Issue No. 19 *1987(A)*

The GENSTAT Newsletter

NAG NUMERICAL ALGORITHMS GROUP

Editors

P W Lane
Rothamsted Experimental Station
Harpenden
Hertfordshire
United Kingdom AL5 2JQ

K I Trinder
NAG Central Office
Mayfield House
256 Banbury Road
Oxford
United Kingdom OX2 7DE

Printed and produced by the Numerical Algorithms Group

ISSN 0269-0764

R W Payne

GENSTAT NEWSLETTER
Issue No. 19

NP1508

1987 March

Contents

			Page
1.	Editorial		3
2.	Genstat 4.03E in China	*E R Williams*	4
3.	Conversion from Genstat 4 to Genstat 5	*A E Ainsley, P G N Digby, S A Harding, P W Lane, R W Payne and H R Simpson*	5
4.	A Genstat 5 Procedure for a First Difference Analysis	*D B Baird*	40
5.	The Use of Pseudo-Factors for a Balanced 6×6 Row-and-Column Design for Nine Treatments	*D A Preece*	48

Enclosure

Genstat Newsletter Display Sheet

Published Twice Yearly by
Rothamsted Experimental Station Statistics Department
and the Numerical Algorithms Group Ltd

Editorial

The latest version of Genstat is called Genstat 5 Release 1.1. General release has been awaiting completion of the Manual: we hope that in October, by which time this Newsletter should have appeared, the Manual will be available and the new version will start to be distributed. Sites with VAX/VMS will get it immediately. The following conversions are also expected to be ready in October: IBM mainframes with MVS, IBM and compatible personal computers with PC-DOS or MS-DOS, SUN workstations with Unix, and PR1ME minicomputers with Primos. Other conversions started in June; progress will depend on the amount of time they can spend, and the difficulties they encounter with each type of computer and operating system, particularly with the Fortran 77 compiler.

The Manual is being published as a bound book by Oxford University Press. It has been completely redesigned, and will provide a much more reliable, informative and comprehensive reference source than its predecessor. In particular, all sections of the Manual now contain worked examples to illustrate the description, rather than relying on a set of examples distributed separately with Genstat itself. The updated Reference Summary is also an appendix to the manual (effectively replacing the old Part II), and contains the same basic information as can be accessed on-line with the HELP directive. The Introduction to Genstat has also been rewritten, and will be published by Oxford University Press late in 1987, entitled 'Genstat 5: an introduction'. A further book has been written, and will be published by Oxford University Press in 1988, entitled 'Genstat 5: a second course'.

Genstat 5 has been tested for many months at Rothamsted and other sites in the Agriculture and Food Research Council, and at some other beta-test sites which have VAX computers. Many changes, corrections and improvements have been made since the first trial version, Release 0, was put together for the Fourth Genstat Conference in York in 1985, progressing from 0.1 to 0.6, then 1.0, 1.1 and 1.2.

The changes to the Genstat command language were described at the Conference in York, but were not published in the Newsletter because the language was still being implemented. The notes in this Issue describe all the major changes, and are intended as a summary to help people to make the change between writing programs for Genstat 4 and Genstat 5.

The whole command language of Genstat has been completely redesigned for Genstat 5. This is the first time for over 10 years that Genstat has been changed in a way that is not upwards-compatible, excepting one or two changes to individual directives. This means that if you want to run an existing Genstat program with the new version of Genstat, you will have to rewrite the program. However, we believe that the benefits of the new language greatly outweigh the inconvenience of 'relearning' Genstat and that users will soon appreciate the advantages of Genstat 5 over earlier releases.

Two other articles in this issue describe applications of Genstat in the area of balanced experiments: both use the Genstat 5 syntax. Articles for future issues using either the old or the new syntax will be welcomed; the Editors would particulary like to hear about any comparisons between Genstat 4 and Genstat 5.

Genstat 4.03E in China

(Reprinted with the permission of the author and the editor, from the DMS Newsletter)

E R Williams
Division of Mathematics and Statistics
CSIRO
GPO Box 1965
Banks Street
Yarralumla
Canberra ACT 2601
Australia

I recently visited China with Colin Matheson of the Division of Forest Research. The purpose of the visit was to give a course on design and analysis of experiments to facilitate the handling of field experiments on Australian species in southern China, and to supply two IBM PC-XT computers for analysis of results. Two courses of two weeks each were given, one at the Academy of Forestry in Beijing and the other at the Research Institute for Tropical Forestry in Guangzhou. The courses covered the use of computers themselves and an introduction to Genstat which is now available on PC's. A number of case studies using field trial data were also presented.

Class sizes ranged between 15-25 at the two centres and were made up of people with all possible combinations of computing, statistics and English expertise. We were pleasantly surprised with how quickly people latched onto using Genstat and at the end of the course in each centre there were about six competent users of Genstat who were in a position to help their colleagues with any difficulties. Admittedly we only covered selected sections of the Genstat package (input-output, graphs, tables, regression and anova) but we felt that to try to include any more in only two weeks would not be wise. We noticed that there was a correlation between Genstat proficiency, statistical knowledge, English competence and age. Many people could understand English, nevertheless the whole course was given through a translator. According to some of the good English speakers statistical points were often lost in the translation, which would have contributed to the aforementioned correlation.

There was no doubting the enthusiasm of the Chinese and their willingness to work very hard to complete recommended exercises between each day of lectures. We would observe that files were being created on the PC very late into each night. This was particularly so in Guangzhou where prior to our arrival the scientists had a choice between hand analysis or sending data to Beijing to be processed in the Computer Science Department. Whilst we were in Guangzhou we analysed a number of data sets for clients. We also created a file of Genstat programs for the examples from one of the Chinese statistics books (as Rothamsted did with the Cochran and Cox examples).

There were a couple of lessons for me from the Chinese experience:

(i) Genstat is not an 'impossible' language to learn; the only prerequisite is some background knowledge of statistics. The manual (together with the Introduction to Genstat book) can be digested. The important thing is not to try and go too fast as this just results in confusion and frustration (I once heard about a one-day course in the complete Genstat).

(ii) IBM compatible PC's with 640Kb and 10Mb hard disc are powerful machines. Possible criticisms of PC's such as excessive time spent writing space saving programs and swapping discs, are not valid for the above configuration. Genstat fits comfortably on the PC, occupying about 2.8mb on the hard disc. The package works well on the PC; I would anticipate that the majority of my consulting work could be handled on the PC. On top of this, transfer of data (say from Division to Division) is made easy with the use of floppy discs, thus removing problems and inconvenience that can occur with magnetic tape transfer to the VAX.

Conversion from Genstat 4 to Genstat 5

A E Ainsley
P G N Digby
S A Harding
P W Lane
R W Payne
H R Simpson

Statistics Department
Rothamsted Experimental Station
Harpenden
Hertfordshire
United Kingdom *AL5 2JQ*

Introduction

The latest release of Genstat is called Genstat 5 Release 1.1. The reason for the change from the previous sequence of release numbers, which ran from 3.01 (in 1973) to 4.04 (in 1983), is that this time we have made major changes to the syntax. Between 1973 and 1983, changes were always made with the aim of ensuring "upwards compatibility": that is, the aim was that programs that worked in one release of Genstat should work in the same way in the next release. Occasionally this was not possible, but in the main there were very few perturbations to the syntax. This was very convenient to existing users, but during that period we became aware that there were many potential users who were put off by the difficulty of learning the Genstat language, and others who disliked the syntax but persisted in using Genstat because of its facilities rather than its ease of use.

The syntax of Genstat 4 contains many inconsistencies. In fact, it is hard to detect many general rules: much of the syntax has to be learned piecemeal with each directive. Compare for example `TABULATE` with `FIT`: in one the parameter lists are in parallel, in the other they are not. Because the facilities had evolved over a period, there were many directives with overlapping facilities, or with conflicting philosophy. For example, `EXTRACT` declares tables implicitly: `TABULATE` does not. Thus there was a general need for revision and rationalization.

In Genstat 5 the syntax has been completely redesigned. There is a set of general "grammatical" rules which are obeyed consistently by every Genstat 5 directive. These rules have all been very carefully planned – we spent over a year in design and discussion before they were finalized. We have also taken the opportunity to standardise the vocabulary of directive, option and parameter names to avoid using the same name for different purposes in different directives or different names for the same purpose. Likewise we have introduced standard prefixes and acronyms. These should all make Genstat 5 easier to learn and remember.

The earlier versions of Genstat were designed at a time when most analyses were run in batch; they make few concessions to interactive working. The compilation and execution of blocks of commands in Genstat 4 is not very convenient in an interactive run; you usually want to see the results from executing one command before typing in the next one. Also the statistical output is mainly produced in rather large sections which often runs off the top of the screen before the top lines can be read.

In Genstat 5, statements are generally compiled and executed one at a time. Also the output has been redesigned to make it more convenient to examine on a v.d.u.

Genstat 5 contains all the facilities in Genstat 4, and there are several extensions. Facilities for high-quality graphics have been much improved (see Section 4). These are now incorporated using routines from the NAG Graphical Supplement, and so should be very much easier to implement than the rather rudimentary interface provided in Genstat 4. New directives are provided for fitting a range of standard curves (see Section 5), and for randomizing experimental designs (see Section 2). The use of pointers has been extended (see Section 1), and several new compound data structures have been introduced to improve the representation of complicated types of data like the SSP. Finally, and perhaps most importantly, the Genstat 4 macro has been replaced by a procedure structure in Genstat 5. The use of a procedure looks identical to the use of a standard Genstat 5 directive, and procedures are accessed automatically from libraries as required. The official procedure library supplied with Genstat 5, and any special site library supported by the local Genstat representative, will be attached to Genstat 5 automatically whenever it is run. Genstat 5 can thus be customized for the users at a site, by providing procedures for their particular needs.

Obviously the programs that you have been running with Genstat 4 will no longer run with Genstat 5. The notes in the subsequent sections are intended to help you convert to the new syntax. They are based on those written for one-day conversion courses that have been presented during 1986 and 1987 at several of the sites that have been helping us with the testing. The initials after each section represent the original authors and speakers and do not necessarily indicate any responsibility for the corresponding parts of Genstat. Most of the Genstat 4 directives map across in a fairly obvious way (see Section 9), and those that have changed most radically are generally those that were least satisfactory in Genstat 4. Consequently the users at the test sites seem to have adapted very quickly to the new syntax – the main comments have been of relief that particular quirks of the previous Genstat have been straightened out.

(RWP)

1. Syntax and Data Structures

The syntax of Genstat 5 has been redesigned throughout, primarily to make it convenient for interactive use. The opportunity has been taken to eliminate most of the irritations and inconsistencies in Genstat 4.04, and to redefine many system words according to a consistent policy. Rather than define the new syntax from scratch, as is done in the new Manual, we illustrate the changes here in the context of a simple example, describing new features as they occur. This should make it easier for you to understand what the changes are, and see how existing programs could be translated, assuming you are already familiar with Genstat 4.

We hope that the initial inconvenience of having to relearn the syntax will be quickly outweighed by the advantages designed into the new syntax:

(a) Interactive working is more convenient.

(b) Standard statistical analysis is simpler to specify.

(c) It is easier for newcomers to learn the syntax, and for everyone to remember it, because it is consistent.

(d) There is more flexibility for programming.

1.1. The Structure of Statements

Here is a simple program in the Genstat 4 command language that produces a line-printer graph and a summary statistic.

```
'REFERENCE' Introduction_to_Genstat_5
'' Broadbalk yield 1981-5 ''
'VARIATE' Year = 1981 ... 1985
: Maxyield = 10.13,8.69,7.91,9.41,9.91
'HEADING' Xt = ''Year''
: T = ''Maximum wheat yield on Broadbalk''
'GRAPH/TITLE=T,ATX=Xt' Maxyield; Year
'RUN'
'SCALAR' Meany
'CALCULATE' Meany = MEAN(Maxyield)
'PRINT' Meany $ 12.2
'RUN'
'STOP'
```

Here is a translation, line-by-line as far as possible, into the Genstat 5 command language, taking no advantage of most of the new features.

```
JOB 'Introduction to Genstat 5'
" Broadbalk yield 1981-5 "
VARIATE [VALUES=1981 ... 1985] Year
& [VALUES=10.13,8.69,7.91,9.41,9.91] Maxyield
TEXT [VALUES='Year'] Xt
& [VALUES='Maximum wheat yield on Broadbalk'] T
GRAPH [TITLE=T; XTITLE=Xt] Y=Maxyield; X=Year
SCALAR Meany
CALCULATE Meany = MEAN(Maxyield)
PRINT Meany; FIELDWIDTH=12; DECIMALS=2
STOP
```

This simple program illustrates the cosmetic changes to the syntax, and also some of the simplifying features, as follows.

(a) Directive names have changed, such as REFERENCE to JOB; also option and parameter names and their settings. (The parameters of a directive were called arguments or nameable lists in Genstat 4.) Each new name has been chosen to be an English word, such as DECIMALS, or a word prefaced by one or two letters to indicate the application of the word when there are alternatives, such as XTITLE. Some common acronyms have been retained, such as the ANOVA directive name. No word is used with a different meaning in two different names, but some letters have alternative meanings: for example, the TMEAN function forms tables of means, and the TDISPLAY directive prints results of time-series analysis.

(b) Directive names have no quotes round them.

(c) Options appear within square brackets and are separated by semicolons, not commas. Some options, like VALUES in the VARIATE directive, can have a list of values as the setting.

(d) All parameters are separated by semicolons; thus, the dollar sign is no longer used to separate identifiers from formats in PRINT, for example. All parameters are nameable, except when a directive has only one parameter (like JOB or CALCULATE, for example).

(e) Text appears in single quotes, not repeated single quotes, and comments appear in double quotes.

(f) Ampersand (&) replaces colon (:) for repeating directive names. Also, you can reset options after using this *repetition symbol.*

(g) There is no RUN directive in Genstat 5. All statements are executed immediately after they are compiled, except within program structures described in Section 2: even there, there is no need to distinguish between compile time and execution time interpretation of the components of a statement.

1.2. General Changes Making Programming More Convenient

Here is an alternative translation of the above program, taking advantage of some more of the new features in Genstat 5.

```
VARIATE [VALUES=1981 ... 1985] Year
& [VALUES=10.13,8.69,7.91,9.41,9.91] Maxyield
TEXT Xt,T; VALUES='Year','Maximum wheat yield on Broadbalk'
GRAPH [TITLE=T; XTITLE=Xt] Maxyield; Year
CALCULATE Mean = MEAN(Maxyield)
PRINT Mean
STOP
```

This program shows the following features of the new language.

(a) The JOB directive is optional.

(b) As well as a VALUES option, declarations have a VALUES parameter to allow different values to be assigned to several structures in one declaration. (There are few other instances in Genstat 5 where a directive has an option and a parameter with the same name.)

(c) The settings of all parameters in a statement are matched in parallel, recycling if necessary to match the length of the list in the first parameter. Thus, the first quoted string, 'Year', is the value of Xt, and the second string is the value of T. This rule holds for all directives, and the first parameter must always have the longest list.

(d) Function names are not reserved words; thus, the identifier Mean is not confused with the function MEAN; nor would it be if the identifier were all in capitals.

(e) Structures need not be declared when their size and type is clear from the context of a statement that assigns their values; for example, the variate Mean is not declared.

(f) Sensible default formats are used by PRINT to display numbers, as in all other directive that print results.

Here is an even shorter version of the program.

```
VARIATE [VALUES=10.13,8.69,7.91,9.41,9.91] Maxyield
GRAPH [TITLE='Maximum wheat yield on Broadbalk'; \
   XTITLE='Year'] Maxyield; !(1981 ... 1985)
CALCULATE Mean = MEAN(Maxyield)
PRINT Mean
STOP
```

(g) Quoted text may appear anywhere in place of the identifier of a text.

(h) A *continuation symbol* (\) is used to continue long statements on to following lines (but see SET in 1.4).

(i) Combined sets of values, using the *combination symbol* (!) and round brackets, may appear anywhere in place of identifiers of variates. In fact, a similar notation can be used for other structures as well: for example, !T() stands for an unnamed text structure with possibly many textual values (more about text in 1.5), and !P() stands for an unnamed pointer.

1.3. New Characters

Some new characters from the ASCII set are allowed, and others have new meanings. Only caret (^), tilde (~) and query (?) are unused.

< > are equivalent to the operators .LT. and .GT.

[ ] are used for options and suffices (and { } are equivalent).

: terminates statements; for example, when two statements are given on one line.

' delimits quoted strings (and ` is equivalent).

" delimits comments.

$ introduces qualifiers of identifiers in expressions (see 2.1).

! combines values into an unnamed structure (and | is equivalent).

& replaces colon as the repetition symbol; it may be followed by options.

\ continues statements onto following lines.

(appearing as the pound sign on some terminals and printers) substitutes values of structures (as used to happen automatically with identifiers defined by 'SET'); ## substitutes text into a program (as used to be done with 'USE' *macro* $)

1.4. The Environment of a Genstat Program

You can alter some of the rules of syntax in a program by using the SET directive; this is not related to the old SET directive, which is not needed now there is an explicit substitution symbol (#). For example,

```
SET [NEWLINE=ignored]
```

makes continuation characters unnecessary; statements must then be terminated explicitly by a colon (:).

```
SET [CASE=ignored]
```

makes case unimportant in identifiers, so that TIME, Time, and time would all refer to the same structure. SET has other options to control the default actions of central parts of Genstat. For example,

```
SET [DIAGNOSTICS=faults]
```

turns off the reporting of warnings, so that diagnostics are printed only for faults. Diagnostic messages in Genstat 5 are much more comprehensive and explicit than in Genstat 4.

There is also a GET directive in Genstat 5, again not related to the GET directive of Genstat 4 (now called RETRIEVE, see 3.4). GET now accesses the state of the environment settings controlled by SET.

System words can be abbreviated in Genstat 5 – often more heavily than in Genstat 4. Option and parameter names can be abbreviated to the minimal ordered form within the set of options or parameters defined for a directive. Usually, this is one letter, but when two options in a directive start with the same first letter (for example LEVELS and LABELS of the FACTOR directive, see 1.5 below), the name of the second option (LABELS) must be given with at least two letters (LA). To avoid confusion in writing and reading programs, you may find it preferable to abbreviate all system words to no fewer than four characters. Settings of options and parameters that are system-defined, such as faults in the setting of the DIAGNOSTICS option of the SET statement above, can also be abbreviated to the minimal ordered form within the set of possible settings.

All standard Genstat documentation will use no abbreviations, except when describing the rules of abbreviations, and when specifying default settings in the Reference Summary. We recommend that if you write an article for the Genstat Newsletter, you should ensure that abbreviation does not obscure the readability of a program.

1.5. Data Structures

Most structures are as in Genstat 4, though the syntax of the declarations has been changed to become consistent. Thus there are scalars, variates, matrices, symmetric matrices, diagonal matrices, and tables. Factors are slightly modified; they can now have a set of numerical levels and a set of corresponding textual labels. There is no integer structure in Genstat 5; but all directives that declare numerical structures have a DECIMALS parameter: setting DECIMALS=0 will result in the values being printed as integers by default in any Genstat statement. Two new structures are available to store expressions and formulae, as could be done in Genstat 4 with 'SET/LIST=E' and 'SET/LIST=M'; they can be declared with the EXPRESSION and FORMULA directives.

The heading and name structures have been replaced by the text structure. This stores a series of strings, in the same way as a variate stores a series of numbers, and a string may contain any number of characters. All strings may be quoted (with single quotes), and the following strings can also be given without quotes:

(a) strings that are system-defined settings of options or parameters, like ignored in

```
SET [CASE=ignored]
```

(b) strings that occur in values lists of text structures – that is, in the setting of the VALUES option of the TEXT directive, or within the brackets of a !T( ) construction – and that start with a letter and consist of letters and digits only;

(c) strings read in fixed format by the READ directive (see 3.1).

Within quotes, a newline is taken to start a new string unless the continuation character (\) is given, and quotes must be duplicated to avoid ending the text or initiating a comment.

The use of pointers has been considerably extended, to make it convenient both to refer to a group of structures with a single identifier, and to access the individual structures within a group. Elements of pointers can be referred to by suffixes to any depth; thus

```
POINTER [VALUES=A,B,C] M
& [VALUES=L,M,N] X
PRINT X[2][1,3]
```

will print the values of A and C. Conversely, whenever suffixes are used, pointers are automatically declared to store identifiers with common name but different suffixes. If you, like most users of Genstat, use suffixes only as an extra labelling device for structures, the only important effect of this automatic declaration is that you cannot use an unsuffixed identifier independently of a suffixed one with the same name. Thus you cannot, for example, use

```
FOR X=X[1....10]
```

in Genstat 5, because X is automatically a pointer storing references to X[1], X[2] ... X[10], and so cannot be a dummy that stores a reference to only one of these structures at each pass of the loop.

The null suffix, [ ], stands for all possible suffixes defined so far in the program for the identifier name it is appended to. Thus X[ ] stands for the list L,M,N in the example above. The substitution symbol (#) also replaces a pointer by its values, but will replace any of the values that are pointers by its values too, and so on. Thus, #X stands for L,A,B,C,N. Textual suffixes can also be defined in the POINTER directive, so you could arrange to refer to a structure as Yield['apples'], for example. Sub-pointers can be constructed by giving identifiers of variates or texts within the square brackets.

There are three special compound structures in Genstat 5. You can set up your own compound structure by defining a pointer to point to the collection of simple structures you want to be able to refer to with a single identifier. However, the SSPM, LRV and TSM structures are needed generally for the regression, multivariate and time-series analysis directives. The SSPM (for Sums of Squares and Products plus Means) directive replaces the old DSSP directive for declaring a structure to hold a symmetric matrix of sums of squares and products, a variate of means, and a scalar for the number of observations. A within-groups SSPM can also be defined. The LRV directive now defines a compound structure pointing to a matrix of latent vectors, a variate of latent roots, and a scalar to hold the trace (the old LRV directive is now called FLRV: see 2.6). The TSM directive defines a time-series model, as in Genstat 4.

1.6. Extensions to the Language

The new command language is designed to be extended. You can effectively define new directives by setting up procedures containing Genstat statements (see 2.9). Alternatively, you are allowed to define new directives that invoke your own Fortran code. Of course, this involves relinking Genstat to include your code, but the mechanism of specifying a new directive is straightforward and a skeleton routine to interpret a statement using a new directive is provided with Genstat.

As in Genstat 4, you can also extend Genstat by using the OWN directive. This now has a SELECT option, to allow convenient use for multiple applications. Its advantage over the definition of new directives is that no trouble need be taken to define and interpret syntax; again, Genstat must be relinked, and a skeleton version of the OWN subprogram is provided.

A new directive has been introduced in Genstat 5 to allow extensions to be made in Fortran without relinking Genstat. The PASS directive invokes an external program and passes information to and from it. The external program should contain as its main routine an edited version of a routine provided with Genstat that deals with communication. There is also a new directive called SUSPEND which suspends Genstat to allow you to give commands to the operating system. These directives may not be able to be implemented in some operating systems.
(PWL)

2. Manipulation and Programming

This section covers the directives for calculations, transfer and manipulation of values, operations on factors, tabulation, matrix operations, text handling, and programming. Many of these are very similar to their counterparts in Genstat 4, and those that have changed most radically (such as COMBINE and RESTRICT) are those that were least satisfactory in Genstat 4. Some directives have been deleted as their facilities are duplicated elsewhere in Genstat: for example, the transfer of values provided by COPY can also be done by the ELEMENTS function of CALCULATE. (The directive called COPY in Genstat 5 now allows you to produce a transcript of the input and output from a job: see 3.5.) There are also some new facilities; for example, the RANDOMIZE directive provides the randomization of experimental designs, and directives EDIT and CONCATENATE provide some of the facilities for text handling.

2.1. Expressions

Arithmetic and logical expressions occur as options or parameters of several of the directives described in this section. The rules for their specification are mainly as in Genstat 4. Some of the function names have changed to become more logical or more memorable; the correspondence between the Genstat 4 functions and those in Genstat 5 is listed in the Conversion Summary (Section 9). Otherwise most Genstat 4 expressions should work unchanged in Genstat 5.

The functions for interpolation, LINT and INLINT, have been removed. Interpolation is provided in Genstat 5 by the new directive INTERPOLATE (see 2.2).

These are some of the new functions:

(a) CIRCULATE moves the values of a variate in a circular fashion. For example,

```
CALCULATE Y = CIRCULATE(X; 1)
```

moves the value in the final unit of X to unit 1, and then shifts all the other values down one place; the results are stored in Y.

(b) SHIFT moves the values up or down a variate. This is not a circular shift; so with a downward shift, for example, units at the top lose their values and are set to missing.

(c) EXPAND forms a logical variate from a list of unit numbers; this has the value 1 where the unit is in the list, and 0 elsewhere.

(d) RESTRICTION forms a logical variate indicating the units to which a vector is currently restricted.

(e) SOLUTION solves linear equations.

(f) UNSET checks whether a dummy structure has been set.

These are the functions whose method of use has changed:

(a) ELEM (now with the full name of ELEMENTS) has an optional third list to allow elements of matrices to be defined. Elements can also be specified within expressions using qualified identifiers: for example

```
CALCULATE A$[1] = B$[3]
```

assigns the third value of variate B to the first unit of A

```
CALCULATE A$[2,3] = M$[1,2; 2]
```

assigns the first and second values in column 2 of matrix M to the second and third units of V.

(b) The variate functions (VSUM, VMEAN, VMAX, VMIN, VVAR and VNMV) now expect a pointer as their argument; the pointer points to the set of variates over which the particular summary is to be formed. This removes one of the inconsistencies in Genstat 4, by which these were the only functions whose arguments were not in parallel with the other lists in the expression. For example, the calculation

```
'CALCULATE' S1,S2 = VSUM(X,Y)
```

in Genstat 4 gives both S1 and S2 the same values, namely the sums of the values in each unit of X and Y. In Genstat 5, this could be written as

```
POINTER [VALUES=X,Y] P
CALCULATE S1,S2 = VSUM(P)
```

or more succinctly, using unnamed pointers, as

```
CALCULATE S1,S2 = VSUM( !P(X,Y) )
```

and the intention is clear. The second argument has been deleted; this again was an inconsistency – these were the only functions where some of the output was via one of the arguments. The numbers of non-missing values in each unit can now be obtained by the new function VNOBSERVATIONS. The other new variate function is VMEDIANS.

There are several new operators available for use in expressions:

(a) *+ provides matrix multiplication: for example A*+B does the same as PDT(A; B) did in Genstat 4 but is easier to use when the expression involves a series of multiplications. (The function for matrix products is still available in Genstat 5, but is renamed PRODUCT.)

(b) .EQS. allows you to test for equality of the units of texts.

(c) `.NES.` tests for inequality of texts.

(d) `.IN.` tests whether the value in each unit of a vector is a member of a specified set of values. For example:

```
Size .IN. !T(small,large)
```

tests whether each unit of the factor or text `Size` has the value small or large. (The second argument `!T(small, large)` is an unnamed text, as described in 1.5)

(e) `.NI.` provides the converse of `.IN.`: that is, it tests for non inclusion in the specified set.

(f) There are now also various synonyms:

==	.EQ.	<	.LT.	<=	.LE.
/=	.NE.	>	.GT.	>=	.GE.
<>	.NE.				

2.2. Numerical Calculations

CALCULATE is still the main directive for numerical calculations; it has a new option, TOLERANCES, which allows you to set the bound used to detect zero values. The other change is that, if the structure on the left-hand side of an assignment has not yet been declared, Genstat 5 will declare it automatically as a structure of a type appropriate for the result of the calculation.

INTERPOLATE replaces the interpolation functions. It allows either direct or inverse interpolation, or the estimation of interpolated values for missing units.

2.3. Transfer and Manipulation of Values

EQUATE still maps the values from one set of structures into another, but the syntax is slightly modified to conform to Genstat 5's general rule of parallelism. If the mapping is from one structure into another, the form is very similar to that in Genstat 4: for example

```
'EQUATE' M = T
```

becomes

```
EQUATE OLDSTRUCTURE=T; NEWSTRUCTURE=M
```

or

```
EQUATE OLD=T; NEW=M
```

if you take advantage of the abbreviation rules for the parameter names (see 1.4). However, if the mapping is into a set of structures, these must be combined into a pointer. Similarly, if the values are to be taken from a set of structures, these also must be grouped together in a pointer. So for example

```
'EQUATE' R1,R2,R3 = M
```

in Genstat 4 becomes

```
EQUATE OLD=M; NEW=!P(R1,R2,R3)
```

This removes an inconsistency of Genstat 4, and also means that you can do several equates in a single statement. The formats are now specified by options, OLDFORMAT and NEWFORMAT, whose values are variates.

RANDOMIZE is a new directive that allows you to randomize the units of vectors, or to randomize the allocation of treatments in an experimental design. For example,

```
RANDOMIZE [SEED=63133] X,Y
```

puts the units of X and Y into an identical random order, using random numbers determined by the seed 63133, and

```
RANDOMIZE [SEED=392181; BLOCKSTRUCTURE=Blocks/Plots] T
```

would do the randomization for a randomized block design with treatment factor T. A further option, EXCLUDE, allows you to restrict the randomization so that no randomization is done over one or more of the factors in the block formula. This might be required if one of the treatment factors is constrained to be in a particular order, for example order in time.

RESTRICT has a new and very much simpler syntax. The restriction is specified by a logical expression in which the units to be excluded have the value zero (or false) and those to be included have a non zero (or true) value. For example,

```
RESTRICT Name; CONDITION=( (Income>10000) .AND. (Region.IN.'South') )
```

restricts the units of Name to those of people in the South with an income of more than 10000.

SORT sorts the units of vectors or pointers according to an index vector which can be either a text or a variate. For example,

```
SORT [INDEX=Age] Age,Income,Name
```

sorts the units of Age, Income and Name into order of ascending ages. Here we have used only the first parameter, and so the structures are sorted in place. (The name of the parameter, OLDVECTOR, has been omitted here because it is the first parameter of the directive, and is specified first.) Alternatively, if you want to retain the original structures with the units in their unsorted order, you can use the NEWVECTOR parameter to specify new structures to store the sorted values; for example:

```
SORT [INDEX=Age] Age,Income,Name; NEWVECTOR=Sortage,Sortinc,Sortname
```

Use of a text as index allows you to sort into alphabetic order. The option DIRECTION allows you to sort into descending instead of ascending order. Other options of SORT allow you to form a factor from a variate or a text (see 2.4).

The Genstat 4 directives COPY and POSITION have no counterparts in Genstat 5. The effect of COPY can be achieved by using a CALCULATE expression with either the ELEMENT function or a qualified identifier (see 2.1). There is still a directive called COPY in Genstat 5, but this is now to allow you to keep a transcript of the job (see 3.5). Positions of particular values within a vector can be obtained using the SAVESET parameter of RESTRICT.

2.4. Operations on Factors

When a factor appears in an expression, Genstat usually takes its (numerical) levels, as in Genstat 4. An exception in Genstat 5 is when the factor occurs as the first operand of either of the inclusion operators .IN. or .NI. and the second operand is a text; then the labels of the factor are used instead. When the factor is on the left-hand side of the equals sign, Genstat 5 (as in 4) checks that each of the results of the calculation is an acceptable level for the factor. This provides one way of defining the values of a factor from a variate but a better way is to use the SORT directive (see below).

GENERATE, as in Genstat 4, is the directive for generating the values of a set of factors in standard order. For this use the syntax is unchanged: the factors to be generated are listed in the first parameter of the directive; for example:

```
GENERATE A,B,C
```

The final three parameters of GENERATE in Genstat 4 become options in Genstat 5, with names TREATMENTS, REPLICATES and BLOCKS.

Factors can be formed from variates or texts using the SORT directive. The identifier of the factor is specified by the GROUPS option, and the variate or text is specified by the INDEX option. The factor will be declared implicitly if you have not declared it already. This use of SORT takes over many of the functions of the Genstat 4 GROUPS directive. If no other option of SORT is specified, the factor is formed with a level for each distinct value of the variate or text; this corresponds to the RANK function of the GROUPS directive. The LIMITS option of SORT allows you to specify a range of values for each level, as in the LIMITS function of GROUPS, while the NGROUPS

option provides the effect of the QUANT function of GROUPS. The INTPT function of GROUPS can also be duplicated by setting DECIMALS=0 in SORT to specify that ranking is to be done on numbers rounded to zero decimal places. There is no direct alternative to the GROUP function within SORT; instead you should use the NEWLEVELS function in CALCULATE; for example,

```
'FACTOR' F$6 = (1...6)3
: G$3
'INTEGER' I = 1,2,-3,4,-5,-6
'GROUPS' G = GROUP(F; I)
```

becomes

```
FACTOR [LEVELS=6; VALUES=(1...6)3] F
FACTOR [LEVELS=3] G
CALCULATE G = NEWLEVELS(F; !(1,1,1,2,2,3))
```

The variate in the second argument of NEWLEVELS is a mapping list that specifies the level to which each level of the original factor is to be assigned: level 1 of G for the first 3 levels of F, level 2 of G for levels 4 and 5 of F, and level 3 for level 6.

2.5. Operations on Dummies and Pointers

The function UNSET allows you to test whether a dummy has been set (see 2.1). This is particularly useful inside a procedure (see 2.9), where you may need to check that an argument has been set by the calling program in order to decide whether the procedure is able to perform the required task.

Values of dummies and pointers can be set by the ASSIGN directive. For example

```
ASSIGN A,B; POINTER=P; ELEMENTS=2,3
```

sets elements 2 and 3 of pointer P to point to the structures A and B; and

```
ASSIGN X; D
```

sets dummy D to point to X. This is a rather different syntax from Genstat 4, where ASSIGN sets a dummy to point to a particular element of a list of structures.

2.6. Operations on Matrices

Some matrix functions have been renamed to avoid cryptic acronyms and there is the new operator *+ for matrix multiplication; these should simplify the writing of matrix expressions. The ELEMENTS function has a third argument to allow you to specify submatrices; qualified identifiers can also be used for this (see 2.1).

There are three directives specifically for matrix operations in Genstat 5: SVD, FLRV and FSSPM. SVD remains much as in Genstat 4. The other two are renamed from Genstat 4 and operate on matrices that are parts of compound structures. FLRV forms latent roots and vectors, and corresponds to the LRV directive of Genstat 4. (In Genstat 5, LRV declares an LRV structure: see 1.5.) FSSPM forms the values of an SSPM structure, and corresponds to the SSP directive in Genstat 4. (SSPM structures, known as SSPs in Genstat 4, are declared by the SSPM directive in Genstat 5: see 1.5.)

2.7. Operations on Tables

The COMBINE directive replaces the Genstat 4 directives COMBINE and OMIT, allowing levels of a factor classifying a table to be either combined or omitted. It is also more powerful in that several dimensions can be modified at a time, thus avoiding the need to define interim tables when more than one classifying factor needs to be changed. Values in the original table can be placed into more than one position in the new table, allowing you to form lines of intermediate summaries. COMBINE can also work on variates and matrices.

MARGIN, as in Genstat 4, is the directive for adding margins to tables, or for recalculating their

values. However, in Genstat 5 the factors that are listed are those classifying the margins to be formed (not those over which the margins are to be formed).

The representation of the body of a table as percentages of one of its margins, provided by the PERCENT directive in Genstat 4, is easily done using CALCULATE. First you need to form a table containing the appropriate marginal values, and then divide it into the full table. However, to simplify this process, it is intended to provide a procedure called PERCENT in the official procedure library.

Some of the options and parameters of TABULATE have been renamed, but all in a fairly obvious way: for example, the cryptic CS option of Genstat 4 becomes CLASSIFICATION in Genstat 5, and ASSCT becomes NOBSERVATIONS. The EMPTY and ZDZ options have been deleted; empty cells in tables of counts or associated counts are filled with zeroes, and empty cells in other types of table (totals, means, minima and so on) are given missing values. TABULATE will now declare the tables implicitly if you have not declared them already.

2.8. Text Handling

There were very few facilities for text handling in Genstat 4. This is an area that is much expanded in Genstat 5, with new directives CONCATENATE and EDIT. There are several other more general directives that can also operate on text. SORT allows units of vectors to be sorted into an order corresponding to alphabetic order of a text (see 2.3) or to form a factor from a text (see 2.4). You can use PRINT to place output into a text, and READ can read from a text. EQUATE will also operate on texts, in exactly the same way as for numerical structures. Thus, for example, you can form a text whose initial units are taken from one text and whose final units are taken from another. You can also omit some of the units or you can recycle them, exactly as you would with the (numerical) values, for example, of variates.

The alternative form of appending, whereby each line of the new text is formed by joining the corresponding lines of several other texts, is provided by CONCATENATE. Characters at the start and end of each component line can be omitted, allowing you to truncate the values of a text either to left or to right.

The EDIT directive is a sub-system for editing texts within Genstat 5. It has its own command syntax allowing you to delete and insert series of characters, or to substitute one series for another, or to delete and insert complete lines, and so on.

2.9. Programming and Control Structures

The implementation of the LABEL and JUMP directives was one reason why instructions in Genstat 4 were formed into blocks that were executed only after a RUN command. This execution of complete blocks of statements is inconvenient when working interactively, and is also a source of confusion for many users; it is abandoned in Genstat 5. Thus LABEL and JUMP are replaced (see below), and there is also no RUN directive. In Genstat 5, statements are executed immediately after they are read. The one exception is with FOR loops when the entire loop is generally read before it is executed; however, even then you can request that the first pass through the loop be executed as you type it. The reading of the statement in the loop does not involve compilation, so there is no cause for concern about compile-time and run-time interpretation.

Loops are still introduced by a FOR statement. However the directive for ending a loop is now called ENDFOR. This follows the general pattern for other types of control structure: for example the contents of a procedure are terminated by ENDPROCEDURE, and a block-if structure is ended by ENDIF. FOR now has an option called NTIMES which allows you to specify the number of times to repeat a loop that has no arguments.

To replace labels and jumps, two possibilities are available in Genstat 5. Multiple-selection control structures can be constructed using the directives CASE, OR, ELSE and ENDCASE, and

block-if control structures can be constructed by the directives IF, ELSIF, ELSE and ENDIF. Current views on good programming practice are that these structured programming concepts lead to code that is easier to follow and less likely to contain mistakes. For example,

```
'JUMP' L1 * (Age.LT.20)
'CALCULATE' Pay = Hours*2.5
'JUMP' L2
'LABEL' L1
'CALCULATE' Pay = Hours*1.75
'LABEL' L2
```

in Genstat 4, can be written as

```
IF Age.LT.20
  CALCULATE Pay = Hours*1.75
ELSE
  CALCULATE Pay = Hours*2.5
ENDIF
```

in Genstat 5. The indentation is of course optional, but greatly helps clarity. Further examples are given in the Genstat 5 Manual.

The use of the macro in Genstat 4 as a sort of "subroutine" is taken over by the procedure in Genstat 5. Procedures are very much easier to use than macros. The syntax for their use is exactly the same as the ordinary Genstat 5 directives, and procedures are accessed from libraries automatically when required. Thus there is no need for the naive user to know whether a particular statement uses a directive or a procedure. (The macro is still available in Genstat 5, using text in conjunction with ##: see 1.3.)

An official library of procedures, checked to ensure that they are useful and reliable, is supplied with Genstat; this library is attached to your job automatically by Genstat. A computer site can also form its own site library which will also be attached automatically. When Genstat finds a statement name that it does not recognize as one of the standard directives it first checks whether you have a procedure of that name already stored in your job. Then it looks to see whether there is a procedure of that name in any of the libraries attached to the job: checking first any libraries that you have attached explicitly, then the site library, and then the official library. Thus the official library allows new facilities to be made available quickly and conveniently to all users, while the site library allows Genstat to be customized to meet the particular requirements of the users at that site.

To define a procedure, you first give a PROCEDURE statement, which gives the name of the procedure; this must not be the same as any of the standard Genstat 5 directives. Then you define the options and parameters of the procedure, using the directives OPTION and PARAMETER. These are followed by the statements that make up the body of the procedure. Finally, you indicate the end of the definition by giving an ENDPROCEDURE statement. Information is transferred to and from the procedure only via its options and parameters: there is no concept of local and global identifiers as in Genstat 4.

To assist with the debugging of procedures, two pairs of directives are available: BREAK and ENDBREAK, and DEBUG and ENDDEBUG. These enable you to interrupt the execution of a procedure so that you can type in other statements to examine the values of variables within the procedure, change their values, and so on. They may also be useful in loops – either for debugging or to allow you to interrupt a loop when you are running Genstat interactively so that you can read one section of output before the next one is produced.

There is also a directive called EXIT which allows any control structure to be abandoned, dependent on a logical condition.

(RWP)

3. The Input and Output of Data

This section describes changes in the transfer of information to and from external files. Such files may be character (formatted) or binary (unformatted) files. The READ and PRINT commands access either kind of file; Sections 3.1 and 3.2 are concerned with character files, and binary files are discussed in Section 3.3. Backing-store operations, for long-term storage and retrieval of information, are described in Section 3.4. The OPEN command, which allows you to attach external files to input channels from within your program, is introduced in Section 5 and finally the use of the INPUT command to switch program control to commands on other channels is explained.

3.1. Reading Data from Character Files

The new syntax of the READ directive is given below, with the corresponding options in Genstat 4.

READ

Reads from a file (formatted or unformatted) or text.

Parameters

STRUCTURE	=	structures to be read
FIELDWIDTH	=	field widths
DECIMALS	=	decimal places for numerical data
SKIP	=	values/characters to skip before reading a value
FREPRESENTATION	=	(labels, levels, ordinals) – how factors are represented

Options

Genstat 4	Genstat 5		
PRINT	PRINT	=	(data, errors, summary) – what to print
	CHANNEL	=	channel number or text structure
FORM	SERIAL	=	yes – if structures are in serial order
NUN	SETNVALUES	=	yes – sets number of values of structures
	LAYOUT	=	fixed – if data is in fixed format
	END	=	*'string'* to define string terminating data
SEQ	SEQUENTIAL	=	scalar to hold number of units read
MODE	ADD	=	yes – accumulates values
MVI	MISSING	=	*'character'* – the character representing missing value
	SKIP	=	characters/values skipped between units
BLK	BLANK	=	(missing, zero, error) – for blank fields
NSB	JUSTIFIED	=	(right, left, both, neither) – for numbers in fixed format fields
ERCT	ERROR	=	number of errors allowed
	FORMAT	=	coded format when the layout is variable
QUIT	QUIT	=	channel number to return to after fatal error
	UNFORMATTED	=	yes – if file is unformatted
	REWIND	=	yes – rewinds file before reading

Since commands are executed as soon as they are compiled, data are expected immediately after the READ statement. In Genstat 4, to read data from a separate file,

```
'INPUT'  2
'READ'   ...
'INPUT'  1
'RUN'
```

was needed; in the new version

```
READ [CHANNEL=2] ...
```

suffices. The channel number must have been associated with a file; this is discussed below.

The end-of-data terminator is colon (:) by default, but any string of up to eight characters may be nominated for this purpose by the END option. The SKIP option and parameter apply only to parallel read; the old FLEV option, which gave the form of the data values for factors, is replaced by the FREPRESENTATION parameter.

Formats are handled by the FIELDWIDTH, DECIMALS and SKIP parameters in simple cases, and by the FORMAT option otherwise.

Strings may be read in free format or fixed format. If free format, they must be quoted (unless they begin with a letter and consist only of letters or digits) and pairs of primes are treated as single primes. In fixed format, the contents of the field are taken as the value with no such processing.

3.2. Writing Data to Text Files

The new form of the PRINT directive is:

PRINT

Outputs to a file (formatted or unformatted) or text.

Parameters

STRUCTURE	=	structures to be printed
FIELDWIDTH	=	field widths
DECIMAL	=	decimal places
SKIP	=	spaces to leave before each value of each structure (* means newline before structure)
JUSTIFICATION	=	left – justifies to the left
MNAME	=	(margin, total, nobserved, mean, minimum, maximum, variance, count, median) – chooses name for table margins

Options

Genstat 4	Genstat 5		
	CHANNEL	=	channel number or text structure
FORM	SERIAL	=	yes – prints structures in serial order
	IPRINT	=	associatedidentifier – instead of table name
LABR	RLPRINT)	=	(labels, integers) – what to print for row
LABC	CLPRINT)		or column labels, when relevant
RLW	RLWIDTH	=	field width for row labels
LHM	INDENTATION	=	left margin width
RHM	WIDTH	=	right margin position
SQUASH	SQUASH	=	yes – omits blank lines
	MISSING	=	*'string'* – to print for missing value
VAR(1)	ORIENTATION	=	across – prints vectors across the page
PERM	PERMUTE	=	permutation vector for tables
VAR(2)	INTERLEAVE	=	level at which multiway structures are to be parallel
NODN	DOWN	=	number of table classifiers to be printed down (offset from zero rather than from default)
PUNK	PUNKNOWN	=	(present, always, zero, missing, never) – when to print unknown cell of tables
	UNFORMATTED	=	yes – if file is unformatted
	REWIND	=	yes – rewinds unformatted file before printing

Output is sent to the current output channel by default. If you want to send it elsewhere, the channel option of PRINT may be set; alternatively

```
OUTPUT 3
```

will send everything to output channel number 3.

Formatting is handled entirely by the FIELDWIDTH, DECIMALS and SKIP parameters. The old FORM=C setting and FMT option have disappeared.

3.3. Reading and Printing to Binary Files

The Genstat 4 commands 'BIN' and 'BOUT' have been included in the new READ and PRINT commands via an UNFORMATTED option. This should be set to yes when you need to transfer data to or from binary files.

Only the CHANNEL, SERIAL and REWIND options are relevant. SERIAL=yes should always be used: it is much faster. Parallel reading should only be necessary when reading data from another program that have been output in this way.

3.4. Storage and Retrieval in Hierarchical Binary Files

The commands PUT and GET, and the simpler features of SAVE and FETCH, have been replaced by two new commands, STORE and RETRIEVE.

STORE

Stores structures in a subfile of a backing-store file.

Parameters

IDENTIFIER	=	structures to be stored
STOREDIDENTIFIER	=	identifiers to be used for the stored structures

Options

Genstat 4	Genstat 5		
PRINT	PRINT	=	(catalogue) – what to print
FILE	CHANNEL	=	channel number
	SUBFILE	=	subfile identifier
LIST	LIST	=	(inclusive, exclusive, all) – how to interpret the list
	METHOD	=	(add, overwrite, replace) – how to append the subfile to the file
	PASSWORD	=	*'string'* – password
	PROCEDURE	=	yes – if subfile is to store procedures

RETRIEVE

Retrieves structures from a subfile

Parameters

IDENTIFIER	=	identifiers to be used for the structures after retrieval
STOREDIDENTIFIER	=	identifiers of the stored structures as stored

Options

Genstat 4	Genstat 5		
FILE	CHANNEL	=	channel number
	SUBFILE	=	subfile identifier
LIST	LIST	=	(inclusive, exclusive, all) – as for STORE
COMP	MERGE	=	yes – if existing structures with the same identifier are not to be overwritten

A password may be stored with a userfile, to guard against deleting information accidentally later.

The PROCEDURE option of STORE distinguishes the storage of data structures (which will be subsequently retrieved) from the storage of procedures (which can then be invoked as described in Section 2.9).

The more complex features of SAVE and FETCH are provided in the MERGE command, which allows subfiles from a number of userfiles to be collected together as a single userfile.

The DISPLAY directive has been renamed CATALOGUE, and new commands RECORD and RESUME allow you to dump and restart jobs (previously done via a COMP=DUMP option in PUT and GET).

3.5. Using External Files in Genstat

In most implementations of Genstat, you should be able to attach external files when you enter the program with a system command or procedure. You can handle files more flexibly from within your Genstat program, by using the OPEN and CLOSE statements.

There are six different types of file. Character files are either `input` (for reading only) or `output` (for writing only); binary files are either `backingstore` or `unformatted`, accessed for both reading and writing. The other two types of file are `graphics` (see 4.5) and `procedurelibrary` (see 2.9).

An example of an OPEN command is

```
OPEN 'DATA','STORE'; CHANNEL=2,1; FILETYPE=input,backing
```

which opens two files, one called DATA on input channel 2 and the other called STORE on backing-store channel 1. The form of a file name is implementation-dependent, and you should find out what the local rules are. Channels are numbered independently from one upwards for each file type. After you have finished with them, these files should be closed by

```
CLOSE 2,1; FILETYPE=input,backing
```

and the channels can later be reopened to access the same or different files.

The INPUT directive switches control immediately to another file, which is expected to contain Genstat commands. (Control can also be switched to a text structure which contains commands by ##, see Section 1.3.) A new directive called RETURN is available to switch control back to a previous channel.

The COPY directive has been introduced to make it easy to keep a record of an interactive session using Genstat. It has a PRINT option to control whether statements or output or both are copied to the specified auxiliary file. (The old COPY directive, which duplicated the action of EQUATE or CALCULATE with the ELEMENTS function, has been removed from Genstat.)

(HRS)

4. Graphics

The facilities for generating graphical output in line-printer style remain much as before with directives GRAPH, HISTOGRAM and CONTOUR. The existing syntax has been updated for Genstat 5 and some new options and parameters added; the style of the output is little changed. The facilities for high-quality graphics have been considerably improved.

4.1. The GRAPH Directive

The options of GRAPH have been made simpler to use and there are two additions: it is now possible to direct the graph to a different output channel using the CHANNEL option, and YINTEGER allows integer markings to be set for the *y*-axis in addition to the existing XINTEGER option. The old BV option for setting axis boundary values has been made much easier to use by replacing it by four separate options YLOWER, YUPPER, XLOWER and XUPPER; also, EQXY has become EQUAL with more flexible choice of settings so axis bounds can be

constrained to be equal at either extreme. Multiple plots are now specified by using the option MULTIPLE which combines the effect of two former options, YCNT and XCNT.

The way in which the plotting method and symbols are chosen has been made much simpler in the new METHOD and SYMBOLS parameters. The method is selected as a string: `point`, `line`, `curve` or `text`. For example, the statement

```
GRAPH Y1,Y2; X1,X2; METHOD=point,curve
```

will produce a point plot of Y1 against X1 and will plot a smoothed curve between the points specified by Y2 and X2. As in Genstat 4, the plotting symbols can be defined by a factor, but they can now also be specified in a text vector containing either text symbols for each point or a single string that is used throughout. The latter method of setting the symbol is particularly useful: in the statement

```
GRAPH Y; X; SYMBOLS='+'
```

each point is marked by a plus sign. Graph titles are equally easy to set as strings in the GRAPH statement itself; for example:

```
GRAPH [TITLE='Residuals'] R; X
```

4.2. The HISTOGRAM Directive

HISTOGRAM has also gained a CHANNEL option for specifying where the output is required. The former GROUP parameter which defined how the histogram was to be generated from the data has become two options in the new syntax, NGROUPS and LIMITS, providing two methods of defining the groups. There is still a GROUPS parameter, but this is now used to save (as a factor) the grouping defined on a variate and the NOBSERVATIONS parameter is an addition that allows the number in each group to be saved in a one-way table of counts.

HISTOGRAM will take as input three kinds of structure: variates, factors, or one-way tables. Variates are grouped first, whereas a factor is taken as defining a grouping of the units and a table gives group counts directly. If a list of structures is supplied they must now all be of the same type and, unlike Genstat 4 where individual histograms would be plotted, a single histogram is formed with each group being represented by a number of parallel bars, one for each structure in the list.

The new parameter SYMBOLS can be used to specify a character to replace the default asterisk in plotting the bars of the histogram. If a parallel histogram is being generated, different symbols can be used for each bar by specifying a list of characters in parallel with the DATA parameter list; for example:

```
HISTOGRAM Observed,Expected; SYMBOLS='O','E'
```

4.3. The CONTOUR Directive

The options of CONTOUR are much as before but have been given more meaningful names: INTERVAL, LOWERCUTOFF and UPPERCUTOFF. INTERVAL can now be set to either a scalar, as before, or a variate giving values of the contours required. The BV option has been replaced by YLOWER, YUPPER, XLOWER and XUPPER and new options TITLE, YTITLE, XTITLE, YINTEGER and XINTEGER have been added; these are all interpreted in the same way as the corresponding options of GRAPH. The Genstat 4 options NR and NC are not applicable, as the method of supplying the grid values in a single variate is no longer used; instead a pointer of variates can be given, with each variate being treated as a column of the data matrix. Alternatively the grid values can be supplied in a matrix or table as before.

4.4. High-quality Graphics

In Genstat 4 the option DEVICE of GRAPH allowed you to direct graphical output to a file that could later be submitted to a plotter or other graphics device. Some of the other options and

parameters gave control over pen colour, symbols and linestyle, but the graph produced was otherwise similar to the line-printer output. In Genstat 5 the provision of high-quality graphics output has been greatly extended by introducing some entirely new directives specifically for use with graphics devices. You can choose between working interactively on a graphics terminal or can save output for subsequent display or plotting. There are directives for drawing graphs, histograms, contours and pie-charts; these are DGRAPH, DHISTOGRAM, DCONTOUR and DPIE respectively (the D standing for 'draw'). Thus, to draw a simple scatter plot of Y versus X, you can type

```
DGRAPH Y; X
```

Default histograms, contours and pie-charts can be generated with similar ease by typing statements such as

```
DHISTOGRAM X
```

and so on.

Flexibility is achieved by allowing control over as many features of the output as possible. Graphs can be scatter plots or the points may be joined with straight lines or curves, with choice of line-style, colour, and plotting symbols. There is a choice of two styles of histogram, and contours can be drawn as straight lines or smoothed curves. You can draw axes in a variety of styles, and supply markings, including textual labels, and you can add titles. Keys can also be provided. A single "frame" of output, corresponding to one v.d.u. screen or sheet of plotter paper, can be divided into several windows containing different plots. You can choose between clearing the screen (or advancing paper) before each piece of output or retaining the current image, writing to a new window or adding items to an existing graph.

There are two means of control over the graphical output. The four directives listed above have a number of options and parameters that allow changes to be made from the default output. In addition, there are four directives that are used to define and modify the graphics "environment": DEVICE, FRAME, AXES and PEN. When you come to produce output, the current settings of the environment will determine the exact appearance of the picture. Genstat has an initial default environment: you can either use this as it stands or modify some settings according to your requirements (the environment directives only set or change those parameters that are specified, any others retaining their existing settings). When you have made the necessary adjustments to the environment you can then start plotting your data. If things are not quite right you can make a small adjustment and then try again – you do not have to respecify all the things that you have set each time you want to plot. If you want to check on what you have set, the HELP directive can give the current settings of the environment and can also tell you what settings are allowed if you wish to make any changes.

4.5. The Graphics Environment

Graphical output is directed to a single device. This may be a graphics terminal if you are working interactively or a *metafile* if you wish to save the output. A metafile is a special kind of output file that contains coded instructions for driving graphics equipment; in general it can be used outside of Genstat for producing output on many kinds of devices. You can name and open the metafile using the OPEN directive (see 3.5), setting FILETYPE=graphics: but only one channel is allowed for graphics files, so you can have only one metafile open at one time. You select the current output device with the DEVICE directive, for example:

```
DEVICE 1
```

You may later decide to change device; for example, by typing DEVICE 0 you switch to the metafile where subsequent output will be stored.

The device numbers will depend on what is available at a particular site. At Rothamsted, device 0 is a metafile and device 1 is a graphics v.d.u. running in Tektronix T4010 mode. A list of the devices supported by a particular implementation of Genstat, with their associated device numbers, can be displayed by HELP.

During a Genstat job you will typically produce a number of plots that are output as a series of frames. Each frame can be divided into windows that are rectangular areas to contain graphical output. The windows can be of any size and located anywhere within the frame and are allowed to overlap. To define a window you use the FRAME directive:

```
FRAME WINDOW=3; YLOWER=0.0; XLOWER=0.0; YUPPER=1.0; XUPPER=1.0
```

This specifies the window number and its upper and lower bounds in the X and Y directions. The co-ordinate system used defines the frame as being a square of size one unit so we have defined window 3 as occupying the entire frame. We could also define window 5, say, by

```
FRAME WINDOW=5; YLOWER=0.0; XLOWER=0.5; YUPPER=0.5; XUPPER=1.0
```

This is the lower right hand quarter of the screen. When you produce output you nominate which window it is to appear in, so by appropriate choice of windows you can fit several graphs into one frame. Keys are also put in windows so you can choose where a key will be drawn relative to its graph.

Associated with each window are a set of parameters defining the style of axes that are to be used when plotting in that window. To set up these parameters for a window you use the AXES directive. The following statement sets *x*-axis and *y*-axis titles for graphs drawn in window 3:

```
AXES WINDOW=3; YTITLE='Temperature'; XTITLE='Time'
```

The axis bounds are normally computed from the data that is being plotted but can be set explicitly:

```
AXES WINDOW=2; YLOWER=0; YUPPER=200; XLOWER=10; XUPPER=50
```

The STYLE parameter controls the drawing of the axes:

```
AXES WINDOW=4; STYLE=xy
```

will draw axes intersecting at the origin. The style box will cause a box to be drawn around the graph and STYLE=grid gives axes with an overlaid grid. You can also use the setting none if you do not want any axes to be drawn.

The XMARKS parameter can be set to a variate giving the positions of the *x*-axis tick marks. In addition the XLABELS parameter can be used to give text labels at these points (you might label the axis by days of the week for example). There are corresponding YMARKS and YLABELS parameters for the *y*-axis.

Genstat has eight notional pens (numbered from one up to eight) that are used for plotting. Each pen has associated with it several attributes which determine how it is to be used; these are set or modified by the PEN directive. It is possible to select plotting method: point, (straight) line or curves; symbols; linestyle (such as continuous or dashed); colour and brush style. For example,

```
PEN NUMBER=5; COLOUR=3; METHOD=point
```

sets up pen 5 so that it can be used for point plots in colour 3. The colours are numbered 1 to 4 within Genstat, but the actual colours used depend on how pens are loaded in the plotter.

After giving the statement

```
PEN NUMBER=3; METHOD=line
```

then if pen 3 is used for plotting, the points will be joined by straight lines. To join the points with smooth curves you can use the method monotonic.

The points are marked with graphics symbols. The SYMBOLS parameter associates a particular symbol with a pen:

```
PEN NUMBER=2; SYMBOL=4
```

Numbers in the range 1...9 can be given for the symbol. You can also use SYMBOL=0 when drawing lines if you do not want individual points to be marked. The symbol can also be set to a factor or text (as in GRAPH).

Linestyle is an integer 0...4, producing a continuous line or various degrees of broken line.

Linestyle 0 produces no line and is the style assumed in point plots. Linestyle 1 is a continuous line, linestyle 4 is a dotted line, and linestyles 2 and 3 are dashed lines.

The brush style is set using the parameter BRUSH to a number in the range 1...16. It defines the pattern used by DPIE and DHISTOGRAM when shading segments of the plot; the higher numbers give a pattern of greater density and hence take longer to produce and plot.

4.6. Graphical Output

Graphical output is produced by the directives DGRAPH, DCONTOUR, DHISTOGRAM and DPIE. Output is sent to two windows, the main window and a key window. For DGRAPH, the main window requires axes to be set; if bounds are not specified they are calculated from the data. If a title is supplied it will be written above the main plot and the key. Text can be provided for a key; alternatively each graph will be identified by the line or symbol used and the variate names. If the symbol is a factor then the factor name and the first level will be given in the key.

All graphical output directives have the following options in common:

WINDOW	integer 0...8	(output can be suppressed by using window 0)
KEYWINDOW	integer 0...8	
TITLE	text string	
SCREEN	keep or clear	(keep means use the existing frame clear means move to the next frame)

They also have the parameters PEN and DESCRIPTION in common. DESCRIPTION is used to supply text for the key and PEN denotes which pen is to be used for drawing the picture. The pen attributes set in previous PEN statements will determine how the pen is used but not all are relevant; for example, the setting of SYMBOL is not used for histograms.

The DGRAPH statement will produce a plot of points specified as variates containing the *x*- and *y*-coordinates. You can also specify windows for the plot and key to appear in and give a pen that is to be used for the plot.

```
DGRAPH [WINDOW=1; KEYWINDOW=2] Y=Yield; X=Nitrogen; PEN=1
```

By default pen 1 is used in window 1, with a key in window 2. If you do not want a key you can use window 0 to suppress it:

```
DGRAPH [WINDOW=4;KEYWINDOW=0] Y=Yield; X=Nitrogen
```

You can supply a title that is printed above the graph and the key

```
TEXT [VALUES='Daily maximum temperature'] Gtitle
DGRAPH [TITLE=Gtitle] Y=Maxtemp; X=Days
```

Several graphs can be drawn in one window by giving lists of variates

```
DGRAPH Y=Y2,Y1; X=X2,X1; PEN=2,1
```

Normally you would want to start each plot in a new frame, so the screen is cleared before the graph is output. You can use several windows in one frame to build a composite picture by plotting to each window in turn, retaining the existing image each time. The SCREEN option gives control over screen clearing.

```
DGRAPH [WINDOW=1; SCREEN=clear] Y=Y1; X=X1; PEN=1
DGRAPH [WINDOW=2; SCREEN=keep] Y=Y2; X=X2; PEN=2
DGRAPH [WINDOW=3; SCREEN=keep] Y=Y3; X=X3; PEN=3
```

Four further parameters of DGRAPH can be used to give error bars to be plotted at the given points.

DHISTOGRAM produces histograms of data stored in variates. Pen 1 is used by default in window 1, as in

```
DHISTOGRAM Heights
```

The BRUSH attribute of the pen determines the shading pattern of the histogram bars.

```
PEN NUMBER=5; BRUSH=6
DHISTOGRAM [WINDOW=1] Heights; PEN=5
```

The data are sorted into categories in the same way as in HISTOGRAM. It is possible to specify the number of groups or group limits by using the NGROUPS and LIMITS options. Data that have already been grouped can be presented as values of a factor or a one-way table of group counts.

If more than one data structure is given, the option APPEND is used to select one of two forms for the plotted histogram. The default setting is APPEND=no; in this case a parallel histogram is plotted with separate bars for each structure in each group. Alternatively APPEND=yes can be used to plot an appended histogram where each group has a single bar divided into separate sections for each structure. To ensure that the plotted histogram can be clearly read different pens should be used for each structure, with different values of BRUSH.

DCONTOUR plots contours as a two-dimensional representation of a three-dimensional surface. A grid of contour heights can be supplied as a matrix or two-way table from which the positions of particular contours are computed for plotting. DCONTOUR can be left to work out default contour heights for itself, otherwise the option INTERVAL can be used to set either the interval between successive contours or the actual contour heights. The options LOWERCUTOFF and UPPERCUTOFF can be used in conjunction with INTERVAL to set minimum and maximum contour heights: these values together with the interval determine the number and height of the contours. For example:

```
DCONTOUR [WINDOW=3; INTERVAL=10] Grid; PEN=4
```

Axes will be drawn according to the settings specified in AXES statements. The choice of pen will determine how the contours are drawn: the relevant settings are METHOD, LINESTYLE and COLOUR. By default the contours are drawn as straight lines joining the interpolated points, but this can be changed to smooth curves by using a different setting of METHOD.

Pie-charts can easily be generated with the DPIE directive. The data are supplied as a list of scalars which represent the size of the segments (or SLICEs). The relevant environment settings are the window and pens. Normally a circular pie-chart is required: this is achieved by plotting in a square window. (Using a rectangular window will produce an elliptical pie-chart.) The segments are shaded according to the BRUSH and COLOUR settings of the pen. Since the PEN parameter list is used in parallel with the SLICE list it is possible to use a different pen for each segment so that there is sufficient contrast between them; for example:

```
PEN 1...4; BRUSH=1,3,5,7
DPIE SLICE=8,12,7,6; PEN=1...4
```

(SAH)

5. Regression Analysis

As well as changes to comply with the new syntax rules and to provide convenient interactive working, the directives for regression analysis have been modified to make simple analysis easier and to provide extra facilities. Advantage has been taken of the change to include extra options to control analysis, and eliminate some restrictions. The facilities for nonlinear regression have been completely integrated with those for linear and generalised linear regression.

5.1. Preliminary Directives

The Y directive is now called MODEL: a MODEL statement must be given first to specify the type of regression. The DISTRIBUTION option specifies the form of the likelihood, and LINK specifies a transformation to be used in a generalised linear model. The WEIGHTS and OFFSET options specify variates of weights and offsets (previously in TERMS), and the numbers of binomial trials for a binomial distribution are specified in the NBINOMIAL parameter. A new option called RMETHOD allows the type of residual to be specified: deviance residuals are now produced by default, but you may ask for Pearson residuals (there is no difference for linear regression), or no residuals at all. Both types of residual are automatically standardised using an estimate of variance for each unit.

The TERMS directive is now optional; if it is given, it must follow the MODEL statement. However, if the model-modification directives ADD, DROP, SWITCH, TRY or STEP are used, TERMS must be given to establish a common set of units on which to base all analyses. The PRINT option of TERMS can still display a correlation matrix, PRINT=correlation; alternatively the CORRELATE directive can now do this. A new option called TOLERANCE allows you to control the detection of aliased parameters.

To fit a simple linear regression in Genstat 5 thus requires only two statements, such as:

```
MODEL Y
FIT X
```

5.2. Fitting Linear or Generalised Linear Models

The PRINT option of FIT, ADD, DROP and SWITCH has been modified. The settings model and summary correspond to a description of the model and a summary analysis of variance plus warnings about influential and outlying observations. A warning is produced whenever a standardised residual or a leverage value is particularly large: warnings can be switched off with the NOMESSAGE option. The settings estimates, correlations and fittedvalues give parameter estimates, a correlation matrix, and a list of fitted values and residuals. All residuals are standardised by their estimated variance. The setting accumulated displays an accumulated summary, including information from all models fitted since the latest TERMS or FIT statement. The old ANDEV option to control formation of this summary has been replaced by the options POOL and DENOMINATOR, but it is no longer necessary to remember to initialize the summary at the start of a sequence of fits.

A new directive called RDISPLAY displays the current regression model. It has an option called PRINT which is set like the PRINT option in the directives FIT, ADD, DROP or SWITCH. The CHANNEL option allows you to send the display to an auxiliary output file.

A new directive RKEEP must be used if results of a fit are to be stored. It has parameters FITTEDVALUES and RESIDUALS to store fitted values and residuals, ESTIMATES to store the estimates of coefficients, DEVIANCE and DF to store the residual sum of squares (or deviance) and the residual degrees of freedom. Two extra sets of results of the analysis can be stored: SE saves the standard errors of the parameters, and TERMS saves the current list of terms in a pointer, or current model formula in a formula structure.

The directives BEST, WORST and MINIMIZE have been combined into STEP.

The PREDICT directive has been extended to help deal with models that include aliased parameters. By default, as in Genstat 4, predictions are not produced from such models; but if the ALIASING option is set to ignore, predictions are produced assuming the value 0 for all aliased parameters. Alternatively, the option COMBINATIONS can be set to present to produce predictions for just those combinations of the classifying factors for which there are observations in the data. This is particularly useful when some factor levels have been excluded from an analysis by restriction, for example. The WEIGHTS option can be set with a table of weights for some or all of the standardising factors, as could be done before in the WEIGHT keyword in Genstat 4; but it can now be classified by the factors classifying the final table of predictions as well, if variable weights are required.

There is still the problem of the large workspace required to form standard errors of predictions from models that have many combinations of factor levels. There is now no effective limit to the number of rows of a symmetric matrix (about 250 on many computers with Genstat 4), but the workspace available in any version of Genstat is still not sufficient to deal, for example, with an analysis involving four factors with 10 levels each. The full variance-covariance matrix of predictions, which must still be formed internally even if prediction is done for only one of the factors in the model, has one row for each combination of factor levels: that is, 10000 rows. It therefore has 10000*10001/2 values; that is, 50,005,000 values.

5.3. Fitting General Nonlinear Models

The directive OPTIMIZE has been renamed FITNONLINEAR. It has been integrated with the linear regression facilities, linking with the directives MODEL, TERMS, ADD, DROP, SWITCH, RDISPLAY and RKEEP in the same way as FIT.

The CALCULATION option specifies an expression structure which stores the required calculations of fitted values. Expression structures are set up explicitly with the EXPRESSION declaration, or implicitly with the !E( ) notation. Several expressions can be specified by setting a pointer in the CALCULATION option.

Alternatively, calculations may be programmed in Fortran, and linked into Genstat as for the OWN directive.

FITNONLINEAR allows separate estimation of linear parameters, associated with variates, as before, but also with one factor – allowing parallel models to be fitted.

A new directive RCYCLE names parameters for nonlinear models. Parameters LOWER, UPPER, STEPS and INITIAL control the iterative search.

For example, to fit an exponential curve (which can actually be fitted much more easily with FITCURVE: see 5.4), the statements in Genstat 4 looked like this:

```
'MODEL' Exp $ Fit = R**X
'OPTIMIZE/LIK=3' Exp; PARAM=R; LOWER=0; Y=Obs; Z=Fit
```

In Genstat 5, the same fit is specified as follows:

```
EXPRESSION [VALUE=Fit=R**X] Exp
MODEL Obs
RCYCLE R; LOWER=0
FITNONLINEAR [CALCULATE=Exp] Fit
```

Though more statements are required in Genstat 5, it is much easier to see how to fit a nonlinear model if you have already fitted linear models.

General functions can still be optimized with FITNONLINEAR, by specifying no response variate in the MODEL statement, and setting the FUNCTION option to a scalar that is to store the function value. For example:

```
MODEL [FUNCTION=F]
RCYCLE A,B
FITNONLINEAR [CALCULATE=!E(F=(10*A-B)**2+B**2)]
```

5.4. Fitting Standard Curves

A new directive FITCURVE fits a range of standard nonlinear curves. The CURVE option specifies the curve to be fitted: exponential, double exponential, critical exponential, line plus exponential, logistic, generalised logistic, Gompertz, linear divided by linear, linear divided by quadratic, and quadratic divided by quadratic. All have been implemented using code from the FITCURVE module of the MLP program.

Options of FITCURVE are as in FIT and FITNONLINEAR, plus CURVE and SENSE to specify the form and shape of the curve.

For each type of curve, four possible models can be fitted for observations that are grouped: common curve, parallel curves, curves with all linear parameters separate, and curves with all parameters separate.

FITCURVE links with the other regression directives, as for FITNONLINEAR. This example carries out an analysis of parallelism for a logistic curve relating Y to X with groups defined by F:

```
MODEL Y
TERMS F*X
FITCURVE [CURVE=logistic] X
ADD F
& F.X
& [PRINT=accumulated; NONLINEAR=separate]
```

(PWL)

6. Analysis of Designed Experiments

The general structure of the directives for analysis of variance remains much the same as in Genstat 4, so there should be little difficulty in adapting to the new syntax. The changes are summarized, and then discussed in more detail.

(a) EXTRACT is renamed AKEEP (corresponding to the directives RKEEP and TKEEP of regression and time series).

(b) A new directive ADISPLAY has been introduced, to simplify the display of further output from an analysis; this corresponds to the directives RDISPLAY and TDISPLAY of regression and time series.

(c) Option and parameter names have been revised to make them more meaningful and easy to remember (but remember that you can always abbreviate these down to four letters at most: see 1.4).

(d) Various (obscure) options like PR and LIMA have been redefined.

(e) There are small changes and extensions in model formulae.

(f) The output has been redesigned to make it easier to read on a video screen; in particular, options PRINT, UPRINT and CPRINT select from nine different components of output instead of the five of their predecessors (PR, PRX and PRYU) and the printing of submodel deviations can be suppressed.

6.1. Model Specification

The directives for specifying the structure of the design and the treatment model to be fitted are BLOCKSTRUCTURE and TREATMENTSTRUCTURE, which abbreviate to the familiar BLOCKS and TREATMENTS. The main expertise required to do an analysis of variance in Genstat 4 has always been in the specification of the block and treatment models. The rules for these model formulae are virtually as before; there is one small change and one extension. The extension is that the formulae can now contain lists of factors; for example:

```
A * B,C * D
```

The list is treated as a series of factors, within brackets and separated by plusses:

```
A * (B+C) * D
```

The change is that the arguments of the functions for specifying contrasts are now separated by semicolons (and so conform to the rules for the functions used in expressions, for example in CALCULATE). Thus, for example,

```
POL(Dose, 2, Doselevs)
```

becomes

```
POL(Dose; 2; Doselevs)
```

in Genstat 5. You can also put lists within the functions: for example

```
REG(A,B; 2; Acont,Bcont)
```

is the same as

```
REG(A; 2; Acont) + REG(B; 2; Bcont)
```

The covariates that are to be fitted in the analysis are specified by the COVARIATE directive, exactly as in Genstat 4.

6.2. Analysis

The name of the directive for doing the analysis is still ANOVA; however some of its options and parameters have been given more appropriate names, and easier methods of specification have been devised.

The correspondence between old and new options and parameters is as follows:

	Genstat 4	Genstat 5
Options:	PR	PRINT) (PFACTORIAL,
	PRX	CPRINT) + (PCONTRASTS and
	PRYU	UPRINT) (PDEVIATIONS
	LIMA	FACTORIAL, CONTRASTS and DEVIATIONS
	WT	WEIGHT
	EFF	TWOLEVEL
	SE	SE
	ACON	DESIGN
	ORTH	ORTHOGONAL
	ALIAS	deleted
	SEED	SEED
	TOL	TOLERANCES and MAXCYCLE
	PROB	FPROBABILITY
Parameters:	VAR	Y
	RES	RESIDUALS
	OUT	SAVE
	FVAL	FITTEDVALUES

One major difference is that you select which components of output are to be displayed by giving a list of strings, instead of the five-digit integer used in the PR, PRX and PRYU options of ANOVA in Genstat 4 (with its cryptic mnemonic "RECAM"). So instead of putting for example

```
PR=10019
```

you would put

```
PRINT=aovtable,means,residuals
```

or more succinctly (as explained in 1.4)

```
PRINT=a,m,r
```

In the Genstat 4 syntax, the order of means, effects or contrasts to be printed was indicated by the value of the corresponding digit of the integer. So in the example 9 was placed in the final digit to ensure that tables of means containing up to nine factors would be printed. (Since there is a limit of 9 on the number of factors in a table, this would cause all the tables of means to be printed.) In Genstat 5, this is controlled by two new options: PFACTORIAL (or PF for short) for tables of means or of effects, and PCONTRASTS (or PC) for tables of contrasts. There is also a further option PDEVIATIONS (or PD) controlling the printing of deviations from fitted contrasts, so if you put

```
PDEVIATIONS=0
```

no deviations will be printed; this is an aspect of the output that could not be controlled in Genstat 4. These three options (PFACTORIAL, PCONTRASTS and PDEVIATIONS) all have 9 as default value.

The use of strings in the PRINT options in Genstat 5 also means that the output can be broken down into more components; the nine available values for PRINT are as follows: aovtable, information, covariates, effects, residuals, contrasts, means, %cv and missingvalues. Thus, to focus on one particular irritation of Genstat 4, the estimates of missing values are no longer tied to the analysis-of-variance table, but are selected separately. Options CPRINT and UPRINT have eight possible values; covariates (which produces the estimated covariate regression coefficients) is not relevant for the analyses of the covariates or for the unadjusted analysis of the *y*-variate.

Another extension to the available output is that the TWOLEVELS option, which takes over from the EFF option in Genstat 4, allows you to request the actual effects instead of the responses given usually given for terms whose factors all have two levels.

Other changes are that the LIMA option of Genstat 4 is split into its three separate parts, instead of being specified as yet another multi-digit integer, and that the aspects of TOL for specifying tolerances for near-zero values and for controlling the maximum number of iterations in the estimation of missing values are specified by two separate options.

6.3. Display of Further Output

To get further output from an analysis in Genstat 4, you needed to save the output structure from the analysis (using the OUT parameter of ANOVA) and then specify the same output structure in the subsequent ANOVA, while omitting to set the VAR parameter. This was not particularly convenient, and has probably not been much used.

In Genstat 5 further output from one or more analyses of variance can be obtained by the new directive ADISPLAY. If you are analysing several *y*-variates at a time, you will still need to store a save structure (as it is now known) for each of them. However the save structure from the last *y*-variate analysed by ANOVA is automatically saved by Genstat and is used as a default by ADISPLAY and also by AKEEP (see below); so if you have only one *y*-variate, it is very simple to look at the output in stages. For example

```
ANOVA [PRINT=aovtable] Gain
ADISPLAY [PRINT=means]
ADISPLAY [PRINT=missingvalues]
```

looks first at the analysis-of-variance table from the analysis of the variate Gain, then (without repeating the calculations) the means are printed, and then the estimates of missing values. This can be made more succinct using the abbreviation rules for strings within an option setting (see 1.5), and by noting also that PRINT is the first option of ANOVA and ADISPLAY:

```
ANOVA [a] Gain
ADISPLAY [m]
& [mi]
```

ADISPLAY has all the options of ANOVA that relate to printed output: PRINT, UPRINT, CPRINT, PFACTORIAL, PCONTRASTS, PDEVIATIONS, FPROBABILITY, SE and TWOLEVELS. There is also an option CHANNEL, like that in PRINT, which allows you to send the output to another output channel. Its parameters are RESIDUALS, FITTEDVALUES and SAVE, so you can also save variates of residuals and fitted values in ADISPLAY as well as in ANOVA.

6.4. Accessing Information from an Analysis

The directive AKEEP replaces the EXTRACT directive of Genstat 4. The general principles for its use are mainly similar to those of EXTRACT. There are five areas of change.

(a) The save structure for the analysis from which the information is to be taken is specified by the SAVE option of AKEEP, instead of by the first parameter as in EXTRACT. The fact that this (unnamed) parameter of EXTRACT was not in parallel with the other parameters caused confusion in Genstat 4. The default for the SAVE option is to take the save structure from the last *y*-variate analysed, thus simplifying the accessing of information when a single *y*-variate is being analysed.

(b) There is a new option FACTORIAL (similar to the option of the same name in ANOVA) which sets a limit on the number of factors in the terms formed from, the model formula defined by TERMS parameter. (This parameter, unnamed in EXTRACT, indicates the model terms for which information is to be obtained.)

(c) Option STRA of EXTRACT, which controlled which strata are to be searched for information, is replaced by two options in AKEEP. Option STRATUM specifies the stratum of interest, by giving its model term (such as Blocks.Wplots) instead of an integer specifying its place in the block formula, as in EXTRACT. A negative value of the integer in the STRA option of EXTRACT indicated that strata above the nominated stratum were not to be searched; in AKEEP this is requested by setting option SUPPRESSHIGHER=yes.

(d) Parameter EFF of EXTRACT, which obtained residuals for block terms and effects for treatment terms, is replaced by two parameters EFFECTS and RESIDUALS.

(e) Names of the options have been expanded or modified to make them more meaningful. The correspondence is as follows:

	Genstat 4 (EXTRACT)	Genstat 5 (AKEEP)
Options:	STRA	STRATUM and SUPPRESSHIGHER
	–	FACTORIAL
	–	SAVE (replaces first parameter)
Parameters:	identifier	deleted (SAVE option)
	model formula	TERMS
	EFF	EFFECTS and RESIDUALS
	MEAN	MEANS
	REP	REPLICATION
	DF	DF
	SS	SS
	VAR	VARIANCE
	SSPM	CSSP
	PEFFECT	PARTIALEFFECT
	COEFFICIENT	CREG

(RWP)

7. Multivariate and Cluster Analysis

The facilities for multivariate analysis in Release 1 of Genstat 5 are largely unchanged from Genstat 4. The CLASSIFY directive used in Genstat 4 to form non-hierarchical groupings is now called CLUSTER, to reflect the formation of separate clusters of units. All the other multivariate directives have the same names as in Genstat 4. Any of the results from an analysis can be saved in output structures. Various minor enhancements have been made to the directives: these are described below.

The facilities for cluster analysis are also largely unchanged. However, the number of directives has been reduced by combining several together; also the directive names have changed. In Genstat 5 there is greater control over the printed output from the directives for cluster analysis; also it is possible to save summary results from a cluster analysis, and information on the minimum spanning tree. These differences are described below.

7.1. Multivariate Analysis

The input to all of the multivariate directives is now supplied as a single data structure. In Genstat 4 the input to some directives consists of several variates, representing a data matrix; in Genstat 5 a pointer, with the variates as its values, must be supplied as input. Thus it is possible in Genstat 5 to do more than one analysis with each directive.

To make directives easier to use, some of the PRINT option settings have been changed for Genstat 5; in particular it is now possible to print either the latent roots or the latent vectors (or both). This is particularly useful for the output from principal co-ordinates analysis, where you might want to inspect all the latent roots, but not have the potentially large quantity of output generated by printing all the co-ordinates. Since the trace – the sum of the latent roots – consists of a single value, it is printed whenever the roots are printed; the trace cannot be printed separately. Requesting that the results for the smallest roots are to be printed is now done with the option setting SMALLEST=yes, rather than by supplying a negative value as the setting of the number of dimensions for which results are to be printed.

Structures supplied to the multivariate directives to save the results from the analyses must all have been declared in advance. Also, the attributes of all such structures that correspond to the number of dimensions must be consistent. However, this number of dimensions may be different from the number printed. Results can only be saved that correspond to the largest latent roots. If you wish to save the latent vectors, roots, and trace from an analysis then you must supply a

structure of type LRV. Similarly, to save the SSP matrix from the PCP directive you must supply a structure of type SSPM.

A PRINT option has been added to the SVD directive. Also, it will now decompose a matrix with fewer rows than columns, in which case the number of singular values is the same as the number of rows of the input matrix. You need not keep all the singular values, and associated left and right vectors; however, the keep structures must have the same number of columns.

7.2. Cluster Analysis

The correspondence between the Genstat 4 directives for hierarchical cluster analysis and those in Genstat 5 is given below.

Purpose	Genstat 5	Genstat 4
To form or print a similarity matrix	FSIMILARITY	SMATRIX, ESMATRIX, SMPRINT
To form or print a reduced similarity matrix	REDUCE	REDUCE
To do hierarchical cluster analysis	HCLUSTER	HIERARCHY
To display results from hierarchical clustering	HDISPLAY	MSBW, NEIGHBOURS, MST, TYPE
To list a data matrix, optionally in groups	HLIST	LIST
To print groups-by-levels tables for each variate	HSUMMARIZE	KEY
To relate variate values to results from PCO	RELATE	RELATE

The printed output from the HCLUSTER directive is in two sections: the first shows the formation of groups; the second is the dendrogram. You can choose to have either (or neither or both) of these printed. The HDISPLAY directive can be used to print or save ancillary results that are relevant to hierarchical cluster analysis; for example, a matrix showing nearest neighbours can be printed.

The *merging clusters* output from HCLUSTER can be saved in a matrix; for example, you could use this to draw a dendrogram with the high-quality graphics facilities. In fact, Genstat 5 procedures have already been written at Rothamsted to do this and will be the subject of an article in a future issue of the Newsletter. The links forming the minimum spanning tree can be saved as matrices from the HDISPLAY directive; for example this would allow you to superimpose the MST on a plot of principal co-ordinate scores for the units.

(PGND)

8. Time-series Analysis

The directives for analysing time series have been changed mainly to bring them into line with the new syntax rules and to make interactive working more convenient. All the time-series directives now obey restrictions imposed by the RESTRICT directive; however, if the restriction is to anything other than a contiguous set of units, you will be faulted. Thus you can, for example, estimate parameters based on the first section of a series by restricting to the first 100 values, say. But you cannot exclude an observation in the middle by restriction: you must do this by replacing it with a missing value.

8.1. Correlation Functions

The directive DERIVE has been split into CORRELATE, producing sample correlation functions from time series, and TSUMMARIZE, producing functions from specified time-series models.

CORRELATE has an option PRINT which can display a correlation matrix from a list of variates, autocorrelations and partial autocorrelations from individual variates, and crosscorrelations from pairs of variates. There is also an option GRAPH to display, in line-printer style, any of these except the correlation matrix. Option MAXLAG specifies the maximum number of lags used in the calculations.

CORRELATE has parameters AUTOCORRELATIONS, PARTIALCORRELATIONS and CROSSCORRELATIONS to store the correlation functions.

8.2. ARIMA Models

The TSM directive is effectively unchanged; it sets up a compound structure defining an autoregressive integrated moving-average model, or a transfer-function model.

The directive PRELIMINARY is now called FTSM (standing for 'Form TSM').

If you want to estimate parameters of transfer-function models, you must specify the transfer-function models using the new directive TRANSFERFUNCTION before using ESTIMATE to estimate the parameters.

A new directive TDISPLAY allows further display of the fit of a model after an ESTIMATE statement has done the calculations. Results from a fit can be stored only with the new directive TKEEP.

The FORECAST directive still provides the calculation of forecasts from a fitted model.

8.3. Spectral Analysis

The directive FOURIER is unchanged, providing cosine or Fourier transforms of real or complex series. There will be a procedure in the standard library for Genstat 5 which calculates the spectral density function of a series.

(PWL)

9. Conversion Summary

This section lists the Genstat 4 directives and their replacements in Genstat 5. Directives appear in the same order as in the Genstat 4 Reference Summary.

9.1. Structures and data handling

Genstat 4	Genstat 5
Structure of Program	
CLOSE	ENDJOB
FOR	FOR
GOTO	†
JUMP	†
LABEL	†
R	-
REFERENCE	JOB
RUN	-
REPEAT	ENDFOR
START	-
STOP	STOP
Data Structures	
ARRAY	VARIATE
DIAGMAT	DIAGONALMATR
DSSP	SSPM
FACTOR	FACTOR
HEADING	TEXT
INTEGER	-
MATRIX	MATRIX
NAME	TEXT
POINTER	POINTER
SCALAR	SCALAR
SYMMAT	SYMMETRICMATRIX
TABLE	TABLE
UNIT	UNITS
VARIATE	VARIATE
Substitution	
ASSIGN	ASSIGN
ENDMACRO	ENDPROCEDURE
LOCAL	-
MACRO *X*	PROCEDURE *X*
SET	POINTER
	EXPRESSION
	FORMULA
	TEXT (# or ##)
USE *X*	*X*

Genstat 4	Genstat 5
Data Handling	
BIN	READ [UNFORMATTED=yes]
BOUT	PRINT [UNFORMATTED=yes]
CALCULATE	CALCULATE / INTERPOLATE / GETATTRIBUTE
CAPTION	PRINT
CLEAR	MERGE
COMBINE	COMBINE
CONTOUR	CONTOUR / DCONTOUR
COPY	CALCULATE (ELEMENTS function)
DESCRIBE	EXTRA and DECIMALS parameters
DEVALUE	DELETE
DISPLAY	CATALOGUE
DUMP	DUMP
ENVIRONMENT	HELP
EOD	: or set by READ [END=]
EQUATE	EQUATE
FETCH	MERGE
GENERATE	GENERATE
GET	RETRIEVE / RESUME
GRAPH	GRAPH / DGRAPH
GROUPS	SORT for values of variates
	CALCULATE for values of factors
HELP	HELP
HISTOGRAM	HISTOGRAM / DHISTOGRAM
INPUT	INPUT / READ [CHANNEL=]
JOIN	CONCATENATE
LINE	SKIP [FILETYPE=out]
MARGIN	MARGIN
OMIT	COMBINE
OUTPUT	OUTPUT
PAGE	PAGE
PERCENT	CALCULATE
POSITION	RESTRICT (SAVESET parameter)
PRINT	PRINT
PUT	STORE / RECORD
READ	READ
RESTRICT	RESTRICT
SAVE	MERGE / STORE
SSP	FSSPM
TABULATE	TABULATE
VALUES	VALUES option or parameter

† Replaced by constructs IF/ELSIF/ELSE/ENDIF , CASE/OR/ELSE/ENDCASE

9.2. Statistical analysis

Designed Experiments

Genstat 4	Genstat 5
ANOVA	ANOVA
BLOCK	BLOCKSTRUCTURE
COVARIATE	COVARIATE
EXTRACT	AKEEP
TREATMENT	TREATMENTSTRUCTURE

Regression

Genstat 4	Genstat 5
ADD	ADD
BEST	STEP
DROP	DROP
FIT	FIT
MINIMIZE	STEP
PREDICT	PREDICT
REGRESS	TERMS
SWITCH	SWITCH
TERMS	TERMS
TRY	TRY
WORST	STEP
Y	MODEL
MODEL	EXPRESSION
OPTIMIZE	FITNONLINEAR

Multivariate

ADPT	ADDPOINTS
CLASSIFY	CLUSTER
CVA	CVA
FACROT	FACROTATE
LRV	FLRV
PCO	PCO
PCP	PCP
ROTATE	ROTATE
SVD	SVD

Cluster Analysis

Genstat 4	Genstat 5
ESMATRIX	FSIMILARITY
HIERARCHY	HCLUSTER
KEY	HSUMMARIZE
LIST	HLIST
MSBW	HDISPLAY
MST	HDISPLAY
NEIGHBOURS	HDISPLAY
REDUCE	REDUCE
RELATE	RELATE
SMATRIX	FSIMILARITY
SMPRINT	FSIMILARITY
TYPE	HDISPLAY

Time Series

DERIVE	CORRELATE
ESTIMATE	ESTIMATE
EXPAND	TSUMMARIZE
FILTER	FILTER
FORECAST	FORECAST
FOURIER	FOURIER
PRELIMINARY	FTSM
TSM	TSM

9.3. New directives

These are the new Genstat 5 directives and their purposes:

ADISPLAY	displays further output from ANOVA
AXES	defines the axes for high-quality graphics
BREAK	temporarily changes the current input channel
CASE	used in the construct CASE/OR/ELSE/ENDCASE
CLOSE	closes the file on a specified channel
COPY	forms a transcript of statements or output or both
DEBUG	gives an implicit BREAK at the end of every statement
DEVICE	switches between graphics devices
DISPLAY	displays the last diagnostic
DPIE	draws a pie-chart
DUMMY	stores an identifier for use in loops and procedures
EDIT	edits text vectors
ELSE	used in CASE/OR/ELSE/ENDCASE and IF/ELSIF/ELSE/ENDIF
ELSIF	used in the construct IF/ELSIF/ELSE/ENDIF
ENDBREAK	ends a BREAK
ENDCASE	used in the construct CASE/OR/ELSE/ENDCASE
ENDDEBUG	ends a DEBUG
ENDIF	used in the construct IF/ELSIF/ELSE/ENDIF
EXIT	jumps to the end of the current control structure
FITCURVE	fits standard nonlinear curves
FRAME	defines the frame for high-quality graphics
GET	accesses details of the current environment
IF	used in the construct IF/ELSIF/ELSE/ENDIF
LRV	declares a structure to store latent vectors,roots and trace
OPEN	opens a file on the specified channel
OPTION	defines the options of a procedure
OR	used in the construct CASE/OR/ELSE/ENDCASE
OWN	used to link in user's own Fortran subprograms (present in Genstat 4, but not in Reference Summary)
PARAMETER	defines the parameters of a procedure
PASS	runs an external program, communicating with Genstat
PEN	sets properties of each pen in high-quality graphics
RANDOMIZE	randomizes the order of a vector/designed experiment
RCYCLE	controls the iterative fitting of GLMs
RDISPLAY	displays the iterative fitting of GLMs
RETURN	returns through input streams
RKEEP	stores regression results
SET	sets details of the current environment
SUSPEND	suspends Genstat to issue operating-system commands
TDISPLAY	displays iterative fitting of time-series models
TKEEP	stores results of fitting time-series models
TRANSFERFUNCTION	specifies model for use in transfer-function modelling

9.4. Functions

These are the Genstat 4 functions and their Genstat 5 equivalents:

Genstat 4	Genstat 5	Genstat 4	Genstat 5
ABS	ABS	NPI	NORMAL
ANG	ANGULAR,ANG	NROW	NROWS
ARCCOS	ARCCOS	NVAL	NVALUES
ARCSIN	ARCSIN	ORDER(*x;y*)	SORT(*x;y*)
CHOL	CHOLESKI	PDT(*x;y*)	PRODUCT(*x;y*)
CHISQ(*p;s*)	CED(*p;s*)	PDTT(*x;y*)	RTPRODUCT(*x;y*)
CORMAT	CORRMAT	RANDU(*seed;s*)	URAND(*seed;s*)
COS	COS	REG	FIT + RKEEP directives
CPROB(*x;s*)	CHISQ(*x;s*)	REPMV	MVREPLACE(*x;y*)
CUM	CUMULATE,CUM	REV	REVERSE
DET	DETERMINANT,DET,D	RSYMRI(*x;y*)	QPRODUCT(*x;y*)
DIFF(*x;s*)	DIFFERENCE(*x;s*)	SIN	SIN
ELEM	ELEMENTS(*x;el;e2*)	SQRT	SQRT
EXP	EXP	SUBMAT	SUBMAT,ELEMENTS
FLOAT	–	SUM	SUM,TOTAL
FPROB(*x;s1;s2*)	FRATIO(*x;s1;s2*)	TMAX	TMAXIMA
	FPROBABILITY(*x;s1;s2*)	TMEAN	TMEANS
FRATIO(*p;s1;s2*)	FED(*p;s1;s2*)	TMED	TMEDIANS
INLINT	INTERPOLATE directive	TMIN	TMINIMA
INTPT	INTEGER,INT	TNMV	TNMV
INV	INVERSE,INV,I	TPDT(*x;y*)	LTPRODUCT(*x;y*)
LINT	INTERPOLATE directive	TRACE	TRACE
LLB(*x;n;p*)	LLB,LLBINOMIAL(*x;n;p*)	TRANS	TRANSPOSE,T
LLG(*x;m;d*)	LLG,LLGAMMA(*x;m;d*)	TSUM	TSUMS
LLN(*x;m;v*)	LLN,LLNORMAL(*x;m;v*)	TTOTAL	TTOTALS
LLP(*x;m*)	LLP,LLPOISSON(*x;m*)	TVAR	TVARIANCES
LOG	LOG	TYPE	GETATTRIBUTE directive
LOG10	LOG10	VAR	VARIANCE,VAR
MAX	MAXIMUM,MAX	VARFAC(*f;x*)	NEWLEVELS(*f;x*)
MEAN	MEAN	VMAX(*x;y*)	VMAXIMA(*x;y*)
MEDIAN	MEDIAN,MED	VMEAN(*x;y*)	VMEANS(*x;y*)
MIN	MINIMUM,MIN	VMIN(*x;y*)	VMINIMA(*x;y*)
NCOL	NCOLUMNS	VNMV(*x;y*)	VNMV(*x;y*)
NED	NED	VREG	FIT + RKEEP directives
NLEV	NLEVELS	VSUM(*x;y*)	VSUMS,VTOTALS(*x;y*)
NMV	NMV	VVAR(*x;y*)	VVARIANCES(*x;y*)

9.5. New Functions

CIRCULATE(*x;s*)
EXPAND
NOBSERVATIONS
RESTRICTION
ROUND
SHIFT(*x;s*)
SOLUTION
TNOBSERVATIONS
TNVALUES
UNSET
VMEDIANS
VNOBSERVATIONS
VNVALUES

(AEA)

A Genstat 5 Procedure for a First Difference Analysis

D B Baird
Agricultural Research Division
M.A.F.
P.O. Box 24
Lincoln
Canterbury
New Zealand

In field experiments the importance of controlling local variation in yields is well recognised and traditionally this has lead to various forms of blocking to control the variation. Although in most instances the traditional methods perform well, sometimes blocking may fail to remove the effects of fertility trends within the site. In these cases the effect of the fertility trends may still be able to be removed by using the neighbouring plots to adjust yields for variation in fertility. A particular neighbour analysis, derived from a simple stochastic model for yields, is the First Difference analysis described by Besag and Kempton [2]. This model, in an alternative form, was first proposed by Williams [4] as the Linear Variance Model.

The model is as follows:

$$Y = Xt + z + e$$

where Y is a $n\times1$ vector of plot yields, t is a $p\times1$ vector of treatment effects with $n\times p$ design matrix X, z is a $n\times1$ vector of trend effects and e is a $n\times1$ vector of independent local errors (white noise). Further the model assumes that the treatment effects are additive (and hence this model does not allow for interplot competition or treatment interference), that the trend term can be described by a random walk process and that the terms t, z and e are all independent.

The analysis of the model is simplified if the differenced yields (i.e. $y_i - y_{i-1}$) are used. If the variance of the innovations of the random walk process is k and the variance of the white noise process is ak (so that a compares the variability of the two processes), then under the assumed model

$$E(DY) = DXt \quad \text{and} \quad Var(DY) = k(I+aDD'),$$

where D is the $n(n-1)$ matrix that forms the differenced yields $y_i - y_{i-1}$.

Thus, apart from the estimation of the nonlinear parameter, a, this is in the form of a standard generalised least-squares analysis. If one makes the further assumption of the the yields being multivariate Normal, then a may be estimated by maximum likelihood. In the procedure below an unbiased estimate of a is obtained by using a profile likelihood, conditioned on the treatment effects. This is equivalent to the restricted maximum likelihood approach of Patterson and Thompson, [3] for estimation of variance parameters. The parameter a is called the tuning parameter as it controls how smooth the behaviour of the estimated trend effects will be. When $a = 0$ the yields are detrended by the the modified second difference operator which forms second differences of yields $(-y_{i+1}+2y_i-y_{i-1})$ except at the end plots. When $a = \infty$ the estimated trend simply becomes the mean of the row of plots and this analysis is in fact equivalent to a classical analysis which removes row effects. The variance k can be estimated by two methods, by the normal generalised least-squares estimator, or by an estimator based on first differences of the error (Besag and Kempton, [2]).

Although the First Difference model may be only a crude representation of the underlying fertility pattern, substantial gains may be still made. This has been examined in a large empirical study of the performance of the First Difference analysis on uniformity data and a large range of simulated yield models (Baird, [1]). This study showed worthwhile gains were often made over classical analyses although the yields did not follow the First Difference model. In all cases examined, the standard errors from the First Difference analysis, based on the first differences of errors estimator of k, were approximately valid under randomisation. Although the standard errors based on the generalised least-squares estimator of k are unbiased under the First

Difference model these could be seriously biased under departures from the model when a was set to zero or an estimate close to zero was obtained in the analysis. However when a was not estimated as being close to zero the estimator of k based on generalised least-squares (with a loss of two degrees of freedom as suggested by Besag and Kempton) was slightly more efficient than the estimator based on first differences of errors.

The procedure FDIFF carries out a one-dimensional (within row) First Difference analysis for a rectangular array of plots. The procedure uses an alternative parameterisation of the tuning parameter, λ. The two parameterisations are related by $\lambda = a/(1+a)$. Thus $\lambda = 0$ corresponds to no white noise component in the model and $\lambda = 1$ corresponds to no random walk component. λ is used as it has better properties in the maximization of the likelihood and remains finite. The value of the tuning parameter, λ, can be determined by the user or else estimated as outlined above.

Options and their defaults:

EST_TP – The tuning parameter to be estimated (YES, NO). Default NO.

PRECISION – The accuracy to which the tuning parameter is to be estimated to. A scalar or number. Default 0.01.

LAMBDA – Value of the tuning parameter to be used if it is not to be estimated. Default 0.

NAMES – Text list of names of the treatment codes (only the first 11 characters of each name will be printed). Default 1,2...NT.

RESIDUALS – Residuals from the analysis to be printed (YES, NO). Default NO.

OUTFILE – Name of output file (set to be output channel 2). Default FDIFF.OUT.

Input parameters (scalars unless specified):

NR – Number of rows of plots.

NC – Number of columns of plots.

NT – Number of treatments.

YIELD – Variate of plot values input row by row (length NR*NC).

TREAT – Variate containing the corresponding plot treatment codes.

Output parameters:

EFFECTS – A variate of length NT containing the centred treatment effects.

SEDGLS – A symmetric matrix containing the standard errors of all pairs of treatment differences calculated from the GLS estimate of variance.

SED2D – The equivalent standard errors calculated from the RMS of first differences.

The procedure FDIFF and a procedure it calls, FITFD may be stored in a procedure library with the following commands

```
OPEN 'PROCEDURE.LIB' ; CHANNEL = 1 ; FILETYPE = BACKINGSTORE
STORE [PROCEDURE=YES ; CHANNEL = 1] FDIFF,FITFD
CLOSE 1 ; FILETYPE=BACKINGSTORE
```

and it may then be used within any other program with the commands

```
OPEN 'PROCEDURE.LIB' ; CHANNEL=1 ; FILETYPE = PROCEDURELIBRARY
FDIFF [ options ] parameters
```

The procedure FDIFF is as follows:

```
PROCEDURE 'FDIFF'
OPTION 'EST_TP','PRECISION','LAMBDA','NAMES','RESIDUALS','OUTFILE' ; \
        MODE = T,V,V,T,T,T ; \
        DEFAULT = 'NO',!(0.01),!(0),'NotSet','NO','FDIFF.OUT'
PARAMETER 'NR' , 'NC' , 'NT' , 'YIELD' , 'TREAT' , \
          'EFFECTS' , 'SEDGLS' , 'SED2D' ; MODE = P
SCALAR N,NC1,NC2,PI,NT1,DF,Lambda "Setup scalars to be used"
CALC N = NR*NC  &  NC1 = NC - 1  &  NC2 = NC - 2  &  PI = ARCCOS(0)*2
  &  NT1 = NT - 1  &  DF = NR*NC1 - NT + 1  &  Lambda = LAMBDA
"Check input of options and parameters"
TEXT [ NT ] Trt_Name ; CHAR=11  "Check if NAMES have been set"
GETATTRIBUTE [ NV ] NAMES ; NVN
IF ( #NVN == NT )
   EQUATE NAMES ; Trt_Name
ELSE "Default Names"
   IF ( #NVN > 1 )
      PRINT 'Incorrect number of names provided, default names will be used'
   ENDIF
   PRINT [ Trt_Name ; SQUASH = Y ] !(1...NT) ; FIELDWIDTH = 11 ; DECIMALS = 0
ENDIF
"Check YIELD for correct number of values and no missing values"
IF ( NOBS(YIELD) <> N ) .OR. ( NMV(YIELD) > 0 )
   PRINT [ SQUASH=Y ; SERIAL=Y ] 'ERROR: Incorrect numbers of values in YIELD',\
         'NOTE : This procedure can not analyse data containing missing values'
     &   [ IPRINT = * ; SERIAL = N ] N ,' values were expected.' ; \
         FIELDWIDTH = 4 ; DECIMALS = 0
   DUMP [ PRINT = ATTRIBUTES,VALUES ] YIELD
   EXIT [ CONTROL = PROCEDURE ]
ENDIF
"Check TREAT for appropriate values"
SORT [ GROUPS = TFACTOR ; INDEX = TREAT ]
IF ( ( NLEVELS(TFACTOR) <> NT ) .OR. ( MAX(TREAT) <> NT ) /
    .OR. ( NMV(TREAT) > 0 ) .OR. ( NOBS(TREAT) <> N ) )
   PRINT [ SQUASH=Y ] 'ERROR: The treatment codes are not of the correct form.'
     &   [ IPRINT=* ] 'This procedure expects the treatment codes to be 1,2...'\
                     ,NT ; FIELDWIDTH = 3 ; DECIMALS = 0
     &  'The treatment codes should be a variate of length ',N ; \
                     FIELDWIDTH = 3 ; DECIMALS = 0
   DUMP [ PRINT = ATTRIBUTES,VALUES ] TREAT
   EXIT [ CONTROL = PROCEDURE ]
ENDIF
OPEN NAME = OUTFILE ; CHANNEL = 2 ; FILETYPE = OUTPUT ; WIDTH = 132
"Set up matrices used in the First Difference analysis"
MATRIX [ NR ; NC ] Yields,Treatmnt ; !( #YIELD ),!( #TREAT )
  &    [ NC1 ; NC ] D ; !(( 1, -1, #NC1(0) )#NC2, 1 , -1) "F. Difference operator"
  &    [ N ; NT1 ] T "Treatment Design Matrix"
SYMM [ NC1 ] DDT,V
  &  [ NT1 ] VarCov
DIAG [ NC1 ] W
VARIATE Index1,Index2,Index3,Index4 ; !(1...NC1),!(1...NC2),!(2...NC),!(1...NT1)
CALC U = Yields$[* ; Index1] - Yields$[* ; Index3]
  &  DDT = QPRODUCT( T(D$[Index2 ; Index1]) ; 1)
  &  W = 2*(1 - COS(PI*Index1/NC)) "CWC is the spectral decomposition of DDT"
  &  C   = SQRT(2/NC)*SIN( PI*( Index1 *+ T(Index1) )/NC )
  &  TCOL[1...NT] = (TREAT == 1...NT)
  &  T$[* ; 1...NT1] = TCOL[1...NT1]
```

```
FOR I = 1...NR "Set up the Differenced Design Matrix F for each row"
   CALC P1 = NC*(I - 1) + 1  &  P2 = NC*I
     &  F[I] = T(D *+(T$[!(P1...P2) ; *]))
ENDFOR
PAGE [ 2 ] "Print out Yields and Treatments"
PRINT [ 2 ; RLWIDTH = 2 ; SERIAL = Y ; SQUASH = Y ] \
      'Data to be analysed by First Difference method', \
      '-----------------------------------------'
  &   [ SQUASH = N ] Yields ; FIELDWIDTH = 7
  &   Treatmnt ; FIELDWIDTH = 4 ; DECIMALS = 0
  &   [ SQUASH = Y ] 'Note that the differences will be taken along rows'
DELETE [ YES ] Yields,Treatmnt,D,T,P1,P2,PI,Index1,Index2,Index3,NVN,NC2,TFACTOR
"Search for the optimum of the modified profile likelihood if the Tuning
 Parameter is to be estimated. (The golden section search procedure is used)."
IF ( EST_TP .EQS. 'YES' )
   CALC Phi2 = ( Phi = ( SQRT(5) - 1 )/2 )**2 "Golden Ratio and its square"
     &  Bounds = !( 0 , Phi2 , Phi , 1 ) "Points bounding the m.l.e. of Lambda"
     &  Likhd = !( * , * , * , * ) "Modified profile likelihood values"
   "Set up initial values of the likelihood"
   FOR K = 1...4
      CALC Lambda = Bounds$[K]
      FITFD Lambda;NR;NT1;NC1;DF;C;W;U;F;V;VarCov;Effects;Likelihood
      CALC Likhd$[K] = Likelihood
   ENDFOR
   "Search for the maximum likelihood and stop when the range < Precision"
   FOR [ 100 ]
      IF ( Likhd$[2] > Likhd$[3] )
         CALC Bounds$[4] = (Bounds$[1] - Bounds$[2] + Bounds$[3])  &  K = 2
      ELSE
         CALC Bounds$[1] = (Bounds$[2] - Bounds$[3] + Bounds$[4])  &  K = 3
      ENDIF
      SORT Bounds,Likhd ; Bounds,Likhd
      CALC Lambda = Bounds$[K]
      FITFD Lambda;NR;NT1;NC1;DF;C;W;U;F;V;VarCov;Effects;Likelihood
      CALC Likhd$[K] = Likelihood
      EXIT (Bounds$[4] - Bounds$[1]) < PRECISION
   ENDFOR
   "Choose Lambda from Bounds to be the value with the maximum likelihood"
   SORT [ INDEX = Likhd ] Bounds
   CALC Lambda = Bounds$[4]
     &  DF = DF - 2 "Subtract 2 degrees of freedom for estimating Lambda"
   DELETE [ YES ] Bounds,Likhd,Phi,Phi2
ENDIF
"Refit the FD model for choosen value of Lambda"
FITFD [ CALC='EFFECTS' ] Lambda;NR;NT1;NC1;DF;C;W;U;F;V;VarCov;Effects;Likelihd
"Calculate the two estimates of the variance"
SCALAR M,Deviance,RSS2 ; 0
IF ( Lambda < .9 )
   CALC M = 2*NR*( NC - 2 )*( 1 + 3*Lambda/(1 - Lambda) )
ELSE
   CALC M = 2*NR*( NC - 2 )*( 2 + 1/Lambda )
ENDIF
FOR I = 1...NR
   CALC E = U$[I ; *] - T(Effects) *+ F[I]
     &  Deviance = Deviance + QPRODUCT( E ; V )
     &  M = M - TRACE( QPRODUCT( F[I] ; DDT ) *+ VarCov )
     &  RSS2 = RSS2 + QPRODUCT( E ; DDT )
ENDFOR
CALC MeanDev = Deviance/DF
  &  RMS2 = RSS2/M
```

```
"Calculate full set of treatment effects (subject to the constraint
 that they sum to zero) and the standard errors of their differences"
TEXT Stat ; !T(Minimum,Mean,Maximum)
DIAG [ NT ] DiagVar
SYMM [ Trt_Name ] J,VarDiff,SDGLS,SD2D
VARIATE [ NT ] C_Effects ; !((0)#NT)
CALC C_Effects$[Index4] = Effects
  & C_Effects = C_Effects - MEAN(C_Effects)
  & VarDiff = 0
  & VarDiff$[Index4,Index4] = VarCov
DELETE [ YES ] Effects,VarCov,Index4
CALC DiagVar = VarDiff  &  J = 1
  & VarDiff = MeanDev*( DiagVar*+J + J*+DiagVar - 2*VarDiff )
  & VarDiff = VarDiff**2/VarDiff "This gives the diagonal missing values"
  & SDGLS = SQRT( VarDiff )
"Calculate summary statistics for the Std Errors of Differences"
  & Min = MIN(SDGLS)  &  Mean = MEAN(SDGLS)  &  Max = MAX(SDGLS)
  & StatGLS = !( Min , Mean , Max )
  & SD2D , Stat2D = ( SDGLS , StatGLS ) * SQRT( RMS2/MeanDev )
"Calculate the alternative parameterisation of the
 tuning parameter used by Kempton and Besag "
IF ( Lambda < 1 )
   CALC Alpha = Lambda/(1 - Lambda)
ELSE
   TEXT Alpha ; 'Infinity'
ENDIF
DELETE [ YES ] J,E,F[1...NR],DDT,V,U,Min,Mean,Max
PAGE [ 2 ] "Print results"
PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ] \
      'RESULTS FROM A FIRST DIFFERENCE ANALYSIS', \
      '=======================================',
SKIP  [ 2 ; FILE = OUTPUT ] 1
PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ] \
      'Value of the tuning parameter ( Note Alpha = Lambda/(1 - Lambda))'
SKIP  [ 2 ; FILE = OUTPUT ] 1
PRINT [ 2 ; SERIAL = N ; SQUASH = Y ] Lambda , Alpha
SKIP  [ 2 ; FILE = OUTPUT ] 2
PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; INDENT = 10 ] \
      'Fitting statistics','-----------------'
SKIP  [ 2 ; FILE = OUTPUT ] 1
PRINT [ 2 ; SERIAL = N ; SQUASH = Y ] Deviance , DF , MeanDev
SKIP  [ 2 ; FILE = OUTPUT ] 1
PRINT [ 2 ; SERIAL = N ; SQUASH = Y ] RSS2 , M , RMS2
SKIP  [ 2 ; FILE = OUTPUT ] 1
PRINT [ 2 ; SERIAL= Y ; SQUASH = Y] \
      'Note if Lambda < .9 , both MeanDev and RMS2 estimate the variance', \
      'of the innovations in the random walk process assumed by the',\
      'First Difference (FD) model, otherwise these estimate the variance of', \
      'the white noise component of the model.', \
      'MeanDev is an estimate based on Generalised Least-squares.', \
      'The degrees of freedom (DF) for the Deviance are reduced by 2', \
      'when the tuning parameter is estimated.', \
      'When Lambda is close to zero and the assumed First Difference model', \
      'is incorrect, the estimate based on GLS can be seriously biased', \
      'RMS2 is an estimate based on First Differences of Errors.', \
      'RMS2 is more robust to departures from the First Difference model',\
      'than MeanDev but for Lambda > .3 may it be a less efficient estimator',\
      'than MeanDev.'
SKIP  [ 2 ; FILE = OUTPUT ] 2
PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; INDENT = 16 ] \
      'Estimated Treatment effects','--------------------------'
SKIP  [ 2 ; FILE = OUTPUT ] 1
PRINT [ 2 ; SERIAL = N ; SQUASH = Y ; ORIENT=ACROSS ] Trt_Name,C_Effects
```

```
SKIP  [ 2 ; FILE = OUTPUT ] 2
PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; INDENT = 10 ] \
      'Standard Errors of Differences of Treatment Effects based on RMS2',\
      '-------------------------------------------------------------'
  &   [ IPRINT = * ; INDENT = 0 ] SD2D
SKIP  [ 2 ; FILE = OUTPUT ] 1
PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; IPRINT = * ; INDENT = 18 ] \
      'Summary Statistics of SEDs','-------------------------'
  &   [ ORIENT = ACROSS ; SERIAL = N ; INDENT = 12 ] Stat,Stat2D
IF ( Lambda > .3 )
   SKIP  [ 2 ; FILE = OUTPUT ] 2
   PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; INDENT = 10 ] \
         'Standard Errors of Differences of Treatment Effects based on MeanDev',\
         '-------------------------------------------------------------'
     &   [ IPRINT = * ; INDENT = 0 ] SDGLS
   SKIP  [ 2 ; FILE = OUTPUT ] 1
   PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; IPRINT = * ; INDENT = 18 ] \
         'Summary Statistics of SEDs','-------------------------'
     &   [ORIENT = ACROSS ; SERIAL = N ; INDENT = 12 ] Stat,StatGLS
ENDIF
DELETE [ YES ] Alpha,Deviance,MeanDev,DF,RSS2,M,RMS2,TrtName,Stat,StatGLS,Stat2D
IF ( RESIDUALS .EQS. 'YES' )
   MATRIX [ N ; NT ] T
   CALC T$[* ; 1...NT] = TCOL[1...NT] "Set up the full design matrix"
     &  E = YIELD - T *+ C_Effects "Calculate the treatment corrected Yields"
   MATRIX [ NR ; NC ] TC_Yields ; !(#E)
   PAGE [ 2 ] "Print out residuals"
   PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; INDENT = 16 ] \
         'Treatment corrected yields' ,'-------------------------',
   SKIP  [ 2 ; FILE = OUTPUT ] 1
   PRINT [ 2 ; IPRINT = * ; SERIAL = Y ; SQUASH = Y ; RLWIDTH = 3 ] \
         TC_Yields ; FIELDWIDTH = 7
   DELETE E,T,TCOL[1...NT]
   IF ( Lambda < .05 )
      SKIP  [ 2 ; FILE = OUTPUT ] 2
      PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; INDENT = 16 ] \
            'As the tuning parameter is close to zero there is no white noise',\
            'component in the model and the estimated trend effects are just', \
            'the same as the treatment corrected yields.'
   ELSE "Calculate the decomposition in to trend and white noise"
      MATRIX [ NR ; NC ] Trend , WNoise
        &     [ NC ; 1 ] One , Position ; !((1)#NC) , !(1...NC)
        &     [ 1 ; NC ] ER
      DIAG [ NC ] Identity ; !((1)#NC)
      SYMM [ NC ] S
      "Calculate the Smoothing Matrix S
       S = I - ((I - J/N)V(I - J/N))+ where + stands for the Moore-Penrose G.I."
      IF ( Lambda > .99 )
         CALC S = One*+T(One)/NC
      ELSE
         CALC V = Identity -ABS(One*+T(Position)-Position*+T(One))*(1/Lambda-1)
           &  S = V - (V*+One)*+T(One/NC) - (One/NC)*+(T(One)*+V) + MEAN(V)
         "Calculate the Moore Penrose Generalised Inverse to obtain S"
         LRV   [ NC ] L
         FLRV S ; L "Calculate Eigenvalue-vector decompostion"
         "Set any eigenvalues < 1 to zero (as all non-zero E. values are > 1)"
         CALC L[2] = L[2]*(L[2] > 0.9)
         "Obtain the reciprocals of eigenvectors setting 1/0 = 0"
           &  [ZDZ=ZERO] L[2] = (L[2]/L[2])/L[2]
           &  S = Identity - QPRODUCT(L[1] ; L[2])
      ENDIF
      DELETE [ YES ] V,Identity,One,Position,L
```

```
      "Form the estimated trend effects (smoothed treatment corrected yields)
       and the white noise residuals for each row"
      FOR I = 1...NR
          CALC ER = TC_Yields$[I ; *]
            &  Trend$[I ; *] = ER*+S
            &  WNoise$[I ; *] = ER - Trend$[I ; *]
      ENDFOR
      SKIP  [ 2 ; FILE = OUTPUT ] 2
      PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; INDENT = 16 ] \
            'Estimated Trend Effects','----------------------'
      SKIP  [ 2 ; FILE = OUTPUT ] 1
      PRINT [ 2 ; IPRINT = * ; SERIAL = Y ; SQUASH = Y ; RLWIDTH = 3 ] \
            Trend ; FIELDWIDTH = 7
      SKIP  [ 2 ; FILE = OUTPUT ] 2
      PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ; INDENT = 16 ] \
            'Residual White Noise Component','----------------------------'
      SKIP  [ 2 ; FILE = OUTPUT ] 1
      PRINT [ 2 ; IPRINT = * ; SERIAL = Y ; SQUASH = Y ; RLWIDTH = 3 ] WNoise
      IF ( Lambda == 1 )
         SKIP  [ 2 ; FILE = OUTPUT ] 2
         PRINT [ 2 ; SERIAL = Y ; SQUASH = Y ] \
         'Note this model is equivalent to a classical analysis which only', \
         'removes row effects (hence the trend is simply the mean of each row)'
      ENDIF
   ENDIF
   DELETE [ YES ] TC_Yields,Trend,WNoise,ER,S
ENDIF
"Output results to any structures which have been set"
IF .NOT.UNSET( EFFECTS ) : CALC EFFECTS = C_Effects : ENDIF
IF .NOT.UNSET( SEDGLS  ) : CALC SEDGLS  = SDGLS     : ENDIF
IF .NOT.UNSET( SED2D   ) : CALC SED2D   = SD2D      : ENDIF
ENDPROCEDURE
```

The procedure FITDF, given below, is called by FDIFF.

```
PROCEDURE 'FITFD' "Procedure to fit the FD model for a given value of Lambda"
OPTION 'CALC' ; MODE = T ; DEFAULT = 'LIKELIHOOD'
PARAMETER 'LAMBDA','NR','NT1','NC1','DF','C','W','U','F', "Input parameters" \
          'V','VARCOV','EFFECTS','LIKELIHOOD' ; MODE = P  "Output parameters"
"Set up the weight matrix V (the inverse of the variance-covariance matrix for
 differenced plot yields)"
IF ( LAMBDA == 0 )
   CALC V = ( W > 0 ) "This gives the identity matrix"
ELSIF ( LAMBDA < .9 )
   CALC V = QPRODUCT( C ; (1/(1 + LAMBDA/(1 - LAMBDA)*W )) )
ELSE
   CALC V = QPRODUCT( C ; (1/( 1/LAMBDA - 1 + W )) )
ENDIF
"Calculate the variance-covariance matrix for the reduced set of parameters"
CALC VARCOV = 0
FOR I = 1...NR
   CALC VARCOV = VARCOV + QPRODUCT( F[I] ; V )
ENDFOR
CALC VARCOV = INV(VARCOV)
"Calculate the reduced ( NT - 1 ) set of treatment effects"
VARIATE [ NT1 ] EFFECTS ; !(#NT1(0))
FOR I = 1...NR
   CALC EFFECTS = EFFECTS + VARCOV *+ F[I] *+ V *+ T(U$[I ; *])
ENDFOR
IF ( CALC .EQS. 'EFFECTS' ) : EXIT [CONTROL=PROCEDURE] : ENDIF
```

```
"Calculate the Modified Profile Likelihood"
MATRIX [NR ; NC1] E "Calculate the residuals of the differenced data"
VARIATE [ NC1 ] ESum
CALC E$[1...NR ; *] = ( (U$[1...NR ; *] - T(EFFECTS) *+ F[1...NR]) *+ C )**2
  &  ESum = T(E) *+ !((1)#NR) "Sum the residuals within a column"
IF ( LAMBDA < .9 )
   CALC LV = SUM( LOG( 1 + LAMBDA/(1 - LAMBDA) * W ) )
     &  LEVE = LOG( SUM( ESum/( 1 + LAMBDA/(1 - LAMBDA) * W ) ) )
ELSE
   CALC LV = SUM( LOG( 1/LAMBDA - 1 + W ) )
     &  LEVE = LOG( SUM( ESum/( 1/LAMBDA - 1 + W ) ) )
ENDIF
CALC LIKELIHOOD = -( NR*LV + DF*LEVE - LOG(DET(VARCOV)) )/2
DELETE [ YES]  LV,LEVE,E,ESum
ENDPROCEDURE
```

References

[1] Baird, D.B.
Neighbour models in field trials.
Unpublished MSc thesis, University of Reading, 1986.

[2] Besag, J.E. and Kempton, R.A.
Statistical analysis of field experiments using neighbouring plots.
Biometrics, **42**, pp. 231-252, 1986.

[3] Patterson, H.D. and Thompson, R.
Recovery of inter-block information when block sizes are unequal.
Biometrika, **58**, pp. 545-554, 1971.

[4] Williams, E.R.
A neighbour model for field experiments.
Biometrika, **73**, pp. 279-287, 1986.

The Use of Pseudo-Factors for a Balanced 6×6 Row-and-Column Design for Nine Treatments

D A Preece
Institute of Horticultural Research
East Malling
Maidstone
Kent
United Kingdom ME19 6BJ

As many Genstat programmers report difficulty in understanding the use of pseudo-factors, it seems worthwhile to describe their use in the analysis of a design chosen for a 1986 experiment in one of the orchards at East Malling. This was a 6×6 row-and-column design similar to a 6×6 Latin square, but it compared nine treatments each replicated four times, not six treatments each replicated six times:

3	1	4	6	7	9
9	6	5	7	3	2
6	8	2	5	4	7
5	7	3	1	8	4
8	3	1	9	2	5
4	2	6	8	9	1

This design is balanced in the sense that all estimated pairwise differences between treatments have the same variance if the estimates are obtained without recovery of inter-row and inter-column information on the treatments. In fact, 345 combinatorially distinct 6×6 designs for nine treatments are known to have such balance (Preece, [1] and [2]). The design used in 1986 was chosen at random from the 345 possibilities and was then properly randomised. The methodology to be described for its analysis would apply equally to any of the other 344 possibilities.

The obvious Genstat factors `Row`, `Column` and `Treatmnt` are easily set up for this design. Then anyone who knows the design to be balanced in the sense described may well be tempted to specify `BLOCK` and `TREATMENT` formulae as follows (Genstat 5 notation):

```
BLOCKS  Row * Column
TREATMENTS  Treatmnt
```

Genstat will indeed accept this but will produce an incorrect analysis of variance with an impossible 8 d.f. for `Treatmnt` in each of the `Row` and `Column` strata. Each of these strata will then find itself with –3 residual d.f.; warning messages about these negative numbers of d.f. will be printed. The information summary will nevertheless report a correct efficiency factor of 0.875 for `Treatmnt` in the `Row.Column` stratum.

The reason for these mistakes in the analysis is failure to recognise the *partial* confounding of the treatments with the rows and with the columns. The easiest way for the user to see how to overcome this is to write, alongside each row and column of the design, the treatments that are *absent* from that row or column:

3	1	4	6	7	9	2	5	8
9	6	5	7	3	2	1	4	8
6	8	2	5	4	7	1	3	9
5	7	3	1	8	4	2	6	9
8	3	1	9	2	5	4	6	7
4	2	6	8	9	1	3	5	7

1	4	7	2	1	3
2	5	8	3	5	6
7	9	9	4	6	8

We note in passing that the twelve sets of three absentees constitute the twelve blocks of a balanced incomplete block design – which explains why the row-and-column design is balanced in the sense described.

A few moments' scrutiny shows that the absentees in rows 1, 3 and 5 constitute a replicate of the treatments, as do the absentees in rows 2, 4 and 6, those in columns 1, 2 and 6 and those in columns 3, 4, and 5. This is the clue that leads us to the definition of the three-level pseudo-factors needed to handle the partial confounding.

Consider first the sets of absentees in rows 1, 3 and 5. The first of these sets is (2, 5, 8), and we let these three treatments be at level 1 of our first pseudo-factor PF1. The second set is (1, 3, 9), which gives us the three treatments for level 2 of PF1. Then we have treatments (4, 6, 7) for level 3 of PF1. In Genstat 5 notation, the pseudo-factor PF1 can thus be set up as follows:

```
FACTOR [LEVELS = 3] PF1
CALCULATE PF1 = NEWLEVELS (Treatmnt; !(2,1,2,3,1,3,3,1,2))
```

Similarly pseudo-factors PF2, PF3 and PF4 can be defined from rows 2, 4 and 6, from columns 1, 2 and 6, and from columns 3, 4 and 5. The Genstat code can be written concisely as follows:

```
FACTOR [LEVELS = 3]  PF1,PF2,PF3,PF4
CALCULATE PF1,PF2,PF3,PF4 = NEWLEVELS(4(Treatmnt); \
                                !(2,1,2,3,1,3,3,1,2) , \
                                !(1,2,3,1,3,2,3,1,2) , \
                                !(1,1,3,2,2,3,1,3,2) , \
                                !(3,2,2,2,3,3,1,1,1) )
```

All that remains is to specify BLOCKS and TREATMENTS formulae as follows:

```
BLOCKS  Row * Column
TREATMENTS  Treatmnt//(PF1 + PF2 + PF3 + PF4)
```

The analysis of variance then correctly has 4 d.f. for Treatmnt in the Row stratum, 4 in the Column stratum, and all 8 in the Row.Column stratum. The Information Summary explains this in detail, with efficiency factors (e.f.):

```
***** Information summary *****

Model term                      e.f.   non-orthogonal terms

Row stratum
  PF1                          0.125
  PF2                          0.125

Column stratum
  PF3                          0.125
  PF4                          0.125

Row.Column stratum
  PF1                          0.875  Row
  PF2                          0.875  Row
  PF3                          0.875  Column
  PF4                          0.875  Column

Aliased model terms
Treatmnt
```

When the (one-way) table of Treatmnt means is printed, the pseudo-factor levels appear for each treatment, and standard errors are given in the following format:

```
*** Standard errors of differences of means ***

Table              Treatmnt
rep.                      4
s.e.d.                2.355
Except when comparing means with the same level(s) of
 PF1                  2.039
 PF2                  2.039
 PF3                  2.039
 PF4                  2.039
```

As *any* two means are at the same level of one of the pseudo-factors, the s.e.d. of 2.355 does not apply to *any* difference between means and should thus be ignored, leaving 2.039 as the sole s.e.d. – which confirms the balance of the design.

With the TREATMENTS formula that has just been used, the pseudo-factors PF1, PF2, PF3 and PF4 exhaust the 8 d.f. for Treatmnt, but the program cannot be expected to recognise this – hence the spurious s.e.d. of 2.355. This awkwardness can be overcome by recognising that the 4 d.f. for PF3 + PF4 are also the 4 d.f. for the interaction between PF1 and PF2. So we can write:

```
BLOCKS  Row * Column
TREATMENTS  Treatmnt //(PF1 * PF2)
```

This gives the correct Information Summary as

```
***** Information summary *****

Model term                          e.f.   non-orthogonal terms

Row stratum
  PF1                              0.125
  PF2                              0.125

Column stratum
  PF1.PF2                          0.125

Row.Column stratum
  PF1                              0.875  Row
  PF2                              0.875  Row
  PF1.PF2                          0.875  Column

Aliased model terms
Treatmnt
```

and the standard errors simply as

```
*** Standard errors of differences of means ***

Table              Treatmnt
rep.                      4
s.e.d.                2.039
```

The term (PF1 * PF2) in the formula could of course have been replaced by (PF3 * PF4). However, the crossing of a Row pseudo-factor with a Column pseudo-factor would wrongly produce 6 d.f. for Treatmnt in each of the Row and Column strata.

References

[1] Preece, D.A.
Balanced 6×6 designs for 9 treatments.
Sankhyā B, **30**, pp. 443-446, 1968.

[2] Preece, D.A.
A second domain of balanced 6×6 designs for nine equally replicated treatments.
Sankhyā B, **38**, pp. 192-194, 1976.