIFETABLE procedure

Calculates the life-table estimate of the survivor function (D.A.Murray).

Options

PRINT = *string tokens*	Controls printed output (lifetable); default life
PLOT = *string tokens*	Type of graph to be plotted (survivor, hazard, pdf); default surv, haza, pdf
INTERVAL = *scalar* or *variate*	A scalar defining the width of the intervals or a variate containing the boundaries of the intervals

Parameters

TIMES = *variates*	Observed timepoints
CENSORED = *variates*	Variate specifying whether the corresponding element of each TIMES variate is censored (1) or represents failures (0)
FREQUENCY = *variates*	Variate containing frequencies for the elements of TIMES; by default these are all assumed to be 1
GROUPS = *factors*	Factor specifying the different groups for which to

	estimate life tables
LIFETABLE = *pointers*	Pointer to variates to save the information from each life table

RLIFETABLE calculates the life-table estimate, or *actuarial* estimate, of the survivor function (see Chapter 4 of Lee 1992). The life-table method requires a fairly large number of observations so that survival times can be grouped into intervals. These are specified using the INTERVALS option. For equal intervals, you can set INTERVALS to a scalar to define their width. Alternatively you can set INTERVALS to a variate containing the lower boundaries of the intervals. The PLOT option can be used to produce plots of the survivor function (survivor), estimated hazard function (hazard) and the probability density function (pdf). You can set the option PRINT=* to suppress printing of the life table; by default PRINT=lifetable.

The observed timepoints (or the timepoints at which censoring took place) are specified using the TIMES parameter. The CENSORED parameter specifies a variate containing the value one if the corresponding element of TIMES is censored or zero if it was not. CENSORED can be omitted if there was no censoring. If there are several observations (all censored or all uncensored) at a time point, you can specify the time point only once and define the number of observations by specifying a variate of counts using the FREQUENCY parameter. This is particularly useful if the contents of the TIMES variate are intended to identify time intervals rather than discrete time points. The GROUPS parameter can be used to request separate life tables for different groups of data. If the input vectors are restricted, the life tables will be based only on the units not excluded by the restriction. The LIFETABLE parameter allows the life table to be saved in a pointer to a set of variates for each of the columns within the table.

Example 8.2.3 and Figures 8.2.3a-c illustrates the use of RSTEST, using data from Lee (1992).

Example 8.2.3

```
  2  " Survival data for 2418 male patients with angina pectoris
 -3    (Lee 1992, page 91)."
  4  VARIATE  [NVALUES=32] Time; VALUES=!((0.5...15.5)2)
  5  &        Censor; VALUES=!(16(0,1))
  6  &        Count; VALUES=!(456,226,152,171,135,125,83,74,51,42,43,34,\
  7           18,9,6,0,0,39,22,23,24,107,133,102,68,64,45,53,33,27,23,30)
  8  RLIFETABLE [INTERVAL=!(0...15)] Time; CENSORED=Censor; FREQUENCY=Count;\
  9           LIFETABLE=Ltab

Life table survival estimates
=============================

                                                  Effective Conditional
 Interval    Midpoint      Events    Censored        size probability
    0.000       0.500         456           0    2418.000       0.189
    1.000       1.500         226          39    1942.500       0.116
    2.000       2.500         152          22    1686.000       0.090
    3.000       3.500         171          23    1511.500       0.113
    4.000       4.500         135          24    1317.000       0.103
    5.000       5.500         125         107    1116.500       0.112
    6.000       6.500          83         133     871.500       0.095
    7.000       7.500          74         102     671.000       0.110
    8.000       8.500          51          68     512.000       0.100
    9.000       9.500          42          64     395.000       0.106
   10.000      10.500          43          45     298.500       0.144
   11.000      11.500          34          53     206.500       0.165
   12.000      12.500          18          33     129.500       0.139
   13.000      13.500           9          27      81.500       0.110
   14.000      14.500           6          23      47.500       0.126
   15.000           *           0          30      15.000       0.000
```

Interval	Midpoint	Survival	Survival s.e.	Hazard	Hazard s.e.
0.000	0.500	1.000	*	0.208	0.010
1.000	1.500	0.811	0.008	0.124	0.008
2.000	2.500	0.717	0.009	0.094	0.008
3.000	3.500	0.652	0.010	0.120	0.009
4.000	4.500	0.579	0.010	0.108	0.009
5.000	5.500	0.519	0.010	0.119	0.011
6.000	6.500	0.461	0.010	0.100	0.011
7.000	7.500	0.417	0.010	0.117	0.014
8.000	8.500	0.371	0.011	0.105	0.015
9.000	9.500	0.334	0.011	0.112	0.017
10.000	10.500	0.299	0.011	0.155	0.024
11.000	11.500	0.256	0.011	0.179	0.031
12.000	12.500	0.214	0.011	0.149	0.035
13.000	13.500	0.184	0.012	0.117	0.039
14.000	14.500	0.164	0.012	0.135	0.055
15.000	*	0.143	0.013	*	*

Interval	Midpoint	pdf	pdf s.e.	Median remaining lifetime	Median remaining lifetime s.e.
0.000	0.500	0.189	0.008	5.331	0.175
1.000	1.500	0.094	0.006	6.250	0.200
2.000	2.500	0.065	0.005	6.343	0.236
3.000	3.500	0.074	0.005	6.226	0.236
4.000	4.500	0.059	0.005	6.219	0.185
5.000	5.500	0.058	0.005	5.908	0.181
6.000	6.500	0.044	0.005	5.596	0.186
7.000	7.500	0.046	0.005	5.167	0.271
8.000	8.500	0.037	0.005	4.942	0.276
9.000	9.500	0.036	0.005	4.826	0.414
10.000	10.500	0.043	0.006	4.689	0.418
11.000	11.500	0.042	0.007	*	*
12.000	12.500	0.030	0.007	*	*
13.000	13.500	0.020	0.007	*	*
14.000	14.500	0.021	0.008	*	*
15.000	*	*	*	*	*

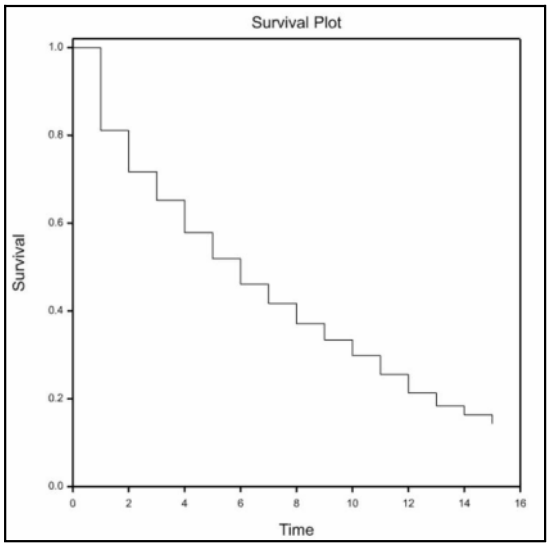


Figure 8.2.3a

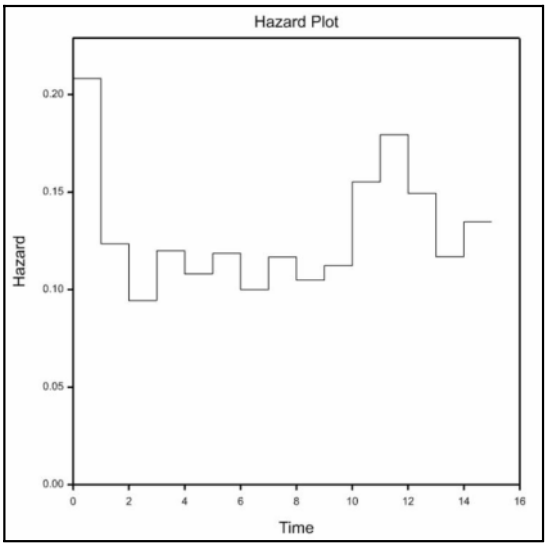


Figure 8.2.3b

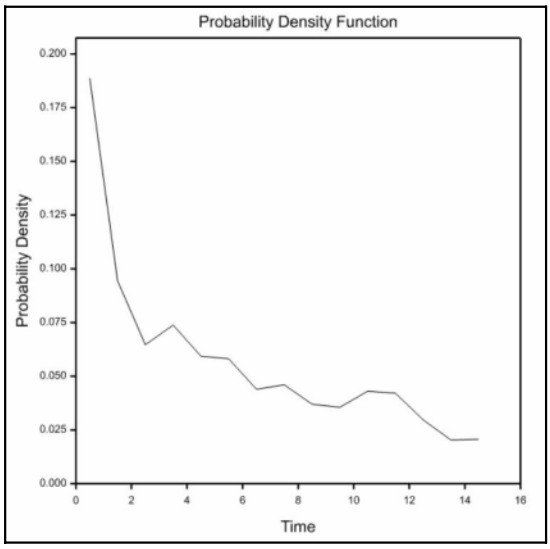


Figure 8.2.3c

8.2.4 Survival distributions

RSURVIVAL procedure

Models survival times of exponential, Weibull, extreme-value, log-logistic or lognormal distributions (R.W. Payne & D.A. Murray).

Options

PRINT = *string tokens*	Controls printed output (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, loglikelihood); default mode, summ, esti
TIMES = *variate*	Time of each observation
DISTRIBUTION = *string token*	Distribution of the survival times (exponential, weibull, extremevalue, loglogistic, lognormal); default expo
CENSORED = *variate*	Indicator for censored observations: 0 if uncensored, 1 if right censored (subject survived the whole trial), −1 if left censored (log-logistic distribution only); default assumes no censored observations
PLOT = *string token*	What to plot (survivorfunction); default *
GRAPHICS = *string token*	Type of graphics (lineprinter, highresolution) default high
ALPHA = *scalar*	Saves the estimated value of the parameter α of the Weibull and extreme-value distributions, if the scalar is input with a non-missing value this provides the initial estimate for α (which will also be the final estimate if MAXCYCLE=1)
_2LOGLIKELIHOOD = *scalar*	Saves −2 multiplied by the log-likelihood
SIGMA = *scalar*	Saves the estimated value of the shape parameter sigma of the log-logistic and lognormal distributions
SURVIVOR = *variate*	Saves estimates of the survivor function
PARAMETERIZATION = *string token*	Controls the parameterization used when saving the survivor function for the Weibull distribution (ph, aft);

	default ph
MAXCYCLE = *scalar*	Maximum number of iterations to use to estimate α; default 20
TOLERANCE = *scalar*	Convergence limit for α; default 10^{-5}
Parameter	
TERMS = *formula*	Defines the model to fit

RSURVIVAL models survival times assuming that they follow either an exponential, Weibull, extreme-value, log-logistic or lognormal distribution, as indicated by the DISTRIBUTION option. It also caters for right-censored observations, where the subject concerned survived the trial: the CENSORED option can be used to specify a variate with an entry for each subject containing one where the subject survived, otherwise zero. The log-logistic caters for left-censored observations, which they can be specified by an entry of −1 in the CENSORED variate. The model to be fitted to the survival times is specified using the TERMS parameter.

The analysis is performed using the Genstat generalized linear models facilities.
For the exponential, Weibull and extreme-value distributions a y-variate (= 1 − CENSORED) is specified indicating whether the subject died or survived, and an offset variate is included which depends on the time variate (see Chapter 6 of Aitkin *et al*. 1989). For the exponential distribution this offset is simply the logarithm of the times. With the Weibull distribution it is the Weibull parameter α multiplied by the logarithm of the times, while for the extreme-value distribution it is the parameter α multiplied by the times. The parameters of the TERMS model and α itself are estimated alternately (with number of cycles controlled by the MAXCYCLE option) until successive estimates are within a tolerance specified by the TOLERANCE option. The ALPHA option can input an initial value for α and save the estimated value. By setting the MAXCYCLE option to one, α can be fixed at the initial value; this is useful for comparing one model with another, when the value of α should be fixed at the value estimated from the more complicated model.

The log-logistic distribution is fitted using a logistic regression model with number of successes $1-c$ and binomial denominator $2-c-b$ (where c is an index for a right-censored observation and b is an index for a left-censored observation) using an offset variate of the logarithm of times divided by σ. The parameters of the TERMS model and σ (shape parameter) are estimated alternately (with number of cycles controlled by the MAXCYCLE option) until successive estimates are within a tolerance specified by the TOLERANCE option.

For the lognormal distribution maximization of the log-likelihood is achieved using an EM algorithm details of which are given in Section 6.19 of Aitkin *et al*. (1989). The SIGMA option can be used to save the estimated value of the shape parameter for both the log-logistic and lognormal distributions. The importance of variables in the lognormal model should be assessed by omitting the variable and comparing −2 times the log-likelihood; this can be saved using the _2LOGLIKELIHOOD option.

The SURVIVOR option allows you to save estimates of the survivor function. For the Weibull distribution the PARAMETERIZATION option can be used to choose whether to produce the estimates for the survivor function using the proportional hazards or accelerated failure time parameterization.

Printed output from the generalized linear model analysis is controlled by the PRINT option with similar settings to those of the FIT directive, except that there is an extra setting loglikelihood to print −2 × the log-likelihood. Further information can be printed subsequently by using RDISPLAY in the usual way. The PLOT option can be set to survivorfunction to produce plots of the empirical survivor function against the value predicted by the model, when the exponential, Weibull and extreme-value distributions are selected (see Aitken *et al*. 1989, pages 275-276). The GRAPHICS option determines the type of

graph, with settings `highresolution` (the default) or `lineprinter`.

Example 8.2.4

```
   2  " Data from Gehan (1965, Biometrika, 52, 203-223)."
   3  VARIATE   [VALUES=1,1,2,2,3,4,4,5,5,8,8,8,8,11,11,12,12,15,17,22,23,\
   4    6,6,6,6,7,9,10,10,11,13,16,17,19,20,22,23,25,32,32,34,35] Time
   5  & [VALUES=24(0),1,0,1,0,1,1,0,0,1,1,1,0,0,1,1,1,1,1] Censor
   6  FACTOR [LABELS=!t(control,'6-mercaptopurine'); VALUES=21(1,2)] Treat
   7  PRINT 'Exponential distribution'

Exponential distribution

   8  RSURVIVAL [TIMES=Time; CENSORED=Censor] Treat

Regression analysis
===================

 Response variate: 1-Censor
     Distribution: Poisson
    Link function: Log
   Offset variate: logtime
     Fitted terms: Constant, Treat


Summary of analysis
-------------------

                                        mean  deviance
Source        d.f.     deviance     deviance     ratio
Regression       1        16.49      16.4852     16.49
Residual        40        38.02       0.9504
Total           41        54.50       1.3293

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.

* MESSAGE: the residuals do not appear to be random;
           for example, fitted values in the range 1.27 to 2.65
           are consistently larger than observed values
           and fitted values in the range 0.12 to 0.15
           are consistently smaller than observed values.

* MESSAGE: the following units have high leverage.
          Unit     Response    Leverage
            20         1.00       0.121
            21         1.00       0.126


Estimates of parameters
-----------------------

                                                 antilog of
Parameter                    estimate         s.e.      t(*)   estimate
Constant                       -2.159        0.218     -9.91     0.1154
Treat 6-mercaptopurine         -1.526        0.396     -3.86     0.2173

* MESSAGE: s.e.s are based on dispersion parameter with value 1.

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
               Treat  control

   9  PRINT 'Weibull distribution'

Weibull distribution

  10  RSURVIVAL [DIST=weibull; TIMES=Time; CENSORED=Censor] Treat
```

```
Regression analysis
===================

 Response variate: 1-Censor
     Distribution: Poisson
    Link function: Log
   Offset variate: alphlogt
     Fitted terms: Constant, Treat


Summary of analysis
-------------------

                                      mean  deviance
Source        d.f.     deviance     deviance     ratio
Regression       2        21.21       10.607     10.61
Residual        39        52.83        1.355
Total           41        74.04        1.806

Dispersion parameter is fixed at 1.00.

* MESSAGE: deviance ratios are based on dispersion parameter with value 1.

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
           21         1.00       0.160


Estimates of parameters
-----------------------

                                                          antilog of
Parameter                   estimate         s.e.      t(*)   estimate
Constant                      -3.071        0.218    -14.07    0.04639
Treat 6-mercaptopurine        -1.731        0.398     -4.35     0.1771

* MESSAGE: s.e.s are based on dispersion parameter with value 1.

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
               Treat  control


Estimated value of alpha
------------------------

alpha = 1.366
```

Full details of the method can be found in Chapter 6 of Aitkin *et al*. (1989). For the exponential distribution (pages 269-270), the survivor function is

$$S(t) = \exp(-\lambda t)$$

with

$$\lambda = \exp(\Sigma(b_i x_i))$$

where b_i are the parameter estimates, x_i are the appropriate values of the explanatory variates, and t is the time. The Weibull distribution (page 280) is defined with density function

$$f(t) = \alpha \lambda t^{**}(\alpha-1) \exp(-\lambda (t^{**}\alpha))$$

and has survivor function

$$S(t) = \exp(-\lambda t^{**}\alpha).$$

The extreme-value distribution (pages 283-284) has survivor function

$$S(t) = \exp(-\lambda \exp(\alpha t)).$$

The loglogistic distribution (pages 295-297) has the survivor function

$$S(t) = 1 / \{ 1 + (t / \theta)^{a} \}$$

with

$$\theta = \exp(\sum(b_i \times x_i))$$

and $a = 1 / \sigma$.

The lognormal distribution (pages 297-300) has survivor function

$$S(t) = 1 - \texttt{CDFNORMAL}(\log((t - \sum(b_i \times x_i)) / \sigma))$$

8.2.5 Proportional hazards model

The data for a proportional hazards model (Cox 1972) consist of a set of subjects observed at one or more times. The final time for each subject is usually at the time of death (or failure). Otherwise, if the subject survives to the end of the trial (or experiment) the observation is said to be *censored*. The model makes the assumption that the subjects have a baseline hazard function which is modified proportionally by the various treatment terms. In Genstat it is assumed that the survival times follow a piecewise exponential distribution (Breslow 1974). This partitions the time axis using a set of discrete cut-points a_i, and assumes a constant baseline hazard γ_i between each one. This corresponds to an exponential distribution with mean $1/\gamma_i$ for the survival times (in the absence of treatments) within each time interval. A cut-point is defined at every time that a death (or failure) occurs and, if the covariates or treatments vary with time, also at every time when the subjects are observed.

To fit a proportional hazards model as a generalized linear model, the variates and factors that make up the treatment terms must be expanded so that, for each subject, there is a unit for every time interval up to the last one during which the subject was observed. If (as usually happens) the subject was not observed at every cutpoint, the covariates and treatments are taken to be constant during the intervals between the times of the observations. The y-variate used within the generalized linear model is an indicator that takes the value 0 if the subject was still surviving within the time interval concerned, otherwise it has the value 1. The model also contains an offset representing the log of the exposure time within each interval.

You can produce the expanded sets of values using procedure `RPHVECTORS`, and then fit models yourself using the standard facilities for generalized linear models (see 3.5). Alternatively, procedures `RPHFIT`, `RPHCHANGE`, `RPHDISPLAY` and `RPHKEEP` will organise this for you automatically. They produce the expanded sets of values, and use them to replace the original values while the model is fitted and displayed. The original values are then reinstated before exit from the procedures, unless a fault has been generated e.g. from the regression directives `FIT` &c. None of the vectors can be restricted (so any restrictions will be cancelled).

`RPHFIT` procedure

Fits a proportional hazards model to survival data as a generalized linear model (R.W. Payne).

Options

`PRINT` = *string tokens*	Controls printed output (`model`, `deviance`, `summary`, `estimates`, `correlations`, `fittedvalues`, `accumulated`, `monitoring`, `loglikelihood`); default `mode`, `summ`, `esti`
`MAXIMALMODEL` = *formula*	Defines the full model to explore (using `RPHCHANGE`); default uses the model defined by the `TERMS` parameter
`SUBJECTS` = *factor*	Subject corresponding to each observation
`TIMES` = *factor* or *variate*	Time of each observation
`CENSORED` = *variate*	Contains the value 1 for censored observations, otherwise 0; if unset it is assumed that there is no censoring
`OFFSET` = *variate*	Offset to include in the model
`POOL` = *string token*	Whether to pool terms in the accumulated summary generated by the fit

Parameter	
TERMS = *formula*	Model to fit

The CENSORED option of RPHFIT provides a variate with an entry for each subject containing one when there is censoring, otherwise zero. If this is not specified, it is assumed that there is no censoring. The SUBJECTS option provides a factor to indicate the subject corresponding to each observation; this can be omitted if there is only one observation per subject. The time at which each observation was made is defined by the TIME option, in either a factor or a variate.

The model to fit is specified by the TERMS parameter. This can be modified later by using procedure RPHCHANGE. However, if you intend to use RPHCHANGE to include additional model terms, you should use option MAXIMALMODEL of RPHFIT to define the largest model that you may want to consider. (This option acts similarly to the TERMS directive in ordinary generalized linear modelling). The OFFSET option allows you to supply an offset to be included in addition to the log of the exposure time within each interval (required to define the proportional hazards model).

The PRINT option controls printed output with similar settings to those of the FIT directive, except that there is an extra setting loglikelihood to print −2 times the log-likelihood (see Example 8.2.5b). The deviance produced for the terms in the regression model can be assessed using chi-square distributions as usual, but the residual deviance is not usable, as the maximal model assumed by the generalized linear models method is inappropriate. So, the residual line is suppressed in the summary and accumulated analysis of deviance (Examples 8.2.5a and 8.2.5b). By default the terms in the model are fitted individually so that they will all have their own lines in an accumulated analysis of deviance. However, you can set option POOL=yes to fit them all at once.

Example 8.2.5a

```
  11  FACTOR      [LEVELS=42; VALUES=1...42] Subject
  12  RPHFIT      [TIMES=Time; SUBJECTS=Subject; CENSORED=Censor] Treat

Regression analysis
===================

 Response variate: _
     Distribution: Poisson
    Link function: Log
  Grouping factor: Interval
     Fitted terms: Treat

Estimates of parameters
-----------------------

                                                          antilog of
Parameter                  estimate         s.e.      t(*)   estimate
Interval 1                   -2.551        0.711     -3.59    0.07799
Interval 2                   -2.470        0.711     -3.47    0.08459
Interval 3                   -3.075        1.000     -3.08    0.04621
Interval 4                   -2.334        0.713     -3.28    0.09689
Interval 5                   -2.232        0.714     -3.13     0.1073
Interval 6                   -1.713        0.588     -2.91     0.1803
Interval 7                    -2.76         1.00     -2.75    0.06346
Interval 8                   -1.357        0.509     -2.67     0.2574
Interval 10                   -2.43         1.01     -2.41    0.08837
Interval 11                  -1.693        0.715     -2.37     0.1839
Interval 12                  -1.465        0.718     -2.04     0.2311
Interval 13                   -1.90         1.01     -1.87     0.1503
Interval 15                   -1.86         1.01     -1.84     0.1555
Interval 16                   -1.69         1.02     -1.67     0.1841
Interval 17                   -1.65         1.01     -1.63     0.1919
Interval 22                  -0.573        0.729     -0.79     0.5638
Interval 23                  -0.151        0.744     -0.20     0.8596
Treat 6-mercaptopurine       -1.509        0.409     -3.69     0.2211
```

```
* MESSAGE: s.e.s are based on dispersion parameter with value 1.

Parameters for factors are differences compared with the reference level:
              Factor  Reference level
               Treat  control


Summary of analysis
-------------------

Source                d.f.     deviance  probability
regression               1      15.2109       <0.001
```

RPHDISPLAY procedure

Prints output for a proportional hazards model fitted by RPHFIT (R.W. Payne).

Option

PRINT = *string tokens*	Controls printed output (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, loglikelihood); default mode, summ, esti

No parameters

You can display further output using procedure RPHDISPLAY. The PRINT option has the same settings as in RPHFIT. Example 8.2.5b prints the accumulated analysis of deviance.

Example 8.2.5b

```
  13  RPHDISPLAY  [accumulated, loglikelihood]

Accumulated analysis of deviance
--------------------------------

Change   d.f.     deviance  probability
+ Treat     1      15.2109       <0.001


Log-likelihood
--------------

-2 x log-likelihood = 172.759
d.f. in fitted model = 1
```

RPHKEEP procedure

Saves information from a proportional hazards model fitted by RPHFIT (R.W. Payne).

Options

RESIDUALS = *variate*	Saves the standardized residuals
FITTEDVALUES = *variate*	Saves the fitted values
ESTIMATES = *variate*	Saves estimates of the parameters
SE = *variate*	Saves standard errors of the estimates
RESPONSE = *variate*	Saves the response variate defined for the generalized linear model
OFFSET = *variate*	Saves the offset variate defined for the generalized linear model
INDEX = *variate*	Index variate used to produce the expanded covariates

	and factors
RISKSET = *factor*	Saves the expanded time factor
_2LOGLIKELIHOOD = *scalar*	Saves −2 × log-likelihood for the fitted model
DFTERMS = *scalar*	Saves the number of d.f. in the model specified by TERMS

No parameters

RPHKEEP allows you to copy information into Genstat data structures from a proportional hazard model that has been fitted by procedure RPHFIT. You do not need to declare the structures in advance; Genstat will declare them automatically to be of the correct type and length.

The RESIDUALS and FITTEDVALUES options save the standardized residuals and the fitted values. The ESTIMATES and SE options save the parameter estimates and their standard errors. The RESPONSE and OFFSET options save the response variate and the offset variate that have been defined for the generalized linear model. The INDEX variate saves the variate of indexes used to construct the expanded x-variates and factors from original variates and factors of the model. The RISKSET option saves a variate indicating the time interval corresponding to each of their units. Finally, the _2LOGLIKELIHOOD option saves −2 times the log-likelihood, and the DFTERMS option saves the number of degrees of freedom in the model specified by TERMS;see Example 8.2.5c.

RPHCHANGE procedure

Modifies a proportional hazards model fitted by RPHFIT (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (model, deviance, summary, estimates, correlations, fittedvalues, accumulated, monitoring, loglikelihood); default mode, summ, esti
METHOD = *string token*	How to change the model (add, drop, switch); default add
POOL = *string token*	Whether to pool terms in the accumulated summary generated by the fit

Parameter

TERMS = *formula*	Model specifying the change

You can use RPHCHANGE to modify the contents of a proportional hazards model that has been fitted by procedure RPHFIT. The change to the model is specified by the TERMS parameter. The setting of the METHOD option specifies how the model is to be changed:

add	adds the terms specified by the TERMS parameter to the fitted model;
drop	drops those terms from the fitted model; and
switch	drops any terms specified by the TERMS parameter that are already in the fitted model, and adds those that are not (i.e. this operates similarly to the SWITCH directive).

The default is METHOD=add. Note, though, that any term that is to be added must have been included in the full model specified by the MAXIMALMODEL option of RPHFIT. The PRINT option controls printed output, and the POOL option controls whether od not each each term will have its own line in an accumulated analysis of deviance, as in RPHFIT.

Example 8.2.5c drops Treat from the model and calculates the change in log-likelihood

(which corresponds to the Change line in the accumulated analysis of deviance in Example 8.2.5b).

Example 8.2.5c

```
  14  RPHKEEP     [_2LOGLIKELIHOOD=llhd1; DFTERMS=df1]
  15  RPHCHANGE   [PRINT=summary,loglikelihood; METHOD=drop] Treat

Summary of analysis
-------------------

Source                d.f.     deviance  probability
regression               0       0.0000            *


Log-likelihood
--------------

-2 x log-likelihood = 187.970
d.f. in fitted model = 0

change in -2 x log-likelihood = 15.211
change in d.f. = -1

  16  RPHKEEP     [_2LOGLIKELIHOOD=llhd2; DFTERMS=df2]
  17  CALCULATE   change = llhd2 - llhd1
  18  &           df = df1 - df2
  19  PRINT       change,df; DECIMALS=3,0

      change          df
      15.211           1
```

8.3 Geostatistics

Geostatistics embodies a suite of techniques for analysing data distributed in a space of one, two or three dimensions and for estimating (predicting, kriging) local values in that space. It is based on the Theory of Regionalized Variables, due largely to Matheron (1965, 1971). Both the theory and the methods were developed for mining, but they are proving just as valuable for estimation and mapping in the earth and environmental sciences generally, especially in two dimensions. The international geostatistics conferences, and especially the European conferences on environmental geostatistics (the geoENV series), demonstrate the scope and development in environmental science (Monestiez *et al*. 2001, Sanchez-Vila 2004). The standard text by Journal & Huijbregts (1978) covers the subject fairly comprehensively in the mining context, while Webster & Oliver (2007) provide sufficient background for the options currently available in Genstat.

In the theory a two-dimensional regionalized variable is regarded as a realization of a random function, Z, with values $Z(\mathbf{x})$ everywhere in the plane, where $\mathbf{x}$ denotes the spatial coordinates $[x_1, x_2]$ or $[x, y]$, depending on convention. In this sense the realization is completely determined, and Z is a mathematical variable. But its complexity is usually such as to defy mathematical description. Add to this that in practice we can never know its values everywhere – we can measure and record it at only a finite number of places – so that the only way forward is to treat data as if they are samples from realizations of random processes. Matheron (1989) discusses the rationale for such an approach.

Geostatistical analysis and estimation require a model. For the current implementation in Genstat the model is

$$Z(\mathbf{x}) = \mu_v + \varepsilon(\mathbf{x}) , \tag{8.3.1}$$

where $Z(\mathbf{x})$ is the value of a random variable, μ_v is the mean of Z in some locality V, and $\varepsilon(\mathbf{x})$ is an autocorrelated random term with a mean of zero and variance defined by

$$\text{var}[\ \varepsilon(\mathbf{x}) - \varepsilon(\mathbf{x}+\mathbf{h})\ ] \ = \ \mathbf{E}[\ \{\varepsilon(\mathbf{x}) - \varepsilon(\mathbf{x}+\mathbf{h})\}^2\ ]\ , \tag{8.3.2}$$

where the vector **h**, the lag, is the spatial separation between **x** and **x** + **h**. The mean is assumed to be locally constant, so that

$$\mathbf{E}[\ Z(\mathbf{x}) - Z(\mathbf{x}+\mathbf{h})\ ] \ = \ 0\ , \tag{8.3.3}$$

and

$$\text{var}[\ Z(\mathbf{x}) - Z(\mathbf{x}+\mathbf{h})\ ] \ = \ \mathbf{E}[\ \{Z(\mathbf{x}) - Z(\mathbf{x}+\mathbf{h})\}^2\ ] \ = \ 2\,\gamma(\mathbf{h}) \tag{8.3.4}$$

depends only on the separation **h** and not on position **x**. The quantity γ is the *semivariance*, and as a function of **h** it is the *variogram*.

These assumptions constitute Matheron's *Intrinsic Hypothesis*, and they are sufficient for very many applications. A somewhat more restrictive assumption is that of second-order stationarity in which the mean of the random process is constant globally, i.e. $\mathbf{E}[\ Z(\mathbf{x})\ ] = \mu$, and the spatial covariance exists and is given by

$$C(\mathbf{h}) \ = \ \mathbf{E}[\ \{Z(\mathbf{x}) - \mu\}\ \{Z(\mathbf{x}+\mathbf{h}) - \mu\}\ ]\ ,$$

with

$$C(\mathbf{0}) \ = \ \text{var}[\ Z(\mathbf{x})\ ] \ = \ \mathbf{E}[\ \{Z(\mathbf{x}) - \mu\}^2\ ]\ . \tag{8.3.5}$$

The spatial covariance is related to the semivariance by

$$\gamma(\mathbf{h}) \ = \ C(\mathbf{0}) - C(\mathbf{h}). \tag{8.3.6}$$

Note that for a variable that is intrinsic in the above sense the semivariance can exist when the covariance does not. This makes the variogram more generally useful than the covariance function for describing spatial variation.

The commonest form of geostatistical estimation is *ordinary kriging*. To estimate the average value of Z in a block B it forms weighted averages of data:

$$\hat{Z}(B) \ = \ \sum_{i=1}^{N} \lambda_i\ z(\mathbf{x}_i)\ , \tag{8.3.7}$$

where λ_i is the weight associated with the ith item of data. The estimation variance is

$$\sigma^2(B) \ = \ 2\sum_{i=1}^{N} \lambda_i \bar{\gamma}(\mathbf{x}_i,\ B) \ - \ \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i\lambda_j\gamma(\mathbf{x}_i,\ \mathbf{x}_j) \ - \ \bar{\gamma}(B,\ B)\ , \tag{8.3.8}$$

where $\gamma(\mathbf{x}_i,\ \mathbf{x}_j)$ is the semivariance of Z between the sampling points $\mathbf{x}_i$ and $\mathbf{x}_j$, $\bar{\gamma}(\mathbf{x}_i,\ B)$ is the average semivariance between the sampling points and the block B being estimated, and $\bar{\gamma}(B,\ B)$ is the within-block variance. The block can be as small as a point, i.e. the same size and shape (support) as that on which the measurements were made, and in that event $\bar{\gamma}(\mathbf{x}_i,\ B)$ reduces to the semivariance between the sampling point $\mathbf{x}_i$ and the estimation point $\mathbf{x}_0$, and the within-block variance, $\bar{\gamma}(B,\ B)$, disappears.

The weights in equation (8.3.7) sum to 1 to avoid bias, and subject to this they are chosen to minimize $\sigma^2(B)$. They must satisfy

$$\sum_{i=1}^{N} \lambda_i\gamma(\mathbf{x}_i,\ \mathbf{x}_j) \ + \ \psi \ = \ \bar{\gamma}\ (\mathbf{x}_j,\ B)$$

and

$$\sum_{i=1}^{N} \lambda_i \ = \ 1 \tag{8.3.9}$$

for all $j = 1, 2, \ldots, N$. The quantity ψ is a Lagrange multiplier introduced for the minimization.

Equations (8.3.9) constitute the kriging system, which may be represented in matrix form by

$$\mathbf{G}\boldsymbol{\lambda} \ = \ \mathbf{b} \tag{8.3.10}$$

where **G** is the augmented matrix, of order $N + 1$, containing the semivariances between sampling points, $\boldsymbol{\lambda}$ is the vector of weights and the Lagrange multiplier, and **b** is the vector containing the average semivariances between the data and the block B. Matrix **G** is inverted and multiplied by **b** to give the weights, which are then inserted into equation (8.3.7) to estimate $Z(B)$. In practice only the nearest few points to B or $\mathbf{x}_0$ carry significant weight, and so **G** can be of order $n + 1$ where $n \ll N$ and typically about 20.

The kriging variance is estimated from

$$\hat{\sigma}^2(B) = \mathbf{b}^T\boldsymbol{\lambda} - \bar{\gamma}(B, B) \; . \tag{8.3.11}$$

Equations (8.3.8) to (8.3.11) contain semivariances. These are obtained from the variogram, for which a mathematical function must therefore be available. The variogram must usually be estimated and computed first.

Thus, starting with a set of data there are three stages in kriging, namely

(1) estimating semivariances at discrete lags to form an ordered set; the sample or experimental variogram,
(2) fitting an allowed model to the experimental variogram, and
(3) the kriging itself.

In many applications the purpose of kriging is to make a map. Values are then kriged at the nodes of a fine grid, through which isarithms, "contours", can then be threaded. Kriging is implemented in Genstat with this in view.

8.3.1 The FVARIOGRAM directive

FVARIOGRAM directive

Forms auto variograms for individual variates or cross-variograms for pairs of variates.

Options

PRINT = *string token*	Controls printed output (statistics); default stat
Y = *variate*	Y positions (needed only for 2-dimensional irregular data)
X = *variate*	X positions or interval (not needed for 2-dimensional regular data i.e. when DATA is a matrix)
YMAX = *scalar*	Maximum lag in the y direction (2-dimensional regular data only)
XMAX = *scalar*	Maximum lag in the x direction
STEPLENGTH = *scalar* or *variate*	Length(s) of the steps in which lag is incremented
METHOD = *string token*	How to estimate the variogram (moments, cressiehawkins, dowd, genton); default mome
DIRECTIONS = *scalar* or *variate*	Directions (degrees) along which to form the variogram (relevant only for 2-dimensional irregular data)
SEGMENTS = *scalar* or *variate*	Angles subtended by the segments (degrees) over which averaging is to be done (relevant only for 2-dimensional irregular data)

Parameters

DATA = *variates* or *matrices*	Measurements as a variate or, for data on a regular grid, as a matrix
VARIOGRAMS = *variates* or *matrices*	Structure to store the sample variogram
COUNTS = *variates* or *matrices*	Numbers of comparisons involved in the calculation of

	each variogram
DISTANCES = *variates* or *matrices*	Mean lag distances at each step
LAGPOINTS = *pointer*	Saves lag classes, indexes to observations and directions to plot in an h-scattergram

The FVARIOGRAM directive forms an experimental variogram from a set of values of a variable, *Z*, distributed in one or two dimensions. By default the variogram is calculated by Matheron's method of moments, as

$$\hat{\gamma}(\mathbf{h}) = \frac{1}{2m(\mathbf{h})} \sum_{i=1}^{m(\mathbf{h})} \{ z(\mathbf{x}_i) - z(\mathbf{x}_i + \mathbf{h}) \}^2 , \tag{8.3.12}$$

where $z(\mathbf{x}_i)$ and $z(\mathbf{x}_i + \mathbf{h})$ are the values at positions $\mathbf{x}_i + \mathbf{h}$, and $m(\mathbf{h})$ is the number of paired comparisons contributing to the estimate. For data on a regular grid or transect $\mathbf{h}$ is an integer multiple of the sampling interval. For irregularly scattered data $\mathbf{h}$ is discretized so that for each nominal lag there is a range of distance equal to the increment and an angular range set by the user. The nominal lag is at the centre of both ranges. However, you can set the METHOD option to calculate robust estimates instead. The cressiehawkins setting uses the estimator of Cressie & Hawkins (1980), which aims to damp the effect of outliers from the secondary process:

$$\hat{\gamma}(\mathbf{h}) = \frac{0.5 \times \{ \frac{1}{m(\mathbf{h})} \sum_{i=1}^{m(\mathbf{h})} | z(\mathbf{x}_i) - z(\mathbf{x}_i + \mathbf{h}) |^{1/2} \}^4}{0.457 + \frac{0.494}{m(\mathbf{h})} + \frac{0.045}{m^2(\mathbf{h})}} \tag{8.3.13}$$

The dowd setting gives Dowd's (1984) estimator, which estimates the variogram for a dominant intrinsic process in the presence of outliers:

$$\hat{\gamma}(\mathbf{h}) = 2.198 \times (\text{median} \{ | z(\mathbf{x}_i) - z(\mathbf{x}_i + \mathbf{h}) |, i = 1, 2 \ldots m(\mathbf{h}) \}) \tag{8.3.14}$$

Similarly the genton setting gives Genton (1978) method:

$$\hat{\gamma}(\mathbf{h}) = 0.5 \times [2.219 \times (\text{order}_k \{ | z(\mathbf{x}_i) - z(\mathbf{x}_i + \mathbf{h}) - z(\mathbf{x}_j) + z(\mathbf{x}_j + \mathbf{h}) | \})]^2 \tag{8.3.15}$$

where order_k denotes the kth order statistic, and k is the number of distinct pairs that can selected from a number of objects equal to the integer part of $1 + m(\mathbf{h})/2$. For further details see Webster & Oliver (2007) pages 67-68 and 115-116.

The data are specified using the DATA parameter. If they are on a regular grid, they should be supplied in a matrix defined with a variate of column labels to provide the x-values and a variate of row labels to provide the y-values. Alternatively, if they are irregularly scattered, then they should be supplied in a variate, and the X and Y options should be set to variates to supply their spatial coordinates.

The experimental variogram is controlled by five options. For irregular data the maximum distance to which the variogram is calculated is set by the XMAX option for all directions. For regular data XMAX defines the maximum lag distance in the X direction, and YMAX must also be given to limit the distance in the Y direction. The increments in distance are set by the STEPLENGTH option, where you can supply a scalar to define equally-spaced steps or a variate to specify the steps themselves. The variogram may be computed in one or more directions. These are given by the DIRECTIONS option in degrees counterclockwise from east in the usual convention. Each direction is at the centre of an angular range, which is defined by the

SEGMENTS option. DIRECTIONS and SEGMENTS should be set to scalars if the variogram is to be calculated for only one direction, or to variates if there are to be several.

A variogram can be computed without regard to direction by setting DIRECTIONS to 0 and SEGMENTS to 180. This is advisable if variation seems to be isotropic, i.e. the same in all directions, or if there are too few data to compute $\hat{\gamma}(\mathbf{h})$ for two or more directions separately. The lag then becomes a scalar $|\mathbf{h}| = h$ in distance only. Experience suggests that some 300 data are needed to distinguish anisotropy.

By default some statistics are printed concerning the variogram, but these can be supressed by setting option PRINT=*. Other information can be saved using the various parameters, in variates if there is a single direction, or in matrices with one column for each direction if there are several: VARIOGRAMS stores the ordered set of semivariances; DISTANCES stores the mean lag distances at which the semivariances have been computed; and COUNTS stores the numbers of paired comparisons from which the semivariances have been computed.

The LAGPOINTS parameter allows you to save a pointer containing lag classes, indexes to observations and directions that can be used to plot an h-scattergram.

Example 8.3.1 forms the variogram for measurements of potassium taken on an incomplete grid at Brooms Barn Experimental Station in Suffolk. The plot from the DGRAPH statement in line 21 is shown in Figure 8.3.1.

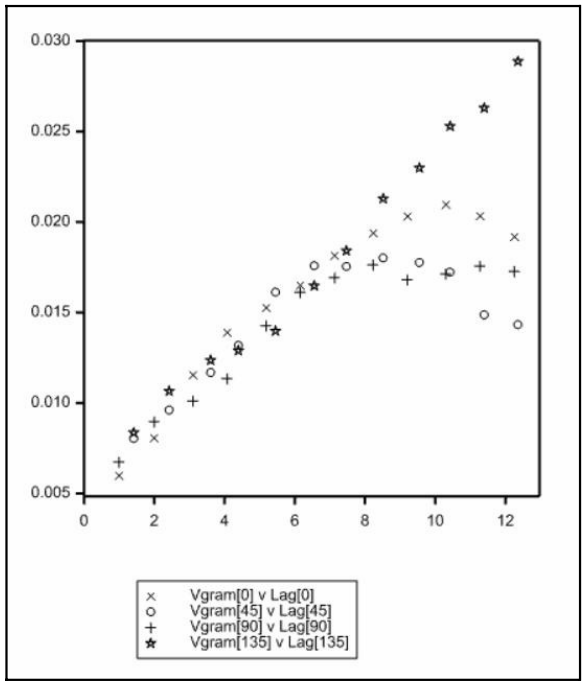


Figure 8.3.1

Example 8.3.1

```
  2  " Data are levels of potassium at Brooms Barn Experimental Station
 -3    (see Webster, R. & Oliver, M.A. 1990, Statistical Methods in Soil
 -4    and Land Resource Survey, Oxford University Press, pages 267-269)."
  5  FILEREAD   [NAME='Broomesb.dat'; PRINT=summary] East,North,K

Summary
-------

The file Broomesb.dat is assumed to contain 3 structure(s), with one value for
each structure on each record.


The file contains 435 values for each of the following structures:
```

```
  Identifier      Type   Missing
        East   variate         0
       North   variate         0
           K   variate         1

   6  CALCULATE  LogK = LOG10(K)
   7  VARIATE    [VALUES=0,45,90,135] Angles
   8  &          [VALUES=45,45,45,45] Segments
   9  FVARIOGRAM [PRINT=statistics; Y=North; X=East; STEP=1; XMAX=13; \
  10             DIRECTIONS=Angles; SEGMENTS=Segments] \
  11             LogK; VARIOGRAM=LogKvar; COUNTS=Kcounts; DISTANCES=Midpoints

Variogram of LogK
=================

General mean:       1.398
General variance:   0.0180


Based on 434 observations
Maximum lag 13

  12  VARIATE    Vgram[#Angles],Lag[#Angles],Count[#Angles]
  13  CALCULATE  Vgram[] = LogKvar$[*; 1...4]
  14  &          Lag[]   = Midpoints$[*; 1...4]
  15  &          Count[] = Kcounts$[*; 1...4]
  16  PRINT      Lag[0],Vgram[0],Count[0],Lag[45],Vgram[45],Count[45]

     Lag[0]    Vgram[0]    Count[0]     Lag[45]   Vgram[45]   Count[45]
          *           *         0.0           *           *         0.0
      1.000     0.00599       396.0       1.414     0.00805       374.0
      2.000     0.00806       362.0       2.425     0.00961      1024.0
      3.107     0.01155       971.0       3.606     0.01169       612.0
      4.081     0.01390       890.0       4.395     0.01319       859.0
      5.190     0.01526      1336.0       5.452     0.01613      1294.0
      6.160     0.01651      1227.0       6.555     0.01759       939.0
      7.138     0.01815      1106.0       7.469     0.01755      1489.0
      8.237     0.01939      1340.0       8.515     0.01802      1307.0
      9.212     0.02031      1157.0       9.551     0.01776      1144.0
     10.310     0.02096      1239.0      10.422     0.01723      1150.0
     11.282     0.02033      1060.0      11.395     0.01487       896.0
     12.259     0.01918       879.0      12.353     0.01433      1238.0

  17  &          Lag[90],Vgram[90],Count[90],Lag[135],Vgram[135],Count[135]

    Lag[90]   Vgram[90]   Count[90]    Lag[135]  Vgram[135]  Count[135]
          *           *           0           *           *           0
      1.000     0.00674         399       1.414     0.00836         376
      2.000     0.00897         375       2.426     0.01065        1032
      3.106     0.01011        1014       3.606     0.01236         620
      4.081     0.01135         968       4.396     0.01289         875
      5.187     0.01427        1490       5.454     0.01398        1336
      6.157     0.01612        1407       6.555     0.01648         989
      7.136     0.01692        1323       7.472     0.01841        1606
      8.232     0.01764        1684       8.519     0.02129        1461
      9.208     0.01681        1599       9.550     0.02299        1326
     10.304     0.01714        1929      10.425     0.02530        1381
     11.278     0.01757        1875      11.395     0.02630        1087
     12.255     0.01728        1808      12.356     0.02888        1522

  18  XAXIS      1; LOWER=0
  19  YAXIS      2; LOWER=0
  20  PEN        1...4; COLOUR='black'; SYMBOL=1...4
  21  DGRAPH     Vgram[]; Lag[]; PEN=1...4
```

8.3.2 The MVARIOGRAM procedure

MVARIOGRAM procedure

Fits models to an experimental variogram (S.A. Harding D.A. Murray & R. Webster).

Options

PRINT = *string tokens*	Controls printed output from the fit (model, summary, estimates, correlations, fittedvalues, monitoring); default mode, summ, esti
MODELTYPE = *string token*	Defines which model to fit (power, boundedlinear, circular, spherical, doublespherical, pentaspherical, exponential, besselk1, gaussian, affinepower, linear, cubic, stable, cardinalsine, matern); default powe
WEIGHTING = *string token*	Method to be used for weighting (counts, cbyvar, equal); default coun
CONSTANT = *string token*	How to treat the constant (estimate, omit); default esti
SMOOTHNESS = *scalar*	Value of power parameter for the stable model, or ν parameter for the Matern model; default * i.e. estimate
ISOTROPY = *string token*	Defines whether to fit an isotropic or geometrical anisotropic model (isotropic, geometrical); default isot
WINDOW = *scalar*	Window in which to plot a graph; default 0 i.e. no graph
TITLE = *text*	Title for the graph
XUPPER = *scalar*	Upper limit for the *x*-axis in the graph
PENDATA = *scalar*	Pen to be used to plot the data; default 1
PENMODEL = *scalar*	Pen to be used to plot the model; default 2

Parameters

VARIOGRAM = *variates* or *matrices*	Experimental variogram to which the model is to be fitted, as a variate if in only one direction or as a matrix if there are several
COUNTS = *variates* or *matrices*	Counts for the points in each variogram (not required if WEIGHTING=equal)
DISTANCE = *variates* or *matrices*	Mean lag distances for the points in each variogram
DIRECTION = *variates*	Directions in which each variogram was computed
INITIAL = *scalars* or *variates*	Scalar defining initial distance parameter for an isotropic model, or variate with two values for a double-spherical isotropic model, or a variate with three values for a geometrical anisotropic model
ESTIMATES = *variates*	Estimated parameter values
FITTEDVALUES = *variates*	Fitted values
EXIT = *scalars*	Exit status from the nonlinear fitting
SAVE = *pointers*	Saves the model name and estimates in a pointer that can be used in KRIGE

Procedure MVARIOGRAM uses the directives FIT, FITCURVE and FITNONLINEAR to fit various models to the experimental variogram. Models must be authorized in the sense that they cannot give rise to negative variances when data are combined. Technically they are conditionally negative semi-definite (CNSD); see Webster & Oliver (1990, 2007) or Journel & Huijbregts

(1978) for an explanation.

The MODELTYPE option can be set to select the following bounded isotropic models with finite ranges – these all take the value $c + c_0$ for $h \geq a$, and the following values for $h < a$

boundedlinear	$c_0 + ch/a$
circular	$c_0 + c\ \{1 - (2/\pi)\arccos(h/a) + (2h/(\pi a))\sqrt{(1-h^2/a^2)}\}$
spherical	$c_0 + c\ \{1.5h/a - 0.5(h/a)^3\}$
doublespherical	$c_0 + c_1\ \{1.5h/a_1 - 0.5(h/a_1)^3\} + c_2\ \{1.5h/a_2 - 0.5(h/a_2)^3\}$ for $h \leq a_1$
	$c_0 + c_1 + c_2\ \{1.5h/a_2 - 0.5(h/a_2)^3\}$ for $a_1 < h < a_2$
	where $c = c_1 + c_2$
pentaspherical	$c_0 + c\ \{1.875h/a - 1.25(h/a)^3 + 0.375(h/a)^5\}$
cubic	$c_0 + c\ \{7(h/a)^2 - 8.75(h/a)^3 + 3.5(h/a)^5 - 0.75(h/a)^7\}$

There are also bounded asymptotic models

besselk1	$c_0 + c\ \{1 - h/a\ K_1(h/a)\}$ (Whittle's elementary correlation, Whittle 1954)
exponential	$c_0 + c\ \{1 - \exp(-h/a)\}$
gaussian	$c_0 + c\ \{1 - \exp(-h^2/a^2)\}$
stable	$c_0 + c\ \{1 - \exp(-(h/a)^b))\}$
matern	$c_0 + c\ \{1 - 1 / (2^{(\nu-1)}\ \Gamma(\nu))\ (h/a)^\nu\ K_\nu(h/a)\}$

unbounded models

power	$c_0 + g\ h^\alpha$ (power function with exponent α strictly between 0 and 2)
linear	$c_0 + c\ h$ which is a special case of the power function with exponent 1

and hole effect models

cardinalsine	$c_0 + c \times (1 - a/h \times \sin(h/a))$.

Geometrically anisotropic models, i.e. ones that might be made isotropic by a simple linear transformation of the spatial coordinates, can be fitted by setting option ISOTROPY=geometrical. The following transformation is used:

$$omega(\theta) = \sqrt{\{\ a^2\cos^2(\theta-\varphi) + b^2\sin^2(\theta-\varphi)\ \}}$$

where θ represents the direction (specified by the DIRECTION parameter) converted from degrees to radians. So, for example, a geometrical anisotropic power model would be

$$c_0 + (\ \sqrt{\{\ a^2\cos^2(\theta-\varphi) + b^2\sin^2(\theta-\varphi)\ \}}\ h\ )^{power}$$

(Note: this particular model can also be defined by setting MODELTYPE=affinepower; the ISOTROPY option is then ignored.)

In all these models, the intercept term (or *nugget variance*) c_0 can be omitted by setting the CONSTANT option to omit; the default is estimate.

For the stable model (or powered exponential model; see Webster & Oliver 2007) the SMOOTHNESS option controls the power parameter for the model. For the matern model it specifies the ν parameter. By default, the parameter is estimated. However, you can supply a value, to fix the parameter for the model fitting.

The data for the procedure can be taken directly from the FVARIOGRAM directive, with parameters DISTANCES, VARIOGRAMS and COUNTS corresponding to those with the same names

in FVARIOGRAM. The data will be in variates if the variogram was calculated in only one direction. If it is in several, they can either be in matrices (as generated by FVARIOGRAM) or in variates. For MODELTYPE=affinepower directions must be supplied, using the DIRECTIONS parameter. These should be in a variate with one value for each column if the other data are in matrices; alternatively, they should be in a variate of the same length as the other variates.

The WEIGHTING option controls the weights that are used when fitting the model. The default setting counts uses the values supplied by the COUNTS parameter, cbyvar uses the COUNTS divided by the values in VARIOGRAM, and equal uses equal weights (of one).

The procedure generates rough starting values for the parameters before calling FITNONLINEAR to convergence. If the solution does not converge there are two likely reasons. The model may be unsuited for the particular experimental variogram. For example, a bounded model is specified when the variogram is clearly unbounded, or *vice versa*. You should choose only models that have approximately the right shape. Alternatively, the starting values may be too far from a sensible solution. You should then supply initial values using the INITIAL parameter. For a double-spherical isotropic model, INITIAL must be set to a variate with two values representing the two distance parameters. For the other isotropic models it should be set to a scalar defining the initial distance parameter. Finally, for a geometrical anisotropic model, it should be set to a variate with three values, defining the initial values for φ, the maximum distance parameter and the minimum distance parameter.

Printed output is controlled by the PRINT option, and includes all the usual settings as in FIT, FITCURVE or FITNONLINEAR. You can also produce a high-resolution graph of the data and the fitted model, by setting the WINDOW option to the number of a suitable window. By default WINDOW is zero, and no graph is produced. The TITLE option can supply a title for the plot. Option XUPPER can define an upper value for the x-axis (i.e. distance), and PENDATA and PENMODEL can supply the numbers of the pens to be used to plot the experimental variogram and the fitted model respectively (by default 1 and 2). Alternatively, you can use the ESTIMATES parameter to save the parameter estimates, and plot the variogram and model later with the DVARIOGRAM procedure (8.3.3).

Example 8.3.2 continues the study of potassium concentrations in the soil at Brooms Barn, and fits and plots linear, spherical and exponential models (Figures 8.3.2a-c). Notice that CALCULATE is used at line 23 to set the counts to zero for the data at distances greater than 11.75 which, from the graph in Figure 8.3.1, would seem to be rather less reliable.

Example 8.3.2

```
 22  " Model the variogram."
 23  CALCULATE Kcounts=Kcounts*(Midpoints<11.75)
 24  FOR Mod='LINEAR','SPHERICAL','EXPONENTIAL'
 25    MVARIOGRAM [MODELTYPE=#Mod; PRINT=model,summary,estimates; \
 26                WEIGHTING=counts] LogKvar; COUNTS=Kcounts; \
 27                DISTANCES=Midpoints; ESTIMATES=est
 28    DVARIOGRAM [MODELTYPE=#Mod; TITLE=Mod] LogKvar; DISTANCES=Midpoints; \
 29                XUPPER=15; ESTIMATES=est
 30  ENDFOR

Variogram model: linear
=======================

y = c0 + c*x

Regression analysis
===================

 Response variate: y
   Weight variate: rwt
     Fitted terms: Constant, x
```

```
Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression       1       0.5758     0.575844     99.55
Residual        42       0.2429     0.005784
Total           43       0.8188     0.019042

Percentage variance accounted for 69.6
Standard error of observations is estimated to be 0.0761.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           44       0.0253        2.45
           46       0.0149       -2.74
           47       0.0176       -2.37

* MESSAGE: the error variance does not appear to be constant;
           large responses are more variable than small responses.

* MESSAGE: the following units have high leverage.
         Unit     Response    Leverage
           47       0.0176       0.119


Estimates of parameters
-----------------------

Parameter      estimate         s.e.     t(42)
Constant       0.007944     0.000925      8.59
x              0.001200     0.000120      9.98


Variogram model: spherical
==========================

y = c0 + c*(1.5*x/a-0.5*(x/a)**3)  for x.lt.a
y = c0 + c                         for x.ge.a


Nonlinear regression analysis
=============================

 Response variate: y
   Weight variate: rwt
Nonlinear parameters: a
  Model calculations: spherical


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression       2       0.6218     0.310915     64.72
Residual        41       0.1970     0.004804
Total           43       0.8188     0.019042

Percentage variance accounted for 74.8
Standard error of observations is estimated to be 0.0693.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           44       0.0253        2.92
           48       0.0263        3.05

* MESSAGE: the error variance does not appear to be constant;
           large responses are more variable than small responses.
```

```
Estimates of parameters
-----------------------

Parameter         estimate         s.e.
a                    10.81         1.19
* Linear
c                   0.01528      0.00139
Constant            0.00460      0.00142


Variogram model: exponential
============================

y = c0 + c*(1-EXP(-x/a))

Nonlinear regression analysis
=============================

 Response variate: y
   Weight variate: rwt
Nonlinear parameters: a
  Model calculations: negex1


Summary of analysis
-------------------

Source        d.f.         s.s.         m.s.      v.r.
Regression       2       0.6162     0.308104     62.36
Residual        41       0.2026     0.004941
Total           43       0.8188     0.019042

Percentage variance accounted for 74.1
Standard error of observations is estimated to be 0.0703.

* MESSAGE: the following units have large standardized residuals.
         Unit     Response    Residual
           44       0.0253        2.85
           46       0.0149       -2.37
           48       0.0263        2.75

* MESSAGE: the error variance does not appear to be constant;
           large responses are more variable than small responses.


Estimates of parameters
-----------------------

Parameter         estimate         s.e.
a                    5.82         2.16
* Linear
c                   0.02054      0.00191
Constant            0.00280      0.00249
```

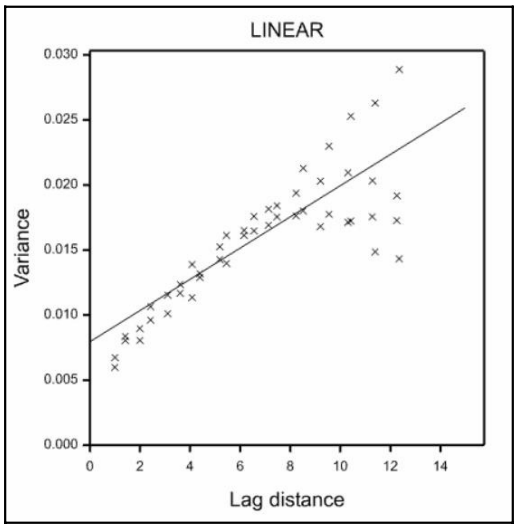


Figure 8.3.2a

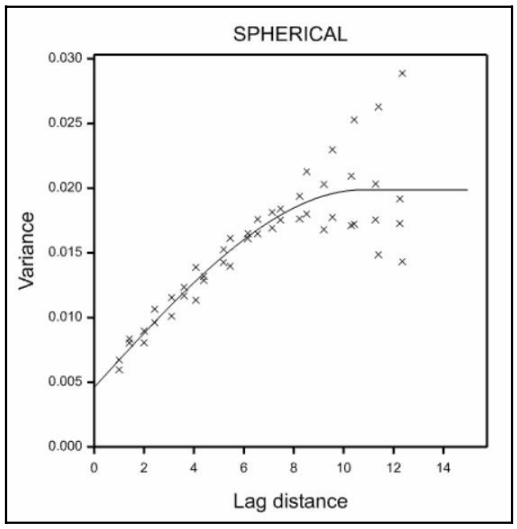


Figure 8.3.2b

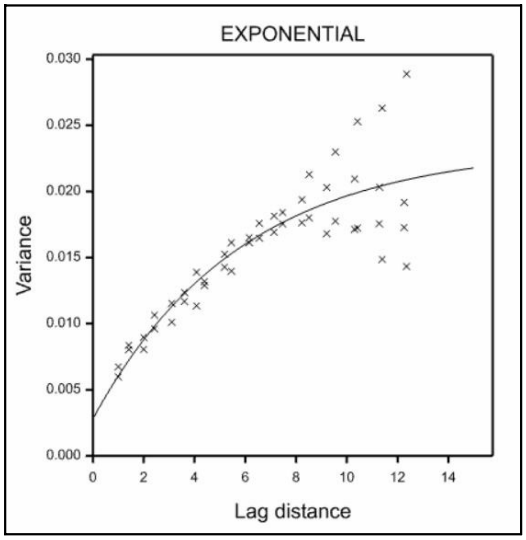


Figure 8.3.2c

From an examination of the graphs and the % variance accounted for, the spherical model seems to describe the variogram best. An alternative, but more time-consuming, method of assessing the models would be to use the `KCROSSVALIDATION` procedure. This uses the variograms for kriging, and sees how well the kriging predicts the true values. The observed value of *z* at each sampling point in the data is omitted in turn from the whole set and predicted from the others. The predictions are compared with the true values to give a mean deviation or error, and the kriging variances are compared with the squared deviations to give a mean squared deviation ratio. This process is known as "cross-validation".

The `SAVE` parameter of `MVARIOGRAM` saves the parameter estimates and associated information required by the `KRIGE` directive. Alternatively, the `ESTIMATES` parameter saves just the estimates themselves, which can be used by `DVARIOGRAM` to plot the fitted model. The `FITTEDVALUES` parameter saves the fitted values, and the `EXIT` parameter saves the exit "status code" from `FIT`, `FITCURVE` or `FITNONLINEAR` (a zero value indicates success; see 3.7.4.).

8.3.3 The DVARIOGRAM procedure

DVARIOGRAM procedure

Plots fitted models to an experimental variogram (S.A. Harding, D.A. Murray & R. Webster).

Options

MODELTYPE = *string token*	Defines which model to plot (power, boundedlinear, circular, spherical, doublespherical, pentaspherical, exponential, besselk1, gaussian, affinepower, linear, cubic, stable, cardinalsine, matern); default powe
ISOTROPY = *string token*	Defines whether this is an isotropic or geometrical anisotropic model (isotropic, geometrical); default isot
WINDOW = *scalar*	Window in which to plot a graph; default 1
TITLE = *text*	Title for the graph

Parameters

VARIOGRAM = *variates*	Experimental variogram to which the model or matrices has been fitted, as a variate if in only one direction or as a matrix if there are several
DISTANCE = *variates*	Mean lag distances for the points in each or matrices variogram
DIRECTION = *variates*	Directions in which each variogram was computed
ESTIMATES = *variates*	Estimated parameter values
XUPPER = *scalar*	Upper limit for the x-axis in the graph
PENDATA = *scalar*	Pen to be used to plot the data; default 1
PENMODEL = *scalar*	Pen to be used to plot the model; default 2

DVARIOGRAM plots fitted models to an experimental variogram using estimates produced by MVARIOGRAM.

The data can be taken directly from the FVARIOGRAM directive and MVARIOGRAM procedure. The parameters DISTANCES and VARIOGRAMS correspond to those with the same names in FVARIOGRAM. The data will be in variates if the variogram was calculated in only one direction. If it is in several, they can either be in matrices (as generated by FVARIOGRAM) or in variates. For the affinepower model, directions must be supplied using the DIRECTIONS parameter. These should be in a variate with one value for each column if the other data are in matrices; alternatively, they should be in a variate of the same length as the other variates.

The MODELTYPE and ISOTROPY options specify the fitted model that is to be plotted, exactly as in the MVARIOGRAM procedure (8.3.2). The estimates for the model parameters are supplied in a variate using the ESTIMATES parameter. These can be taken directly from MVARIOGRAM using the ESTIMATES parameter. The number of values within the variate for the estimates will depend on the model that has been fitted (see 8.3.2).

The placement of the graph within the graphical frame can be controlled using the WINDOW option. The TITLE option can supply a title for the plot. Option XUPPER can define an upper value for the x-axis (i.e. distance), and PENDATA and PENMODEL can supply the numbers of the pens to be used to plot the experimental variogram and the fitted model respectively (by default 1 and 2).

The use of DVARIOGRAM was illustrated in lines 27 and 28 of Example 8.3.2.

8.3.4 The KRIGE directive

KRIGE directive

Calculates kriged estimates using a model fitted to the sample variogram.

Options

PRINT = *string token*	Controls printed output (description, search, weights, monitor, data); default desc
Y = *variate*	Y positions (not needed for 2-dimensional regular data i.e. when DATA is a matrix)
X = *variate*	X positions (needed only for 2-dimensional irregular data)
YOUTER = *variate*	Variate containing 2 values to define the Y-bounds of the region to be examined (bottom then top); by default the whole region is used
XOUTER = *variate*	Variate containing 2 values to define the X-bounds of the region to be examined (left then right); by default the whole region is used
YINNER = *variate*	Variate containing 2 values to define the Y-bounds of the interpolated region (bottom then top); no default
XINNER = *variate*	Variate containing 2 values to define the X-bounds of the interpolated region (left then right); no default
BLOCK = *variate*	Dimensions (length and height) of block; default !(0, 0) i.e. punctual kriging
RADIUS = *scalar*	Maximum distance between target point in block and usable data
SEARCH = *string token*	Type of search (isotropic, anisotropic); default isot
MINPOINTS = *scalar*	Minimum number of data points from which to compute elements; default 7
MAXPOINTS = *scalar*	Maximum number of data points from which to compute elements (2 < MINPOINTS ≤ MAXPOINTS < 41); default 20
NSTEP = *scalar*	Number of steps for numerical integration; (3 < NSTEP < 11); default 8
DRIFT = *string token*	Amount of drift (constant, linear, quadratic); default cons
YXRATIO = *scalar*	Ratio of Y interval to X interval; default 1.0
INTERVAL = *scalar*	Distance between successive interpolations; default 1.0

Parameters

DATA = *variates* or *matrices*	Observed measurements as a variate or, for data on a regular grid, as a matrix
ISOTROPY = *string tokens*	Form of variogram (isotropic, Burgess, geometrical); default isot
MODELTYPE = *string tokens*	Model fitted to the variogram (power, boundedlinear, circular, spherical, doublespherical, pentaspherical, exponential, besselk1, gaussian, cubic, stable, cardinalsine, matern); default powe
NUGGET = *scalars*	The nugget variance

SILLVARIANCES = *variates*	Sill variances of the spatially dependent component; default none
RANGES = *variates*	Ranges of the spatially dependent component; default none
GRADIENT = *variates*	Slope of the unbounded component; default none
EXPONENT = *variates*	Power of the unbounded component or power for the stable model; default none
SMOOTHNESS = *scalar*	Value of ν parameter for the Matern model; defalt none
PHI = *variates*	Phi parameters of an anistropic model (ISOTROPY = Burg or geom)
RMAX = *variates*	Maximum gradient or distance parameter of an anistropic model
RMIN = *variates*	Minimum gradient or distance parameter of an anistropic model
PREDICTIONS = *matrices*	Kriged estimates
VARIANCES = *matrices*	Estimation variances
LAGRANGEMULTIPLIER = *matrices* or *pointers*	Saves the Lagrange multipliers from each kriging solution
MEASUREMENTERROR = *scalar*	Specifies the variance of the measurement error
SAVE = *pointers*	Supplies the model name and estimates, as saved from MVARIOGRAM

The KRIGE directive computes the ordinary kriging estimates of a variable at positions on a grid from data and a model variogram by solving the kriging system, equations (8.3.9) above.

The data must be supplied, using the DATA parameter, in one of the two forms as for FVARIOGRAM: i.e. for data on a regular grid, in a matrix defined with a variate of column labels to provide the x-values and a variate of row labels to provide the y-values or, for irregularly scattered data, as a variate with the X and Y options set to variates to supply the spatial coordinates.

By default all data are considered when forming the kriging system. However, a subset of the data may be selected by limiting the area to a rectangle defined by XOUTER and YOUTER options. Each of these should be set to a variate with two values to define lower and upper limits in the x (East-West) and y (North-South) directions respectively.

The positions at which Z is predicted (estimated) are contained in a rectangle defined by the XINNER, YINNER and INTERVAL options. XINNER and YINNER are set to variates similarly to XOUTER and YOUTER, and their limits should not lie outside those of XOUTER and YOUTER. INTERVAL is set to a scalar to define the distance between the successive positions in the rows and columns of the grid at which kriging is to be done, specified in the same units as the data. However, if the aim is to make a map, INTERVAL should be chosen so that it represents no more than 2 mm on the final printed document. The optimality of the kriging will then not be degraded noticeably by the subsequent contouring.

Kriging may be either punctual, i.e. at "points" which have the same size and shape as the sample support, or on bigger rectangular blocks. The size of the blocks is specified by the BLOCK option, in a variate whose two values define the length of the block first in the x direction (eastings) and then in the y direction (northings). By default the BLOCK variate contains two zero values, to give punctual kriging. The average semivariances between point and block, $\bar{\gamma}(\mathbf{h})$ and $\bar{\gamma}(B, B)$ in equations (8.3.9) and (8.3.11), are computed by integrating the variogram numerically over the block. The number of steps in each direction is defined by the NSTEP option. The default of 8 is recommended as a compromise between speed and accuracy. The kriging may be accelerated at the expense of accuracy by reducing NSTEP, or accuracy gained by increasing it.

The minimum is 4 and the maximum 10.

The minimum and maximum number of points for the kriging system, *n* in equations (8.3.9), are set by the MINPOINTS and MAXPOINTS options. There is a minimum limit of 3 for MINPOINTS and a maximum of 40 for MAXPOINTS, and MINPOINTS must be less than or equal to MAXPOINTS. The defaults are 7 and 20 respectively. Data points may be selected around the point or block to be kriged by setting the RADIUS option to the radius within which they must lie. If the variogram is anisotropic, the search may be requested to be anisotropic by setting option SEARCH to anisotropic; by default SEARCH=isotropic.

Universal kriging may be invoked by setting the DRIFT option to linear or to quadratic, i.e. to be of order 1 or 2 respectively. By default is DRIFT=constant, to give ordinary kriging. For data in a regular grid that is not square, the ratio of the spacing in the y direction to that in the x direction is given by the YXRATIO option. The default is 1.0 for square.

The variogram is specified by its type and parameters. The model and estimates can be saved using the SAVE parameter of MVARIOGRAM, and passed on to KRIGE using its SAVE parameter. Alternatively, they can be supplied as follows.

The model can be defined by setting the MODELTYPE option to either power, boundedlinear (one dimension only), circular, spherical, doublespherical, pentaspherical, exponential, besselk1 (Whittle's function), gaussian, cubic, stable (i.e. powered exponential), cardinalsine or matern, as defined in 8.3.2. All models may have a nugget variance, supplied using the NUGGET option; this is the constant estimated by MVARIOGRAM. For punctual kriging, you can specify the variance of any measurement error using the MEASUREMENTERROR parameter. The parameters of the power function (the only unbounded model) are defined by the GRADIENT and EXPONENT parameters. The parameter for the power of the stable model is supplied using the EXPONENT parameter. The parameter ν for the matern function is supplied using the SMOOTHNESS parameter. The simple bounded models, i.e. all other settings of MODELTYPE except doublespherical, require the SILLVARIANCES (the sill of the correlated variance) and RANGES parameters. The latter is strictly the correlation range of the boundedlinear, circular, spherical and pentaspherical models, while for the asymptotic models it is the distance parameter of the model. The doublespherical model requires SILLVARIANCES and RANGES to be set to variates of length two, to correspond to the two components of the model.

The ISOTROPY parameter allows the variation to be defined to be either isotropic or anisotropic in one of two ways: either Burgess anisotropy (Burgess & Webster 1980) or geometric anisotropy (Journel & Huijbregts 1978, Webster & Oliver 1990). The anisotropy is specified by three parameters, namely PHI, the angle in radians of the direction of maximum variation, RMAX, the maximum gradient or distance parameter of the model, and RMIN, the minimum gradient or distance parameter. the power, stable, exponential, Gaussian, pentashperical, spherical, cubic, and circular functions may be anisotropic.

KRIGE calculates two matrices, one of predictions (or estimates), which can be saved using the PREDICTIONS parameter, and the other of the prediction (estimation or kriging) variances saved using the VARIANCES parameter. The matrices are arranged with the first row of each matrix at the bottom following geographic rather than mathematical convention. You can save the Lagrange multipliers from the kriging solution using the LAGRANGEMULTIPLIER parameter. For ordinary Kriging the Lagrange multipliers are saved in a matrix (with a multiplier for each point). For universal Kriging a pointer of matrices is saved, where a matrix to save the Lagrange multipliers of each equation term.

The PRINT option can be set to data to print the data (2-dimensional regular data only). It also allows intermediate results to be printed. The setting search lists the results of the search for data around each position to be kriged, weights lists the kriging weights at each position and monitor monitors the formation and inversion of the kriging matrices for each position. These options enable you to check that the kriging is working reasonably. However, they can

produce a great deal of output, and should not be requested when kriging large matrices, such as might be wanted for mapping.

Example 8.3.4 completes the examination of the Brooms Barn data by using KRIGE to produce predictions of the potassium levels on a regular grid. First a small grid of values is produced for printing, then a finer grid is produced for contouring (Figures 8.3.4a and 8.3.4b).

Example 8.3.4

```
  31  " Produce matrices of predictions Kest and prediction variances Kvar."
  32  KRIGE  [PRINT=d;  X=East; Y=North; YOUTER=!(1,30); XOUTER=!(1,18); \
  33         YINNER=!(1,30); XINNER=!(1,18); BLOCK=!(1.0,1.0); RADIUS=4.75;\
  34         MINPOINTS=7; MAXPOINTS=20; INTERVAL=2] \
  35         LogK; ISOTROPY=isotropic; MODELTYPE=spherical;  NUGGET=0.0046; \
  36         SILL=0.01528; RANGE=10.81; PREDICTIONS=Kest; VARIANCES=Kvar

Kriging of irregularly spaced data
==================================

Data rectangle:          1 to 18 in the X direction
                         1 to 30 in the Y direction
Interpolated rectangle:  1 to 18 in the X direction
                         1 to 30 in the Y direction
Block size:              1 by 1
Interpolation grid:      15 rows, 9 columns
Interpolation interval:  2


Number of points required for interpolation

Minimum:  7
Maximum: 20


Data: LogK, 434 sites, initial search radius 4.75


Isotropic spherical model
-------------------------

Parameters:

Nugget variance                  0.004600
Sill variance                    0.015280
Range                             10.8100
Within-block variance            0.005712

  36  PRINT  Kest,Kvar; FIELD=7; DECIMALS=4

               Kest
              1.000  3.000  5.000  7.000  9.000 11.000 13.000 15.000 17.000

       30.00 1.3899 1.2306 1.1859 1.2975 1.3343 1.3367 1.3288 1.3210 1.3850
       28.00 1.3327 1.2228 1.2407 1.2172 1.2801 1.3976 1.4479 1.4927 1.4785
       26.00 1.2777 1.2074 1.2055 1.1949 1.2833 1.4003 1.3761 1.4376 1.4034
       24.00 1.3286 1.2377 1.1964 1.2080 1.3078 1.3519 1.3763 1.4095 1.4170
       22.00 1.3994 1.3113 1.2769 1.4288 1.6294 1.4297 1.4088 1.4187 1.4203
       20.00 1.4991 1.4199 1.3595 1.4777 1.5787 1.4379 1.4070 1.3699 1.4202
       18.00 1.5298 1.4277 1.4249 1.5298 1.5659 1.5382 1.5154 1.4733 1.4965
       16.00 1.4978 1.4149 1.3911 1.3519 1.4168 1.5365 1.5763 1.5680 1.5383
       14.00 1.4227 1.3754 1.2784 1.2846 1.4344 1.5506 1.4918 1.4896 1.4553
       12.00 1.4823 1.4692 1.2811 1.2742 1.4166 1.4729 1.4530 1.4497 1.4335
       10.00 1.4585 1.3819 1.2920 1.3201 1.3211 1.3669 1.4894 1.5313 1.4571
        8.00 1.4628 1.4633 1.3861 1.3741 1.3245 1.3236 1.4547 1.5048 1.4826
        6.00 1.3894 1.3603 1.3613 1.3713 1.3085 1.3334 1.4412 1.5032 1.4951
        4.00 1.3882 1.3463 1.3712 1.3869 1.3630 1.3895 1.4611 1.5369 1.5650
        2.00 1.3917 1.3963 1.3914 1.4226 1.4604 1.5063 1.5630 1.6051 1.5988
```

```
                Kvar
              1.000  3.000  5.000  7.000  9.000 11.000 13.000 15.000 17.000

        30.00 0.0014 0.0010 0.0011 0.0013 0.0013 0.0013 0.0013 0.0013 0.0027
        28.00 0.0013 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010
        26.00 0.0013 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010
        24.00 0.0016 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010
        22.00 0.0044 0.0027 0.0026 0.0025 0.0013 0.0010 0.0010 0.0010 0.0010
        20.00 0.0051 0.0027 0.0026 0.0025 0.0013 0.0010 0.0010 0.0012 0.0012
        18.00 0.0056 0.0014 0.0010 0.0010 0.0010 0.0010 0.0010 0.0012 0.0012
        16.00 0.0068 0.0027 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010
        14.00 0.0067 0.0016 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010
        12.00 0.0063 0.0013 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010
        10.00 0.0061 0.0013 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010
         8.00 0.0061 0.0013 0.0014 0.0010 0.0010 0.0010 0.0010 0.0010 0.0010
         6.00 0.0061 0.0013 0.0010 0.0019 0.0012 0.0010 0.0010 0.0010 0.0010
         4.00 0.0062 0.0013 0.0014 0.0033 0.0015 0.0010 0.0010 0.0010 0.0010
         2.00 0.0068 0.0014 0.0034 0.0054 0.0037 0.0014 0.0013 0.0016 0.0032

  38  KRIGE  [PRINT=d;  X=East; Y=North; YOUTER=!(1,30); XOUTER=!(1,18); \
  39         YINNER=!(1,30); XINNER=!(1,18); BLOCK=!(1.0,1.0); RADIUS=4.75;\
  40         MINPOINTS=7; MAXPOINTS=20; INTERVAL=0.5] \
  41         LogK; ISOTROPY=isotropic; MODELTYPE=spherical;  NUGGET=0.0046; \
  42         SILL=0.01528; RANGE=10.81; PREDICTIONS=Egrid; VARIANCES=Vgrid

Kriging of irregularly spaced data
==================================

Data rectangle:          1 to 18 in the X direction
                         1 to 30 in the Y direction
Interpolated rectangle:  1 to 18 in the X direction
                         1 to 30 in the Y direction
Block size:              1 by 1
Interpolation grid:      59 rows, 35 columns
Interpolation interval:  0.50


Number of points required for interpolation

Minimum:  7
Maximum: 20


Data: LogK, 434 sites, initial search radius 4.75


Isotropic spherical model
-------------------------

Parameters:

Nugget variance                  0.004600
Sill variance                    0.015280
Range                             10.8100
Within-block variance            0.005712

  43  GETATTRIBUTE [ATTRIBUTE=rows,columns] Egrid; SAVE=Dim
  44  CALCULATE Dim['rows'] = REVERSE(Dim['rows'])
  45  &         Nrow = NVALUES(Dim['rows'])
  46  MATRIX    [ROWS=Dim['rows']; COLUMNS=Dim['columns']] Ergrid,Vrgrid
  47  CALCULATE (Ergrid,Vrgrid)$[Nrow...1;*] = (Egrid,Vgrid)$[1...Nrow;*]
  48  " produce a contour map"
  49  FRAME     WINDOW=1,2; YLOWER=0; YUPPER=0.97,0.9; \
  50            XLOWER=0,0.65; XUPPER=0.65,0.99
  51  XAXIS     [RESET=yes] 1; LOWER=0.5; UPPER=18.5
  52  YAXIS     [RESET=yes] 1; LOWER=0.5; UPPER=30.5
  53  PEN       2,3; COLOUR='white','blue'
  54  DCONTOUR  [TITLE='Brooms Barn LogK'] Ergrid; PENFILL=!(2,3)
  55  DCONTOUR  [TITLE='LogK estimation variance'] Vrgrid; PENFILL=!(2,3)
```

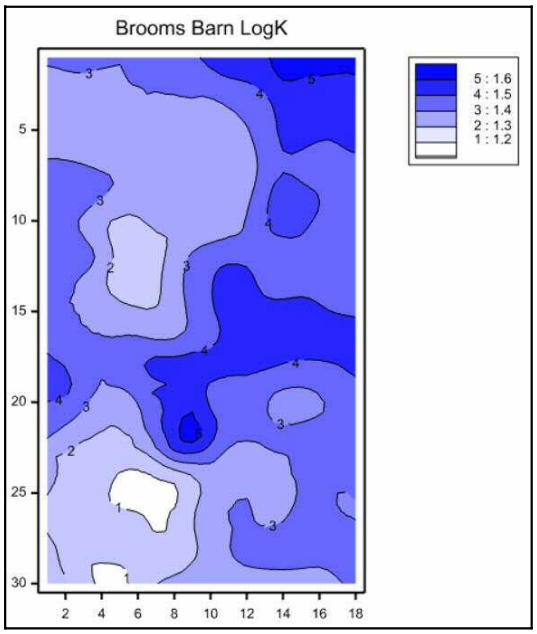


Figure 8.3.4a

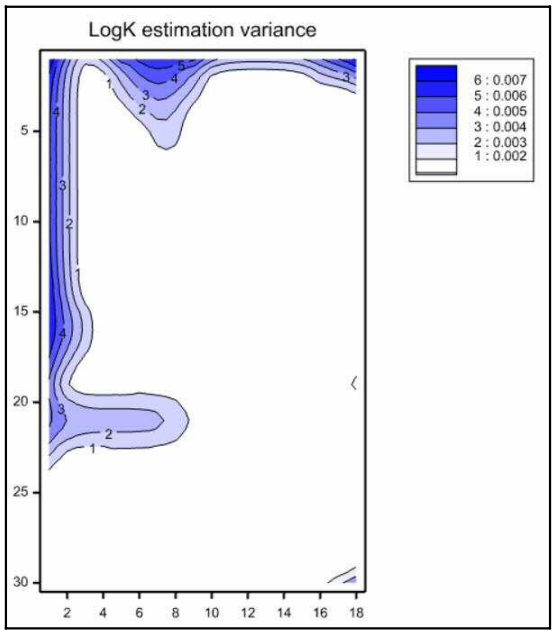


Figure 8.3.4b

8.3.5 Coregionalization and cokriging

Genstat has four commands, FCOVARIOGRAM, MCOVARIOGRAM, DCOVARIOGRAM and COKRIGE that can be used to model the spatial behaviour of several variables at once. These have been produced in collaboration with Andreas Papritz (Institute of Terrestrial Ecology, ETH Zurich). This section describes the underlying theory. The commands themselves are then described in Sections 8.3.6 - 8.3.9. Further information can be found in Chapter 10 of Webster & Oliver (2007).

Two or more random variables may be "coregionalized" in the sense that they are spatially correlated individually (regionalized in the sense above) and spatially correlated with one another. The ideas are formalized for two variables, $Z_u(\mathbf{x})$ and $Z_v(\mathbf{x})$, denoted u and v henceforth, and both obeying Matheron's intrinsic hypothesis as set out at the start of this section.

In the augmented notation the expected difference for u at lag $\mathbf{h}$ is

$$\mathbf{E}[\, Z_u(\mathbf{x}) - Z_u(\mathbf{x}+\mathbf{h}) \,] = 0\,, \tag{8.3.16}$$

and the variogram, specifically the *autovariogram* of u, is

$$\gamma_{uu}(\mathbf{h}) = \tfrac{1}{2}\,\mathbf{E}[\, \{Z_u(\mathbf{x}) - Z_u(\mathbf{x}+\mathbf{h})\}^2 \,]\,. \tag{8.3.17}$$

The reason for the double subscript uu will become apparent presently. Similar expressions hold for variable v, the autovariogram of which is $\gamma_{vv}(\mathbf{h})$.

The two variables have a *cross-variogram*, $\gamma_{uv}(\mathbf{h})$, defined as

$$\gamma_{uv}(\mathbf{h}) = \tfrac{1}{2}\,\mathbf{E}[\, \{Z_u(\mathbf{x}) - Z_u(\mathbf{x}+\mathbf{h})\}\ \{Z_v(\mathbf{x}) - Z_v(\mathbf{x}+\mathbf{h})\} \,]\,. \tag{8.3.18}$$

This function describes the way in which u is related spatially to v.

If both variables are second-order stationary, then both will have covariance functions. That for $C_{uu}(\mathbf{x})$ is

$$C_{uu}(\mathbf{h}) = \mathbf{E}[\, \{Z_u(\mathbf{x}) - \mu_u\}\ \{Z_u(\mathbf{x}+\mathbf{h}) - \mu_u\} \,]\,. \tag{8.3.19}$$

where μ_u is the mean of u. The covariance function of v, $C_{vv}(\mathbf{x})$, is defined similarly. The two variables have a cross-covariance function:

$$C_{uv}(\mathbf{h}) = \mathbf{E}[\, \{Z_u(\mathbf{x}) - \mu_u\}\ \{Z_v(\mathbf{x}+\mathbf{h}) - \mu_v\} \,]\,. \tag{8.3.20}$$

This function is related to the cross-variogram by

$$\gamma_{uv}(\mathbf{h}) = C_{uv}(\mathbf{0}) - \tfrac{1}{2}\,\{C_{uv}(\mathbf{h}) + C_{uv}(-\mathbf{h})\} \tag{8.3.21}$$

Note, however, that $C_{uv}(\mathbf{h})$ is in general different from $C_{uv}(-\mathbf{h})$, whereas

$$\gamma_{uv}(\mathbf{h}) = \gamma_{uv}(-\mathbf{h}) \tag{8.3.22}$$

for all **h**.

The cross-variogram is estimated from data in a way analogous to that for the autovariogram by the method of moments:

$$\hat{\gamma}_{uv}(\mathbf{h}) = \frac{1}{2m(\mathbf{h})} \sum_{i=1}^{m(\mathbf{h})} \{Z_u(\mathbf{x}_i) - Z_u(\mathbf{x}_i+\mathbf{h})\}\,\{Z_v(\mathbf{x}_i) - Z_v(\mathbf{x}_i+\mathbf{h})\} \tag{8.3.23}$$

where the $Z_u(\mathbf{x}_i)$ and $Z_v(\mathbf{x}_i)$ are the measured values of u and v at $\mathbf{x}_i$, and $Z_u(\mathbf{x}_i+\mathbf{h})$ and $Z_v(\mathbf{x}_i+\mathbf{h})$ are those at $\mathbf{x}_i+\mathbf{h}$. Note that there must be measurements of both u and v at some places. When there are only a small number of matching locations or no common locations, Genstat provides an alternative algorithm, described by Künsch, Papritz & Bassi (1997), which estimates the generalized cross-covariances.

The models available for cross-variograms are the same as those for autovariograms. To describe the coregionalization, however, the models must combine in a coherent way such that the combination cannot give rise to "negative variances". For this one adopts the linear model of coregionalization. In it the variogram for any pair of variables u and v is the sum of two or more, $K \geq 2$, basic functions, $g_k(\mathbf{h})$, multiplied by appropriate coefficients:

$$\gamma_{uv}(\mathbf{h}) = \sum_{k=1}^{K} b^k_{uv}\, g_k(\mathbf{h}) \ . \tag{8.3.24}$$

The coefficients b^{kuv}, in which the k is simply an index, not a power, are the variances and covariances, i.e. nugget (the b^1_{uv}) and sill variances of independent components of the cross-variogram if they are bounded. The $g_k(\mathbf{h})$ are basic variogram functions of correlated random variables with mean 0 and variance 1 and distance parameters to be determined. Thus, a basic isotropic spherical function, for example, is

$$\begin{aligned} \gamma_k(\mathbf{h}) &= \frac{3a_k}{2h} - \frac{1}{2}\left(\frac{h}{a_k}\right)^3 \qquad \text{for } h \leq a_k \\ &= 1 \qquad\qquad\qquad\quad \text{for } h > a_k \ , \end{aligned} \tag{8.3.25}$$

where $h = |\mathbf{h}|$. Its sole parameter is a_k, the range for the kth component. For unbounded variograms the b^1_{uv} are the nugget variances and the b^k_{uv} for $k>1$ are the gradients.

The coefficients $b^k_{uv} = b^k_{vu}$ for all k, and for each k the matrix of coefficients

$$\begin{bmatrix} b^k_{uu} & b^k_{uv} \\ b^k_{vu} & b^k_{vv} \end{bmatrix}$$

must be positive definite. The matrix is symmetric, and so it is sufficient that $b^k_{uu} \geq 0$ and $b^k_{vv} \geq 0$ and that its determinant is positive or zero:

$$|b^k_{uv}| = |b^k_{vu}| \leq \sqrt{(b^k_{uu}\, b^k_{vv})} \tag{8.3.26}$$

This is Schwarz's inequality.

For V coregionalized variables the full matrix of coefficients, $[b^k_{ij}]$, is of order V, and all its principal minors must be positive or zero.

Schwarz's inequality has the following consequences for each pair of variables.

1. Every basic variogram function, $g_k(\mathbf{h})$, represented in a cross-variogram must also appear in the two autovariograms, i.e. $b^k_{uu} \neq 0$ and $b^k_{vv} \neq 0$ if $b^k_{uv} \neq 0$. If a basic $g_k(\mathbf{h})$ is absent from either autovariogram then it may not be present in the cross-variogram.
2. The reverse is permissible; b^k_{uv} may be zero when either b^k_{uu} or b^k_{vv} or both exceed zero;

i.e. structures may appear in the autovariograms without their being present in the cross-variogram.

Genstat ensures that the model fitted to the coregionalization is conditional semi-definite (CNSD) by using the algorithm of Goulard & Voltz (1992). As a further check on the model one can plot the cross experimental variogram for any pair of variables and the model for them on a graph with the limiting values that would hold if correlation were perfect. This last condition gives the hull of perfect correlation (Wackernagel 1995), which is obtained from the b^k_{uu} and b^k_{vv} by

$$\text{hull}[\gamma_{uv}(\mathbf{h})] = \pm \sum_{k=1,K} \{ \sqrt{(b^k_{uu}\, b^k_{vv})}\, g_k(\mathbf{h}) \} \qquad (8.3.27)$$

The line for the fitted model must lie within the hull to be acceptable. It also reveals the strength of the cross correlation. If it lies close to either bound of the hull then the corelation is strong. If, in contrast, the line lies far from both bounds then the correlation is weak.

Cokriging is an elaboration of the corresponding form of autokriging in which the additional information in the cross correlations with subsidiary variables is taken into account in the predictions.

Suppose there are V regionalized variables, l=1, 2, ... V, of which variable u, the target variable, is to be predicted. Typically u will have been sampled less densely than the others. In ordinary cokriging an estimate of u in a block B is the linear sum

$$\hat{Z}_u(B) = \sum_{l=1}^{V} \sum_{i=1}^{n_l} \lambda_{il} z_l(x_i) \ , \qquad (8.3.28)$$

where the subscript l refers to the variables, and i refers to the sampling points of which there are n_l where variable l has been measured. The λ_{ij} are weights satisfying

$$\begin{aligned} \sum_{i=1}^{n_l} \lambda_{il} &= 1 \quad \text{if } l = u \ , \\ &= 0 \quad \text{if } l \neq u \ , \end{aligned} \qquad (8.3.29)$$

These are the non-bias conditions, and subject to them the prediction variance of $Z_u(B)$ for a block, B, is minimized by solution of the kriging system:

$$\begin{aligned} &\sum_{l=1}^{V} \sum_{i=1}^{n_l} \lambda_{il} \gamma_{lv}(\mathbf{x}_i \mathbf{x}_j) + \psi_v = \bar{\gamma}_{uv}(\mathbf{x}_j, B) \ , \\ &\sum_{i=1}^{n_l} \lambda_{il} = 1 \quad \text{if } l = u \ , \\ &\qquad = 0 \quad \text{if } l \neq u \ , \end{aligned} \qquad (8.3.30)$$

for all v=1, 2, ... V and all j=1, 2 ... n_v. The quantity $\gamma_{lv}(\mathbf{x}_i, \mathbf{x}_j)$ is the (cross) semivariance between variables l and v at sites i and j, separated by the vector $\mathbf{x}_i - \mathbf{x}_j$; $\bar{\gamma}_{uv}(\mathbf{x}_j, B)$ is the average (cross) semivariance between a site j and the block B, and ψ_v is the Lagrange multiplier for the vth variable. If l=v or u=v, then the semivariances are the autosemivariances. This set of equations is the extension of the autokriging system.

Solving these Equations 8.3.30 gives the weights, λ_{il}, which are inserted into Equation 8.3.28 to estimate $Z_u(B)$. The cokriging variance is obtained from

$$\sigma_u^2(B) = \sum_{l=1}^{V} \sum_{j=1}^{n_l} \lambda_{jl} \bar{\gamma}_{ul}(\mathbf{x}_j, B) + \psi_u - \bar{\gamma}_{uu}(B, B) ,$$

(8.3.31)

where $\bar{\gamma}_{uu}(B, B)$ is the integral of $\gamma_{uu}(\mathbf{h})$ over B, i.e. the within-block variance of u.

The equations are represented in matrix form for only two variables, u and v, for simplicity. Let $\mathbf{\Gamma}_{uv}$ denote a matrix of semivariances (including cross semivariances where $u \neq v$) between sampling points in a neighbourhood, and suppose that there are n_u places at which variable u was measured and n_v where v was measured.

The matrix, of order $n_u \times n_v$, is

$$\mathbf{\Gamma}_{uv} = \begin{bmatrix} \gamma_{uv}(\mathbf{x}_1, \mathbf{x}_1) & \gamma_{uv}(\mathbf{x}_1, \mathbf{x}_2) & \cdots & \gamma_{uv}(\mathbf{x}_1, \mathbf{x}_{n_v}) \\ \gamma_{uv}(\mathbf{x}_2, \mathbf{x}_1) & \gamma_{uv}(\mathbf{x}_2, \mathbf{x}_2) & \cdots & \gamma_{uv}(\mathbf{x}_2, \mathbf{x}_{n_v}) \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \gamma_{uv}(\mathbf{x}_{n_u}, \mathbf{x}_1) & \gamma_{uv}(\mathbf{x}_{n_u}, \mathbf{x}_2) & \cdots & \gamma_{uv}(\mathbf{x}_{n_u}, \mathbf{x}_{n_v}) \end{bmatrix}$$

(8.3.32)

Denote by $\mathbf{b}_{uu}$ and by $\mathbf{b}_{uv}$ the vectors of autosemivariances for variable u and cross semivariances:

$$\mathbf{b}_{uu} = \begin{bmatrix} \bar{\gamma}_{uu}(\mathbf{x}_1, B) \\ \bar{\gamma}_{uu}(\mathbf{x}_2, B) \\ \cdot \\ \cdot \\ \cdot \\ \bar{\gamma}_{uu}(\mathbf{x}_{n_u}, B) \end{bmatrix}$$

(8.3.33)

$$\mathbf{b}_{uv} = \begin{bmatrix} \bar{\gamma}_{uv}(\mathbf{x}_1, B) \\ \bar{\gamma}_{uv}(\mathbf{x}_2, B) \\ \cdot \\ \cdot \\ \cdot \\ \bar{\gamma}_{uv}(\mathbf{x}_{n_v}, B) \end{bmatrix}$$

(8.3.34)

The matrix equation is then

$$\begin{bmatrix} & & & 1 & 0 \\ & & & 1 & 0 \\ \boldsymbol{\Gamma}_{uu} & & \boldsymbol{\Gamma}_{uv} & \cdot & \cdot \\ & & & \cdot & \cdot \\ & & & \cdot & \cdot \\ & & & 1 & 0 \\ & & & 0 & 1 \\ & & & 0 & 1 \\ \boldsymbol{\Gamma}_{vu} & & \boldsymbol{\Gamma}_{vv} & \cdot & \cdot \\ & & & \cdot & \cdot \\ & & & \cdot & \cdot \\ & & & 0 & 1 \\ 1 & 1 \ \ldots \ 1 & 0\,0 \ \ldots \ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_{1u} \\ \lambda_{2u} \\ \cdot \\ \cdot \\ \cdot \\ \lambda_{n_u u} \\ \lambda_{1v} \\ \lambda_{2v} \\ \cdot \\ \cdot \\ \cdot \\ \lambda_{n_v v} \\ \psi_u \\ \psi_v \end{bmatrix} = \begin{bmatrix} \\ \\ \mathbf{b}_{uu} \\ \\ \\ \\ \mathbf{b}_{uv} \\ \\ \\ 1 \\ 0 \end{bmatrix} \tag{8.3.35}$$

Denote the augmented matrix of $\boldsymbol{\Gamma}$s by $\mathbf{G}$, the vector of weights and Lagrange multipliers by $\boldsymbol{\lambda}$, and the right hand side vector by $\mathbf{b}$; then the solution of the equation is succinctly

$$\boldsymbol{\lambda} = \mathbf{G}^{-1}\,\mathbf{b}\ . \tag{8.3.36}$$

The cokriging (prediction) variance is given by

$$\hat{\sigma}_u^{\,2}(B) = \mathbf{b}^{\mathrm{T}}\boldsymbol{\lambda} - \bar{\gamma}_{uu}(B, B)\ . \tag{8.3.37}$$

As in autokriging the block B may be of any reasonable size and shape, and it may be reduced to a point, $\mathbf{x}_0$, having the same dimensions as the support on which the data were obtained. In these circumstances the averages $\bar{\gamma}_{uv}(\mathbf{x}_j, B)$ become $\gamma_{uv}(\mathbf{x}_j, \mathbf{x}_0)$, and $\bar{\gamma}_{uu}(B, B)$ is zero and hence disappears, so that

$$\mathbf{b}_{uu} = \begin{bmatrix} \gamma_{uu}(\mathbf{x}_1, \mathbf{x}_0) \\ \gamma_{uu}(\mathbf{x}_2, \mathbf{x}_0) \\ . \\ . \\ . \\ \gamma_{uu}(\mathbf{x}_{n_u}, \mathbf{x}_0) \end{bmatrix} \tag{8.3.38}$$

$$\mathbf{b}_{uv} = \begin{bmatrix} \gamma_{uv}(\mathbf{x}_1, \mathbf{x}_0) \\ \gamma_{uv}(\mathbf{x}_2, \mathbf{x}_0) \\ . \\ . \\ . \\ \gamma_{uv}(\mathbf{x}_{n_v}, \mathbf{x}_0) \end{bmatrix} \tag{8.3.39}$$

and

$$\hat{\sigma}_u^2(\mathbf{x}_0) = \mathbf{b}^{\mathrm{T}}\boldsymbol{\lambda} \, . \tag{8.3.40}$$

8.3.6 The FCOVARIOGRAM directive

FCOVARIOGRAM directive

Forms a covariogram structure containing auto-variograms of individual variates and cross-variograms for pairs from a list of variates.

Options

PRINT = *string token*	Controls printed output (statistics, variograms, autovariograms); default stat
METHOD = *string token*	Specifies what to do when the measurements are not all made at the same locations (allwithcrossnugget, allnocrossnugget, commonpoints); default comm
COVARIOGRAM = *pointer*	Pointer to store the variograms, cross-variograms and associated information for use in MCOVARIOGRAM
MAXLAG = *scalar*	Maximum lag in all directions
STEPLENGTHS = *scalar* or *variate*	Length of the step or steps in which lag is incremented
DIRECTIONS = *scalar* or *variates*	Directions along which to form the variogram, scalar for a single direction in 2 dimensions, variate for several directions in 2 dimensions, and pairs of variates for 3 dimensional data
SEGMENTS = *scalar*	Angle subtended by each segment along the DIRECTIONS

COORDSYSTEM = *string token*	Coordinate system used for the geometry for discretizing the lag (mathematical, geographical); default math
MAXCONEDIAMETER = *scalar*	Diameter at which the segments over which averaging is to be done should cease to expand; default * implies no limit
MINCOUNT = *scalar*	Minimum number of points required at a particular lag point for the cross-variogram to be estimated there; default 1
DRIFT = *string token*	Mean function (constant, linear, quadratic); default cons

Parameters

DATA = *variates*	Measurements as a variate
X1 = *variates*	Locations of each set of measurements in the first dimension
X2 = *variates*	Locations of each set of measurements in the second dimension (if recorded in more than 1 dimension)
X3 = *variates*	Locations of each set of measurements in the third dimension (if recorded in 3 dimensions)

To perform cokriging in Genstat, you must first form a covariogram structure containing the necessary auto- and cross-variograms, using the FCOVARIOGRAM directive.

The data are supplied as a list of variates (one for each variable of interest) using the DATA parameter. The locations of the measurements are supplied using the parameter X1 for data in one dimension only, or X1 and X2 for two dimensions, or X1, X2 and X3 for three dimensions. Any restrictions on the variates are ignored.

The METHOD option specifies how to calculate the cross-variograms. The setting commonpoints specifies that only those points in common in every sample are to be included; Equation 8.3.23 is then used (see Section 8.3.5). Alternatively, the setting allnocrossnugget can be used when the sampling locations do not match. This uses an algorithm outlined in Künsch, Papritz & Bassi (1997) that performs least-squares fitting of the cloud of products of differences to estimate the expected value of these products. If there are no common points, the nugget variance cannot be calculated. However, if there is partial sampling (some common points), the setting allwithcrossnugget can be used to shift the cross-variograms by the semivariance at the origin to estimate the nugget effect.

The maximum lag distance in all directions to which the variograms are calculated is set by the MAXLAG option. The increments in distance are set by the STEPLENGTH option, where you can supply a scalar to define equally-spaced steps or a variate to specify the steps themselves. The directions along which to form the variograms are supplied in degrees using the DIRECTIONS option. The geometry used for the directions is given by the COORDSYSTEM option: the setting mathematical specifies directions counter-clockwise from east, and geographical specifies clockwise from north (for the first direction only in three dimensions). Each direction is at the centre of an angular range. The angle is the same in every direction, and is defined by the SEGMENTS option. For a single direction in two dimensions the DIRECTIONS option should be set to a scalar, while for several directions it should be set to a variate. For directions in three dimensions, DIRECTIONS should specify a pair of variates. The MAXCONEDIATMETER option can be used to specify a diameter at which the segments cease to expand. For cross-variograms that are formed using all points the minimum number of points required at each lag can be specified using the MINCOUNT option.

The DRIFT option can be used to calculate the variograms after removing a systematic

component. Setting the DRIFT option to linear or quadratic will fit a regression to the observations and then form the variograms on the residuals.

The COVARIOGRAM option allows you to specify pointer to save the auto-variograms, cross-variograms and associated information. Its elements contain:

1 a matrix with columns of variograms and cross-variograms and rows indexed by lags within directions;
2 a variate of counts at the lags in each direction;
3 distances of the lags in each direction;
4 horizontal angles;
5 vertical angles;
6 variances;
7 distance classes;
8 method;
9 pointer containing identifiers of the DATA variates;
10 number of dimensions.

This structure provides the information required to fit models to the covariogram using the directive MCOVARIOGRAM.

The PRINT option can be set to statistics to display statistics for each of the variates. The setting variograms displays each of the auto- and cross-variograms, while the setting autovariogram displays only the auto-variograms.

In Example 8.3.6 the experimental auto- and cross-variograms of cadmium, zinc and nickel taken on an incomplete grid at Swiss Jura are estimated where the cross-variograms have been formed from common points, e.g. sites where both variables have been measured. The results are saved into a pointer called save_cov for use within the MCOVARIOGRAM directive and to extract values for plotting using the DGRAPH statement in lines 20 and 21 (Figures 8.3.6).

Example 8.3.6

```
  2  " Data are measurements of concentrations of trace metals in the topsoil
 -3    of the Swiss Jura. Data analyzed are Cadmium, Nickel and Zinc taken
 -4    from Goovaerts prediction subset. See Goovaerts (1997) Geostatistics
 -5    for Natural Resources Evaluation."
  6  FILEREAD     [PRINT=summary; NAME=\
  7               '%GENDIR%/Examples/GuidePart2/Goovaerts.dat']X1,X2,Cd,Ni,Zn

Summary
-------

The file %GENDIR%/Examples/GuidePart2/Goovaerts.dat is assumed to contain 5
structure(s), with one value for each structure on each record.


The file contains 259 values for each of the following structures:

  Identifier      Type   Missing
          X1   variate         0
          X2   variate         0
          Cd   variate         0
          Ni   variate         0
          Zn   variate         0

  8  FCOVARIOGRAM [PRINT=statistics; MAXLAG=2.1; STEP=0.1; DIRECTIONS=0;\
  9               SEGMENTS=180; MAXCONE=500; MINCOUNT=1; \
 10               COVARIOGRAM=Save_cov] DATA=Cd,Ni,Zn; X1=X1; X2=X2
```

```
Sample statistics
-----------------

Variate        Mean    Variance     No.Obs.
     Cd       1.309       0.838         259
     Ni      19.730      67.780         259
     Zn      75.078     842.119         259
```

```
  11  " Plot the variograms and covariograms."
  12  GETATTRIBUTE [ATTRIBUTE=columns] Save_cov['semivar']; Lab
  13  FRAME        11...16; YLOWER=2(0.66,0.33,0); YUPPER=2(0.98,0.65,0.32);\
  14               XLOWER=(0,0.5)3; XUPPER=(0.5,1)3
  15  TEXT         scr; VALUE='clear'
  16  PEN          1; SYMBOL='circle'
  17  XAXIS        11...16; TITLE='Lag distance/km'; LOWER=0; LROTATION=45
  18  YAXIS        11...16; TITLE='Semi-variance'; LOWER=0
  19  FOR [INDEX=i; NTIMES=6]
  20    DGRAPH     [WINDOW=i+10; KEY=0; TITLE=Lab['columns']$[i]; SCREEN=#scr] \
  21               Save_cov['semivar']$[*;i]; Save_cov['distances']$[*;i]
  22    TEXT       scr; VALUE='keep'
  23  ENDFOR
```

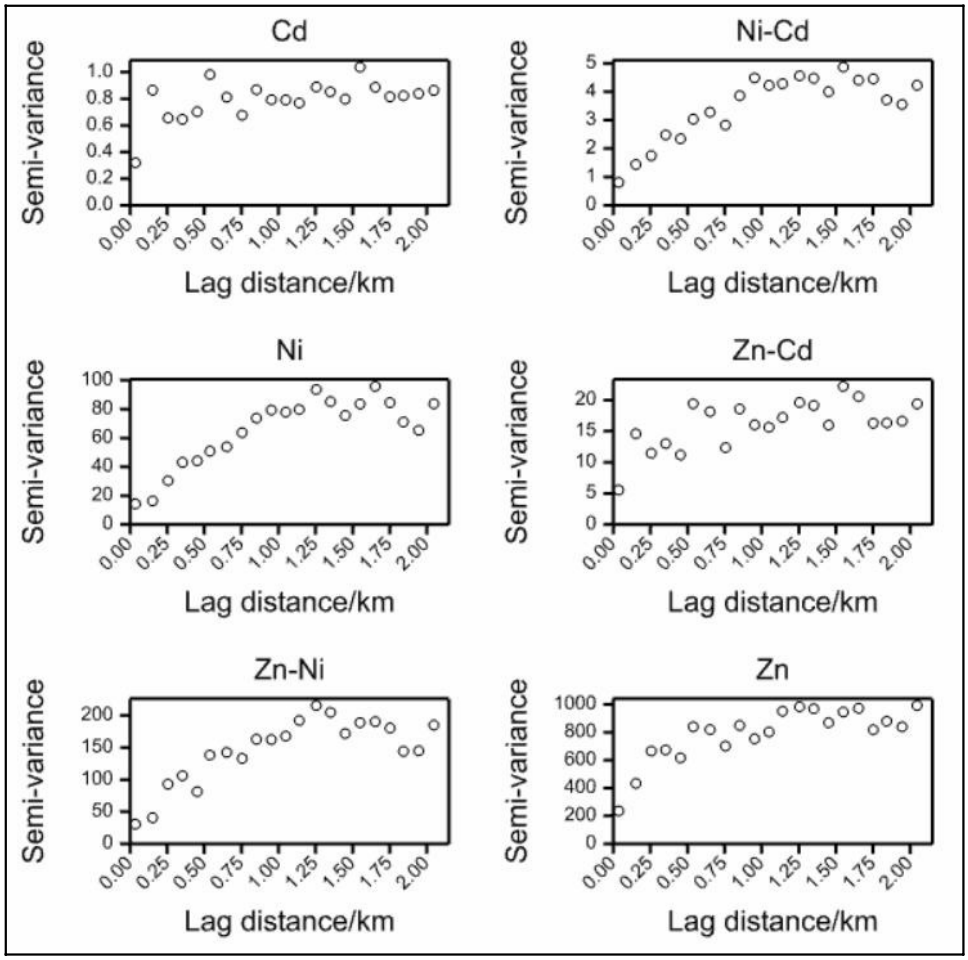


Figure 8.3.6

8.3.7 The MCOVARIOGRAM directive

MCOVARIOGRAM directive

Fits models to sets of variograms and cross-variograms.

Options

PRINT = *string tokens*	Controls printed output from the fit (model, summary, estimates, fittedvalues, monitoring); default mode, summ, esti
WEIGHTING = *string token*	Method to be used for weighting (counts, equal); default coun
MAXLAG = *scalar*	Maximum lag distance of points to be included in the modelling
MINCOUNT = *scalar*	Minimum number of points required at a particular lag point for a pair of variables for this to be used to model their cross-variogram; default 30 for equal weighting and 10 for counts
MAXCYCLE = scalar	Maximum number of iterations for model fitting; default 30
TOLERANCES = *variate*	Tolerances for model fitting; default * i.e. appropriate default values
COORDSYSTEM = *string token*	Coordinate system used for the geometry for discretizing the lag (mathematical, geographical); default math
COVARIOGRAM = *pointers*	Experimental variograms, cross-variograms and associated information defining the data for fitting the model

Parameters

MODELTYPE = *string tokens*	Defines the model structures to be fitted (nugget, power, boundedlinear, circular, spherical, pentaspherical, cubic, stable, besselk1, cardinalsine, dampenedcosine); no default i.e. must be specified
INITIAL = *scalars* or *variates*	Scalar defining the initial distance parameter for fitting an isotropic model structure or a variate defining initial values for an anisotropic ellipse or ellipsoid for fitting an geometrical anisotropic model
ISOTROPY = *string tokens*	Specifies the zonal anisotropy to be used for model structure (isotropic, x, y, z, xy, xz, yz); default isot
ESTIMATES = *pointers*	Structures to store the estimated nonlinear parameters and sill values
LOWER = *scalars*	Lower bound for each nonlinear distance parameter
UPPER = *scalars*	Upper bound for each nonlinear distance parameter
STEPLENGTH = *scalars*	Initial step length for each nonlinear distance parameter
SMOOTHNESS = *scalars*	Value of exponent parameter for the power and stable models, or theta parameter for the dampened-cosine model

The next step is to use the MCOVARIOGRAM directive to fit models to the auto- and cross-variograms formed by FCOVARIOGRAM (8.3.6). These are transferred using the COVARIOGRAM options of the two directives.

You can specify a combination of basic variogram functions to model the variograms, for example, nugget plus spherical. MCOVARIOGRAM uses the algorithms from the directives FIT and FITNONLINEAR to estimate the model parameters for the combination of basic variogram functions. It then fits a linear model of coregionalization using the Goulard & Voltz (1992) algorithm, where each step of the solution is checked for conditional semi-definiteness. The two-step process is iterated until convergence.

The MODELTYPE parameter selects the combination of model structures to be used in the model:

nugget	c_0
boundlinear	ch/a for h ≤ a, otherwise 0
circular	$c\ \{1 - (2/\pi)\arccos(h/a) + (2h/(\pi a))\sqrt{(1-h^2/a^2)}\}$ for h ≤ a, otherwise 0
spherical	$c\ \{1.5h/a - 0.5(h/a)^3\}$ for h ≤ a, otherwise 0
pentaspherical	$c\ \{1.875h/a - 1.25(h/a)^3 + 0.375(h/a)^5\}$ for h ≤ a, otherwise 0
cubic	$c\ \{7(h/a)^2 - 8.75(h/a)^3 + 3.5(h/a)^5 - 0.75(h/a)^7\}$
stable	$c\ \{1 - \exp(-(h/a)^b)\}$ for $0 \le b \le 2$
besselk1	$c\ \{1 - h/a\ k_1(h/a)\}$
cardinalsine	$c\ \{1 - a/h \sin(h/a)\}$
dampenedcosine	$c\ \{1 - \exp(-h/(as))\cos(h/a)\}$
power	gh^{α}

Initial values for the model structures should be supplied using the INITIAL parameter. For an isotropic model the initial value should be specified as a scalar. You can specify a geometrically anisotropic model by supplying the values within a variate. In two dimensions the variate should contain three values that define an anisotropy ellipse. The first value should define the first axis direction. This is the angle for the main direction of continuity (least change with separating distance) measured in degrees, counter-clockwise from East if option COORDSYSTEM is set to mathematical or clockwise from North if COORDSYSTEM is set to geographical. The second value should contain the initial value for the distance parameter of the first axis, and the last value of the variate should be the anisotropy ratio between the distance parameters along the first axis (principal direction of continuity) and the second axis.

In three dimensions the variate should contain six values that define an anisotropy ellipsoid. The first value defines the angle for the first axis (principal direction of continuity) which is measured in degrees, counter-clockwise from East if COORDSYSTEM is set to mathematical or clockwise from North if COORDSYSTEM is set to geographical. The second value defines the dip angle for the first axis (rotation angle around the y-axis) which is measured in degrees up from horizontal. The third value defines the rotation angle of the second and third axis around the first axis (defined by the two previous angles). The fourth value should contain the initial value for the distance parameter along the first axis. The fifth value defines the anisotropy ratio between distance parameters along the first and second axis of the ellipsoid. The last value of the variate defines the anisotropy ratio between the distance parameters along the second and third axis of the ellipsoid.

Another form of anisotropy can occur when the sill of a semi-variogram varies in different directions. This is known as zonal anisotropy and you can set a model structure to be zonal in particular directions using the ISOTROPY parameter. A model structure can be zonal and geometrically anisotropic.

For the power and stable models the SMOOTHNESS option controls the power parameter for the model. By default, the parameter is estimated, however, you can supply a value to fix the parameter for the model fitting.

The WEIGHTING option controls the weights that are used when fitting the model. The default setting counts uses the values supplied for the counts within the COVARIOGRAM option, and equal uses equal weights (of one).

The MAXLAG option can be used to specify the maximum lag distance of points to be included in the modelling. The MINCOUNT option specifies the minimum number of points to be used to model the variograms at a particular lag.

The TOLERANCES option controls the criterion for convergence of the nonlinear regression and Goulard & Voltz algorithm. The values should be supplied in a variate where the first value is the criterion for the nonlinear regression and the second value is the criterion for the Goulard & Voltz algorithm. The option MAXCYCLE can be used to change the maximum number of iterations performed by the nonlinear regression from the default of 30.

The geometry used for the directions supplied using the COVARIOGRAM option is given by the COORDSYSTEM option, where the setting mathematical specifies directions counter-clockwise from East, and geographical clockwise from North (for the first angle only in 3 dimensions).

The ESTIMATES parameter allows you to specify an identifier to save the estimated nonlinear parameters, sill values and associated information. This structure stores the information required by the DCOVARIOGRAM procedure (8.3.8) or the COKRIGE directive (8.3.9).

The PRINT option controls the output to be displayed, with settings:

model	description of the models fitted,
summary	summary of analysis,
estimates	parameter estimates,
fittedvalues	fitted semi-variances,
monitoring	monitoring information at each iteration of the nonlinear regression.

Example 8.3.7 models the coregionalization in Example 8.3.6, using double spherical functions with nugget variances. For initial values the parameters obtained by Goovaerts (1997) are used. The experimental auto- and cross-variograms along with other information defining the data for fitting the model are supplied in a pointer saved from the FCOVARIOGRAM directive. The model estimates are saved in a pointer called Save_est for use by DCOVARIOGRAM (8.3.8) and COKRIGE (8.3.9).

Example 8.3.7

```
 24  " Model the coregionalization."
 25  MCOVARIOGRAM [PRINT=summary,estimates; WEIGHTING=counts;\
 26               MAXLAG=3; MINCOUNT=20; COVARIOGRAM=Save_cov]\
 27               MODELTYPE=nugget,spherical,spherical; INITIAL=*,0.2,1.3;\
 28               ESTIMATES=Save_est

Summary
-------

Number of adjustable parameters: 20
Residual Sum of Squares:         280780.


Model Estimates
---------------

Number of elementary correlation structures: 3

Correlation structure 1

  Anisotropy:  Isotropic
  Model:       Pure nugget
```

```
Sill variance(s):

          Cd     0.05172
          Ni     0.05265     0.09138
          Zn     0.05170     0.04123     0.05511
                      Cd          Ni          Zn

Correlation structure 2

  Anisotropy:  Isotropic
  Model:       Spherical

Distance parameter(s):

  Distance:  0.1096

Sill variance(s):

          Cd         0.6
          Ni         1.3        19.8
          Zn        10.9        57.8       515.3
                     Cd          Ni          Zn

Correlation structure 3

  Anisotropy:  Isotropic
  Model:       Spherical

Distance parameter(s):

  Distance:  1.566

Sill variance(s):

          Cd         0.2
          Ni         3.0        64.0
          Zn         7.3       126.7       403.9
                     Cd          Ni          Zn
```

8.3.8 The DCOVARIOGRAM procedure

DCOVARIOGRAM procedure

Plots models fitted to 2-dimensional auto- and cross-variograms (D.A. Murray).

Options

PLOT = *string token*	Controls how to display the plotted variograms (separate, scattermatrix); default scat
ESTIMATES = *pointer*	Pointer containing model estimates saved from MCOVARIOGRAM

Parameter

COVARIOGRAM = *pointer*	Pointer to supply the semi-variances, distances and associated information as saved from FCOVARIOGRAM

DCOVARIOGRAM plots 2-dimensional auto- and cross-variograms using data generated by FCOVARIOGRAM (8.3.6). DCOVARIOGRAM can also be used to display the fitted model for isotropic models using estimates generated from MCOVARIOGRAM (8.3.7).

The data should be supplied in a pointer that has been saved using the COVARIOGRAM option from FCOVARIOGRAM. This pointer provides the auto-variograms, cross-variograms and

associated information required for the plots. The ESTIMATES option can be used to plot an isotropic fitted model of coregionalization where the estimates are taken directly from MCOVARIOGRAM. Graphical output is controlled using the PLOT option. The setting separate produces each auto- and cross-variogram on a separate plot. Alternatively, they can be combined onto a single scatter matrix using the scattermatrix setting.

Figure 8.3.8 shows a plot of the models fitted by MCOVARIOGRAM in Example 8.3.7, using the command:

```
DCOVARIOGRAM [ESTIMATES=Save_est] Save_cov
```

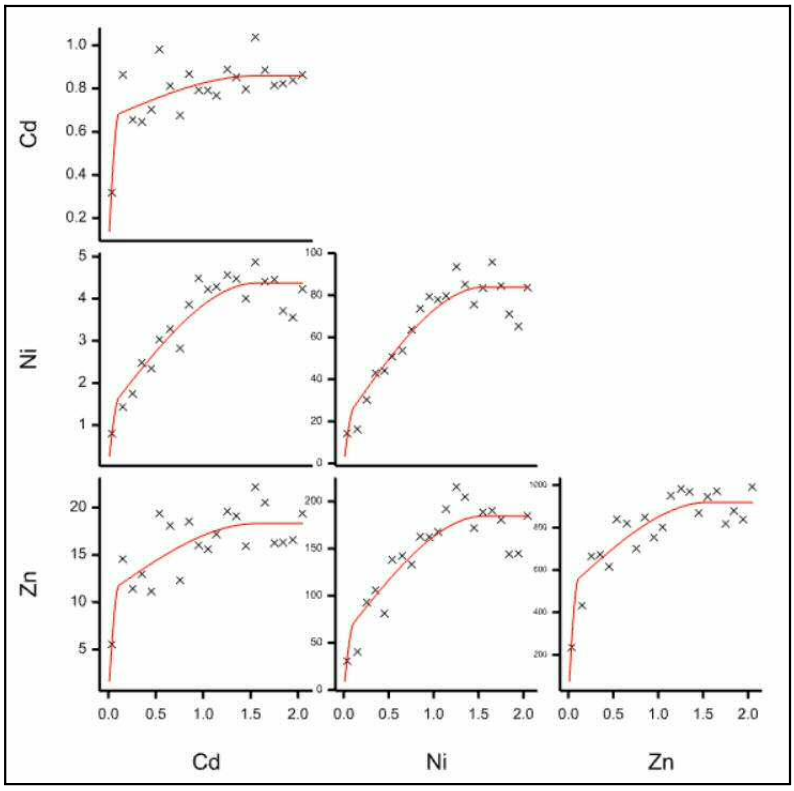


Figure 8.3.8

8.3.9 The COKRIGE directive

COKRIGE directive

Calculates kriged estimates using a model fitted to the sample variograms and cross-variograms of a set of variates.

Options

PRINT = *string token*	Controls printed output (description, search, weights, conditionalprobabilities, quantiles, crossvalidations); default desc
Y = *variate*	Variate to predict in the cokriging
METHOD = *string token*	Type of kriging (Normal, LogNormal); default Norm
X1OUTER = *variate*	Variate containing 2 values to define the bounds of the region to be examined in the first direction; by default the whole region is used

`X2OUTER` = *variate* Variate containing 2 values to define the bounds of the region to be examined in the second direction; by default the whole region is used

`X3OUTER` = *variate* Variate containing 2 values to define the bounds of the region to be examined in the third direction; by default the whole region is used

`X1INNER` = *variate* Variate containing 2 values to define the bounds of the interpolated region in the first direction; no default

`X2INNER` = *variate* Variate containing 2 values to define the bounds of the interpolated region in the second direction; no default

`X3INNER` = *variate* Variate containing 2 values to define the bounds of the interpolated region in the third direction; no default

`X1INTERVAL` = *scalar* Distance between successive interpolations in the first direction; default 1.0

`X2INTERVAL` = *scalar* Distance between successive interpolations in the second direction; default 1.0

`X3INTERVAL` = *scalar* Distance between successive interpolations in the third direction; default 1.0

`POINTS` = *matrix* Allows the point where predictions are required to be specified explicitly if the `X1-3INNER` and `X1-3INTERVAL` options are unset, otherwise if these are set, saves the locations of the prediction points

`BLOCKDIMENSIONS` = *variate* or *matrix* Dimensions of the block(s) in the 3 directions, a variate defines identical blocks for each prediction point, a matrix can be used to define different block sizes for each point when the points are defined by the `POINTS` option; default `!(0,0,0)` i.e. punctual kriging at every point

`POOLRADIUS` = *scalar* Specifies the minimum distance for which points are pooled; default `*` i.e. no pooling

`SEARCHNEIGHBOURHOOD` = *string token* Search neighbourhood to be used (`global`, `local`); default `glob`

`MINPOINTS` = *scalars* Minimum number of data points from which to compute elements

`MAXPOINTS` = *scalars* Maximum number of data points in each direction from which to compute elements

`RADII` = *scalars* or *variates* Scalar defining the maximum distance between target point in block and usable data for each variable in 1 dimension, or radii of the ellipse or ellipsoid enclosing the usable points in 2 or 3 dimensions

`ELLIPSEAXIS` = *scalar* or *variate* Angle or angles defining the direction of the axis of the ellipse or ellipsoid, scalar for 2 dimensions and variate containing 3 values for 3 dimensions

`DRIFT` = *string token* Mean function for universal cokriging (`constant`, `linear`, `quadratic`, `polygon`); default `cons`

`X1EXV` = *variate* Variate containing locations of the explanatory model in the first dimension

`X2EXV` = *variate* Variate containing locations of the explanatory model in the second dimension (if recorded in 2 or 3 dimensions)

X3EXV = *variate*	Variate containing locations of the explanatory model in the third dimension (if recorded in 3 dimensions)
TERMS = *variates*	List of variates for explanatory model; default * i.e. none
POLYGONCOORDINATES = *pointer*	Pointer containing the coordinates of polygons in 2 variates and the map unit numbers within a factor
COORDSYSTEM = *string token*	Coordinate system used for the geometry for discretizing the lag (mathematical, geographical); default math
CPTHRESHOLD = *scalar or variate*	Threshold(s) for calculating the conditional probabilities
PERCENTQUANTILES = *scalar or variate*	Percentage points for which quantiles are required; default 5 and 95
LOGBASE = *string token*	Base of antilog transformation to be applied to the predictions and variances for lognormal (co)kriging (ten, e); default * i.e. none

Parameters

DATA = *variates*	Measurements as one or more variates
X1 = *variates*	Locations of the measurements in the first dimension
X2 = *variates*	Locations of the measurements in the second dimension (if recorded in 2 or 3 dimensions)
X3 = variates	Locations of the measurements in the second dimension (if recorded in 3 dimensions)
PREDICTIONS = *variate*	Kriged estimates
VARIANCES = *variate*	Estimation variances
MEASUREMENTERROR = *scalars*	Variance of measurement error for punctual (co)kriging
ESTIMATES = *pointers*	Estimates for the model structure
CONDITIONALPROBABILITIES = *pointers*	Structure to save conditional probabilities
QUANTILES = *pointers*	Structure to save estimated quantiles
SAMPLESUPPORT = *scalars*	Sampling size (length, area or volume according to the dimensionality of the data) of the data points

The COKRIGE directive computes kriged estimates using a model fitted by MCOVARIOGRAM (8.3.7) to the sample auto- and cross-variograms of a set of variates. These are transferred using the ESTIMATES options of the two directives.

The data are supplied using the DATA, X1, X2 and X3 parameters, as in the FCOVARIOGRAM directive (8.3.6). The target variable to predict is supplied using the Y option. Note that the target variable must also be present in the list of variates supplied with the DATA parameter.

The METHOD option allows you to specify whether to perform Normal or logNormal cokriging. The lognormal setting is only available for punctual cokriging. For logNormal cokriging the LOGBASE option allows you to specify the base of the logarithms (ten or e) for back transforming the kriged predictions and variances.

By default, Genstat uses global prediction where, for each prediction, all the data values are used. However, it is often desirable to use a subset in a (spatial) neighbourhood around the prediction location. This could be for computational reasons, or to assume local first-order stationarity. You can choose whether to use a global or local search using the SEARCHNEIGHBOURHOOD option.

You can select a subset of the data to be considered when forming the cokriging system by specifying the area or volume defined by X1OUTER, X2OUTER and X3OUTER. Each of these

should be set to a variate with two values to define the lower and upper limits in each direction.

You can supply the positions at which the target variable is predicted (estimated) in two ways. The first way is to generate the locations using the X1-3INNER and X1-3INTERVAL options. X1INNER, X2INNER and X3INNER are set to variates with two values to define the lower and upper limits in each direction, and the limits should not lie outside those of X1OUTER, X2OUTER and X3OUTER. X1INTERVAL, X2INTERVAL and X3INTERVAL are set to scalars to define the distance between the successive positions in the first, second and third direction. The intervals should be specified using the same units as the data. You can save the generated locations by supplying an identifier in the POINTS option. The second way is to explicitly supply the points where predictions are required. If the X1-3INNER and X1-3INTERVAL options are unset then you can use the POINTS option to supply a matrix of prediction locations.

By default the cokriging is punctual, i.e. at points that have the same size and shape as the sample support. The BLOCKDIMENSIONS option can be used to specify block cokriging. You can either specify a variate containing the dimensions of the block(s) in the three directions or alternatively supply a matrix defining different block sizes for each point when points are supplied using the POINTS option. For punctual cokriging, you can specify the variance of any measurement error using the MEASUREMENTERROR parameter.

The minimum and maximum number of points used for the kriging are set by the MINPOINTS and MAXPOINTS options, respectively.

The RADII option defines the maximum distance between the target point in a block and usable data. For an isotropic search you should supply a scalar to define the maximum distance or radii of the ellipse (two dimensions) or ellipsoid (three dimensions). For an anisotropic search you should supply the distances for each axis of the ellipse of ellipsoid. For an anisotropic search the angle or angles defining the direction of the axes of the ellipse or ellipsoid for the search are supplied using the ELLIPSEAXIS option. For two dimensions you should supply a scalar containing the angle for the first axis which is measured in degrees, counter-clockwise from East if option COORDSYSTEM is set to mathematical, or clockwise from North if COORDSYSTEM is set to geographical. For three dimensions the first value defines the angle for the first axis which is measured in degrees, counter-clockwise from East if COORDSYSTEM is set to mathematical, or clockwise from North if COORDSYSTEM is set to geographical. The second value defines the dip angle for the first axis (rotation angle around the y-axis) which is measured in degrees up from horizontal. The third value defines the rotation angle of the second and third axis around the first axis (defined by the two previous angles). The POOLRADIUS option allows you to specify a minimum distance for which points can be pooled.

The PRINT option controls the printed output with settings:

description	description of the length, area or volume being kriged and the model that is used,
search	the results of the search for data around each position that is kriged,
weights	the kriging weights at each position,
crossvalidation	cross-validation statistics for punctual cokriging (the cross-validation is calculated by estimating each sample point from the data after excluding the sample value),
conditionalprobabilities	conditional probabilities for the values specified by the CPTHRESHOLD option,
quantiles	quantiles for the values specified by the PERCENTQUANTILES option.

Universal kriging may be invoked by setting the DRIFT option to linear or to quadratic, i.e. to be of order 1 or 2. The default is DRIFT=constant, to give ordinary cokriging. You can include explanatory variables in the mean function by listing explanatory variates with the TERMS option, and their associated coordinates using the X1EXV, X2EXV and X3EXV options. For

two-dimensional cokriging, the DRIFT=polygon option allows you to specify categorical variables defined by one or more closed polygons (map units). The map units and polygons should be supplied in a pointer using the POLYGONCOORDINATES option. The pointer should contain the coordinates of the polygons in two variates (x- and y-positions) and a factor where each level defines a different map unit. If there is more than one polygon within a map unit these should be separated with a row of missing values.

You can specify the sampling support size (length, area or volume) of the data points using the SAMPLESUPPORT parameter.

The PERCENTQUANTILES option can specify percentage values for which to compute quantiles for the conditional distributions. The quantiles can be saved using the QUANTILES parameter.

The CPTHRESHOLD option allows you to specify thresholds for calculating conditional probabilities. The conditional probabilities can be saved using the CONDITIONALPROBABILITIES parameter.

The kriged predictions and variances can be saved using the PREDICTIONS and VARIANCES parameters. If a grid or volume of points has been generated using the X1-3INNER and X1-3INTERVAL options, the corresponding prediction locations can be saved using the POINTS option.

Example 8.3.9 uses COKRIGE to produce predictions for cadmium (Cd) as the target variable. The model estimates are supplied in a pointer saved from the MCOVARIOGRAM directive in Example 8.3.7. The predictions are formed using punctual cokriging. The resulting predictions and estimation variances for cadmium are plotted as shade diagrams in Figures 8.3.9a and 8.3.9b.

Example 8.3.9

```
 30  " Read the locations of the prediction points."
 31  MATRIX       [ROWS=1547; COLUMNS=2] Mpoints
 32  OPEN         '%GENDIR%/Examples/GuidePart2/Mpoints.dat'; CHANNEL=2
 33  READ         [CHANNEL=2] Mpoints

   Identifier   Minimum      Mean   Maximum    Values   Missing
      Mpoints     18.00     38.23     58.20      3094         0

 34  CLOSE        2; FILETYPE=input
 35  " Produce predictions and variances for target variable Cadmium."
 36  COKRIGE  [PRINT=description; Y=Cd; POINTS=Mpoints; RADII=20;\
 37           SEARCHNEIGHBOURHOOD=local] Cd; X1=X1; X2=X2;\
 38           ESTIMATES=Save_est; PREDICTIONS=Predictions;\
 39           VARIANCES=Variances

Co-Kriging of Cd
================

Support Variables:
  Cd 259 support points

Number of points to be kriged:  1547

Number of points required for interpolation

Minimum:       7
Maximum:      20

Number of elementary correlation structures: 3

Correlation structure 1

  Anisotropy:  Isotropic
  Model:       Pure nugget
```

```
Sill variance(s):

           Cd     0.05172
                      Cd

Correlation structure 2

  Anisotropy:  Isotropic
  Model:       Spherical

Distance parameter(s):

  Distance:  0.1096

Sill variance(s):


           Cd      0.6109
                      Cd

Correlation structure 3

  Anisotropy:  Isotropic
  Model:       Spherical

Distance parameter(s):

  Distance:  1.566

Sill variance(s):


           Cd      0.1965
                      Cd
```

```
  40  " Plot the predictions and variances."
  41  VARIATE  [NVALUES=NROWS(Mpoints)] Xpos,Ypos
  42  EQUATE   T(Mpoints); !p(Xpos,Ypos)
  43  GROUPS   [REDEFINE=yes] Xpos,Ypos; FACTOR=Xfac,Yfac; LEVELS=Xlevs,Ylevs
  44  TABULATE [CLASSIFICATION=Yfac,Xfac] Predictions,Variances;\
  45           MEANS=Zvals,Zvars
  46  MATRIX   [ROWS=!(#Ylevs); COLUMNS=!(#Xlevs)] Mpredictions; !(#Zvals)
  47  MATRIX   [ROWS=!(#Ylevs); COLUMNS=!(#Xlevs)] Mvariances; !(#Zvars)
  48  XAXIS    [RESET=yes] 1
  49  YAXIS    [RESET=yes] 1
  50  PEN      2,3; COLOUR='azure','midnightblue'
  51  DSHADE   [TITLE='Cokriged estimates for cadmium in the Swiss Jura';\
  52           YORIENTATION=normal; GRIDMETHOD=*] Mpredictions; PEN=!(2,3)
  53  DSHADE   [TITLE='Cokriging variances for cadmium in the Swiss Jura';\
  54           YORIENTATION=normal; GRIDMETHOD=*] Mvariances; PEN=!(2,3)
```

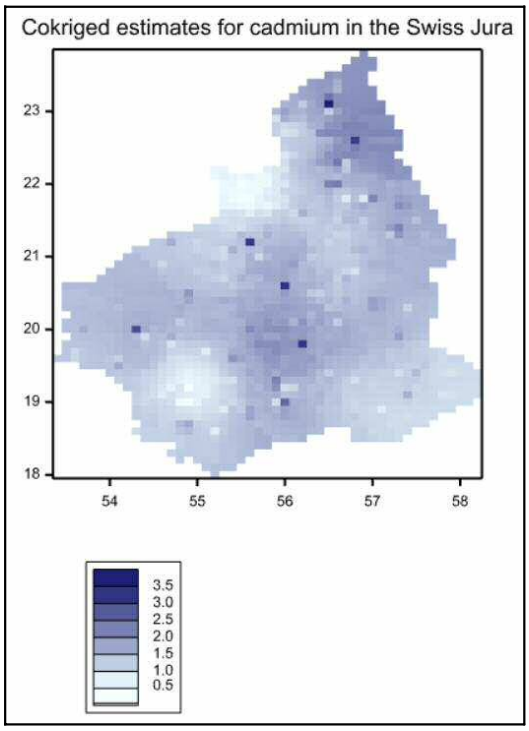


Figure 8.3.9a

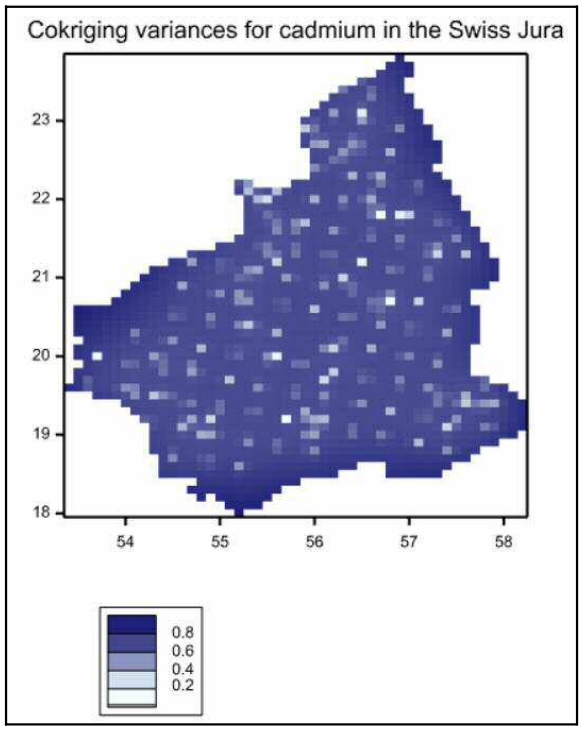


Figure 8.3.9b

8.4 Analysis of spatial point patterns

Spatial point patterns are sets of *n* coordinates (in 2-D) representing locations of some objects of interest (e.g. events, patients, aphids, trees, diseased plants etc). Genstat has several procedures, listed at the start of this chapter, for plotting and manipulating spatial point patterns. Also, if the points were recorded at different times, you can investigate their clustering in both space and time. Some of these procedures use the `PASS` directive (1: 5.7.2) to link to Fortran programs from the Splancs system of Rowlingson & Diggle (1993). This facility may therefore not be available in some Genstat implementations.

The procedures are all described in Part 3 of the *Genstat Reference Manual,* or in Genstat's on-line help. Alternatively, details can be be displayed in any implementation, using procedure `LIBHELP`. You can also obtain an example of the use of any of the procedures, using procedure `LIBEXAMPLE`. These are illustrated in Example 8.4, which shows the help information for procedure `KSTHAT` (see Figure 8.4a), and then runs an example that plots the spatial and temporal K functions (Figures 8.4b and 8.4c).

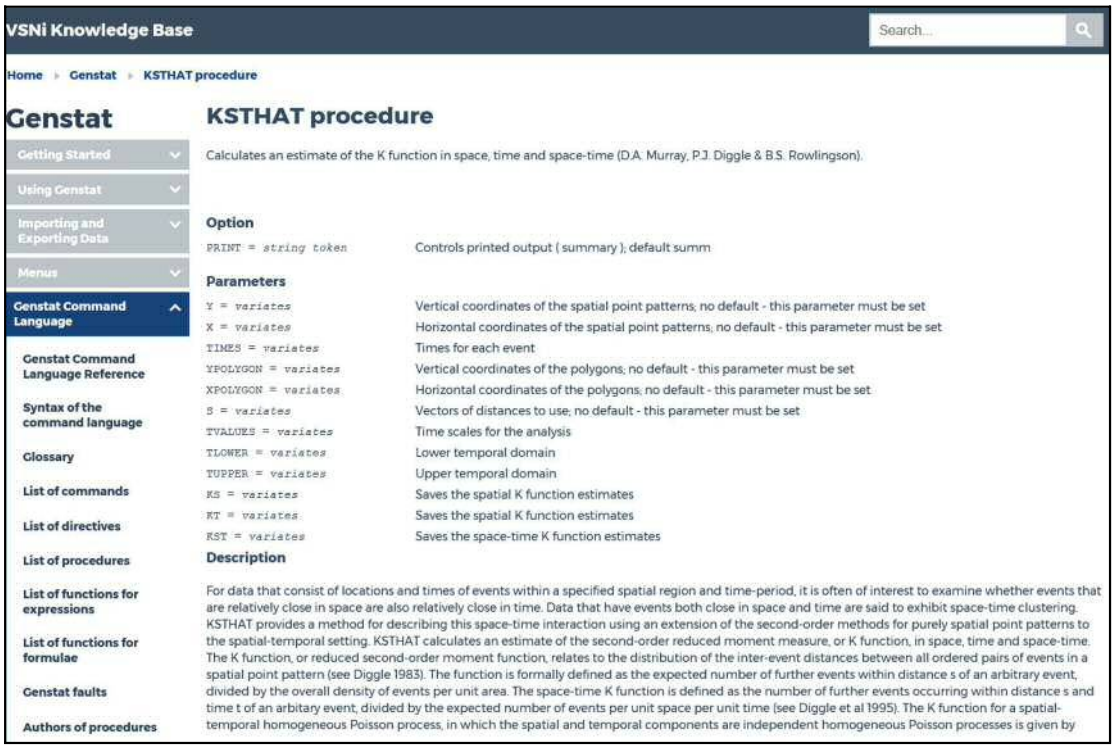


Figure 8.4a

Example 8.4

```
  2  LIBHELP 'KSTHAT'
  3  LIBEXAMPLE 'KSTHAT'; EXAMPLE=KSTex
  4  SET [INPRINT=statements,macros]
  5  ##KSTex
    1  CAPTION   'KSTHAT example'; STYLE=meta

KSTHAT example
==============

     2  VARIATE [NVALUES=188] X
     3  READ X

    Identifier   Minimum      Mean   Maximum    Values   Missing
             X     255.0     286.3     335.0       188         0

    14  VARIATE [NVALUES=188] Y
    15  READ Y

    Identifier   Minimum      Mean   Maximum    Values   Missing
             Y     247.0     338.8     399.0       188         0

    26  VARIATE [NVALUES=188] Times
    27  READ Times

    Identifier   Minimum      Mean   Maximum    Values   Missing
         Times     413.0      3530      5775       188         0

    41  VARIATE [NVALUES=353] Xpoly
    42  READ Xpoly

    Identifier   Minimum      Mean   Maximum    Values   Missing
          Xpoly     246.4     296.6     341.0       353         0

    71  VARIATE [NVALUES=353] Ypoly
    72  READ Ypoly
```

```
 Identifier   Minimum      Mean   Maximum    Values   Missing
      Ypoly     237.6     325.3     419.4       353         0

  101  VARIATE [VALUES=1,3...39] S
  102  VARIATE [VALUES=100,200...1500] T
  103  KSTHAT  Y=Y; X=X; TIMES=Times; YPOLYGON=Ypoly; XPOLYGON=Xpoly; S=S;\
  104          TVALUES=T; TLOWER=400; TUPPER=5800; KS=KS; KT=KT
```

Spatial K function

S	K
1.00	18
3.00	88
5.00	230
7.00	413
9.00	628
11.00	899
13.00	1220
15.00	1562
17.00	1896
19.00	2272
21.00	2665
23.00	3110
25.00	3565
27.00	4023
29.00	4481
31.00	4912
33.00	5369
35.00	5863
37.00	6317
39.00	6824

Temporal K function

T	K
100.0	226
200.0	461
300.0	663
400.0	862
500.0	1053
600.0	1266
700.0	1494
800.0	1736
900.0	1973
1000.0	2189
1100.0	2416
1200.0	2620
1300.0	2827
1400.0	3040
1500.0	3256

Space-time K function

	1	2	3	4	5
1	13560	27120	30510	38985	42375
2	31646	71534	88484	117299	144675
3	72325	163506	212954	272279	330165
4	121059	266984	364034	473568	561964
5	211471	414296	558506	717432	840072
6	321008	601279	798735	999270	1162227
7	401927	764611	1032106	1310538	1570385
8	491360	928899	1247441	1584600	1922218
9	558238	1043774	1463272	1858625	2261568
10	633818	1231797	1723439	2180036	2635022

11	692946	1384710	1936022	2450990	2985651
12	785320	1578652	2207477	2765070	3363766
13	912745	1801883	2503223	3150485	3850284
14	1028906	2045878	2864760	3590862	4373671
15	1169958	2309693	3185024	3999037	4889442
16	1277868	2504828	3484788	4410355	5385120
17	1401769	2722282	3769501	4781656	5839445
18	1540372	3030228	4192284	5288243	6441146
19	1619397	3259043	4524680	5706358	6960270
20	1748590	3556102	4943481	6212366	7592532

	6	7	8	9	10
1	45765	45765	52545	62715	67800
2	151455	163633	200923	231433	253468
3	381748	407486	475286	547241	614455
4	644954	713415	840085	973667	1079950
5	963405	1055615	1245244	1449271	1634507
6	1348989	1508612	1773266	2053339	2283814
7	1842692	2111956	2437802	2788218	3091193
8	2255625	2647771	3074807	3519430	3879727
9	2678507	3158503	3634971	4160086	4567508
10	3108551	3685846	4236599	4840989	5346470
11	3576984	4228887	4879133	5560113	6130294
12	4061343	4805047	5563718	6339193	7035254
13	4653177	5491869	6363403	7265948	8081045
14	5282999	6243677	7220190	8218300	9082406
15	5886555	6971003	8062950	9173806	10177653
16	6480578	7650908	8886269	10098853	11165138
17	7041779	8323368	9715337	11056902	12219495
18	7785361	9223000	10710866	12174099	13433192
19	8446162	9996160	11645932	13222026	14559360
20	9228670	10908768	12731137	14394844	15841520

	11	12	13	14	15
1	67800	67800	71190	71190	76275
2	263638	268723	285673	300927	321267
3	670904	691663	746713	814513	841883
4	1165214	1249650	1347075	1455676	1520817
5	1755366	1869207	2011366	2164866	2277939
6	2447948	2590640	2776083	2993173	3193883
7	3309830	3496445	3760902	4035104	4296265
8	4148178	4365840	4685369	5041273	5374663
9	4904134	5184892	5563122	6022638	6407442
10	5792264	6151752	6589434	7112142	7563974
11	6637467	7056101	7572675	8172865	8698543
12	7636659	8128673	8714670	9381837	9987458
13	8774975	9341165	10009526	10798399	11489539
14	9857284	10561470	11357751	12274112	13093763
15	11028839	11817614	12690811	13701981	14622649
16	12117373	12983612	13926902	15042238	16054708
17	13244569	14171773	15247780	16425534	17511140
18	14570310	15575822	16695009	17983031	19171060
19	15840481	16942006	18132757	19457213	20785120
20	17208996	18417740	19705253	21084172	22515530

```
105  YAXIS   1; TITLE='Estimate K'
106  XAXIS   1; TITLE='distance'
107  DGRAPH  [WINDOW=1; TITLE='Spatial K function'] KS; S
108  XAXIS   1; TITLE='time'
109  DGRAPH  [WINDOW=1; TITLE='Temporal K function'] KT; T
```

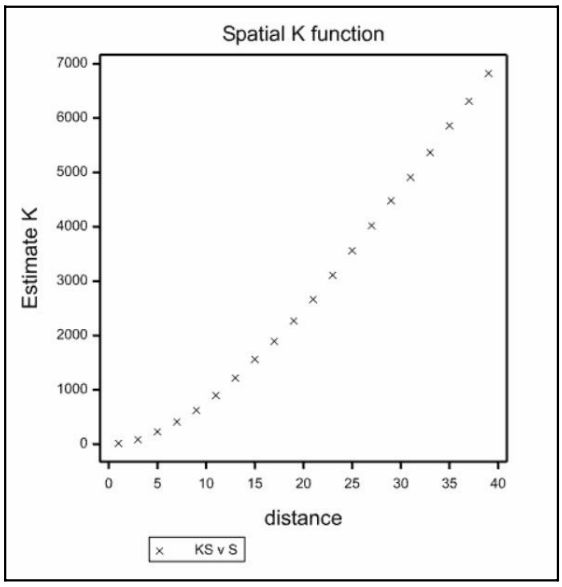


Figure 8.4b

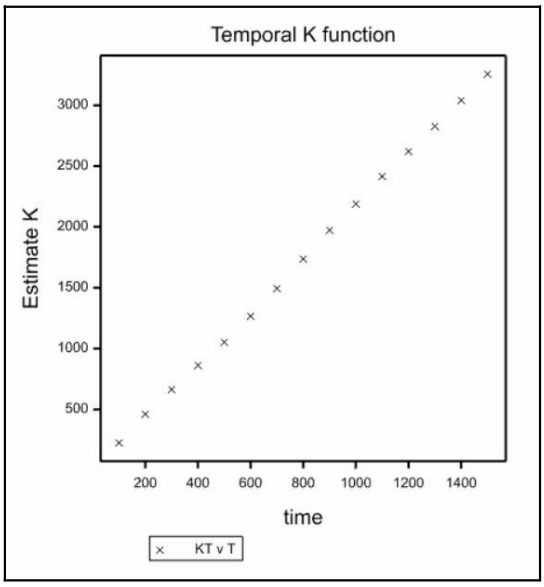


Figure 8.4c

8.5 Clusters of points in multi-dimensional space

You can use the PTFCLUSTERS procedure (8.5.1) to form clusters of points from their densities in multi-dimensional space. The PTFILLCLUSTERS procedure (8.5.2) can then fill any holes within the clusters. These procedures are useful, in particular, for the cluster analysis of very large data sets. For example, the PCPCLUSTER procedure (6.20.3) uses PTFCLUSTERS to form clusters in a space defined by dimensions from a principal coordinates analysis.

PTFCLUSTERS and PTFILLCLUSTERS use two utility procedures: ADJACENTCELLS finds cells adjacent to other cells in a multi-dimensional array, and NEIGHBOURS finds the neighbours of cells in a multi-dimensional array. Details are in Part 3 of the *Genstat Reference Manual,* or in Genstat's on-line help.

8.5.1 The PTFCLUSTERS procedure

PTFCLUSTERS procedure

Forms clusters of points from their densities in multi-dimensional space (R.W. Payne).

Options

PRINT = *string tokens*	What to print (cellclusters, density, summary); default summ
PLOT = *string tokens*	What to plot (cellclusters, density, histogram, summary); default cell, dens, hist
CLUSTERS = *pointer*	Saves variates defining the clusters for each minimum number of points
CELLCLUSTERS = *pointer*	Saves tables containing the clusters of cells for each minimum number of points
DENSITY = *table*	Saves or supplies the table of cell densities
SUMMARY = *pointer*	Saves the summary table
INITIALCELLCLUSTERS = *table*	Defines clusters of cells to use to start the clustering
MINPOINTS = *variate* or *scalar*	Minimum numbers of points within cells at which to form clusters

Parameters

DATA = *variates*	Coordinates of the points
NPARTITIONS = *scalars*	Numbers of partitions in each dimension; default 10

The PTFCLUSTERS procedure forms clusters of points in multi-dimensional space by finding contiguous regions where the density of points exceeds thresholds specified by the MINPOINTS option. The points in these regions will be connected to each other in a similar way to the units in a hierarchical cluster analysis. Note, though, that points in sparsely populated parts of the space will not be allocated to any cluster. These points can be thus be identified as unusual or aberrant.

PTFCLUSTERS divides the space into cells, and uses TABULATE to calculate the number of points in each cell. PTFCLUSTERS starts with the first MINPOINTS value and finds a cell containing more than that number of points. This is the starting point for the first cluster. Additional cells are added to the cluster if they are neighbours of cells in the cluster containing more than that minimum number of points. When this cluster is complete, PTFCLUSTERS looks for a cell that is not in the cluster but which contains more than the minimum number of points. This provides the starting point for another cluster. The process continues until all the cells with more than that minimum number of points have been allocated to a cluster. PTFCLUSTERS then takes the next MINPOINTS value and expands the clusters to contain neighbours with that smaller minimum number of points, merging clusters if they become neighbours. For each MINPOINTS value, PTFCLUSTERS records the number of clusters, the mean number of points within cells inside and outside the clusters, the mean number within cells just inside and just outside the cluster boundaries, the minimum number within cells on the boundaries, and the maximum number for within just outside the boundaries. These should help to assess which MINPOINTS value gives the best set of clusters.

As mentioned above, the MINPOINTS option specifies the minimum numbers of points that are used to form the clusters. The default is to use a list of values calculated as the maximum density multiplied by 0.8, 0,75, 0.7, 0.65, 0.6, 0.55, 0.5, 0.45, 0.4, 0,35, 0.3, 0.25 and 0.2.

The DATA parameter supplies a list of variates containing the coordinates of the points in the various dimensions. The NPARTITIONS parameter supplies a list of scalars indicating the number of partitions to make along each dimension in order to form the multi-dimensional cells; default 10.

The PRINT option controls the printed output, with settings:

cellclusters	shows how the cells are clustered for each minimum number of points,
density	prints the table showing the number of points in each cell,
summary	prints the summary information recorded for each minimum number of points (default).

The PLOT option specifies how the replications are plotted, with settings:

cellclusters	this displays the clustering of the cells for each minimum number of points as a shade plot or as a 3-d graph if there are 2 or 3 dimensions respectively,
density	displays shade plots showing the numbers of points in each pair of dimensions,
histogram	plot a histogram for the numbers of points in the cells,
summary	plots the summary information against the minimum number of points.

The default is to plot all of these.

The CLUSTERS option can save a pointer containing details of the clusters of points formed at each MINPOINTS value. The clusters have integer numbers, from one upwards. The pointer contains a variate for each MINPOINTS value with a unit for every point. These contain either

cluster numbers, or missing values for points in cells that have not been allocated to any cluster.

The CELLCLUSTERS option can similarly save a pointer containing details of the clusters of cells formed at each MINPOINTS value. The pointer contains a table for each MINPOINTS value. These contain either a cluster number, or missing values for cells that have not been allocated to any cluster.

The DENSITY option can save the table containing the number of points within each cell. Alternatively, if you do not set the DATA parameter, it can be used to supply a previously calculated density table. This is useful to save computing time if you want to make several attempts to find clusters with the same data set.

The SUMMARY option can save the summary table, in a pointer with elements labelled 'Min. no. points', 'No. clusters', 'Mean inside clusters', 'Mean outside clusters', 'Mean on boundary', 'Mean outside boundary', 'Min. on boundary' and 'Max. outside boundary'.

The INITIALCELLCLUSTERS option can supply a table of cell cluster allocations, to act as a starting point for the clustering. For example, you could specify a table previously saved by the CELLCLUSTERS option, if you wanted to expand those clusters with some different values of MINPOINTS.

A final ,important point is that PTFCLUSTERS does not form a units-by-units similarity matrix, as in ordinary cluster analysis, but instead works with a (small) density table. It is therefore suitable for clustering(very) large data sets.

Example 8.5.1 uses the GRSCR procedure to generate some random points: first some points scattered over the plane (lines 4-5), then two overlapping polygons (lines 6-9), and then a separate polygon (lines 10-11). Figure 8.5.1a shows their distribution in the plane. PTFCLUSTERS partitions the plane into an 8 × 8 grid of cells and forms these into clusters.

The summary from the first clustering, in lines 15-16, suggests that the best clusters have not yet been found. There is still a fairly small difference between the minimum number of points in cells on the boundary and the maximum number of points in cells just outside the boundary. So, in lines 17-19, PTFCLUSTERS uses the density table and the clustering of cells with 39 as the minimum of points (saved in line 16) to continue the process with 39 down to 10 as the minimum number of points. Some of these minimum numbers of points produce identical clusters to the previous number and, in particular, it looks as though improvements have stopped at 20. So, in lines 20-22, the clustering of cells with 20 minimum points is defined as the input cluster of cells, and MINPOINTS is set only to 20 so that no further clustering takes place. Lines 20-22 therefore simply print and plot the clusterings of the cells (Figure 8.5.1b), and save the clusters of the points in the variate clust[20].

Example 8.5.1

```
 2  VARIATE     xhexagon; VALUES=!(0.3,0.0,0.3,0.7,1.0,0.7)
 3  &           yhexagon; VALUES=!(0.0,0.5,1.0,1.0,0.5,0.0)
 4  GRCSR       [PRINT=*] YPOLYGON=yhexagon; XPOLYGON=xhexagon;\
 5              NPOINTS=100; YCSR=ycsr[0]; XCSR=xcsr[0]; SEED=35719
 6  GRCSR       [PRINT=*] YPOLYGON=yhexagon/2; XPOLYGON=xhexagon/2;\
 7              NPOINTS=600; YCSR=ycsr[1]; XCSR=xcsr[1]
 8  GRCSR       [PRINT=*] YPOLYGON=0.4+yhexagon/2; XPOLYGON=xhexagon/2;\
 9              NPOINTS=700; YCSR=ycsr[2]; XCSR=xcsr[2]
10  GRCSR       [PRINT=*] YPOLYGON=yhexagon/3; XPOLYGON=0.65+xhexagon/3;\
11              NPOINTS=800; YCSR=ycsr[3]; XCSR=xcsr[3]
12  VARIATE     [VALUES=#xcsr[]] xvar
13  VARIATE     [VALUES=#ycsr[]] yvar
14  DPTMAP      [YLOWER=0; YUPPER=1; XLOWER=0; XUPPER=1] Y=yvar; X=xvar
15  PTFCLUSTERS [PRINT=density,summary; PLOT=*; CLUSTERS=clust;\
16            CELLCLUSTERS=cellclust; DENSITY=density] yvar,xvar; NPARTITIONS=8
```

```
Clustering based on densities of points in 2 dimensions
=======================================================


             density
        xvar     1     2     3     4     5     6     7     8
        yvar
           1     1    32    35     6     4    31   125    42
           2    20    68    70    43     1   106   195   119
           3    33    58    59    50     1    29   130    34
           4     5   105   108    30     2     3     3     2
           5    24    73    66    44     3     4     2     2
           6    36    74    87    61     0     3     0     0
           7     7    62    60    27     3     1     0     0
           8     0     1     3     1     5     1     0     0


* MESSAGE: clustering with 146 is identical to the previous minimum number.
* MESSAGE: clustering with 137 is identical to the previous minimum number.
* MESSAGE: clustering with 88 is identical to the previous minimum number.

Summary
-------

(Minimum numbers of points where there was no change are omitted.)

   Minimum     Number       Mean       Mean       Mean       Mean        Min        Max
    number         of    inside    outside         on    outside         on    outside
 of points   clusters  clusters  clusters   boundary   boundary   boundary   boundary
       156          1     195.0      31.83      195.0      77.00        195        130
       127          1     162.5      30.24      162.5      49.40        130        125
       117          1     142.2      27.18      142.2      31.25        119        106
       107          2     135.4      25.81      135.4      45.94        108        106
        97          2     126.9      23.02      126.9      29.60        105         73
        78          3     121.9      21.87      121.9      37.82         87         74
        68          3     105.0      18.08      105.0      31.69         68         66
        58          2      90.3      12.48       90.3      17.31         58         50
        49          2      88.2      11.64       88.2      14.94         50         44
        39          2      82.0       9.40       84.0      11.97         43         36

  17  PTFCLUSTERS [PRINT=summary; PLOT=*; CLUSTERS=clust; CELLCLUSTERS=cellclust;\
  18              DENSITY=density; INITIALCELLCLUSTERS=cellclust[39];\
  19              MINPOINTS=!(39...10)]

Clustering based on densities of points in 2 dimensions
=======================================================

* MESSAGE: clustering with 38 is identical to the previous minimum number.
* MESSAGE: clustering with 37 is identical to the previous minimum number.
* MESSAGE: clustering with 28 is identical to the previous minimum number.
* MESSAGE: clustering with 26 is identical to the previous minimum number.
* MESSAGE: clustering with 25 is identical to the previous minimum number.
* MESSAGE: clustering with 23 is identical to the previous minimum number.
* MESSAGE: clustering with 22 is identical to the previous minimum number.
* MESSAGE: clustering with 21 is identical to the previous minimum number.
* MESSAGE: clustering with 19 is identical to the previous minimum number.
* MESSAGE: clustering with 18 is identical to the previous minimum number.
* MESSAGE: clustering with 17 is identical to the previous minimum number.
* MESSAGE: clustering with 16 is identical to the previous minimum number.
* MESSAGE: clustering with 15 is identical to the previous minimum number.
* MESSAGE: clustering with 14 is identical to the previous minimum number.
* MESSAGE: clustering with 13 is identical to the previous minimum number.
* MESSAGE: clustering with 12 is identical to the previous minimum number.
* MESSAGE: clustering with 11 is identical to the previous minimum number.
* MESSAGE: clustering with 10 is identical to the previous minimum number.

Summary
-------
```

```
(Minimum numbers of points where there was no change are omitted.)
```

Minimum number of points	Number of clusters	Mean inside clusters	Mean outside clusters	Mean on boundary	Mean outside boundary	Min on boundary	Max outside boundary
39	2	82.05	9.405	83.95	11.969	43	36
36	2	80.04	8.756	81.77	11.194	36	35
35	2	78.17	8.100	79.74	10.400	35	34
34	2	76.40	7.436	76.04	9.586	34	33
33	2	74.73	6.763	74.25	8.750	33	32
32	2	73.15	6.081	72.56	7.889	32	31
31	2	71.64	5.389	68.80	7.000	31	30
30	2	70.21	4.686	65.96	6.080	30	29
29	2	68.83	3.971	58.74	5.000	29	27
27	2	67.48	3.273	56.13	4.120	27	24
24	2	66.12	2.625	54.79	3.292	24	20
20	2	64.73	2.065	53.40	2.565	20	7

```
  20  PTFCLUSTERS [PRINT=cellclusters; PLOT=cellclusters; CLUSTERS=clust;\
  21              DENSITY=density; INITIALCELLCLUSTERS=cellclust[20];\
  22              MINPOINTS=20]

Clustering based on densities of points in 2 dimensions
=======================================================


Minimum number of points 20
---------------------------

          xvar   1   2   3   4   5   6   7   8
          yvar
             1   *   2   2   *   *   1   1   1
             2   2   2   2   2   *   1   1   1
             3   2   2   2   2   *   1   1   1
             4   *   2   2   2   *   *   *   *
             5   2   2   2   2   *   *   *   *
             6   2   2   2   2   *   *   *   *
             7   *   2   2   2   *   *   *   *
             8   *   *   *   *   *   *   *   *
```

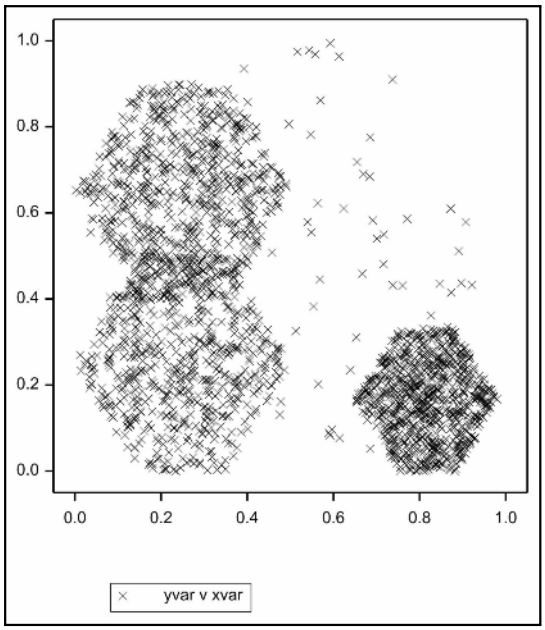


Figure 8.5.1a

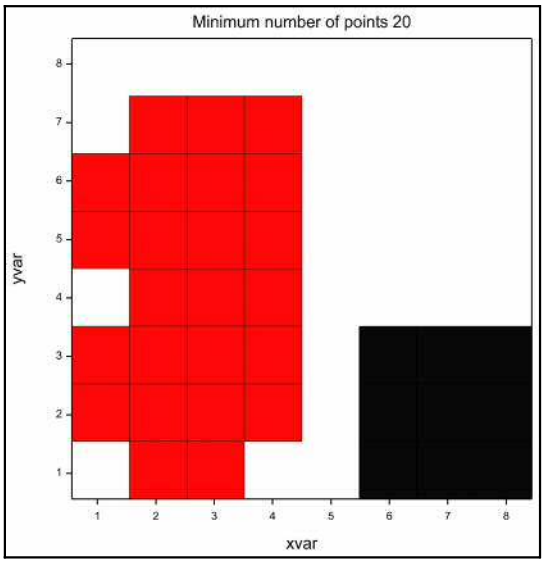


Figure 8.5.1b

8.5.2 The PTFILLCLUSTERS procedure

PTFILLCLUSTERS procedure

Fills holes within clusters of points in multi-dimensional space (R.W. Payne).

Options

PRINT = *string tokens*	Controls printed output (cellclusters); default * .e. none
DIAGONALS = *string token*	Whether to include diagonal cells (include, exclude); default incl
DISTANCE = *scalar*	Maximum distance between cells and adjacent cells; default 1
NUNCLASSIFIED = *scalar*	How many adjacent cells may be unclassified; default 0
NNEWCELLS = *scalar*	Saves the number of cells that have been added to clusters

Parameters

CELLCLUSTERS = *tables*	Clusters of cells containing holes to be filled
NEWCELLCLUSTERS = *tables*	Clusters with filled holes; if unset, the CELLCLUSTERS table itself is updated

The PTFILLCLUSTERS procedure can be used to fill holes within the clusters produced by the PTFCLUSTERS procedure (85.1). That procedure partitions a multi-dimensional space into cells, and then clusters the cells according to the density of the points that they contain. The clusterings of cells can be saved in a table classified by factors indexing the dimensions of the space. This contains either a cluster number, or missing values for cells that have not been allocated to any cluster. PTFILLCLUSTERS finds unallocated cells that are adjacent to the cells of a cluster, and allocates them to that cluster.

The DISTANCE option specifies how close the cells need to be for them to be classed as adjacent. The default of one indicates that they must be alongside each other. Setting DISTANCE=2 means that there can be an intervening cell, and so on. The default is to include cells that are diagonal to each other, but you can set option DIAGONALS=exclude to exclude these. The NUNCLASSIFIED option specifies how many cells adjacent to an unclassified cell may also be unclassified. This means that cells that are not completely surrounded by a cluster (like cells in am indentation at the edge of the cluster) can still be allocated to the cluster.

The CELLCLUSTERS parameter specifies the tables containing holes to be filled. The new tables can be saved using the NEWCELLCLUSTERS parameter. If this is unset, the CELLCLUSTERS tables are updated. The NNEWCELLS option can save a scalar containing the number of unallocated cells that now belong to a cluster. You can print the updated tables by setting option PRINT=cellclusters.

This is illustrated in Example 8.5.2. In lines 21-22, PTFILLCLUSTERS allocates the cell in row 1 of column 10 to cluster 1, and the cell in row 9 of column 3 to cluster 2. In lines 23-24, diagonal cells are excluded (i.e. ignored) and so the cell in row 13 of column 8 can be allocated to cluster 4. In lines 25-26, cells up to a distance of 2 are considered, but there can be one cell in that zone that is not allocated to a cluster. As a result the cells in rows 5 and 7 of column 3, and the cell in row 15 of column 1, can now be allocated to cluster 2.

Example 8.5.2

```
2  FACTOR        [LEVELS=15] rows
3  &             [LEVELS=10] columns
4  TABLE         [CLASS=rows,columns] cellclusters
```

```
  5  READ          [PRINT=data,errors] cellclusters

  6     *   *   *   *   *   *   *   1   1   *
  7     *   *   *   3   3   *   *   1   1   1
  8     *   *   *   *   *   *   *   1   1   *
  9     *   2   2   2   *   *   *   *   *   *
 10     *   2   *   2   *   *   *   *   *   *
 11     *   2   *   2   *   *   *   *   *   *
 12     *   2   *   2   *   *   *   *   *   *
 13     *   2   2   2   *   *   *   *   *   *
 14     *   2   *   2   *   *   *   *   *   *
 15     *   2   2   2   *   *   *   *   *   *
 16     *   *   2   2   2   *   *   *   *   *
 17     *   2   2   2   2   *   *   4   *   *
 18     *   2   2   2   2   *   4   *   4   *
 19     *   2   *   *   2   *   *   4   *   *
 20     *   2   *   *   2   *   *   *   *   *   :
 21  PTFILLCLUSTERS [PRINT=cellclusters]\
 22                 cellclusters; NEWCELLCLUSTERS=newclus

Clusters with holes filled
--------------------------

Negative cluster numbers are printed in the filled holes.

            Clusters
     columns   1   2   3   4   5   6   7   8   9  10
        rows
           1   *   *   *   *   *   *   *   1   1  -1
           2   *   *   *   3   3   *   *   1   1   1
           3   *   *   *   *   *   *   *   1   1   *
           4   *   2   2   2   *   *   *   *   *   *
           5   *   2   *   2   *   *   *   *   *   *
           6   *   2   *   2   *   *   *   *   *   *
           7   *   2   *   2   *   *   *   *   *   *
           8   *   2   2   2   *   *   *   *   *   *
           9   *   2  -2   2   *   *   *   *   *   *
          10   *   2   2   2   *   *   *   *   *   *
          11   *   *   2   2   2   *   *   *   *   *
          12   *   2   2   2   2   *   *   4   *   *
          13   *   2   2   2   2   *   4   *   4   *
          14   *   2   *   *   2   *   *   4   *   *
          15   *   2   *   *   2   *   *   *   *   *


 23  PTFILLCLUSTERS [PRINT=cellclusters; DIAGONALS=exclude]\
 24                 cellclusters; NEWCELLCLUSTERS=newclus

Clusters with holes filled
--------------------------

Negative cluster numbers are printed in the filled holes.

            Clusters
     columns   1   2   3   4   5   6   7   8   9  10
        rows
           1   *   *   *   *   *   *   *   1   1  -1
           2   *   *   *   3   3   *   *   1   1   1
           3   *   *   *   *   *   *   *   1   1   *
           4   *   2   2   2   *   *   *   *   *   *
           5   *   2   *   2   *   *   *   *   *   *
           6   *   2   *   2   *   *   *   *   *   *
           7   *   2   *   2   *   *   *   *   *   *
           8   *   2   2   2   *   *   *   *   *   *
           9   *   2  -2   2   *   *   *   *   *   *
          10   *   2   2   2   *   *   *   *   *   *
          11   *   *   2   2   2   *   *   *   *   *
          12   *   2   2   2   2   *   *   4   *   *
          13   *   2   2   2   2   *   4  -4   4   *
          14   *   2   *   *   2   *   *   4   *   *
          15   *   2   *   *   2   *   *   *   *   *
```

```
  25  PTFILLCLUSTERS [PRINT=cellclusters; DISTANCE=2; NUNCLASSIFIED=1]\
  26                 cellclusters; NEWCELLCLUSTERS=newclus

Clusters with holes filled
--------------------------

Negative cluster numbers are printed in the filled holes.

             Clusters
      columns   1   2   3   4   5   6   7   8   9  10
         rows
             1   *   *   *   *   *   *   *   1   1  -1
             2   *   *   *   3   3   *   *   1   1   1
             3   *   *   *   *   *   *   *   1   1   *
             4   *   2   2   2   *   *   *   *   *   *
             5   *   2  -2   2   *   *   *   *   *   *
             6   *   2   *   2   *   *   *   *   *   *
             7   *   2  -2   2   *   *   *   *   *   *
             8   *   2   2   2   *   *   *   *   *   *
             9   *   2  -2   2   *   *   *   *   *   *
            10   *   2   2   2   *   *   *   *   *   *
            11   *   *   2   2   2   *   *   *   *   *
            12   *   2   2   2   2   *   *   4   *   *
            13   *   2   2   2   2   *   4   *   4   *
            14   *   2   *   *   2   *   *   4   *   *
            15  -2   2   *   *   2   *   *   *   *   *
```

References

Chapter 2

Aitchison J.A. (1986). *The Statistical Analysis of Compositional Data*. Chapman & Hall, London.

Altman, D.G. & Bland, J.M. (1983). Measurement in medicine: the analysis of method comparison studies. *Statistician*, **32**, 307–317.

Andrews D.F., Gnanadesikan R. & Warner J.L. (1973). Methods for assessing multivariate normality. In: *Multivariate Analysis III* (ed. P.R. Krishnaiah) 95-116.Academic Press, New York.

Arimitage, P., Berry, G. & Matthews, J.N.S. (1994). *Statistical Methods in Medical Research*. Blackwell Science, Oxford.

Bland, J.M. & Altman, D.G. (1986). Statistical methods for assessing agreement between two methods of clinical measurement. *Lancet*, **i**, 307–310.

Bland J.M. & Altman D.G. (1995). Comparing methods of measurement – why plotting difference against standard method is misleading. *Lancet*, **346**, 1085–1087.

Bland, J. M. & Altman, D. G. (1999). Measuring agreement in method comparison studies. *Statistical Methods in Medical Research*, **8**, 135–160.

Bland, J.M. & Altman, D.G. (2007). Agreement between methods of measurement with multiple observations per individual. *Journal of Biopharmaceutical Statistics*, **17**, 571–582.

Bliss, C.I. (1953). Fitting the negative binomial distribution to biological data. *Biometrics*, **9**, 176-196.

Carvalho, L. (2015). An improved evaluation of Kolmogorov's distribution. *Journal of Statistical Software*, **65**(3), 1-7.

Chao, A. (1987). Estimating the population size for capture-recapture data with unequal catchability. *Biometrics*, **43**, 783-791.

Collett, D. (1991). *Modelling Binary Data*. Chapman & Hall, London.

Colwell, R.K., Mao, C.X. & Chang, J. (2004). Interpolating, extrapolating comparing incidence-based species accumulation curves. *Ecology*, **85**, 2717-2727.

Conover, W.J. (1971). *Practical Nonparametric Statistics*. Wiley, New York.

Csörgő, S. & Faraway, J.J. (1996). The exact and asymptotic distributions of Cramér-von Mises statistics. *Journal of the Royal Statistical Society, Series B*, **58**, 221-234.

Eid, M.T., Black, C.A., Kempthorne, O. & Zoellner, J.A. (1954). Significance of soil organic phosphorus to plant growth. *Iowa Agricultural Experiment Station Research Bulletin* **406.**

Fisher, N.I. (1993). *Statistical Analysis of Circular Data*. Cambridge University Press, Cambridge.

Heck, K.L., van Belle, G. & Simberloff, D. (1975). Explicit calculation of the rarefaction diversity measurement and the determination of sufficient sample size. *Ecology*, **56**, 1459-1461.

Helshe, J.F. & Forrester, N.E. (1983). Estimating species richness using the jackknife procedure. *Biometrics*, **36**, 1-19.

Hoffman, M.S. (editor) (1992). *World Almanac and Book of Facts*. Pharos Books, New York.

Hogg, R. V. & Klugman, S. A. (1984). *Loss Distributions*. John Wiley & Sons, New York.

Johnson, N.L. & Kotz, S. (1969). *Discrete Distributions*. Houghton Mifflin, Boston, Massachussets.

Johnson, N. L., Kotz, S. & Balakrishnan N. (1994). *Continuous Univariate Distributions, Volume 1, 2nd edition*. John Wiley & Sons, New York.

Johnson, N. L., Kotz, S. & Balakrishnan N. (1995). *Continuous Univariate Distributions, Volume 2, 2nd edition*. John Wiley & Sons, New York.

Jones, M.C., Marron, J.S. & Sheather, S.J. (1996). Progress in data-based bandwidth selection for kernel density estimation. *Computational Statistics*, **11**, 337-381.

Kempton, R.A. & Taylor, L.R. (1974). Log-series and log-normal parameters as diversity determinants for the Lepidoptera. *Journal of Animal Ecology*, **43**, 381-399

Krouwer, J.S. (2008). Why Bland–Altman plots should use X, not (Y+X)/2 when X is a reference method. *Statistics in Medicine*, **27**, 778–780.

Landis J,L., Heyman, E,R. & Koch, G.G. (1978). Average Partial Association in Three-way Contingency Tables: a Review and Discussion of Alternative Tests. *International Statistical Review*, **46**, 237-254.

Lin, L.I. (1989). A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, **45**, 255-268.

Lin, L.I. (2000). A note on the concordance correlation coefficient. *Biometrics*, **56**, 324-325.

Magurran, A.E. (2003). *Measuring Biological Diversity*. Blackwell, Oxford.

Mantel N. & Haenszel W. (1959). Statistical Aspects of the Analysis of Data From Retrospective Studies

of Disease. *Journal National Cancer Institute*, **22(4)**, 719-748.
Marsaglia, G. & Marsaglia, J. (2004). Evaluating the Anderson-Darling distribution. *Journal of Statistical Software*, **9**(2), 1-5.
McCullagh, P. & Nelder, J.A. (1989). *Generalized Linear Models (second edition)*. Chapman & Hall, London.
Michael, J. R. (1983). The stabilized probability plot. *Biometrika*, **83**, 11-17.
Montgomery, D.C. (1985). *Introduction to Statistical Process Control*. Wiley, New York.
Nelson, L.S. (1982). Control charts. In: *Encyclopedia of Statistical Sciences* (ed. S. Kotz, N.L. Johnson & C.B. Read), Volume 2, 176-183. Wiley, New York.
Ross, G.J.S. (1987). *Maximum Likelihood Program*. Numerical Analysis Group, Oxford.
Ross, G.J.S. (1990). *Nonlinear Estimation*. Springer-Verlag, New York.
Sheather, S.J. & Jones, M.C. (1991). A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society, Series B*, **53**, 683-690.
Shewhart, W.A. (1931). *Economic Control of Quality of Manufactured Product*. Van Nostrand, New York.
Siegel, S. (1956). *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill, New York.
Silverman, B.W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman & Hall. London
Smith, E.P. & van Belle, G. (1984). Nonparametric estimation of species richness. *Biometrics*, **40**, 119-129.
Smith, F.B. & Brown, P.E. (1933). The diffusion of carbon dioxide through soils. *Soil Science* **35**, 413-421.
Snedecor, G.W. & Cochran, W.G. (1989). *Statistical Methods (eighth edition)*. Iowa State University Press, Ames, Iowa.
Steel, R.G.D. (1959). A multiple comparison rank sum test: treatments versus control. *Biometrics*, **15**, 560-572.
Stephens M.A. (1974). EDF statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association*, **69**, 730-737.
Tokeshi, M. (1993). Species Abundance Patterns and Community Structure. *Advances in Ecological Research*, **24**, 111-186.
Tokeshi, M. (1996). Power fraction: a new explanation of relative abundance patterns in species-rich assemblages. *Oikos*, **75**, 543-550.
Tufte, E.R. (1983). *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, Connecticut.
Tukey, J.W. (1977). *Exploratory Data Analysis*. Addison-Wesley, Reading, Massachussets.
Wang, J., Tsang, W.W. & Marsaglia, G. (2003). Evaluating of Kolmogorov's distribution. *Journal of Statistical Software*, **8**(18), 1-4.
Welch, B.L. (1947).The generalization of 'Student's' problem when several different population variances are involved. *Biometrika*, **34**, 28-35.
Wilson, J.B. (1991). Methods for fitting dominance/diversity curves. *Journal of Vegetation Science*, **2**, 35-46
Youden, W.J. & Beale, H.P. (1934). A statistical study of the local lesion method for estimating tobacco mosaic virus. *Contributions from Boyce Thompson Institute*, **6**, 437-454.
Zhang (2002). Powerful goodness-of-fit tests based on the likelihood ratio. *Journal of the Royal Statistical Society, Series B*, **64**, 281-294.

Chapter 3

Bouvier, A., Gelis, F., Huet, S., Messean, A. & Neveu, P. (1985). *CS-NL*. Laboratoire de Biometrie, INRA-CNRZ, Jouy-en-Josas, France.
Breslow, N.E. & Clayton, D.G. (1993). Approximate inference in generalized linear mixed models. *Journal of the American Statistical Association*, **88**, 421, 9-25.
Brown, M.B. (1976). Screening effects in multidimensional contingency tables. *Applied Statistics*, **25**, 37-46.
Carr, N.L. (1960). Kinetics of catalytic isomerization of *n*-Pentane. *Industrial and Engineering Chemistry*, **52**, 391-396.
Cleveland, W.S. (1979). Robust Locally Weighted Regression and Smoothing Scatterplots. *Journal of the American Statistical Association*, **74**, 829-836.
Cleveland, W.S. & Grosse, E. (1991). Computational Methods for Local Regression. *Statistics and Computing*, **1**, 47-62.
Cleveland, W.S. & Devlin, S.J. (1988). Locally-weighted regression: an approach to regression analysis by local fitting. *Journal of the American Statistical Association*, **83**, 596-610.

Cleveland, W.S., Devlin, S.J. & Grosse, E. (1988). Regression by local fitting: methods, properties, and computing. *Journal of Econometrics*, **37**, 87-114.
Cook, R.D. & Weisberg, S. (1982). *Residuals and Influence in Regression*. Chapman & Hall, New York.
Dobson, A.J. (1990). *An Introduction to Generalized Linear Models*. Chapman & Hall, London.
Draper, N.R. & Smith, H. (1981). *Applied Regression Analysis* (second edition). Wiley, New York.
Finney, D.J. (1971). *Statistical Method in Biological Assay (third edition)*. Griffin, London.
Flack, V.F. & Chang, P.C. (1987). Frequency of selecting noise variables in subset regression analysis: a simulation study. *The American Statistician*, **41**, 84-86.
Forbes, J.D. (1857). Further experiments and remarks on the measurement of heights by the boiling point of water. *Transactions of the Royal Society of Edinburgh*, **21**, 235-243.
Goorin, A.M., Perez-Atayde, A., Gebhardt, M., Andersen, J.W., Wilkinson, R.H., Delorey, M.J., Watts, H., Link, M., Jaffe, N., Frei, E. III & Abelson, H.T. (1987). Weekly high-dose Methotrexate and Doxorubicin for Osteosarcoma: the Dana-Farber Cancer Institute/The Children's Hospital – Study III. *Journal of Clinical Oncology*, **5**, 1178-1184.
Grewal, R.S. (1952). A method for testing analgesics in mice. *British Journal of Pharmacology and Chemotherapy*, **7**, 433-437.
Hastie, T.J. & Tibshirani, R.J. (1990). *Generalized Additive Models*. Chapman & Hall, London.
Kenward, M.G., Lesaffre, E. & Molenberghs, G. (1994). An application of maximum likelihood and generalized estimating equations to the analysis of ordinal data from a longitudinal study with cases missing at random. *Biometrics*, **50**, 945-953.
Kenward, M.G. & Smith, D.M. (1995a). Computing the generalized estimating equations for repeated measurements. *Genstat Newsletter*, **32**, 50-62.
Kenward, M.G. & Smith, D.M. (1995b). Computing the generalized estimating equations for repeated ordinal, categorical measurements. *Genstat Newsletter*, **32**, 63-70.
Koenker, R. (2005). *Quantile Regression*. Cambridge University Press, New York.
Koenker, R.W. & D'Orey, V. (1987). Algorithm AS229 computing regression quantiles. *Applied Statistics*, **36**, 383-393.
Lane, P.W. & Hastie, T.J. (1992). Providing for the analysis of generalized additive models within a system already capable of generalized linear and nonlinear regression. *Computational Statistics: Proceedings of the 10th Symposium on Computational Statistics, Volume 1*, 391-396. Physica-Verlag, Heidelberg.
Lane, P.W. & Nelder, J.A. (1982). Analysis of covariance and standardization as instances of prediction. *Biometrics*, **38**, 613-621.
Lee, Y., & Nelder, J.A. (1996). Hierarchical generalized linear models (with discussion). *Journal of the Royal Statistical Society, Series B*, **58**, 619-678.
Lee, Y., & Nelder, J.A. (2001a). Hierarchical generalized linear models: a synthesis of generalised linear models, random-effect models and structured dispersions. *Biometrika*, **88**, 987-1006.
Lee, Y. & Nelder, J.A. (2001b). Modelling and analysing correlated non-normal data. *Statistical Modelling*, **1**, 3-16.
Lee, Y. & Nelder, J.A. (2006). Double hierarchical generalized linear models (with discussion). *Appl. Statist.*, **55**, 139-185.
Lee, Y., Nelder, J.A. & Pawitan, Y., (2006). *Generalized Linear Models with Random Effects: Unified Analysis via H-likelihood*. CRC Press, London.
Liang, K.-Y., Zeger, S.L. & Qaqish, B. (1992). Multivariate regression analyses for categorical data (with discussion). *Journal of the Royal Statistical Society, Series B*, **54**, 3-40.
Liang, K.-Y. & Zeger,S.L. (1986). Longitudinal data analysis using generalized linear models. *Biometrika*, **73**, 13-22.
Martin, J.T. (1940). The problem of the evaluation of Rotenone-containing plants. V. The relative toxicities of different species of derris. *Annals of Applied Biology*, **27**, 274-294.
McCullagh, P. & Nelder, J.A. (1989). *Generalized Linear Models (second edition)*. Chapman & Hall, London.
Miller, A.J. (1990). *Subset Selection in Regression*. Chapman & Hall, London.
Payne, R.W. (2014). Hierarchical generalized nonlinear models. In: *Statistical Modelling in Biostatistics and Bioinformatics* (ed. G. MacKenzie & D. Peng), 111-124. Springer, New York.
Ratkowsky, D.A. (1983). *Nonlinear Regression Analysis*. Dekker, New York.
Ratkowsky, D.A. (1990). *Handbook of Nonlinear Regression Models*. Dekker, New York.
Ross, G.J.S. (1987). *Maximum Likelihood Program*. Numerical Analysis Group, Oxford.
Ross, G.J.S. (1990). *Nonlinear Estimation*. Springer-Verlag, New York.

Schall, R. (1991) Estimation in generalized linear models with random effects. *Biometrika*, **78**, 719-727.
Seber, G.A.F. (1977). *Linear Regression Analysis*. Wiley, New York.
Seber, G.A.F. & Wild, C.J. (1989). *Nonlinear Regression*. Wiley, New York.
Snedecor, G.W. & Cochran, W.G. (1989). *Statistical Methods (eighth edition)*. Iowa State University Press, Ames.
Sochett, E.B., Daneman, D., Clarson, C. & Ehrlich, R.M. (1987). Factors affecting and patterns of residual insulin secretion during the first year of Type 1 (insulin-dependent) diabetes mellitus in children. *Diabetologia*, **30**, 453-459.
Sprent, P. (1969). *Models in Regression and Related Topics*. Methuen, London.
Thompson, M.L. (1978). Selection of variables in multiple regression: Part I. A review and evaluation. *International Statistical Review*, **46**, 1-19.
Watson, G.S. & Hannan, E.J. (1956). Serial correlation in regression analysis II. *Biometrika*, **43**, 436-445.
Weisberg, S. (1985). *Applied Linear Regression*. Wiley, New York.
Woodley, W.L., Simpson, J., Biondini, R. & Berkeley, J. (1977). Rainfall results, 1970-1975: Florida Area Cumulus Experiment. *Science*, **195**, 735-742.
Woods, H., Steinour, H.H. & Starke, H.R. (1932). Effect of composition of Portland cement on heat evolved during hardening. *Industrial and Engineering Chemistry*, **24**, 1207-1214.

Chapter 4

Armitage, P. (1974). *Statistical Methods in Medical Research*. Blackwell, Oxford.
Atkinson, A.C. & Donev, A.N. (1992) *Optimum Experimental Designs*. Oxford University Press, Oxford.
Azais, J.M-. (1987). Design of experiments for studying intergenotypic competition. *Journal of the Royal Statistical Society, Series B*, **49**, 334-345.
Azais, J.M-., Bailey, R.A. & Monod, H. (1993). A catalogue of efficient neighbour designs with border plots. *Biometrics*, **49**, 1252-1261.
Bailey, R.A. (1984). Quasi-complete Latin squares: construction and randomization. *Journal of the Royal Statistical Society Series B*, **46**, 323-334.
Bailey, R.A. (1988). Semi-Latin squares. *Journal of Statistical Planning and Inference*, **8**, 299-312.
Bailey, R.A. (1992). Efficient semi-Latin squares. *Statistica Sinica*, **2**, 413-437.
Bartlett, M.S. (1937). Some examples of statistical methods of research in agriculture and applied biology (with discussion). *Journal of the Royal Statistical Society, Supplement 4*, 137-183.
Bechhofer, R.E., Santner, T.J. & Goldsman, D.M. (1995). *Design and Analysis of Experiments for Statistical Selection, Screening, and Multiple Comparisons.* Wiley, New York.
Berger, M.L. & Hsu, J.C. (1996). Bioequivalence trials, intersection-union tests and equivalence confidence sets. *Statistical Science*, **11**, 283-319.
Brien, C.J. (1983). Analysis of variance tables based on experimental structure. *Biometrics*, **39**, 53-59.
Brien, C.J. & Bailey, R.A. (2006). Multiple randomizations. *Journal of the Royal Statistical Society, Series B*, **68**, 571-608.
Brien, C.J. & Payne, R.W. (1999). Tiers, structure formulae and the analysis of complicated experiments. *The Statistician*, **48**, 41-52.
Cochran, W.G. & Cox, G.M. (1957). *Experimental Designs (second edition)*. Wiley, New York.
Cox, D.R. (1958). *Planning of Experiments*. Wiley, New York.
Davis, A.W. & Hall, W.B. (1969). Cyclic change-over designs. *Biometrika*, **56**, 283-293.
Dunnett, C.W. (1955). A multiple comparison procedure for comparing several treatments with a control. *Journal of the Americal Statistical Association*, **50**, 1096-1121.
Dunnett, C.W. (1989). Algorithm AS251 Multivariate normal probability intervals with product correlation structure. *Applied Statistics*, **38**, 564-579.
Fedorov, V.V. (1972). *Theory of Optimal Experiments*. Academic Press, New York & London.
Franklin, M.F. (1985). Selecting defining contrasts and confounded effects in p^{n-m} factorial experiments. *Technometrics*, **27**, 165-172.
Franklin M.F. & Bailey R.A., (1977) Selection of defining contrasts and confounded effects in two-level experiments. *Applied Statistics*, **26**, 321-326.
Franklin, M.F. & Mann, A.D. (1986). *DSIGNX a Program for the Construction of Randomized Experimental Plans*. Scottish Agricultural Statistics Service, Edinburgh (revised edition).
Hall, W.B. & Williams, E.R. (1973). Cyclic superimposed designs. *Biometrika*, **60**, 47-53.
Healy, M.J.R. & Westmacott, M.H. (1956). Missing values in experiments analysed on automatic computers. *Applied Statistics*, **5**, 203-206.

Hedayat, A. & Wallis, W.D. (1978). Hadamard matrices and their applications. *Annals of Statistics*, **6**, 1184-1238.

Hsu, J.C. (1996). *Multiple Comparisons Theory and Methods*. Chapman & Hall, London.

James, A.T. & Wilkinson, G.N. (1971). Factorisation of the residual operator and canonical decomposition of non-orthogonal factors in analysis of variance. *Biometrika*, **58**, 279-294.

John, J.A. (1987). *Cyclic Designs*. Chapman & Hall, London.

John, J.A. (1981). Efficient cyclic designs. *Journal of the Royal Statistical Society, Series B*, **43**, 76-80.

John, J.A., Wolock, F.W. & David, H.A. (1972). *Cyclic Designs*. National Bureau of Standards, Applied Mathematics Series 62.

John, J.A. & Quenouille, M.H. (1977). *Experiments: Design and Analysis*. Griffin, London.

John, P.W.M. (1971). *Statistical Design and Analysis of Experiments*. Macmillan, New York.

Kempthorne, O. (1952). *The Design and Analysis of Experiments*. Wiley, New York.

Kobilinsky, A. (1995). PLANOR : programme de génération automatique de plans d'expériences réguliers. INRA, Versailles.

Lamacraft, R.R. & Hall, W.B. (1982). Tables of incomplete cyclic block designs: *r=k*. *Australian Journal of Statistics*, **24**, 350-360.

Laycock, P.J. & Rowley, P.J. (1995). A method for generating and labelling all regular fractions or blocks for q^{n-m} designs. *Journal of the Royal Statistical Society, Series B*, **57**, 191-204.

Lin, L.I. (1989). A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, **45**, 255-268.

Lin, L.I. (2000). A note on the concordance correlation coefficient. *Biometrics*, **56**, 324-325.

Lin, C.S. & Poushinsky, G. (1983). A modified augmented design for an early stage of plant selection involving a large number of test lines without replication. *Biometrics*, **39**, 553-561.

Maindonald, J.H. & Cox, N.R. (1984). Use of statistical evidence in some recent issues of DSIR agricultural journals. *New Zealand Journal of Agricultural Research*, **27**, 597-610.

McCullagh, P. (1980). Regression models for ordinal data. *Journal of the Royal Statistical Society Series B*, **43**, 109-142.

Mead, R. (1988). *The Design of Experiments Statistical Principles for Practical Application*. Cambridge University Press, Cambridge.

Nelder, J.A. (1965a). The analysis of randomized experiments with orthogonal block structure. I Block structure and the null analysis of variance. *Proceedings of the Royal Society, Series A*, **283**, 147-162.

Nelder, J.A. (1965b). The analysis of randomized experiments with orthogonal block structure. II Treatment structure and the general analysis of variance. *Proceedings of the Royal Society, Series A*, **283**, 163-178.

Nelder, J.A. (1976). Discussion on papers by Wynn, Bloomfield, O'Neill & Wetherall. *Journal of the Royal Statistical Society, Series B*, **33**, 244-246.

Patterson, H.D. (1976). Generation of factorial designs. *Journal of the Royal Statistical Society, Series B*, **38**, 175-179.

Patterson, H.D. & Williams E.R. (1976). A new class of resolvable incomplete block designs. *Biometrika*, **63**, 83-92.

Patterson, H.D. & Bailey, R.A. (1978). Design keys for factorial experiments. *Applied Statistics*, **27**, 335-343.

Patterson, H.D., Williams E.R. & Hunter, E.A. (1978). Block designs for variety trials. *Journal of Agricultural Science, Cambridge*, **90**, 395-400.

Payne, R.W. & Wilkinson, G.N. (1977). A general algorithm for analysis of variance. *Applied Statistics*, **26**, 251-260.

Payne, R.W. (1990). Remark AS R82 A remark on AS65: Interpreting structure formulae. *Applied Statistics*, **39**, 167-175.

Payne, R.W. & Welham, S.J. (1990). A comparison of algorithms for combination of information in generally balanced designs. *COMPSTAT 90 Proceedings in Computational Statistics*, 297-302. Physica-Verlag, Heidelberg.

Payne, R.W. & Tobias, R.D. (1992). General balance, combination of information and the analysis of covariance. *Scandinavian Journal of Statistics*, **19**, 3-23.

Payne, R.W. (2004). Confidence intervals and tests for contrasts between combined effects in generally balanced designs. *COMPSTAT 2004 Proceedings in Computational Statistics*, 1629-1636. Physica-Verlag, Heidelberg.

Perry, J.N. (1986). Multiple-comparison procedures: a dissenting view. *J. Econ. Entomol.*, **79**, 1149-1155.

Plackett, R.L. & Burman, J.P. (1946). The design of optimum factorial experiments. *Biometrika*, **33**, 305-325 & 328-332.
Preece, D.A. (1971). Iterative procedures for missing values in experiments. *Technometrics*, **13**, 743-753.
Rogers, C.E. (1973). Algorithm AS 65: Interpreting structure formulae. *Applied Statistics*, **22**, 414-424.
Snedecor, G.W. & Cochran, W.G. (1980). *Statistical Methods (seventh edition)*. Iowa State University Press, Ames.
Whitehead, J. (1993). Sample size calculations for ordered categorical data. *Statistics in Medicine*, **12**, 2257-2271.
Wilkinson, G.N. (1957). The analysis of covariance with incomplete data. *Biometrics*, **13**, 363-372.
Wilkinson, G.N. (1970). A general recursive algorithm for analysis of variance. *Biometrika*, **57**, 19-46.
Wilkinson, G.N. & Rogers, C.E. (1973). Symbolic description of factorial models for analysis of variance. *Applied Statistics*, **22**, 392-399.
Williams, E.J. (1949). Experimental designs balanced for the estimation of residual effects of treatments. *Australian Journal of Scientific Research Series A*, **2**, 149-168.
Williams, E.R. (1975). *A New Class of Resolvable Block Designs*. Ph.D. Thesis, University of Edinburgh.
Yates, F. (1935). Complex experiments (with discussion). *Supplement to the Journal of the Royal Statistical Society*, **2**, 181-247. [Reprinted (without Discussion) in Yates, F. (1970). *Experimental Design: Selected Papers*, 69-117. Griffin, London.]
Yates, F. (1936). Incomplete randomized blocks. *Annals of Eugenics*, **7**, 121-140.
Yates, F. (1937). *The Design and Analysis of Factorial Experiments*. Technical Communication No. 35 of the Commonwealth Bureau of Soils. Commonwealth Agricultural Bureaux, Farnham Royal.

Chapter 5

Cox, D.R. & Snell, E.J. (1981). *Applied Statistics: Principles and Examples*. Chapman & Hall, London.
Cullis, B.R. & Gleeson, A.C. (1991). Spatial analysis of field experiments – an extension to two dimensions. *Biometrics*, **47**, 1449-1460.
Dempster, A.P., Selwyn, M.R., Patel, C.M. & Roth, A.J. (1984). Statistical and computational aspects of mixed model analysis. *Applied Statistics*, **33**, 203-214.
Gilmour, A.R., Thompson, R. & Cullis, B.R. (1995). Average Information REML, an efficient algorithm for variance parameter estimation in linear mixed models. *Biometrics*, **51**, 1440-1450.
Gilmour, A.R., Cullis, B.R. & Verbyla, A.P. (1997). Accounting for natural extraneous variation in the analysis of field experiments. *JABES*, **2**, 269-273.
Gilmour, A.R, Gogel, B.J., Cullis, B.R., Welham, S.J. & Thompson, R. (2002). *ASReml User Guide Release 1.0*. VSN International, Hemel Hempstead.
Gilmour, A.R, Gogel, B.J., Cullis, B.R. & Thompson, R. (2006). *ASReml User Guide Release 2.0*. VSN International, Hemel Hempstead.
Gumedze, F.N., Welham, S.J., Gogel, B.J. & Thompson, R. (2010). A variance shift model for detection of outliers in the linear mixed model. *Computational Statistics and Data Analysis*, **54**, 2128-2144.
Kenward, M.G. & Roger, J.H. (1997). Small sample inference for fixed effects from restricted maximum likelihood. *Biometrics*, **53**, 983-997.
Patterson, H.D. & Thompson, R. (1971). Recovery of inter-block information when block sizes are unequal. *Biometrika*, **58**, 545-554.
Piepho, H.P. & Williams, E.R. (2010). Linear variance models for plant breeding trials. *Plant Breeding*, **129**, 1-8.
Robinson, D.L., Thompson, R. & Digby, P.G.N. (1982). REML – a program for the analysis of non-orthogonal data by restricted maximum likelihood. *Compstat 1982 Proceedings in Computational Statistics, Part II (supplement)*, 231-232. Physica-Verlag, Vienna.
Robinson, D.L. (1987). Estimation and use of variance components. *The Statistician*, **36**, 3-14.
Searle, S.R. (1971). *Linear Models*. Wiley, New York.
Snedecor, G.W. & Cochran, W.G. (1989). *Statistical Methods (eighth edition)*. Iowa State University Press, Ames.
Snell, E.J. & Simpson. H.R. (1991). *Applied Statistics: a Handbook of Genstat Analyses*. Chapman & Hall, London.
Thompson, R. (1977). The Estimation of Heritability with Unbalanced Data I. Observations Available on Parents and Offspring. *Biometrics*, **33**, 485-495.
Webster, R. & Oliver, M.A. (2007). *Geostatistics for Environmental Scientists, 2nd edition*. Wiley, Chichester.

Welham, S.J. & Thompson, R. (1997). Likelihood ratio tests for fixed model terms using residual maximum likelihood. *Journal of the Royal Statistical Society, Series B*, **59**, 701-714.
Williams, E.R. (1986). A neighbour model for field experiments. *Biometrika*, **73**, 279-87.
Verbyla, A.P. (1995). *A Mixed Model Formulation of Smoothing Splines and Testing Linearity in Generalised Linear Models*. Dept of Statistics Research Report 95/5, University of Adelaide.
Verbyla, A.P., Cullis, B.R., Kenward, M.G. & Welham, S.J. (1999). The analysis of designed experiments and longitudinal data using smoothing splines (with discussion). *Applied Statistics*, **48**, 269-311.

Chapter 6

Arnold, G.M. (1988). Comparisons of algorithms for generalized Procrustes analyses. *Genstat Newsletter*, **22**, 7-11.
Arnold, G.M. (1992). Scaling factors in generalized Procrustes analysis. *Computational Statistics, Volume 1, Proceedings of the 10th Symposium on Computational Statistics, COMPSTAT, Neuchatel, Switzerland, August 1992*, 61-66.
Barnett, J.A., Payne, R.W. & Yarrow, D. (2000). *Yeasts: Characteristics and Identification (third edition)*. Cambridge. Cambridge University Press.
Bartlett, M.S. (1938). Further aspects of the theory of multiple regression. *Proceedings of the Cambridge Philosophical Society*, **34**, 33-40.
Bladon, S. (1986). *The New Observer's Book of Automobiles*. Frederick Warne, Harmondsworth.
Breiman, L., Friedman, J.H., Olshen, R.A. & Stone, C.J. (1984). *Classification and Regression Trees*. Wadsworth, Monterey.
Campbell, N.A. (1980). Robust procedures in multivariate analysis. I: Robust covariance estimation. *Applied Statistics*, **29**, 231-237.
Chatfield, C. & Collins, A.J. (1986). *Introduction to Multivariate Analysis (revised edition)*. Chapman & Hall, London.
Chatterjee, S. & Price, B. (1991). *Regression Analysis by Example (second edition)*. New York, Wiley.
Clarke K,R. (1993). Non-parametric multivariate analyses of changes in community structure. *Australian Journal of Biology*, **18**, 117-143.
Cooley, W.W. & Lohnes, P.R. (1971). *Multivariate Data Analysis*. Wiley, New York.
Critchley, F. (1983). *Ziggurats and Dendrograms*. Report No. 43, Department of Statistics, University of Warwick.
Digby, P.G.N. (1985). Graphical displays for classification. *PACT Journal of the European Study Group on Physical, Chemical and Mathematical Techniques Applied to Archaeology*.
Digby, P.G.N. & Kempton, R.A. (1987). *Multivariate Analysis of Ecological Communities*. Chapman & Hall, London.
Doran, J.E. & Hodson, F.R. (1975). *Mathematics and Computers in Archaeology*. Edinburgh University Press.
Eckart, C. & Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, **1**, 211-218.
Friedman, H.P. & Rubin, J. (1967). On some invariant criteria for grouping data. *Journal of the American Statistical Association*, **62**, 1159-1186.
Gabriel, K.R. (1971). The biplot – graphic display of matrices with application to principal component analysis. *Biometrika*, **58**, 453-467.
Gordon, A.D. (1981). *Classification: Methods for the Exploratory Analysis of Multivariate Data*. Chapman & Hall, London.
Gower, J.C. (1966). Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, **53**, 325-338.
Gower, J.C. (1967). Multivariate analysis and multidimensional geometry. *The Statistician*, **17**, 13-25.
Gower, J.C. (1968). Adding a point to vector diagrams in multivariate analysis. *Biometrika*, **55**, 582-585.
Gower, J.C. (1971). A general coefficient of similarity and some of its properties. *Biometrics*, **27**, 857-871.
Gower, J.C. (1974). Maximal predictive classification. *Biometrics*, **30**, 643-654.
Gower, J.C. (1975a). Algorithm AS 82: The determinant of an orthogonal matrix. *Applied Statistics*, **24**, 150-153.
Gower, J.C. (1975b). Generalized Procrustes analysis. *Psychometrika*, **40**, 33-51.
Gower, J.C. (1977). The analysis of asymmetry and orthogonality. In: *Recent Developments in Statistics* (editors Barra et al.) 109-123. North Holland, Amsterdam.
Gower, J.C. (1985a). Multivariate analysis: ordination, multidimensional scaling and allied topics. In:

Handbook of Applicable Mathematics (editor W. Ledermann), Statistics Vol. VIB (editor E. Lloyd). Wiley, Chichester.

Gower, J.C. (1985b). Measures of similarity, dissimilarity, and distance. In: *Encyclopaedia of Statistical Sciences, Volume V* (editors S. Kotz, N.L. Johnson & C.B. Read), 397-405. Wiley, New York.

Gower, J.C. & Hand, D.J. (1996). *Biplots*. Chapman & Hall, London.

Gower, J.C. & Krzanowski, W.J. (1999) Analysis of distance for structured multivariate data and extensions to multivariate analysis of variance. *Applied Statistics*, **48**, 505-519.

Gower, J.C. & Legendre, P. (1986). Metric and Euclidean properties of dissimilarity coefficients. *Journal of Classification*, **3**, 5-48.

Gower, J.C. & Payne, R.W. (1975). A comparison of different criteria for selecting binary tests in diagnostic keys. *Biometrika*, **62**, 665-671.

Gower, J.C. & Ross, G.J.S. (1969). Minimum spanning trees and single linkage cluster analysis. *Applied Statistics*, **18**, 54-64.

Greenacre, M. (1984). *Theory and Applications of Correspondence Analysis*. Academic Press, London.

Hartigan, J.A. (1975). *Clustering Algorithms*. Wiley, New York.

Helland, I.S. (1988). On the structure of partial least squares regression. *Commun, Statist.-.Simula.Comput.*, **17**, 581-607.

Hoerl, A.E. & Kennard, R.W. (1970). Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, **12**, 55-67.

Hoskuldsson, A. (1988). PLS Regression Methods, *Journal of Chemometrics*, **2**, 211-228.

Kendall, D.G. (1977). On the tertiary treatment of ties. *Proceedings of the Royal Society of London, Series A*, **354**, 407-423.

Klecka, W.R. (1980). *Discriminant Analysis (Quantitative Applications in the Social Sciences)*. Sage Publishing, Newbury Park, California.

Krzanowski, W.J. (1988). *Principles of Multivariate Analysis: a User's Perspective*. Oxford University Press.

Legendre, P. & Legendre, L. (1998). *Numerical Ecology, Second English Edition*. Elsevier, Amsterdam.

Manly, B.F.J. (1986). *Multivariate Statistical Methods: a Primer*. Chapman & Hall, London.

Manly, B.F.J. (1991). *Randomization and Monte Carlo Methods in Biology*. Chapman & Hall, London.

Mantel, N. (1967). The detection of disease clustering and a generalized regression approach. *Cancer Research*, **27**, 209-220.

Mardia, K.V., Kent, J.T. & Bibby, J.N. (1979). *Multivariate Analysis*. Academic Press, London.

Naes, T. & Martens H. (1989). *Multivariate Calibrarion*. John Wiley, Chichester.

Nathanson, J.A. (1971). An application of multivariate analysis in astronomy. *Applied Statistics*, **20**, 239-249.

Osten, D.W. (1988). Selection of Optimal Regression Models Via Cross-Validation. *Journal of Chemometrics*, **2**, 39-48.

Payne, R.W. & Preece, D.A. (1980). Identification keys and diagnostic tables: a review (with discussion). *Journal of the Royal Statistical Society, Series A*, **143**, 253-292.

Payne, R.W. (1981). Selection criteria for the construction of efficient diagnostic keys. *Journal of Statistical Planning and Inference*, **5**, 27-36.

Payne, R.W., Yarrow, D. & Barnett, J.A. (1982). The construction by computer of a diagnostic key to the genera of yeasts and other such groups of taxa. *Journal of General Microbiology*, **128**, 1265-1277.

Payne, R.W. & Thompson, C.J. (1989). A study of selection criteria for constructing identification keys containing tests with different costs. *Computational Statistics Quarterly*, **5**, 43-52.

Rayner, J.H. (1966). Classification of soils by numerical methods. *Journal of Soil Science*, **17**, 79-92.

Taylor, P.C. & Silverman, B.W. (1993). Block diagrams and splitting criteria for classification trees. *Statistics & Computing*, **3**, 147-161.

TenBerge, J.M.F. (1977). Orthogonal Procrustes rotation for two or more matrices. *Psychometrika*, **42**, 267-276.

Vinod, H.D. (1976). Application of new ridge regression methods to a study of Bell system scale economies. *Journal of the American Statistical Association*, **71**, 835-841.

Chapter 7

Anderson, O.D. (1976). *Time Series Analysis and Forecasting*. Butterworths, London.

Bloomfield, P. (1976). *Fourier Analysis of Time Series: an Introduction*. Wiley, New York.

Box, G.E.P. & Jenkins, G.M. (1970). *Time Series Analysis, Forecasting and Control*. Holden-Day, San

Francisco.
de Boor, C. (1980). FFT as nested multiplication, with a twist. *SIAM Journal of Scientific and Statistical Computing*, **1**, 173-178.
Jenkins, G.M. & Watts, D.G. (1968). *Spectral Analysis and its Applications*. Holden-Day, San Francisco.
Nelson, C.R. (1973). *Applied Time Series Analysis for Managerial Forecasting*. Holden-Day, San Francisco.
Shi-fang, L., Shi-guang, L., Shu-hua, Y., Shao-zhong, Y. & Yuan-xi, L. (1977). Analysis of periodicity in the irregular rotation of the Earth. *Chinese Astronomy*, **1**, 221-227.
Tunnicliffe Wilson, G. (1989). On the use of marginal likelihood in time-series model estimation. *Journal of the Royal Statistical Society, Series B*, **51**, 15-27.

Chapter 8

Aitkin, M., Anderson, A., Francis, B. & Hinde, J. (1989). *Statistical Modelling in GLIM*. Oxford University Press, Oxford.
Box, G.E.P. (1950). Problems in the analysis of growth and wear curves. *Biometrics*, **6**, 362-389.
Breslow. N. (1974). Covariance analysis of censored survival data. *Biometrics*, **30**, 89-99.
Burgess, T.M. & Webster, R. (1980). Optimal interpolation and isarithmic mapping of soil properties. I. The semi-variogram and punctual kriging. *Journal of Soil Science*, **31**, 315-331.
Cressie, N. & Hawkins, D.M. (1980). Robust estimation of the variogram. *Journal of the International Association of Mathematical Geology*, **12**, 115-125.
Gabriel, K.R. (1961). The model of ante-dependence for data of biological growth. *Bulletin Institut International Statistique (Paris)*, **39**, 253-264, (33rd session).
Gabriel, K.R. (1962). Ante-dependence analysis of an ordered set of variables. *Annals of Mathematical Statistics*, **33**, 201-212.
Cox, D.R. (1972). Regression models and life tables (with discussion). *Journal of the Royal Statistical Society, Series B*, **34**, 187-220.
Dowd, P.A. (1984). The variogram and kriging: robust and resistant estimators. In: *Geostatistics for Natural Resources Characterization* (ed. G. Verly, M. David, A.G. Journel & A. Marechal), 91-106. D. Reidel, Dordrecht.
Genton, M.G. (1998). Highly robust variogram estimation. *Mathematical Geology*, **30**, 213-221.
Goovaerts, P. (1997). *Geostatistics for Natural Resources Evaluation*. Oxford University Press, Oxford.
Goulard, M. & Voltz, M. (1992). Linear coregionalization model: tools for estimation and choice of cross-variogram matrix. *Mathematical Geology*, **24**, 269-286.
Greenhouse, S.W. & Geisser, S. (1959). On methods in the analysis of profile data. *Psychometrika*, **24**, 95-112.
Journel, A.G. & Huijbregts, C.J. (1978). *Mining Geostatistics*. Academic Press, London.
Kenward, M.G. (1987). A method for comparing profiles of repeated measurements. *Applied Statistics*, **36**, 296-308.
Künsch, H.R., Papritz, A. & Bassi, F. (1997) Generalized cross-covariances and their estimation. *Mathematical Geology*, **29**, 779-799.
Matheron, G. (1965). *Les Variables Régionalisées et Leur Estimation*. Masson, Paris.
Matheron, G. (1971). *The Theory of Regionalized Variables and its Applications*. Cahiers du Centre de Morphologie Mathématique, Ecole des Mines de Paris, Fontainebleau, No 5.
Matheron, G. (1989). *Estimating and Choosing*. Springer Verlag, Berlin.
Monestiez, P, Allard, D. & Froidevaux, R. (editors) (2001). geoENV III – Geostatistics for Environmental Applications. Kluwer Academic Publishers, Dordrecht.
Rowlingson, B.S. & Diggle, P.J. (1993). Spatial point pattern analysis code in S-Plus. *Computers and Geosciences* **19**, 627-655.
Sánchez-Vila, X., Carrera, J. & Gómez-Hernández, J.J. (editors) (2004). geoENV IV – Geostatistics for Environmental Applications. Kluwer Academic Publications, Dordrecht.
Wackernagel, H. (1995). *Multivariate Geostatistics*. Springer-Verlag, Berlin.
Webster, R. & Oliver, M.A. (1990). *Statistical Methods in Soil and Land Resource Survey*. Oxford University Press, Oxford.
Webster, R. & Oliver, M.A. (2007). *Geostatistics for Environmental Scientists, 2nd Edition*. Wiley, Chichester.
Whittle, P. (1954). On stationary processes in the plane. *Biometrika*, **41**, 434-449.

Index

units factor 408, 434, 467, 654
3-tier analysis 437, 439
  further output 439
A2DISPLAY procedure 76
A2KEEP procedure 77
A2RESULTSUMMARY procedure 78
A2WAY procedure 74
Abbreviation
  of option name 2
  of string token 3
  rules 2
ABC curve 140
ABC plot 139
ABLUPS procedure 434
Absolute residual plot 410
Absolute-residual plot 675
Absorbing factor 652, 666
  in REML 716
Absorption in regression 165
Abundance/biomass comparison 140
Accumulated analysis of deviance 272
Accumulated analysis of variance 188, 196
Accumulated summary of regression 192
ACE measure of species richness 151
ACHECK procedure 418
ACONFIDENCE procedure 421
Actuarial estimate 1056
ADD directive 193
Added factors 628, 630
Adding extra units to a design 617, 618
Adding points to a principal coordinates analysis 848
Adding regression variables 194
Additive model 237, 243
ADDPOINTS directive 848
Adequacy of a model 168
ADETECTION procedure 596
ADISPLAY directive 401
Adjusted analysis of variance 443, 445
Adjusted means 404
Adjusted R-squared 168
Adjusted R2 statistic 168
Adjusted Rand index 919
Adjusted response variate 273
Adjustment term in canonical variate analysis 805, 806
ADPOLYNOMIAL procedure 456
ADSPREADSHEET 574
AFALPHA procedure 546
AFAUGMENTED procedure 621
AFCOVARIATES procedure 445
AFCYCLIC procedure 551
AFIELDRESIDUALS procedure 411
AFMEANS procedure 417
AFMINABERRATION directive 640
AFNONLINEAR procedure 564
AFRESPONSESURFACE directive 560
AGALPHA procedure 544
AGBIB procedure 546
AGBOXBEHNKEN procedure 556
AGCENTRALCOMPOSITE procedure 554
AGCYCLIC procedure 549
AGDESIGN procedure 523
Agglomerative method 899
AGHIERARCHICAL procedure 514
AGLATIN procedure 527
AGLOOP procedure 569
AGMAINEFFECT procedure 558
AGNEIGHBOUR procedure 552
AGQLATIN procedure 533
AGRAPH procedure 413
Agreement between two methods
  Bland-Altman plot 111
AGREFERENCE procedure 567
AGSEMILATIN procedure 537
AGSQLATTICE procedure 541
AGYOUDENSQUARE procedure 534
Akaike information coefficient 707, 709, 710, 712
Akaike information criterion 208
AKEEP directive 458
AKEY procedure 614
Algorithm for nonlinear regression 364
Aliased model terms 470
Aliasing 193, 392
  in ANOVA 402
  in prediction 226
  in regression 169, 187, 189, 198, 216, 217
  in REML 653
  of treatment terms 638
  of treatments generated by design key 638
All subsets regression 206
Alpha design 390, 392, 512, 544, 545
Alternative parameterization 215, 357
Amalgamation 900
Amalgamations matrix
  forming from a minimum spanning tree 905
AMCOMPARISON procedure 423
AMERGE procedure 617
AMTDISPLAY procedure 437, 439
AMTIER procedure 437
Analysable design 476, 482
Analysing an experimental design 399
Analysis of contrasts of repeated measurements

1035
Analysis of covariance 440, 442, 446
method for 445
Analysis of deviance 269
individual terms in 258
Analysis of distance of multivariate data 829
Analysis of similarities 793
Analysis of unbalanced designs 486, 490, 500, 502
saving results in a spreadsheet 504
Analysis of variance 74, 76, 77, 437
advice about unbalanced designs 509
analysing the data 397
BLUPs for block terms 434
comparisons within tables of means 234
current model and y-variate 467
defining covariates 442, 445, 446
defining treatment model 395
detectable effect 391, 596
displaying further output 401
for single channel microarray 390
forming tables of means 417
in regression 167, 171, 188, 196, 269
multi-tiered 439
of a covariate 445
one-way 71
output to a text 402
output to another channel 402
parallel 390, 643
percentage sum of squares accounted for 388
percentage variance accounted for 388
permutation tests 419
plot polynomial contrast 456
power in 391, 593
repeated measures 1037-1039
saving results in a spreadsheet 468
screening test 483
summary of results 78, 408
table 510
two-way 74
Analysis-of-variance table 394, 402
saving for a regression model 176
saving from ANOVA 461
with covariates 443
Anderson-Darling test 61
Animal breeding models 753
Anisotropic variation 1081
Anisotropic variogram 1081
ANOSIM 21, 136, 793
ANOVA directive 398
ANOVA save structure
adjusting d.f. of repeated measurements 1039
Ante-dependence 1030, 1041, 1043-1045
estimating the order 1041
estimation of missing values 1044
model in REML 723
order 1041-1043, 1045
structure 389
testing 1043
Anti-end-cut factor 943
ANTMVESTIMATE procedure 1044
ANTORDER procedure 1041
ANTTEST procedure 1043
AONEWAY procedure 71
AOVANYHOW 506
APERMTEST procedure 419
APLOT procedure 410
APOLYNOMIAL procedure 455
APOWER procedure 593
Appending
into a text 9
Approximate F statistic for REML fixed terms 681
Approximation in generalized linear model 269, 272
APRODUCT procedure 619
ARANDOMIZE procedure 581
AREPMEASURES procedure 1037
ARESULTSUMMARY procedure 408
ARIMA model 986
back-forecasts 1000
bias in least-squares method 1000
Box-Cox transformation 992
changing values 991, 993
default parameter values 994
definition 992
estimation 993
evaluation of likelihood 999
exact likelihood method 1000
fitting 989
forecasting 1005
forecasts 1009, 1010
invertibility of 992, 994
lags of 992
least-squares likelihood method 1000
likelihood function 999
marginal likelihood method 1015, 1020
missing values 995
multiple seasonal 993
non-seasonal 992
orders of 992
parameters of 992
preliminary parameter estimation 1025
seasonal 992
selection 968
specification of 991
starting-value problem 999
stationarity of 992, 994
technical information 999
testing nested models 1001
ASAMPLESIZE procedure 589
ASCREEN procedure 483
ASPREADSHEET procedure 468, 504
Assign values

to dummies and pointers 10
Association between similarity matrices 791
Association measure 779
Assumptions
in analysis of variance 393, 420
ASTATUS procedure 467, 516
Asymmetric matrix 886
Asymmetry coefficient 21, 136, 154
Asymptote of curve 352
Asymptotic regression 346
Attributes of data structures 4
AUDISPLAY procedure 490
Audit trail 1
Augmented design 621
AUGRAPH procedure 497
AUKEEP procedure 500
AUNBALANCED procedure 486
AUPREDICT procedure 502
AUSPREADSHEET procedure 504
Auto-regressive model 729, 736
in REML 720-722
Auto-regressive moving average model
in REML 723
Auto-variogram
plotting 2d 1096
Autocorrelation 966, 969
Autocorrelation function 966
sample 967
saving 967
testing 967
theoretical 1024, 1027
Automatic stepwise regression 203, 204
Autoregressive prediction equation 969
Average Information algorithm 660, 717
Average-linkage clustering 901
Averaging of effects 224, 487, 503, 507
Back-fitting 246, 288
Backing-store 6
Backward elimination 203, 204, 206
Backward shift operator 992
Balance 447, 473, 475, 482
Balanced design 388, 509
Balanced repeated measurements 1033
Balanced-incomplete-block design 390, 428, 512, 546
Banded covariance model
in REML 723
Barchart 27
Bartlett's test 73
Barycentric coordinates 17, 19
Basic contrasts 630
of a model term 640
Basic factors 628, 630
Basic statistics 19, 22
BCDISPLAY procedure 944
BCIDENTIFY procedure 948
BCKEEP procedure 949
BCLASSIFICATION procedure 942
BCVALUES procedure 946
Bernoulli distribution
in regression 269
Best linear unbiased predictor 649, 650, 664
Beta distribution 58
Beta-binomial distribution 58
Bias correction
for standard deviation 127, 129, 131
Bias in maximum likelihood estimates 650
Bimension 886
Binary file
close 15
current position 15
open 15
write data to 15
Binary response variable 269
Binomial distribution
in regression 262, 264, 265, 269, 272, 368
Binomial test 81
sample size for 599, 601
Binomial testing for defective items 131
Bioassay 288
Biplot 778, 878
correlation 881
correspondence analysis 866
distance 881
ordination 883
Bit map 17
Bit pattern 12
BJESTIMATE procedure 988
BJFORECAST procedure 1009
BJIDENTIFY procedure 968
BKDISPLAY procedure 953
BKEY procedure 950
BKIDENTIFY procedure 954
Bland-Altman plot 111
BLANDALTMAN procedure 111
Block design
complement 393
minimum aberration 518
Block factors 628
Block formula 434
for randomization 578
saving 461
Block model of an experimental design 427
Block structure of an experimental design 427
Block term 428
Block-if structure 13
BLOCKSTRUCTURE directive 427
BLUP 649, 650, 664
BLUPs for ANOVA block terms 434
BNTEST procedure 81
Bonferroni test 425
Boolean arithmetic 7
Bootstrap 137, 155
for critical values in a REML analysis 689

for fixed effects in REML 685
in cluster analysis 921
in quantile regression 382
Botanical key 950
Bounded linear covariance model
in REML 723
Bounds
in curve fitting 358
in nonlinear regression 364, 367
Box and Jenkins methods 963
Box-and-whisker plot 29, 30
Box-Behnken design 390, 512, 556, 557
Box-Cox transformation 282
estimation of 994, 1000
in time series 992
in transfer function modelling 1014
Boxplot 29
BOXPLOT procedure 29
BPRUNE procedure 375
Bradley-Terry model 159
BRDISPLAY procedure 374
Breeding value 645
BREGRESSION procedure 372
Breslow test 1054
BRKEEP parameter 379
Broken-stick regression model 160
BRPREDICT procedure 378
BRVALUES procedure 375
CABIPLOT procedure 866
Calculations 7
Calibration 171
CANCORRELATION procedure 839
Canonical correspondence analysis 873, 874
Canonical efficiency factor 389
Canonical link 265
Canonical relationships between projectors 389
Canonical variate scores 807, 809
graphs 808
Canonical variates analysis 803, 811, 839, 888
trellis plot of bar or pie charts 810
Capability statistics 135
Carry-over effects 390, 392, 511, 530, 550
CART 372, 374, 375, 378, 379, 942, 944, 946, 948, 949
Catalogue of balanced-incomplete-block designs 548
Cate-Nelson graphical analysis 159
CATRENDTEST procedure 123
CDESCRIBE procedure 24
Censored data 160, 292, 312, 644, 1050, 1054, 1056, 1059
left-censoring 1059
right-censoring 1054
Central composite design 390, 512, 554, 555
Centre point 554
Centroid 845
Centroid clustering 899
Centroid configuration 893
Change in regression model 196
Changing order of function in regression 240, 245, 246
Channel 6
Chao measure of species richness 151
Chi-square
from regression 175, 269, 273
Chi-square test 114, 115
for independence 116
CHIPERMTEST procedure 117
CHISQUARE procedure 115
Circular covariance model
in REML 723
Circular data 24
plots 46
Circular regression 26
City-block coefficient 790
adding points 786
Class predictor 924, 932
Classification 899, 924
Classification tree 942, 944, 946, 948, 949
values 946
CLASSIFY procedure 934
Climatic data 47
Cluster analysis 776, 899
for large data sets 936
bootstrap analysis 921
comparing groupings 918
displaying results 902
filling holes in clusters in multi-dimensional space 1112
forming the full set of clusters 901
hierarchical 899
of points using their densities in multi-dimensional space 1107
saving results 902
using the density of PCP scores 936
CLUSTER directive 925
CMHTEST procedure 121
Cochran's Q test 20, 120
Cochran-Armitage test 123
Cochran-Mantel-Haenszel test 20, 121, 122
Coefficient of variation 263, 271, 402, 408
COKRIGE directive 1097
Coleman curve 149
Collectors curve 149
Collinearity 189
Collinearity in multiple regression 831
Colour
standard for graphics 16
Combination of information
in REML 642
Combined analysis of several experiment 760
Combined estimates
of covariate regression coefficients 461
of effects 475

of effects, saving 466
of means 402, 405, 479
of means, saving 466
of treatment effects 402
Combining information 466
Comma in formula 214, 239
Command to generate a design 513
Communality 813
Comparison contrasts 449
COMPARISON function 449
Comparison of curves 352
Comparison of effects with standard 261
Comparison of means 234
from REML! 751
Comparisons between treatments 448
Complementary log-log 263-265, 276
Complete Latin square 511, 533
Complete-linkage 901
Completely randomized design 386, 391, 434, 513, 515
Component of variance 223
Component of variation 156
Compositional data 17, 19
Compound symmetry 1037
Concordance correlation coefficient 109, 610
Concurrence matrix 393
Condensed data matrix 906
Conditional test 211, 483, 484, 489
Confidence ellipse 17
Confidence interval
Dunnett's 426
Confidence limits 228
Confounded model terms 628
Confounding 392, 429, 470, 476, 511, 524
with blocks 523
Conjugate hierarchical generalized linear model 316
Constant
ignoring in regression 171, 216
in canonical variate analysis 805
in curve fitting 352
in regression 169, 171, 215, 216, 366
Constrained curve through origin 351
Constraint
in analysis of variance 406
in nonlinear regression 364
on parameters in regression 215
on variance components 655
Contingency table 114, 117, 262
Continuation character 1
Continuous probability distribution 50
Contrast 455
amongst regression means 229, 230
between treatment effects 402, 448
comparison within table of means 234
in regression 237
plotting 456
saving from ANOVA 464
Control mortality 278, 283
Control treatment 440, 517, 619
Conventions 3
Convergence
in iterative model 273
in nonlinear regression 364
of generalized linear model 275
of iterative model 356
of missing-value estimation 448
Cook's statistic 181
CORANALYSIS procedure 860
CORRELATE directive 965
Correlated error structures
in REML 718
Correlated error term 764
Correlated errors 156
Correlation 100, 963, 965
between parameters in curve fitting 355
between parameters in generalized linear model 272
between parameters in polynomial regression 239, 242
between parameters in regression 170
between regression variables 189, 192
between REML model terms 726, 743, 745
between units in regression 344
biplot 881
canonical 839
coefficient 103, 965
forming matrix 100
in curve fitting 355
in principal components analysis 799
matrix, between variables 965
matrix, saving 966
parameters in nonlinear regression 362
sample size to detect 608
Correlogram 966
Correspondence analysis 840, 860
biplot 866
multiple 864
Cosine transformation
inverse 975
of time series 973
Counts
analysis of 262
of distinct values in a vector) 37
Covariance
in principal components analysis 799
Covariance efficiency factor 442, 443
Covariance model 296
Covariance models in REML 722
Covariate 387, 442, 445, 446
centring within REML 653, 768
in REML 653
COVARIATE directive 442
Covariate regression coefficients 402, 445

saving from ANOVA 463
Covariogram
forming 1089
modelling 1093
COVDESIGN procedure 443
Cpk index 136
Cramér-von Mises test 61
CRBIPLOT procedure 881
Critical exponential curve 348, 350
Critical values
for fixed terms in a REML analysis 689
Cross-correlation 970
saving 971
Cross-correlation function 966
sample 971
testing 971
Cross-over design 392
Cross-over trial 530
Cross-product operator 214
Cross-spectral analysis 975
Cross-validation
in kriging 1077
Cross-variogram 1089
plotting 2d 1096
Crossed experimental design 433
Crossing and nesting in an experimental design 434
Crossing operator
in ANOVA 396
in randomization 578
CRTRIPLOT procedure 883
Cubic smoothing spline 243
in REML 756
Cumulative sum 128
Curve fitting 345
Curves
with AR1 errors 1048
with common nonlinear parameters 359
with power-distance correlation model 1048
Customized data structure 4
CUSUM table 128
Cut-point 276
CVA directive 803
CVAPLOT procedure 808
CVASCORES procedure 807
CVATRELLIS procedure 810
Cyclic change-over design 550
Cyclic design 390, 392, 512, 549-551
Cyclic superimposed design 550
Czekanowski coefficient 790
Data manipulation 7
Data structure 2, 3
Data structures in store 4
DBIPLOT procedure 878
DCIRCULAR procedure 46
DCOVARIOGRAM procedure 1096
DDENDROGRAM procedure 911
DDESIGN procedure 572
Debugging programs 14
Declaration of a data structure 4
Default font for graphics 16
Default output
from ANOVA and ADISPLAY 402
Degree of smoothness 243
Degrees of freedom 394
for smoothing 243, 244
in regression 169, 175, 176
of REML fixed model 767
of REML random model 767
saved from time series 1003
saving from ANOVA 463
Deletion residuals 181
Demonstration experiment 517, 542
Dendrogram 787, 899, 900, 924
plotting 911-914
Density estimation 43
Density plot 17
Dependence 156
Derivative
in nonlinear regression 364
of fitted values 355
of function 371
of link function 278
DESCRIBE procedure 22
Design
analysable by ANOVA 388
minimum aberration 640
unbalanced 642
Design generation 390, 511
Design generator 392, 513
Design key 392, 513, 523, 612-615, 617, 640
constructing all possible 630
construction of 627-634, 636
for designs with several strata 631
forming pseudo-factors from 636
inverting 636
to extend an existing design 634
with factors constrained to higher strata 633
Design matrix
in regression 161, 169, 175, 187, 193, 243
saving in regression 175
Design of experiments 390, 391, 511, 593, 612, 613
augmented design 621
D-Optimal design 564
for generalized linear model 564
for nonlinear model 564
non-orthogonal split--plot design 391
space filling design 391
spreadsheet of plan and data6 574
Youden square 534
DESIGN structure in ANOVA 399, 400, 448
Design tools 612
Designed experiment

randomization 578, 581, 582
Designs analysable by ANOVA 475, 476, 478
Designs to estimate main effects of 2-level factors 558
Designs with several error terms 427
Detectable effect
in analysis of variance 391, 596
Determinant
minimization of 924
Deterministic model
validate against observed data 160
Deviance 116
from regression 269
in regression 175, 269, 278, 362, 369
in REML 659, 730
in time series 1000, 1003, 1015
residuals 165, 272
Deviations from fitted contrasts 451, 452
DFOURIER procedure 979
DHISTOGRAM directive 27
Diagnostic table 959
Diagonal matrix 4
Dichotomous variable 784
Differencing operator 992
Diffusion model 367
Digit 2
Dilution assay 263, 265
Dimensionality of data 776
Direct product construction of covariance models 745
Directive 1
Discrete probability distribution 50, 54
Discriminant analysis 805, 814, 815, 817, 820, 823
stepwise 818
DISCRIMINATE procedure 815
Discrimination
quadratic 821
Dispersion parameter 262, 264, 269, 272
Displaying field plans of designs 571
Dissimilarity 784, 790
Distance 790
Distance biplot 881
Distance matrix 840
Distinct values
tally table of 36
DISTRIBUTION directive 48
Distribution fitting 368
Distribution of response 165, 258, 262, 351, 367, 368
Divergence 271
Diversity index 136, 137
Diversity statistics 137, 141, 143
DMST procedure 916
Dominance preemption model 143
Dot product 214
Dot-plot 38
DOTPLOT procedure 38
Double exponential curve 348, 350
Double Fourier curve 349, 351
Double Gaussian curve 349, 351
Double hierarchical generalized linear model 315
analysing 317
defining the fixed model 313
defining the random model 314
displaying 325
predictions 326
saving information from 330
Double Normal distribution 57
DPARALLEL procedure 98
DPROBABILITY procedure 40
DPTMAP procedure 1103
DREPMEASURES procedure 1033
DROP directive 193
Dropping regression variables 194
DSTTEST procedure 584
Dummy 4
Dummy analysis 388, 399, 482
for orthogonal designs 482
Duncan's multiple range test 425
Dunnett's test 388, 426
DVARIOGRAM procedure 1078
ECABUNDANCEPLOT procedure 139
ECDIVERSITY procedure 137
ECFIT procedure 141
ECNICHE procedure 143
ECNPESTIMATE procedure 150
Ecological coefficient 786
Ecological data 136
Ecology 141, 143, 145, 154
Economics 154
ECRAREFACTION procedure 145
EDFTEST procedure 61
Effect of a treatment 394
Effective degrees of freedom 475
in smoothing 243
Effective dose 290
Effective standard error 406
approximate for unbalanced design 487
for contrast in ANOVA 452
Effects
in REML 664
Efficiency
of effects in cross over designs 393
Efficiency factor 443, 470, 473, 475, 478
saving from ANOVA 463
Elimination of effects 165
Emax curve 349
Empirical distribution 26
End-effects in curve fitting 243
Environment of the Genstat job 15
Environmental gradient 840
Equal weights in prediction 224, 487, 503

508
Equation of a polynomial contrast 456
Equation order in REML 717
Equivalence test 583, 584
  in ANOVA 590, 594, 597
  in regression 184
  with t-test 70
Error structure
  in REML 649
Error term in regression 156
Error terms in ANOVA 428
ESTIMATE directive 993, 1014, 1020
Euclidean coefficient 786, 790
Euclidean distance 790, 924
Euclidean space 776, 784, 841
Exact probability 102
Exact test
  Fisher's exact test 20, 116
  for analysis of similarities 794
  for Cochran's Q statistic 121
  for one-way anova 72
  for t-statistic 72
  in analysis of variance 830
  in analysis of variance& 420
  in regression and generalized linear models 186, 309, 694
  Steel's test 96
  to compare groupings 919
  with McNemar's test 119
Excess zeros in count data 338, 342
Exclusion of units in regression 191
Execute other programs 15
Exit code
  from ANOVA 483
  from regression 274, 356, 365
Expansion of formula 214
Experimental design
  plotting 572
Experimental layout 572
Explanatory variate 156, 167, 192, 194
Exploratory analysis 19
Exploratory data analysis 22
Exploratory plots 19
Exponential curve 346, 348, 349
Exponential distribution 57, 1059
  in regression 264
Exponential family 262, 264, 363
Exponentially weighted moving average control chart 130
Expression 2, 4
Extended quasi likelihood 319
Extending a design 634
Extending a design key 634
Extreme observation 168, 269, 270
Extreme-value distribution 1059
F distribution 394
  approximate for REML fixed terms 681
F statistic
  in regression 168, 189, 202
F-statistic
  from similarity matrix 850
FACROTATE directive 811
Factor 4
  in nonlinear regression 366
  merging labels 9
  merging levels 9
  standardize levels or labels 9
  values in systematic order 612
Factor analysis 853
Factor analytic model
  in REML 724
Factor rotation 811
Factorial design 523
Factorial design with confounding 526
Factorial experiment 386
FACTORIAL option
  in ANOVA 396
Factorial plus added control 397, 440
Failure to fit regression model 274
FCA directive 853
FCOVARIOGRAM directive 1089
FDESIGNFILE procedure 526
Feasible sets of contrasts 630
FEXACT2X2 procedure 116
FIELLER procedure 261
Fieller's theorem 261
Files 6
FILTER directive 1022
Filtering 1021
Finding strings within the lines of a text 9
First-order balance 388, 473, 475, 482
Fisher scoring
  in REML 714
Fisher's exact test 20, 116
Fisher's LSD test 669
Fisher's Protected Least Significant Difference 425
Fisher's Unprotected Least Significant Difference 425
Fisher-scoring 288
FIT directive 166, 188
FITCURVE directive 347
FITINDIVIDUALLY procedure 273
FITMULTINOMIAL procedure 276
FITNONLINEAR directive 365
Fitted values
  in regression 170, 278, 367
  in REML 659
  initial for generalized linear model 275
  saving from ANOVA 400, 460
  saving from regression 164, 175, 193
Fitting a distribution 368
Fitting a generalized linear model one term at a time 273

Fitting curves
  with AR1 errors 1048
  with power-distance correlation model 1048
Fixed effects
  definition 647
  estimates 649, 650, 664
  parameterization of 653
Fixed terms
  definition 652
  testing in REML 681
FKEY directive 627
FLC test 643
Fletcher-Powell algorithm 364
Fletcher-Powell optimization 288
Folded replicate 558, 559
FOR loop 13
FORECAST directive 1005, 1015, 1020
Forecasting
  from time series models 1005
Formatted output 5
Forming a variogram 1068
Forming pseudo-factors 636
Formula 4
  in ANOVA 395
  in regression 192, 195, 214, 239
  in REML 648
Fortran 281
Forward selection 203, 204, 206
Fourier curve 349, 351
FOURIER directive 972
Fourier transformation 972
  definition 973, 974, 976
  fast 973
  frequencies 972
  index 963, 972
  inverse 975
  lag window 973
  of a complex series 974
  of conjugate sequence 975
  of real series 973
  order 973, 974, 976
  restrictions on units 972
  smooth spectrum estimate 973
  to calculate convolutions 975
FPSEUDOFACTORS directive 636
Fractional factorial design 390, 511, 638
  minimum aberration 518
FRIEDMAN procedure 94
Friedman's test 93, 94
FSIMILARITY directive 784
FSSPM directive 780
FTSM directive 1025
Function in regression 237
  changing order 240
  list within 239
Function minimization 369
Function of parameters 357
Functional relationship model 779
Further output for an unbalanced design 491
Furthest-neighbour clustering 901
FVARIOGRAM directive 1068
G5XZXO subroutine 367
Galois field 513, 529
Gamma distribution 57
  in regression 263, 264, 269, 271, 278, 368
Gamma ratios 649
Gamma statistic 108
Gauss-Newton algorithm 364
Gauss-Newton optimization 288
Gaussian curve 349, 351
GEE procedure 331
Generalized additive model 275, 280
  stepwise regression 204
Generalized emax curve 349
Generalized estimating equations 331, 1030
Generalized least squares 343, 649, 650
  estimates 664
Generalized linear mixed model 292, 312-314, 317, 325, 327, 329, 330, 388
  further output 298
  likelihood tests for random terms 310
  predictions 301, 326
  residual plots 300
  saving results 304
  tests for fixed terms 323
  tests for random terms 321
Generalized linear model 162, 258
  antilog of estimates 265, 272
  dispersion 272
  fitting individual terms 255
  hierarchical 312-315, 317, 325-327, 329, 330
  in survival analysis 1059
  monitoring information 285
  plotting 261
  random permutation test 185, 307
  search for best model 204
  stepwise regression 204
  structured dispersion model 313, 314
  weighted 345
Generalized linear models
  profile likelihood confidence intervals 159
Generalized logistic curve 349, 350
Generalized nonlinear additive model 285
Generalized nonlinear model 316
Generalized Procrustes rotation 892, 893
Generally-balanced design 388, 390, 473, 475
GENERATE directive 612
Generating factor values
  in systematic order 612, 615
  using a design key 615
Generation of pseudo-factors 614
Generation of values of block factors 613, 615
Genomic prediction 645
Genotype × environment interaction 389

GENPROCRUSTES procedure 892
Genstat Design System 390, 511
Genstat for Windows 1
Genstat spreadsheet file 6
Geometric distribution 55
  in regression 263
Geometric series 141
Geostatistics 1066
Gini coefficient 21, 136, 154
Gini information 943
GLDISPLAY procedure 298
GLKEEP procedure 304
GLMM procedure 292
GLPLOT procedure 300
GLPREDICT procedure 301
Gompertz curve 349, 350, 352
Goodness of fit 274
Goodness-of-fit
  for continuous distributions 62
Gradient of curve 357
Graeco-Latin square 391, 513, 528, 529, 616
Grand mean 394
Graph
  density plot 17
  of HGLM model 328
  of regression model 178
Graphics 16
  default font 16
Graphics environment 13
Graphs of tables of means 413, 414
Greenhouse-Geisser epsilon 1037
GRIB2 meteorological data file 7
Grid evaluation of function 371
Grid evaluation of likelihood 365, 367
Group similarity 788
Group-average clustering 901
Grouped variable
  in regression 212
Grouping factor
  in regression 165, 192
Groupings
  comparing 918
Groups
  comparing 66
Growth curve 350
GSTATISTIC procedure 108
Hadamard matrix 12, 392, 513, 546
Half-Normal plot 181, 300, 410, 675
Harmonic analysis 979
HBOOTSTRAP procedure 921
HCLUSTER directive 900
HCOMPAREGROUPINGS procedure 918
HDISPLAY directive 902
Heterogeneity factor 264, 290
Heterogeneity of variance in REML 730
Heteroscedasticity 168
HGANALYSE procedure 317
HGDISPLAY procedure 325
HGDRANDOMMODEL procedure 315
HGFIXEDMODEL procedure 313
HGFTEST procedure 323
HGKEEP procedure 330
HGNONLINEAR procedure 316
HGPLOT procedure 327
HGPREDICT procedure 326
HGRANDOMMODEL procedure 314
HGRTEST procedure 321
HGWALD procedure 324
Hierarchical cluster analysis 899
Hierarchical clustering 787
Hierarchical generalized linear model 312-315, 317, 321, 323, 325, 327, 330
  analysing 317
  conjugate 316
  defining the fixed model 313
  defining the random model 314
  displaying 325
  displaying the model definitions 317
  double 315
  graph of fitted model 328
  model-checking plots 327
  predictions 326
  saving information from 329, 330
  structured dispersion model 313, 314
  tests for fixed terms 323
  tests for random terms 321
  Wald test 324
  with nonlinear parameters in fixed model 316
Hierarchical generalized nonlinear model 316
Hierarchical tree 899
Higher-order term in ANOVA 396
Histogram 26-28
  of residuals 300, 410, 675
Histogram of residuals 181
HLIST directive 907
Hodges-Lehmann estimate 91
Hot-deck imputation 12
HSUMMARIZE directive 910
HTML 5, 6
Hyperbola 351
ICE measure of species richness 151
Identification 779, 958
  interactive 956
  using a classification tree 948
Identification key 950
  construction 950
  display 953
  identification by 954
  saving information 955
Identifier 2
IDENTIFY procedure 956
Identity function 264

Immunity 288
Impulse-response function 1028
Imputation 12
Indented tree 944
Index of gamma distribution 263
Index plot 410, 675
Individual-based rarefaction 146
Inequality within a distribution 136, 154
Infinite parameter estimate 270
Inflated Latin square 537
Influence in regression 168, 272, 357
Information matrix
  from REML 713
Information summary 402, 470, 510
Initial block of a cyclic design 549
Initial classification 929
Initial classification for non-hierarchical
    clustering 934, 935
Initial fitted values
  for generalized linear model 275
Initial value
  for parameter 358, 364, 368
  for REML covariance model 724
Innovation variance 988
Innovations
  as prediction errors 986
Input and output 5
Input log 1
  adding information there 15
Interaction 232, 386, 394, 396, 442
  between contrasts 451, 452
  in curve fitting 352
  in nonlinear regression 366
  in regression 214
  involving function 214, 237
Interactive identification 956
Intercept 171
Interference between plots 511
  in a design 533
Interleaving Latin square 537
Interpretation of parameters 215
Intersection-union test 593
Interval data 276
Intervention analysis 1013
Interwoven loop design 569
Inverse
  relationship matrix 755
Inverse matrix
  in regression 169
Inverse matrix in regression 175
Inverse Normal distribution
  in regression 263, 269
Inverse polynomial model 263
Inverse relationship matrix 726, 753
Inverse-Normal distribution
  in regression 368
Inverting a design key 636
IRREDUNDANT directive 958
Irredundant test set 779, 958
Isotropic models 1073
Isotropic scale change 893
Isotropic variation 1070, 1081
Items 2
Iterative fitting
  of additive model 243, 246
  of curves 345, 350, 355
  of generalized linear model 271
  of nonlinear model 361
Iterative model 271
Iterative scaling 856
Iterative weights 271-273
Jaccard coefficient 785, 790
Jaccard index 919
Jackknife 137
Jittering of rugplots 33
Job 13
K-dominance plot 139, 140
K-means clustering 924
Kaplan-Meier estimate of survivor function 1050
KAPLANMEIER procedure 1050
Kappa coefficient 106
KAPPA procedure 106
KCONCORDANCE procedure 105
Kendall's coefficient of concordance 105
Kendall's rank correlation coefficient 20, 103
Kenward & Roger degrees of freedom 681
Kernel density
  estimation 43
  for circular data 47
KERNELDENSITY procedure 43
Key
  identification 950, 953, 954
KOLMOG2 procedure 92
Kolmogorov-Smirnov test 61, 89, 91
KRIGE directive 1079
Kriging 1031, 1066, 1067
  cokriging 1097
  cross validation 1077
  modelling a covariogram 1089, 1093
  variance 1068
KRUSKAL procedure 93
Kruskal-Wallis test 93
KTAU procedure 103
Kurtosis 50
Labels for estimates 175
Lagrange multiplier 1067
Large data set
  density plot 17
Large dataset 193
Large residuals
  in ANOVA 402
Lasso 160
Latent root 796-798, 841, 845, 848
  difference table 801, 802

scree diagram 801, 802
Latent variable
in factor analysis 853
LaTeX 5, 6
Latin square 391, 433, 434, 511, 513, 526-528, 530, 615
randomization 582
with split plots 434, 619
Lattice design 391, 478, 512, 513, 526, 541
Lattice square 390, 434, 512, 526
Law of diminishing returns 349
Law-like relationship 156
LCONCORDANCE procedure 109
LD50 261, 290
Least significant difference 404, 405, 462
for predictions 222
in REML 667
Least significant interval 390
Least squares 393
Least-squares scaling 889, 890
Lethal dose 290
Letter 2
Levene test 168
Leverage 168, 170, 175, 181, 193, 272, 355
Life-table estimate 1056
of survivor function 1055
Likelihood
explicit calculation 369
in curve fitting 345
in generalized linear model 269
in regression 264, 269, 362
in REML 650
in time series modelling 999, 1014, 1020
Likelihood in regression 269
Likelihood ratio test statistic for fixed model terms 683
Limit
on number of cycles 275
on order of contrasts in ANOVA 452
Limiting the order of treatment terms fitted by ANOVA 399
Lin's concordance correlation coefficient 20, 109, 610
Line-by-tester trial 644
Line-plus-exponential curve 348, 350
Line-printer graphics 16
Linear component
of smoothing spline 244
Linear contrast 450
Linear functional relationship model 160
Linear mixed model 647
definition for REML 651
general form of 648
properties of 649
Linear model
in analysis of variance 393
Linear parameter
in nonlinear regression 363, 366
Linear predictor 262, 273, 274, 278
Linear regression 156, 262
Linear relationship between explanatory variables 189
Linear variance model
in REML 723, 726
Linear-divided-by-linear curve 349, 350
Link function 258, 262, 264
Linkage disequilibrium 645
Loading
canonical variate 805, 813, 848
principal component 796, 799, 848
Local minimum 858
Locally weighted regression 246, 249 237
Loess 246, 249
separate lines for different groups 255
Log link function 265, 272
Log Normal distribution 57
Log series 141, 143
Log-likelihood ratio 269, 362, 369
Log-linear model 228, 262, 265, 276
Log-logistic distribution 1059
Logarithmic series distribution 55
Logistic curve 349, 350, 358
Logistic regression 262
Logit 262-264, 276
Logit link function 265, 272
Lognormal distribution 50, 1059
Longitudinal data 331, 1030
Loop design 569
Lorenz curve 21, 136, 154
LRV 4
LRVSCREE procedure 801
MacArthur fraction model 143
Mahalanobis distance 782, 783, 803, 805, 806, 841, 847, 888, 924, 926
Main effect 386, 396
in regression 213, 214
Main-effects design 558
Mallows Cp 208
Manhattan coefficient 786
Manipulating data 7
Manipulation of formulae 10
Mann-Whitney test 89, 90
sample size for 606
MANNWHITNEY procedure 90
MANOVA procedure 824
MANTEL procedure 791
Mantel test 791
Mantel-Haenszel statistic 122
Marginal test 211, 483, 484, 489
Marginal weights 224
Marginality 214, 217
Mass spectra 17, 19
Matern model

in Kriging 1073
Matrix 4
Maximal model 192, 193, 269
Maximal predictive classification 924, 926, 930, 932
Maximum likelihood 264, 345, 362, 924
Maximum Likelihood Program 362
MCNEMAR procedure 119
McNemar's test 20, 119, 121
sample size for 604
MCOVARIOGRAM directive 1093
MCROSSPECTRUM procedure 983
MDS directive 856
Mean 50
Mean posterior improvement 943
Mean square 394
Means
in an SSPM 780
in analysis of variance 393, 402, 404
in REML 665
replication of 405
Measure of association 779
Median sorting 901
Mega-environment 389
Message
display in dialog 15
Messages
about large residuals 402, 431
from regression 169, 171, 189, 356
in regression 192
Meta analysis 193, 760
random model for REML' 761
Method of analysis
in ANOVA 481
Metric scaling 840, 856
Michaelis-Menten 159
Michaelis-Menten law 351
Microarray
experiment 567, 569
Minimal cost complexity pruning 375
Minimizing a function 369
Minimum aberration design 518, 640
Minimum detectable effect 391, 596
Minimum spanning tree 901, 905
plotting 916, 917
Minkowski distance 790
Missing degrees of freedom 448
Missing factor combination 223, 226, 487, 503, 507
in prediction 227
Missing treatment effects 448
Missing value
estimation in ANOVA 448
in a stratified design 448
in analysis of covariance 447
in ANOVA 447
in ARIMA modelling 988, 995, 1003
in cluster analysis 900
in regression 162, 170, 191-193
in REML 296, 659, 756
in repeated measurements 1044
in similarity calculation 784
in time series 967, 971, 973, 988, 995, 1003, 1022, 1023
marker score 646
replacing 8
with ante-dependence 1044
Mitscherlich curve 349
Mixed model
definition for REML 651
equations 664, 715
general form of 648
properties of 649
Mixture design 563
Model checking 169
for generalized linear model 180
for regression 180
MODEL directive 162
Model for analysis of variance 387
Model formula
in ANOVA 396
in regression 214, 239
in REML 648
Model formulae in ANOVA 395
Model term 396
Model-checking
for hierarchical generalized linear model 327
Modelling a variogram 1072
Modelling variance structures
in REML 718
Modes
tables of 11
Monitoring
of iterative model 172, 246, 273, 355
Monotonic regression 857
Moving average model
in REML 722
MST 905
Multi-environment trial 644
Multi-site analysis 764
Multi-tiered analysis 437, 439
further output 439
Multidimensional scaling 840, 841, 856
Multinomial distribution 276
in regression 262, 263, 265, 269, 272, 368
Multiple comparison test 423, 491, 669
against control 388
Multiple correspondence analysis 864
biplot 866
Multiple linear regression 187
Multiple Procrustes analysis 896
Multiple responses 19
Multiple-response factor 11
Multiple-selection structure 13

Multiplicative effects
in generalized linear model 265, 272
Multivariate analysis 776
canonical variates analysis 810
in REML 722
parallel coordinates 19
Multivariate analysis of covariance 824, 825
Multivariate analysis of distance 829
Multivariate analysis of variance 824, 825, 1030, 1040, 1041
Multivariate graphics
parallel coordinates 19
Multivariate linear regression 210
Multivariate Normality
test of 60
Multivariate regression 824, 827, 1030
Mutually orthogonal Latin squares 527
MVAOV procedure 829
MVARIOGRAM procedure 1072
NAG Library 7, 12
Natural immunity 290
Natural mortality 288, 290
Nearest neighbours 903
Negative binomial distribution 52, 54, 55, 141
Negative variance components 649, 655, 709
Neighbour-balanced design 391, 512, 551-553
Nelder-Mead simplex algorithm 160
Nested models 269
Nested treatment effects 396
Nested-product operator 214
Nesting an experimental design within another design 619
Nesting operator
in ANOVA 397
in randomization 578
Newton algorithm 364
Newton-Raphson optimization 288
Neyman Type A distribution 55
Niche apportionment model 143
Niche division 143
Niche-apportionment 143
Niche-based model 136, 143
NLAR1 procedure 1048
Nominal data 276
Non-constant variance 165
Non-hierarchical classification 924
Non-inferiority test 584
in ANOVA 590, 594, 597
in regression 184
with t-test 70
Non-metric multidimensional scaling 856
Non-metric scaling 841, 856
Non-negligible model terms 628
Non-orthogonal design 399
Non-orthogonal split-plot design 391
Non-orthogonality 473, 480
in ANOVA 402
in regression 204
Non-standard distribution 278
Non-standard link function 278
Non-superiority test
with t-test 70
Nonlinear contrasts
in ANOVA 455
Nonlinear parameter 349
in generalized linear model 281
Nonlinear regression 361
Nonparametric analysis of variance 93
Nonparametric regression 243
Nonparametric tests 66, 89, 103, 108, 120, 154, 604, 606, 1054
for survival data 1054
Normal distribution 56
in nonlinear regression 363
in regression 156, 258, 263, 269, 366, 368
tests for 59
Normal plot 181, 300, 410, 675
Normal probability density 351
Normal quantile 410, 676
Normality
Shapiro-Wilk test for 59
testing 60
NORMTEST procedure 60
Np chart 131
Nugget variance 1073
Null model in regression 192, 194
Number of binomial successes 263
Number of binomial trials 263
Number of factors in ANOVA 399
Number of iterations
for generalized linear model 288
in missing-value estimation 448
Number of units used to form an SSPM 780
Numerical Algorithms Group 7
Octaves 141
Offset variate 165, 192, 263, 265, 352, 366
One-sample test 86
One-way analysis of variance 71
Operators in formulae
in ANOVA 395
Optimization 361
Option 1
name 2
settings 1
Order of contrast
in ANOVA 452
Order of fitting terms 188
Order of options 2
Ordering objects 840
Ordinal data 108
gamma statistic for 108
Ordinal response 263, 272, 276
Ordination 840, 874, 916
adding points 848

biplot 883
Orthogonal block structure 389, 483
Orthogonal contrasts 448, 452, 455
Orthogonal decomposition of design space 389
Orthogonal design 399
Orthogonal hierarchical design 390, 511, 513, 515
Orthogonal Latin squares 528
Orthogonal partial least squares 838
Orthogonal polynomial 239, 241, 450
Orthogonal polynomial contrasts over time 1035
Outlier 168, 783, 801
Output style 5, 6
Output to a text
from ADISPLAY 402
Over-parameterization 216
Overdispersion 264
Own code for nonlinear models 367
P chart 131
Paired samples 68
Papadakis analysis 389
Parallel coordinates 98, 99
Parallel curve analysis 352
Parallel list of parameters 2
Parallel nonlinear regression 366
Parallel regression lines 212
Parallelism in additive model 254
Parameter 1
name 2
settings 1
Parameter constraints
in curve fitting 350
Parameter estimate 175
Parameter in regression 161
Parameterization
in fixed effect models 653
of random effect models 654
of regression model 227, 357
Parametric bootstrap
for critical values in a REML analysis 689
Pareto
chart 11
distribution 58
optimal set 12
Partial aliasing 482
in regression 216
Partial autocorrelation 969
Partial autocorrelation function 966, 968, 970
sample 969
Partial canonical correspondence analysis 874
Partial confounding 470, 482
Partial effects 481
saving from ANOVA 462
Partial least squares 834, 835
orthogonal 838
Partial test 211
Partially-balanced designs
generation of factors for 614
Pattern in residuals 169
Pause execution 15
PCO directive 842
PCOPROCRUSTES procedure 896
PCP directive 795
PCPCLUSTER procedure 936
PDESIGN procedure 571
Pearson chi-square
from regression 175, 273
Pearson residuals 165, 272
Pedigree 755
checking 755
Percentage sum of squares 168, 388
Percentage variance accounted for 168, 171, 271, 388
Percentages
table of 11
Periodic behaviour 351
Periodogram 968, 973
Permutation test 419
for analysis of similarities 794
for analysis of variance- 420
for MANOVA 825
for one-way anova 72
for t-statistic 68, 70
to compare groupings 919
Peto-Prentice test 1054
pi-weights 1028
Pivot in ANOVA 482
Plackett Burman design 512, 558, 559
Plain-text output 5
Plan of a design 391, 526, 528, 538, 545, 548, 550, 555, 571-574
Plant breeding models 753
Plot factors 613
PLS procedure 834
PNTEST procedure 84
Pointer 4
duplicating 4
Poisson data
censored 160
Poisson distribution 55
in nonlinear regression 363
in regression 262-264, 269, 366, 368
Poisson test 84
Poisson-lognormal distribution 56, 141
Poisson-Pascal distribution 56
POL function
in ANOVA 450
Pólya-Aeppli distribution 56
Polynomial contrast 450, 455, 456
plotting 456
Polynomial ratio 350
Polynomial regression 237
Polytomous data 276
Pooling of deviance 272

Pooling sums of squares in regression 189
Power 391, 582
  in analysis of variance 391, 590, 593
  in regression 182
  of a t-test 584
  of an analysis of variance 591
  of contrasts in cross over designs 393
Power distance model
  in REML 723
Power fraction model 143
Power link function 264
Power model 729
Ppk index 136
Pre-whitening time series 1022
PREDICT directive 218, 271
Prediction 219, 222, 503
  after smoothing 246
  comparison of 229
  from generalized linear model 274
  from polynomial model 238, 243
  from regression tree 378
  from REML 747
  from unbalanced ANOVA 502
  in regression with ARIMA errors 1013
  in REML 665
  in time series modelling 1005, 1015
  missing factor combination 227
Principal component regression 801, 831
Principal components analysis 795, 811, 840, 841, 888
  number of significant components 645
  Tracy-Widom statistic 645
Principal coordinates analysis 840-842
  adding points 848
PRINT directive 1, 2
PRINT option
  in ANOVA 510
  in REML 296
  of ANOVA and ADISPLAY 402
Printed output
  from ANOVA and ADISPLAY 402
Printing a design 571
Probabilities in analysis-of-variance table 394
Probability distributions 17, 19, 26, 40
  estimation of parameters 48-59
  plotting 40
Probability of detection
  in regression 182
Probability plot 17, 19, 40
Probability-probability plot 41
Probit 258
Probit analysis 258, 262, 264, 278, 288
  with control mortality 283
PROBITANALYSIS procedure 288
Procedure 1, 14
  called by another procedure 14
  definition 14
Procrustes rotation 888, 896, 897
  generalized 892, 893
Product of two designs 392
Product of variates 214
Product-moment correlation 100
Products of experimental designs 619, 620
Profile plots
  of repeated measurements 1033
Program 1
Programming 13
Projection matrix 12
Properties of a design 483
Proportional hazards model 1062
  displaying output 1064
  fitting 1062
  modifying the model 1065
  saving information 1064
Proportional replication 454
Proportional weighted replication 399
Proportional-hazards model 263, 276
Proportional-odds model 263, 276
Proportions
  analysis of 262
Pruning a classification tree 946
Pruning a tree 375
Pseudo-factor 392, 476, 478, 482, 524, 537, 613, 636, 637
  for design 616
  generating 480, 614
  to represent basic contrasts 640
Pseudo-factorial operator 476
Pseudo-term 476, 478, 479
psi-weights 1028
PTFCLUSTERS procedure 1107
PTFILLCLUSTERS procedure 1112
Punctual kriging 1080
Pythagorean coefficient 786
Pythagorean distance 888
Q-method 784
Q-Q plot 41
Q-techniques 776
QDISCRIMINATE procedure 821
QTL
  eigenvalue analysis 645
  Flapjack project file creation 645
  missing marker score 646
  Tracy-Widom statistic 645
Quadratic contrast 450
Quadratic surface
  stationary point 124
Quadratic-divided-by-linear curve 349, 351
Quadratic-divided-by-quadratic curve 349, 351
Qualitative variable 212, 784
Quantal response 258
Quantile regression 159, 380, 381
Quantile-quantile plot 41
Quantitative variable 784

Quartimax rotation 811
Quasi-complete Latin square 533
Quasi-likelihood 264
R-squared statistic 168
R-techniques 776
R0KEEP procedure 342
R2 statistic 168
Rand index 919
Random coefficient regression 721, 742, 756
Random effects
  definition 647, 654
  estimates 649, 650, 664
Random faction model 143
Random order 578
Random permutation of basic contrasts 630
Randomization 386, 392, 428, 577, 578, 581, 582
  of a design 578, 581, 582, 619
  of a Latin square 582
  test 792
RANDOMIZE directive 578
Randomized block design 391, 427, 428, 434, 440, 511, 513
  randomization 578, 579
Randomized complete block design 515
Randomness of a sequence of observations 88
Rank
  Steel's test for 93, 95, 427
Rank correlation 102
Rank correlation coefficient 20
  Kendall's 103
Rank/abundance plot 139, 140
RAR1 procedure 1045
Rarefaction 136, 145
Ratio of polynomials 350
Rational function 350
Rayleigh's test of uniformity 25
RCHECK procedure 180
RCOMPARISONS procedure 229
RCURVECOMMONNONLINEAR procedure 359
RCYCLE directive 274, 358
RDA procedure 869
RDESTIMATES procedure 235
RDISPLAY directive 171, 188
Re-analysis sweep in ANOVA 482
Rectangular hyperbola 351
REDUCE directive 788
Reduced sampling effort 146
Reduced similarity matrix 789
Redundancy analysis 869
Reference level 215
Reference-level design 567
Reflection 889, 891
REG function
  in ANOVA 452
Regionalized variables 1066
Regression 156
  abbreviated output 169
  comparisons within tables of means 234
  contrasts amongst means 229
  design matrix 169, 193
  diagnostics 180, 185
  exact test 186
  exclusion of units 191
  fitting individual terms 255
  inverse matrix 169
  missing value 191
  model, plotting 178
  model, saving 176
  monotonic 857
  nonlinear parameters 281
  permutation test 185
  plotting 261
  plotting estimates 235
  power in 182
  principal component 801
  profile likelihood confidence intervals 159
  random permutation test 185, 307
  saving estimates 189
  saving results 190
  saving results in a spreadsheet 176
  structured dispersion model 313, 314
  variance inflation factor 169
  Wald test 198
  weighted 343
  with AR1 errors 1045
  with correlated (ARIMA) errors 1011
  with power-distance correlation model 1045
  zero-inflated 338, 342
Regression save structure 166, 172, 229
Regression tree 372, 374, 375
  constructing 372
  displaying 374
  forming values for 375
  prediction from 378
  pruning 375
  saving information 379
RELATE directive 850
Related samples 20, 120
Relating groups to variables 909
Relationship matrix 755
Relative abundance of species 143
Relative potency 261
REML 642, 658
  Akaike information coefficient 707, 709, 710, 712
  algorithm 717
  algorithm, controlling 717
  all subsets of the fixed terms 643, 697
  approximate stratum variances 713
  average-information algorithm 660, 717
  bootstrap for fixed effects 685
  canonical decomposition of the information matrix 713

censored data 644
checking pedigree 642, 755
checking standardized residuals 697
checks of random effects 699
comparison of predicted means& 751
convergence criterion 717
effect of absorbing factor on standard errors 666
effects 664
equation ordering 717
error structure 649
F-test of random effects 643
Fisher method 714
fitted values 659
investigating the fixed model 643, 697
large residuals 697, 699
least significant differences 667
likelihood function 650
likelihood ratio test for fixed terms 683
likelihood ratio test for variance component 709
line-by-tester analysis 644
method 650
method for residuals and fitted values 660
missing values 659
model 642
model-definition structure 644
outlier 701
output 659
permutation tests for random terms 644
plotting effects 673
plotting means 669
predicted means 665
prediction from 747
random model for meta analysis' 761
random permutation test 692
recycled estimation 684
residuals 659
residuals in field layout 677
restriction of units 659
save structure 659, 663, 684, 767
saving fitted values and their s.e.'s 771
saving fixed tests 774
saving results from 255, 468, 504, 765, 771, 773, 774
saving results from the analysis of a series of trials 644
saving results in a spreadsheet 773
Schwarz information coefficient 707, 709, 710
screening tests 696
shrinkage 665
steplengths 717
testing fixed terms 678
testing nested models 683
testing of submodels 660
Wald test for fixed terms 681
weighted analysis 660
REML directive 658
Removing regression variables 194
Reparameterization of nonlinear model 364
Repeated measurements 331, 332, 389, 1030, 1033, 1040, 1046
analysis of polynomial contrasts 1035
analysis of variance 1037-1039
ante-dependence 1041, 1043
balanced 1033
missing values 1044
profile plots 1033
with REML 719, 727, 745
Repertoire of designs 526
Replacing strings within a text structure 9
Replicated Latin squares 429, 434
Replication 405
in regression 222, 503
required in analysis of variance 391
required in ANOVA 589
saving from ANOVA 462
Required model terms 628
Residual component 654
Residual maximum likelihood 642
Residual plots 487
from ANOVA 410, 411
from regression 180, 185
from REML 675, 677
Residual sum of squares 175, 888
Residual term
in REML 746
Residual variance
as estimated innovation variance 988
Residual variation in REML 654
Residuals
as estimated innovations 986
canonical variate 805
from all strata in ANOVA 460
in analysis of variance 402
in ANOVA 393, 407
in field layout 411, 432
in principal components analysis 799
in regression 164, 165, 170, 193, 272
in REML 659
in time series 994, 1003
principal component 797, 798
saving from ANOVA 400, 460
saving from regression 175
spatial pattern of 411
Residuals in field layout 677
Resolution number 638
Resolution of a design 638
Response
saving from ANOVA 462
Response category 276
Response surface design 124, 392, 511, 560, 564
Response to a treatment 407
Response variate 156, 170, 192, 271, 363, 369
Restricted maximum likelihood 642
Restricted vectors

in ANOVA 399
in principal components analysis 801
in regression 162, 170, 192
in REML 659
in time series 967, 971, 972, 988
Restriction 8
RFUNCTION directive 357
RGRAPH procedure 178, 261
RIDGE procedure 831
Ridge regression 193, 831
RKEEP directive 172, 271, 355
RKESTIMATES directive 189
RLIFETABLE procedure 1055
RLOESSGROUPS procedure 255
RMULTIVARIATE procedure 827
ROBSSPM procedure 782
Robust estimate of sum-of-squares-and-products matrix 779, 782, 783
Rose diagram 47
ROTATE directive 889
Rotation
Procrustes 888
Rotation of factor loadings 811
RPERMTEST procedure 185
RPHCHANGE procedure 1065
RPHDISPLAY procedure 1064
RPHFIT procedure 1062
RPHKEEP procedure 1064
RPOWER procedure 182
RSCREEN procedure 210
RSPREADSHEET procedure 176
RSTEST procedure 1054
RSURVIVAL procedure 1058
RTCOMPARISONS procedure 234
RTF 5, 6
Rugplot 32
RUGPLOT procedure 32
Rules of syntax 1
Runs test 88, 168
RUNTEST procedure 89
RWALD procedure 198
Ryan/Einot-Gabriel/Welsch multiple range test 425
Sample autocorrelation function 967, 968
Sample cross-correlation function 969, 971
Sample size 20, 391, 582
for a sign test 603
for a t-test 583
for analysis of variance 391, 589
for binomial test 599, 601
for Lin's concordance correlation coefficient 610
for Mann-Whitney test 606
for McNemar's test 604
to detect correlation 608
Sample spectrum 968, 969
Sample statistics 50
Sample-based rarefaction 146
Sampling effort 146
Satterthwaite's method 431
Save structure
for REML 659, 663, 684, 767
for time series 996, 1002, 1005, 1014
in ANOVA 400, 402, 445, 460
Saving fitted values
from ANOVA 400, 460
Saving information
from ANOVA 458
Saving output from unbalanced anova 500
Saving residuals
from ANOVA 400, 460
Saving results
from regression 190, 228, 504
saving from regression 174
Saving results from regression 273
SBNTEST procedure 599
Scalar 4
Scaled deviance 269
Scaling in curve fitting 350
Scatterplot 97, 98
Scheffe test 425
Schematic boxplot 31
Schwarz information coefficient 707, 709, 710
Schwarz information criterion 208
Score
canonical correlation 839
canonical variate 805
principal component 796, 799
principal coordinate 845, 848, 888
SCORRELATION procedure 608
Scree diagram 799, 843
of latent roots 801, 802
Screening test 210, 389, 483, 487, 489
Screening tests 696
SDISCRIMINATE procedure 818
Seasonal ARIMA model 992
Seasonal autoregression 993
Seasonal differencing 993
Seasonal moving average 993
Seasonal period 993
Seasonal transfer function model 1019
Seed for randomization 580
Select Design menu 513
Semi-Latin square 391, 512, 537, 538
Semivariance 1067
Sense of curve 348
Sensory analysis 893
Separate constant terms 212
Separate slopes 212, 217
Separation plot 160
Sequential breakage model 143
Sequential formation of an SSPM 193, 781
Serial correlation 345
Seriation 840
Set calculations 7

Shapiro-Wilk test 59, 390
Shepard diagram 858
Short wordlengths 2, 3
Shrinkage
  in REML 665
Sidak test 425
Sigmoid curve 350
Sign test 87
  sample size for 603
Significance of changes 119
Significance test for latent roots 805
Significance test of latent roots 797
SIGNTEST procedure 87
Similarity 784, 790, 899
  between groups 904
Similarity coefficient 784
Similarity level 900
Similarity matrix 840, 842
  reduced 789
Simple lattice 478
Simple matching coefficient 786, 790
Simplex method 372
Sine curve 351
Single-linkage 899-901
Singular value decomposition 888
Site scores
  in redundancy analysis 870
Six sigma 21, 124
Skeleton analysis-of-variance table 400
Skew-symmetry 886, 890
Skewness 50
SKEWSYMMETRY procedure 886
SLCONCORDANCE procedure 610
Slow convergence 271
SMANNWHITNEY procedure 606
SMCNEMAR procedure 604
Smoothed effects
  list of 246
  nonlinear components 246
Smoothed spectrum estimates of time series 976, 977
Smoothing
  in time series 1021
Smoothing spline 243, 249, 288
Smoothness 243
SMOOTHSPECTRUM procedure 976
SP plot 41
Space filling design 391
Sparse matrix methods for REML 660
Spatial analysis of field experiments 720, 735
Spatial covariance 1067
Spatial modelling 1066
Spatial point patterns 1031, 1103, 1107
Spatial statistics 1031, 1103, 1107
SPCAPABILITY procedure 135
SPCCHART procedure 133
SPCUSUM procedure 128
SPEARMAN procedure 102
Spearman's correlation coefficient 102
Species abundance 21, 136, 139, 141, 143, 145
Species accumulation curve 149
  plotting 21, 136, 148
Species diversity 21
Species richness 21, 136, 149, 150
Spectography 351
Spectral analysis 972
  of multiple time series- 983
SPEWMA procedure 130
Spherical covariance model
  in REML 723
Spline smoothing 243
Split-line model 160
Split-plot design 397, 428, 511, 513, 515
  randomization 578
Split-split-plot design 511, 515
SPNTEST procedure 601
SPPCHART procedure 131
Spreadsheet
  plan and data of experimental design4 574
SPSHEWHART procedure 125
Square lattice design 391, 541
SSIGNTEST procedure 603
Sspider-web plot 17
SSPM 4, 192, 193, 216, 799, 854
  forming 780
  means 780
  sums of squares and products 780
  within groups 780
SSPM directive 779
Stabilized probability plot
  41
Stable parameterization 345
Stacking sets of vectors 8
Stagewise regression 202
Standard curve 345
Standard Design menu 513
Standard deviation
  bias correction for 127, 129, 131
  in Gaussian curve 351
Standard errors 473
  for contrasts 452
  in analysis of covariance 443
  in generalized linear model 272
  in generalized linear models 264
  in linear regression 169
  in nonlinear regression 367, 371
  in regression 175
  of differences between means 404, 431, 473
  of differences for predictions 222
  of mean, saving 462
  of parameters 169
  of predictions 223, 274
Standard order of factor values 612
Standardization

of effects 221, 224
of matrix 890
of residuals in regression 165, 272
of similarity 790
Standardized residuals
in regression 272
Star plot 17
Star point 554
Starting value 345
Stationary point of quadratic surface 124
Statistical process control 124
c chart 133
capability statistics 135
CUSUM table 128
exponentially weighted moving average control chart 130
mean chart 125
np chart 131
p chart 131
range chart 125
Shewhart chart 125
standard deviation chart 125
u chart 133
Status of ANOVA 467
STEEL procedure 95
Steel's test 21, 95, 389, 427
Steepest descent algorithm 857, 858
STEM procedure 34
Stem-and-leaf plot 34
STEP directive 200
Step length 364
Stepwise regression 202, 204, 206
Stratified experimental designs 427
Stratum 428
Stratum variance 402
estimation in ANOVA 475
saving from ANOVA 466
Stress 857
String token 2
Structure
of an experimental design 386
Structured dispersion model 313, 314
STTEST procedure 583
Student Version 4
Student's t-test 68
Student-Newman-Keuls test 425
Studentized range 425
Sub-plot 429
Subset 8
Sum of squares 394
due to treatments 394
for non orthogonal treatment terms 480, 481
for pseudo-term 476
in regression 192
Summarizing regression 219
Summary statistics 22, 24, 50, 81, 84
mode 11
Summation of effects 228
Summation operator 214
Sums of squares
saving from ANOVA 463
Sums of squares and products 780
between covariates 464
Superbinomial distribution 264
Supermultinomial distribution 264
SuperPoisson distribution 264
Survey
merging strata 12
Survival analysis 1030, 1050, 1054, 1055
exponential distribution 1058
extreme-value distribution 1058
Kaplan-Meier estimate 1050
log-logistic distribution 1058
lognormal distribution 1058
proportional hazards model 1062
Weibull distribution 1058
Survivor function
life-table estimate for 1055
Suspend execution 15
Sweep in ANOVA 481, 482
SWITCH directive 193
Symmetric matrix 4
Syntax 1
System word 2
Systematic order of factor values 612
t-statistic
in regression 169, 188, 215
t-test 66
plot power and significance 584
sample size for 583
Table 4
Table manipulation 11
Table of effects
saving from ANOVA 462
Table of means
for non orthogonal treatment terms 481
for pseudo-term 476
in ANOVA 473
plotting 413, 414, 497, 669
saving from ANOVA 462
Table of modes 11
Table of percentages 11
Table of residuals
in ANOVA 407, 431
saving from ANOVA 462
Tables
plotting 11
Tables of modes 11
TALLY procedure 36
Tally table 36
Tarone-Ware test 1054
Tau 103
Taxon 959
TDISPLAY directive 1001

Temporary change to model 197
Temporary file 15
Terminology 1
TERMS directive 191
Test for equivalence 583, 584
  in ANOVA 590, 594, 597
  in regression 184
Test for non-inferiority 584
  in ANOVA 590, 594, 597
  in regression 184
Tests for Normality 59
Tests of univariate and multivariate normality 60
Text 4
  changing case 9
  forming from row or column labels of matrix 11
  forming from scalars, variates, texts, factors or pointers 9
Tied data 857
Time series 963
  autocorrelation function 1027
  calculation of deviance 997
  constraining parameters 995, 1015, 1020
  convergence criteria 996
  cosine transformation 973
  deviance 1000, 1003, 1015
  estimation 1012, 1014, 1020
  exact likelihood method 1000
  filtering 1021
  forecasting 1005, 1020
  forecasts from VARMA model 964
  forming preliminary estimates 1024
  generalized form of model 1028
  harmonic analysis! 979
  impulse-response function 1028
  index 963
  initialization for forecasting 997, 1006
  least-squares likelihood method 1000
  marginal likelihood method 1015, 1020
  missing values 967, 971, 973, 988, 995, 1003
  model checking 988, 1008
  moment estimators 1024
  one-step estimation 998
  order and speed 973
  output 995
  output to other channels 1002
  parameter reference numbers 996, 1015
  pi-weights 1028
  pre-whitening 1022
  prediction 1015
  psi-weights 1028
  recycled estimation 995
  residuals 994, 1003
  restriction on units 967, 988
  restrictions on units 971, 972
  revising forecasts 1007
  save structure 996, 1002, 1005, 1014
  save structure, accessing 997
  save structure, resetting 997
  scores 1003
  smoothing 1021
  spectral analysis of multiple time series9 983
  standardized forecast errors 1008, 1009
  testing nested models 1015
  tests of model parameters 998, 1004
  tolerance 996
  univariate 963
  VARMA model 964
  weights 995
  with explanatory variables 1011
  zero-step estimation 998
Time-course microarray experiment 569
TKEEP directive 1002, 1020
Tobit method
  for linear mixed model 644
  for censored Poisson data 160
  Poisson-log generalized linear mixed model 292
  to fit a Poisson-log hierarchical generalized linear model 312
Tolerance
  for collinearity 193
  for zero sums of squares in ANOVA 482
  in ANOVA 400
  in time series modelling 996
TOST procedure 593
Transfer-function model 1011
  accessing components 1020
  bias in estimates 1015
  Box-Cox transformation 1014
  definition 1016
  delay parameter 1018
  errors for explanatory variables 1014
  estimation 1012, 1014, 1020
  evaluation of likelihood 1014
  forecasting 1013, 1015, 1020
  lags of 1019
  likelihood function 1014
  marginal likelihood method 1014, 1020
  minimizing transients 1019
  multi-input 1016, 1019
  non-seasonal 1018
  orders of 1019
  parameter estimation 1012
  parameters of 1019
  preliminary parameter estimation 1026
  seasonal 1019
  specification of 1018
  specifying input series 1014
TRANSFERFUNCTION directive 1013, 1019
Transformation
  back onto the natural scale 274
  of parameters 362
  of predictions 274
  of response 162, 258, 262
Treatment effects 402

Treatment factors 628
Treatment formula 395, 396
  saving 461
Treatment model 386, 395
Treatment term 396
Treatments 386
TREATMENTSTRUCTURE directive 395
Tree
  hierarchical 899
  in indented form 374
  pruning 375
  utility procedures 12
Trend
  Cochran-Armitage test for 123
Triangle inequality 784
Triplot 883
Trojan square 537
TRY directive 197
Trying effect of variables 197
TSM 4
  in ARIMA modelling 987, 991
  in filtering 1022
  printing 991
  transfer-function modelling 1018
TSM directive 991, 1018
TSUMMARIZE directive 1027
TTEST procedure 66
Tukey confidence intervals 425
Two-colour microarray experiment 568, 569
Two-dimensional power model 736
Two-phase experiment 436
Two-sample test 89
Two-straight-line model 160
Two-way analysis of variance 74
Two-way anova 74
  further output 76
  saving information 77
Two-way table 114
Typical unit 932
Unadjusted analysis of variance 445, 485, 486, 488-490
Unbalanced design 75, 78, 388, 478, 485, 486, 488-490, 642
  advice about possible causes 509
Underdispersion 264
Unequal variances in REML 724
Uniform covariance structure 1037
Unit labels
  for a design 392
  in regression 170
Unit score 814
Units structure
  in regression 170
Units used in regression 191
Unrandomized factors 579
Unstable model 239
VAIC procedure 710
Variance 50
  known in generalized linear model 264
  of residuals 168
  of response 164
  saving from ANOVA 463
Variance component 296
Variance components 475, 642
  constraints on 649, 652, 656
  fixing 655
  initial values 652, 655
  linear equality constraints 656
  negative 649, 655, 709
Variance function 262, 263, 278
Variance inflation factor 169, 831
Variance model
  in REML 718
Variance of response 223
Variance ratio 196, 394, 429
  in regression 189
Variance shift outlier model 701
Variance-covariance matrix 175
  forming 11, 777
Variance-mean relationship 264
Variate 4
  forming from a matrix 11
Varimax rotation 811
Variogram 1067
  forming 1068
  in REML 742
  modelling 1072
  plotting 2d 1096
  plotting fitted models 1078
VARMA model 964
  forecasts 964
  plotting 964
VBOOTSTRAP procedure 685
VCHECK procedure 697
VCOMPONENTS directive 651
VCRITICAL procedure 689
VCYCLE directive 717
VDEFFECTS procedure 673
VDISPLAY directive 662
Vector autoregressive moving average model 964
  forecasts 964
Versions of a design 524
Vertical asymptote 356
VFIXEDTESTS procedure 774
VFPEDIGREE procedure 755
VFRESIDUALS procedure 771
VGRAPH procedure 669
VKEEP directive 765
VLSD procedure 667
VMCOMPARISON procedure 669
Von Mises distribution 25
VORTHPOLYNOMIAL procedure 1035
VPEDIGREE directive 753
VPERMTEST procedure 692

VPLOT procedure 675
VPREDICT directive 747
VRACCUMULATE procedure 711
VRCHECK procedure 699
VRESIDUAL directive 763
VSCREEN procedure 696
VSOM procedure 701
VSPREADSHEET procedure 773
VSTATUS directive 727
VSTRUCTURE directive 718
VTCOMPARISONS procedure 751
W statistic 59
Wadley's problem 289, 291
Wald test 296
  for hierarchical generalized linear model 324
  for regression 198
  in REML 681
Warnings about large residuals
  in ANOVA 402
Wavelength 351
Weibull distribution 57, 1059
Weight
  in ANOVA 509
Weight matrix in regression 343
Weight variate
  saving from ANOVA 461
Weighted analysis of variance 399
Weighted replication 405
Weights
  in curve fitting 351
  in prediction 224, 225
  in regression 164, 170, 192, 272
  in REML 660
  in time series 995
Welch's analysis of variance 70
Welch's t-test 70
Whittaker plot 139
Whole-plot 429
WILCOXON procedure 86
Wilcoxon test 86, 87
  for survival data 1054
Wilks Lambda 827
Wind-rose diagram 47
Wine tasting 436
Within-group means in an SSPM 780
Within-group SSPM 780, 803, 841
Within-groups analysis 192
Wordlengths 2, 3
Workspace 4
  for REML 660
WSTATISTIC procedure 59
Yates definition of response 407
Youden square 534
Zero in generalized linear model 269
Zero-inflated regression 338, 342
Zero-inflated regression models 159
Ziggurat 913
Ziggurat-degree 914
Zigzag method 788
Zipf model 141
Zipf-Mandelbrot model 141