GENSTAT NEWSLETTER
NO. 5
JANUARY 1979

This newsletter is being distributed at the same time as Genstat 4.02. The most important enhancement in the new release is the inclusion of optimization procedures which were available in Genstat 4.01 on a trial basis and described in Newsletter No. 4. There have been alterations to the regression output (see article by Peter Lane), reductions in the size of user files (see article by Alan Todd), and several new functions can be used with the CALCULATE directive. Nine new macros have been added to the macro library (see article by Colin Banfield). Complete specifications for these new macros, and for the old ones, can be supplied by the Genstat Secretary; as there is only a limited supply you will be asked to make your own copies.

Since the last newsletter Genstat has been converted for use on DEC and Burroughs computers; also the UNIVAC conversion for the ASCII Fortran compiler was carried out at the University of Copenhagen. We are very grateful to the following for their help:-

Mr. John Byrne	University of York
Dr. Brian Cox	University of Otago
Mr. Jerzy Wasniewski	University of Copenhagen

SOME CHANGES IN RELEASE 4.02

Smaller userfiles

When Genstat data structures were stored on userfiles in the past, space was reserved in the file catalogue for all the data structures used in the job even if only a small proportion of those data structures were stored. In 4.02 catalogue space is reserved for the stored structures only and, in jobs where a large number of identifiers are used but only a few structures stored, there can be a large reduction in file space required.

This saving in space does not occur with files created by setting the option COMP = DUMP.

Alan Todd
R. E. S.

Changes to regression facilities

Several changes have been made in the regression section, mostly affecting the form in which results are output from the x-set statements 'FIT', 'ADD' etc.

One change which will be immediately noticeable to many users concerns the representation of factorial effects in regression models, e.g. when a grouped regression or an unbalanced factorial experiment is analysed. In previous releases, the printed factor effects have been differences between each level of the factor and the last level. Thus if TREAT is a factor with 3 levels, the statement

```
'FIT/PRINT = C'  TREAT
```

would have led to 3 coefficients being printed, e.g.

```
CONSTANT   4.6
TREAT 1   -1.2
TREAT 2    2.3
```

In the new release, the comparison is made with the first level of the factor instead of with the last. There are three reasons for this change:-

(1) When a standard or control treatment is included in an experiment, it is natural to represent it as the first level of a factor, and comparisons will usually be required with the control.

(2) The program GLIM (release 3) has also made this change.

(3) The new form is slightly more efficient computationally.

The statement above will now produce coefficients as follows:-

```
CONSTANT   3.4          (= 4.6        -1.2)
TREAT 2    3.5          (= 4.6  -3.4  +2.3)
TREAT 3    1.2          (= 4.6  -3.4       )
```

and similar differences will be noticed in interaction terms. If the option setting DVSET = F has been used, then there will be no change in the form of output. I hope these modifications will not inconvenience users unduly.

A further change has been made simply to make the output more informative. Previously, regression coefficients for factors (as above) have not been labelled with level names, even if the user specified such names. This omission has now been rectified, so that for example if the factor above was declared by the statements

```
'NAME'  NT = NONE, DRUG, PLACEBO    'FACTOR'  TREAT $ NT
```

the regression coefficients from the FIT statement above would be

```
CONSTANT          3.4
TREAT DRUG        3.5
TREAT PLACEBO     1.2
```

Another change is one that will not affect most users. A new option, ORDER, is now available for the x-set statements. If ORDER = MAX is specified, this will result in regression coefficients, and the corresponding rows and columns of inverse and variance-covariance matrices, being printed (or saved) in the order in which terms were specified in the maximal models (i.e. in the 'REGRESS' or 'TERMS' statement). Coefficients for terms which are not in the current model (e.g. the y-variate) are represented by zeroes.

Lastly, the RECYCLE option of the x-set directives has been modified. It is no longer necessary to save the linear predictor and iterative weight variates for a generalized linear model when a second model is to be fitted using RECYCLE = Y. The fitted-values variate is now sufficient.

Peter Lane
R. E. S.

NEW LIBRARY MACROS

Of the nine additions to the Macro Library, five are useful in discriminant analysis. These are ALLOCATE, DSQUARE, MISALLOG, MISALLOP and JACKNIFE. ALLOCATE is used for allocating units to groups by finding to which group they have smallest Mahalanobis distance, assuming the within-group covariance matrices to be homogeneous. DSQUARE obtains the asymmetric matrix of Mahalanobis squared distances, D^2 when the within-group covariance matrices are not assumed to be homogeneous. MISALLOG and MISALLOP check whether a given allocation of a number of units to a number of groups is correct by reallocating the units using a modified form of the Mahalanobis squared distance. MISALLOG assumes that the within-group covariance matrices are not homogeneous, whereas MISALLOP assumes the contrary. JACKNIFE also checks for misallocation using a jack-knifing technique.

The other four new macros are CVAID, D3PLOT, CORRESP and ASYMANAL. CVAID produces considerably more printed output for a canonical variate analysis than the CVA directive alone. This includes sums-of-squares and products matrices, tests of homogeneity, within-group principal components, tests for misallocation, and canonical-variate plots. D3PLOT plots the values of three variates within a single frame so as to give the appearance that the points of the plot are located in three-dimensional space. Perpendiculars from each point have to be drawn onto a base drawn by the macro so as to give the sense of perspective. CORRESP obtains a correspondence analysis, which is a single ordination of both the n row and v column entities indexing an n x v contingency table. ASYMANAL analyses the skew-symmetric component of an asymmetric matrix such that the skew-symmetry values are represented graphically by the areas of triangles.

Colin Banfield
R. E. S.

HINTS ON THE USE OF GENSTAT

Orthogonal polynomials

We recently wanted to use orthogonal polynomials to describe and compare the response surfaces of different measurements in a storage experiment, in which there were three factors (first and second storage times = x_1 and x_2, second temperature = x_3). Because the design was non-orthogonal we were unable to use ANOVA with a polynomial sub-model. The orthogonal polynomials were therefore derived by the transformation $P = L^{-1}X'$, where X is the matrix (N x K) of regressor variates before orthogonalisation and L is a lower triangular matrix (K x K) satisfying $LL' = X'X$. These operations are easily carried out in Genstat using the matrix function CHOL. The method is as follows. First define the regressor variates X(1...K) of length N. These must be in the order in which one wishes to orthogonalise, usually with linear terms first and the least important higher order terms last. In our example they were x_1, x_2 followed by the calculated variates x_2x_3, x_1x_2, $x_1x_2x_3$, x_1^2, x_2^2, $x_1x_2^2$ and $x_1^2x_3$ (K = 9). The Genstat steps are then:

```
'SET'  SX = X(1...K)  :  SPOL = P(1...K)     ''orthogonal polynomials''
'DSSP'  M1 $ SX
'SYMM'  M2 $ K
'MATR'  M3 $ K, K  :  M4, M5 $ K, N
'SSP'  M1
'EQUA'  M2 = M1  :  M4 = SX
'CALC'  M5 = PDT((M3 = INV(CHOL(M2))) ;  M4)
'EQUA'  SPOL = M5
'PRINT/P'  SPOL
'PRINT' M3
'REGR'  SY, SPOL        ''SY = set of response variates''
'Y'  SY
'FIT'  SPOL             ''Regression of each response variate on the
                          K orthogonal polynomials''
```

The polynomial values are scaled to give unit sums of squares. Consequently the regression coefficients of a given Y on P(1...K) all have a common standard error (= residual SE of y). This can be useful in saving space when the coefficients and their SE's are tabulated. To obtain the coefficients in original units, divide each by the corresponding diagonal element of matrix M3.

It is not always necessary or profitable to orthogonalise all the regressor variates. In our example we found it more informative to start with a straight multiple regression of y on x_1 and x_2 and then orthogonalise the rest (starting with x_2x_3). The X-variate list in REGR and FIT was thus altered to X(1,2), P(3...K).

Chris Baines
L. A. R. S.

Using the 'FOR' directive

The 'FOR' directive is intended to allow a set of statements to be repeated several times, but referencing a different set of structures within these statements each time. For example, the statements

```
'FOR'  Y = V(1...10) ;  X = W(1...10) ;  HY = HV(1...10) ;  HX = HW(1...10)
'GRAPH/HY, HX'  Y ;  X
'REPEAT'
```

can be used to produce ten separate graphs, rather than giving a sequence of ten 'GRAPH' statements. It is sometimes required to use a 'FOR' loop to repeat statements without any change in the structures referenced each time, for example

```
'FOR'  I = 1...10
'BEST'  X(1...20)
'REPEAT'
```

can be used to fit the 'best' ten covariates in a regression model. The numbers 1 up to 10 in the 'FOR' statement are stored by the Genstat compiler as unnamed scalar structures, since it expects to find structure identifiers on the right hand side of the forlist. This can sometimes be a problem, since the number of identifiers may become large. To avoid this, replace the list of numbers by any list of identifiers, or, in particular, one identifer repeated many times. In the example above, if Z is the identifier of some structure (other than type SET or DUMMY) then the forlist may be replaced by

```
'FOR'  I = 10(Z)
```

Peter Lane
R. E. S.

Restricting the unrestrictable - again

In newsletter No. 2 I gave a technique for producing subsets of data for use in directives that do not acknowledge the concept. Since then certain developments have permitted a much neater solution to the problem:

```
'INTEGER'  I
'SCALAR'  N
'RESTRICT'  X $ .... ;  I
'CALC'  N = NVAL (I)
.
.
.
'RUN'
'VARIATE'  Y $ N
'COPY'  Y = X $ I
```

Howard Simpson
R. E. S.

Controlling printing in macros

A disadvantage of Genstat macros is the difficulty of controlling printed output. A macro containing directives having the PRINT = letters facility can be controlled by allowing the letters specified to be changed either by a SET or ASSIGN substitution. However, much macro printing is done through a series of PRINT directives and can be controlled only by jumping round the PRINT directives, as specified by parameters given before entering the macro..

Four methods, each using a different amount of store, have been suggested:-

1.
```
'SET'  PR(1, 3, 5, 8, 10) = YES
-------------------------------
'JUMP'  L*(PR(1) .ISNT. YES)
'PRINT' .........
-------------------------------
```

As many identifiers PR(1, 2, 3...) as are referred to in the macro plus an entry for YES will be set up in the directory. Identifiers not referred to in the SET directive (e.g. PR(2)) will be assumed to identify variates of standard length so the relevant JUMP statements will operate correctly.

```
2.      'SCALAR'  ONE = 1
        'VALUES'  VAR = 1, 0, 4(1), 3(0), 1
        ---------------------------------
        'JUMP'  L*(ELEM(VA ;  4) .NE. ONE)
        'PRINT' .........
        ---------------------------------
```

This is simple and requires only one structure, the variate VAR, to be set up in the directory. There is no printing if all values of VAR are declared to be 0.

```
3.      'ASSIGN'  PRINT = A, B, C, D, E, F $ IPRIN
        ----------------------------------------
        'JUMP'  L*(PRINT .ISNT. A)
        'PRINT' .........
        ----------------------------------------
```

Possibly useful for selecting just one of many possible options governed by a computed scalar IPRIN. The identifiers A, B, ... are in the directory.

```
4.      'SET'  PRINT(1) = LS
        -------------------
        'PCP/PRINT = PRINT(1) ...
        -------------------
```

This is the standard way of setting print-options of directives used within macros. There is no conflict when a SET name is the same as a key-word. For example in the above, PRINT(1) could be replaced by PRINT.

John Gower
R. E. S.

Analysis of Covariance for each level of a factor

The analysis of covariance assumes that the relationship between the variate and the covariate is the same over all levels of factors in the model. It is useful, therefore, to be able to calculate separate regression coefficients for each level of a treatment factor. If the regression coefficients associated with each level are the same within the limits of error the user can proceed in the knowledge that his assumptions are valid.

This splitting of the analysis of covariance can be done by using RESTRICT so that the analysis of covariance is carried out for each factor level in turn but this results in separate analyses spread over several sheets of paper making it inconvenient to compare the regression coefficients.

A look at a model for analysis of covariance

$$y_{ij} = m + b_i + t_j + a(x_{ij} - x..) + e_{ij}$$

suggests a neater method. From the equation it can be seen that, for any value of the covariate equal to the mean value (x..), there will be no covariance adjustments. So, if we can set up dummy covariates, one for each level of the factor such that the values for units not associated with the current level are equal to the mean for that dummy covariate we have restricted the covariance adjustment to values associated with the current level of the factor only. Genstat statements for generating such dummy covariates for a factor with 10 levels are shown below:-

```
'FACT'  BLOCK $ 6  :  VARIETY $ 10
'READ'  YIELD, NUMBER, BLOCK, VARIETY
'FOR'  I = 1...10 ;  J = COVAR(1...10)
'REST'  NUMBER $ VARIETY = I
'CALC'  J = MEAN(NUMBER)
:       J = NUMBER
'REPE'
'BLOCK'  BLOCK
'TREA'  VARIETY
'COVAR'  COVAR(1...10)
'ANOVA/PRX = 0'  YIELD
```

Any of the ten dummy covariates will have the values of NUMBER for the current level and the values of the group mean (= dummy covariate mean) elsewhere.

Similar results can be obtained by putting covariate values not associated with the current level to zero but then the block means for any covariate will not be equal and regression coefficients will be obtained for the block stratum also. These are of no interest.

Norman Alvey
Roger Payne
R. E. S.