THE DESIGN AND MIXED-MODEL ANALYSIS OF EXPERIMENTS

Appendix A Introduction to GENSTAT

Genstat is a comprehensive statistical package. It was originally a programming language and this aspect still survives. Consequently, analyses can be specified using commands. However, because it includes all the standard programming constructs, it is extensible. With the Windows implementation came the ability to use menus to specify analyses and this makes it much easier especially for new and casual users. The menus actually generate the commands that are then executed by Genstat. There is also a Unix version of Genstat, although a linux version is not due for release until later this year — I am not sure about how extensive the menu system is under Unix although I think that it can be run using X-windows.

Its capabilities include:

- Data entry via its own or Excel spreadsheets
- Summarize data in tables and diagrams
- Perform calculations on data
- Regression including multiple linear, generalized linear, generalized additive and nonlinear regression
- Design and analysis of experiments
- Multivariate analyses
- Time series analysis
- Analysis of spatial data

The Windows version

Starting Genstat

You start Genstat for Windows by double-clicking on the Genstat icon, either on the desktop or in the *Start > Programs* menu. The Genstat *Server* starts first — this is program that is going to perform the actions you specify. Second, the Genstat window is opened. It is in this window that you specify what you want the server to do for you. It has a menu bar, a tool bar and, at the start, two other sub-windows. The sub-window that we are most interested in is the Output window. Also, note Help on the menu bar — it gives you access to extensive on-line documentation of the Genstat system.

Data Entry

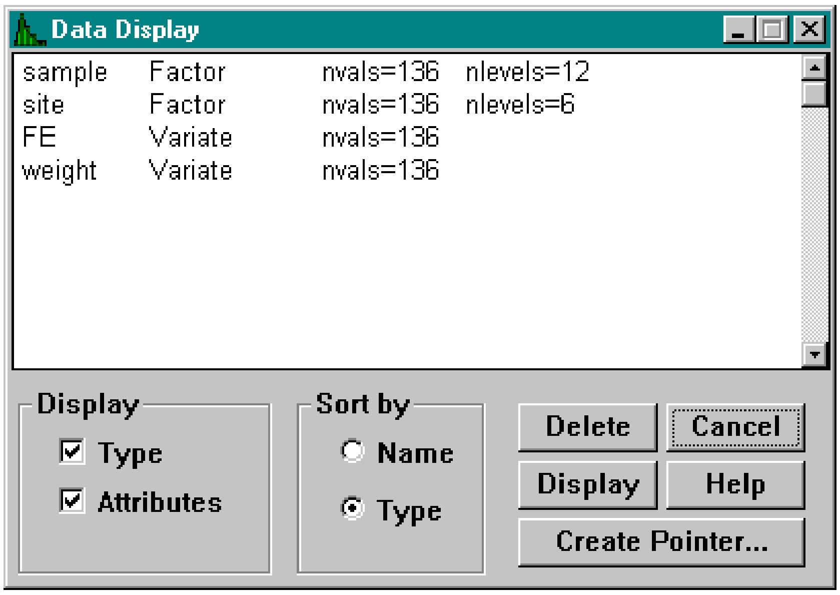
You have two options:

1. use *Data > Load > Data file* to load the data from a file, perhaps an ASCII file or a Genstat or Excel spreadsheet;
2. use *Spread > New* to create a blank Genstat spreadsheet and enter the data into it.

Note that Genstat spreadsheets differ from specialist spreadsheet systems like Excel:

1. It is not the central storage mechanism for Genstat — it is primarily a data entry tool with data for analysis being transferred to the server and stored within it. The transfer is automatic when you switch from the spreadsheet window to another window.
2. Calculations are not done within the spreadsheet — they are done on the data stored internally within the server.

For example Genstat comes with some data in spreadsheet files in \GEN541\COURSE\EXAMPLES. We will load IRON.GSH and use *Data > Display* to find out what is stored in the spreadsheet file. The following screen shows that it contains four sets of information or **data structures**:



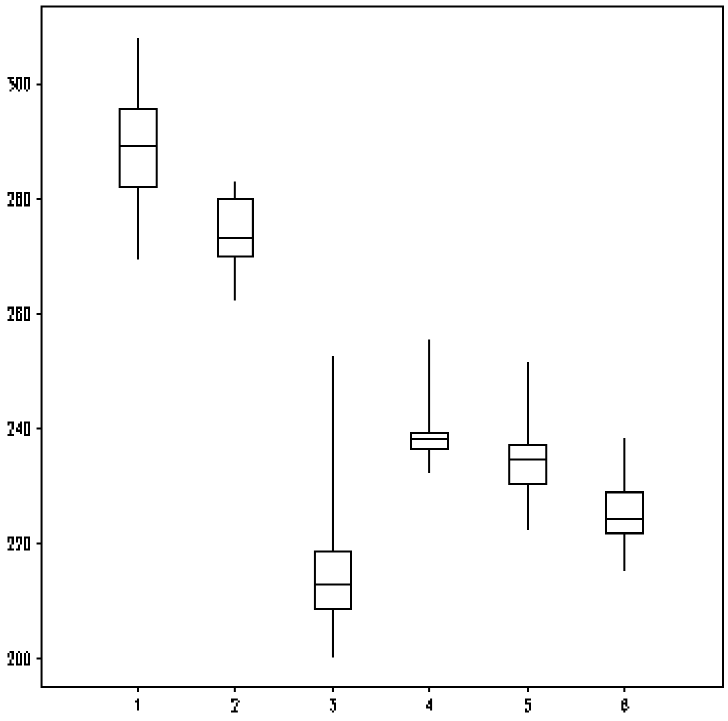
```
sample   Factor    nvals=136  nlevels=12
site     Factor    nvals=136  nlevels=6
FE       Variate   nvals=136
weight   Variate   nvals=136
```

Display: [✓ Type] [✓ Attributes]

Sort by: Name [✓ Type]

The data is from 136 soil samples. The first two data structures give for each sample, the sample (a number 1–12) and the site (a number 1–6) numbers. This kind of categorical information Genstat stores in **factors**. The last two data structures contain the amount of iron (in ppm) and the weight of each sample. This kind of measurement information is stored in **variates**.

We might want to do a boxplot of FE by site. To do this Use *Graphics > Boxplot* and we get the plot given below. This picture can be saved as a Windows bitmap (*bmp* file) using *File > Save As* in the graphics window and then imported into Word using *Insert > Picture > From File*.



Entering commands directly

Actually when Genstat does something like the boxplot it creates a command to send to the Server to do this. If you look in the output window you will see something like:

```
44  BOXPLOT [ SCREEN=clear; METHOD=boxandwhisker; BOXWIDTH=fixed] FE; site
```

This is the command that Genstat generated and you could enter this command yourself. How? Well you need to use *File > New > Text Window* to open an *Input window*.

Then you would type this command into the Input window. To get Genstat to execute the command use *Run > Submit ...* (Line, Selection or Window).

I prefer to use this method as I can save the commands in the input window in a *gen* file as documentation of what I did and for rerunning. Also I usually copy and paste output to Word where I assign it the Courier New font.

For example here is a GENSTAT 5 program to enter temperature and thrust measurements for a rocket engine, produce a scatter diagram and compute and display the MnThrust.

GENSTAT 5 Program

```
"Rocket Performance data"

VARIATE [VALUES=19,15,35,52,35,33,30,57,49,26] Temp

&
[VALUES=1.2,1.5,1.5,3.3,2.5,2.1,2.5,3.2,2.8,1.5] \
                          Thrust

TEXT [VALUES='Rocket Performance'] T

&    [VALUES='Temperature'] Xt

GRAPH [TITLE=T; XTITLE=Xt] Thrust; Temp

SCALAR MnThrust

CALCULATE MnThrust=MEAN(Thrust)

PRINT MnThrust; FIELDWIDTH=12; DECIMALS=2
```

I have saved this file in *Rocket.gen*, which can be loaded and run.

The Genstat language

COMMANDS

Genstat’s language is command-based (not function-based)

Command

$$\boxed{\text{PRINT}\ \ \text{Thrust}}$$

$$\underset{\substack{\uparrow\\ \text{directive name}\\ =\text{verb}}}{\text{PRINT}}\quad \underset{\substack{\uparrow\\ \text{primary parameter}\\ =\text{subject}}}{\text{Thrust}}$$

Commands operate on data structures;
usually, they can work on several

$$\text{PRINT}\ \underset{\text{list}}{\boxed{\text{Thrust, Temp}}}$$

Commands can be modified if necessary

$$\text{PRINT}\ \underset{\substack{\uparrow\\ \text{option}\\ =\text{adverb}}}{\text{[SERIAL=yes]}}\ \text{Thrust, Temp}$$

OPTIONS & PARAMETERS

More than one modification may be made

PRINT [SERIAL = yes; CHANNEL = 2] Thrust

$\uparrow$

option sequence

Some modifications can be made differently
for each subject of the command
parameter sequence

PRINT [SERIAL = yes] Thrust, Temp; DECIMALS = 2,0

$\uparrow$

auxiliary parameter
= adverb

RULE

Settings of auxiliary parameters are matched in parallel with those of the primary parameter

SYSTEM WORDS

Directive names, Option names, Parameter names, and function names form the Vocabulary

Most are English-language words

PRINT SERIAL DECIMALS CHANNEL

Some have prefixes for different applications of the same idea

$$\begin{tikzcd} \text{DISPLAY} & \text{RDISPLAY} & \text{ADISPLAY} \\ & \text{Regression} \arrow[u] & \text{Analysis-of-variance} \arrow[u] \end{tikzcd}$$

Some are acronyms

ANOVA PCP

RULES

1. A word is used for one idea consistently

2. Options & parameter with the same name have the same type of setting

3. All options with common settings have the same default
   ('no' is always the default for yes/no alternatives)

ITEMS

Numbers

```
VARIATE [VALUES=10.13, 8.69, 7.91, 9.41, 9.91] Thrust
```

Strings

```
TEXT [VALUES = ’Maximum wheat yield’] Title
                               ↑
                         quoted string


READ [PRINT = data, summary] Thrust
                       ↑
                unquoted strings
```

RULES

1. Can always quote strings, except in fixed-format read
2. Never need to quote system-defined strings
3. Do not have to quote strings in list of values for a text structure, if letter comes first & all alphanumeric
4. To put quotes in a string, double them up

Identifiers

The name given to a particular data structure in Genstat

PRINT $\boxed{\text{\%yield_i}}$n_tonnes, Temp, YEAR[1981]

$\uparrow$

8 characters stored

RULES

1. Letter first (including %_) then alphanumeric
2. Case is significant, and only first 8 characters significant
3. Suffix in square brackets

--> NO SYSTEM-RESERVED WORDS

Operators in expressions

simple

+   -   *   /   =   <   >

compound

**   *+

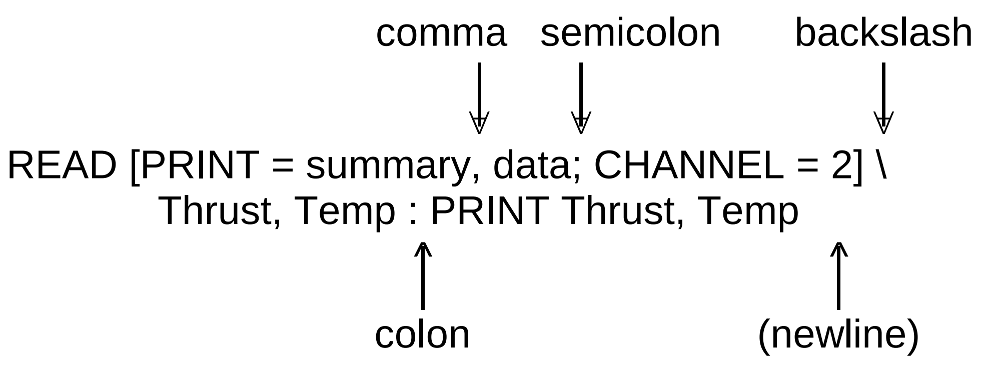
.EQ.   .NE.   .LE.   .LT.   .GT.   .GE.   .IS.   .ISNT.

.AND.   .OR.   .EOR.   .NOT.   ==   /=   <>   <=   >=

.EQS.   .NES.   .IN.   .NI.

PUNCTUATION

Commands use spoken language-style punctuation, with end-of-line meaning end-of-command if no continuation character is given.



```
READ [PRINT = summary, data; CHANNEL = 2] \
        Thrust, Temp : PRINT Thrust, Temp
```

comma	separates items
semicolon	separates lists
backslash	continues command onto next line
colon	indicates the end of command
newline	(end of line) indicates the end of command

RULES

1. The repetition symbol (&) repeats directive name and options;

2. no colon is necessary before &

3. All options whose setting is one or more system-defined strings can be set to * to mean "no strings"

```
READ [PRINT = *] Thrust & Temp
```

LISTS OF ITEMS

Items separated by commas form a list

Elements of a list can be repeated
using premultipliers and postmultipliers

2(A,B,C) is the same as A, A, B, B, C, C
↑
Pre-multiplier

('a','b')2 is the same as 'a', 'b', 'a', 'b'
↑
Post-multiplier

an arithmetic progression can be compacted
using an ellipsis

1980,1985...2000   means   1980, 1985, 1990, 1995, 2000
↑ ↑
the difference between these two is the increment

2(1...3)3   means   1,1,2,2,3,3,1,1,2,2,3,3,1,1,2,2,3,3
↑
no second number is taken as 1
(or -1 for a decreasing progression)

DATA STRUCTURES

Structures of various types to contain data

We will use the following types:

FACTOR	-	a series of values which takes only a limited set of values, called levels.
MATRIX	-	a rectangular array of numbers.
SCALAR	-	a single number
TABLE	-	a multidimensional array of numbers, each dimension classified by a factor.
TEXT	-	a series of strings, each one representing a line of textual information.
VARIATE	-	a series of numbers.

Unnamed structures

Unamed structures can be used to avoid having to declare names for single-use structures

$$\text{PRINT } \underset{\text{unnamed variate}}{\underset{\uparrow}{\boxed{\text{!V(1980...1985)}}}}\text{,Salary}$$

RULES

1. An unnamed structure is preceded by an exclamation mark, followed by the type code, and then the contents contained in round brackets.

2. The type code will usually be T or V for text or variate, respectively; if no code is given, variate is assumed by default.

ABBREVIATION

RULES

1. All system words and system-defined strings can be abbreviated to four characters

2. Further abbreviation is possible, if you want it

```
READ [PRINT = data, summary; SERIAL = yes; \
                SETNVALUES = yes] Thrust

--> READ [P = d, s; S = y; SET = y] Thrust
                               ↑
                     to distinguish from
                  preceeding SERIAL option

--> READ [d,s; ; y; y] Thrust
               ↑
   CHANNEL option not set
```

3. If a directive has a PRINT option, it is first

4. Parameter names can be abbreviated similarly

SHORTENED PROGRAM

```
VARI [VAL=19,15,35,52,35,33,30,57,49,26] Temp

&    [V=1.2,1.5,1.5,3.3,2.5,2.1,2.5,3.2,2.8,1.5] \
                           Thrust

GRAPH [TIT='Rocket Performance'; XTIT='Temperature'] \
                           Thrust; Temp

CALC MnThrust=MEAN(Thrust)

PRINT MnThrust
```

Example II.1 House price (continued)

For this example, involving the response variable Price and the explanatory variables Age and Area, the data was as follows:

Price \$000 (y)	Age years (x_1)	Area 000 feet^2 (x_2)
50	1	1
40	5	1
52	5	2
47	10	2
65	20	3

One might first enter this data into a Genstat spreadsheet. I have already done this and it is called *HousePrice.gsh*.

A Genstat program to obtain the regression analysis is as follows:

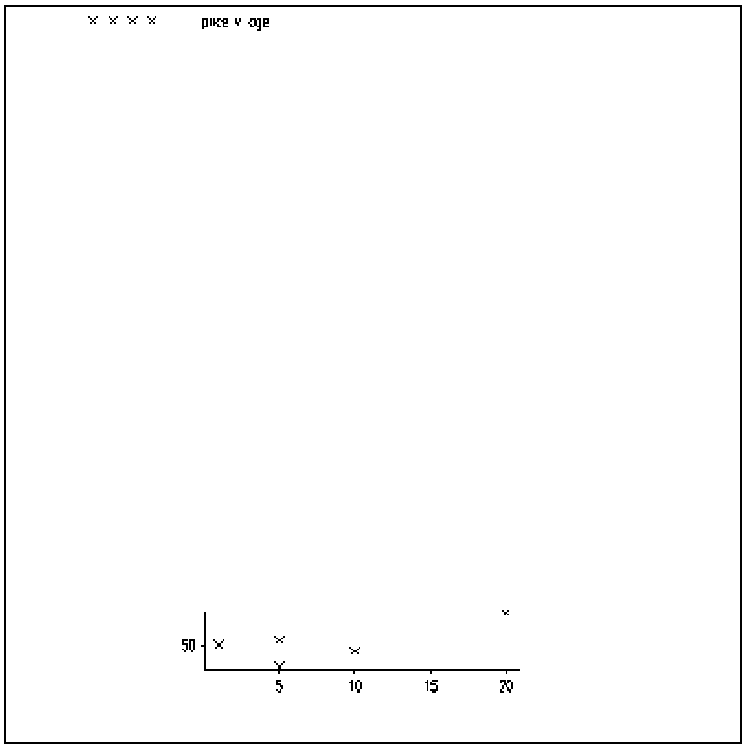
```
"Load data from HousePrice.gsh"

model price; res=res; fit=fit
terms age,years
fit [fprob=y] age, years
dgraph [keywindow=1; window=2] price; age
&      [window=3] price; years
print price,fit,res,age,years; dec=0,1,4,0,0
rkeep inverse=inv
print inv
```

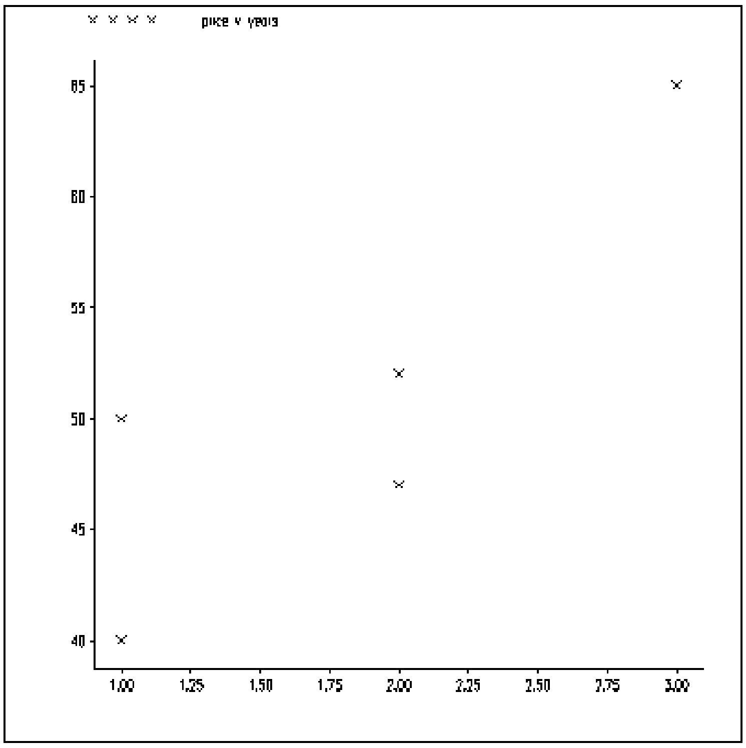
Note:

- use of & to repeat a command
- use of rkeep to extract results from previous regression, in this case $(\mathbf{X}'\mathbf{X})^{-1}$.

Genstat output

```
Genstat 5  Release 4.1  (PC/Windows NT)               17 March 2000 05:17:19
Copyright 1998, Lawes Agricultural Trust (Rothamsted Experimental Station)

                 ______________________________________________

                 Genstat 5 Fourth Edition - (for Windows)
                 Genstat 5 Procedure Library Release PL11
                 ______________________________________________

   3  "Data taken from File: D:/ANALYSES/LM/HOUSEPRICE.GSH"
   4  DELETE [redefine=yes] age,price,years
   5  VARIATE [nvalues=5] age
   6  READ age

    Identifier   Minimum      Mean   Maximum    Values   Missing
           age     1.000     8.200    20.000         5         0

   8  VARIATE [nvalues=5] price
   9  READ price

    Identifier   Minimum      Mean   Maximum    Values   Missing
         price     40.00     50.80     65.00         5         0

  11  VARIATE [nvalues=5] years
  12  READ years

    Identifier   Minimum      Mean   Maximum    Values   Missing
         years     1.000     1.800     3.000         5         0

  14
  15  "Load data from HousePrice.gsh"
  16
  17  graph [nrows=20; ncol=63] price; age


         -
         I                                                  *
    64.0 I
         I
         I
         I
         I
         I
    56.0 I
         I
         I
         I             *
         I   *
         I
    48.0 I
         I                         *
         I
         I
         I
         I
    40.0 I             *
         -+---------+---------+---------+---------+---------+---------+---
        0.0       4.0       8.0      12.0      16.0      20.0      24.0

                         price  v. age using symbol *
  18  &                        price; years


         -
         I                                                        *
    64.0 I
         I
         I
         I
         I
         I
    56.0 I
         I
         I
         I                         *
         I     *
         I
    48.0 I
         I                         *
         I
         I
         I
         I
    40.0 I     *
         -+---------+---------+---------+---------+---------+---------+---
         0.8       1.2       1.6       2.0       2.4       2.8       3.2

                       price  v. years using symbol *

  19  dgraph [keywindow=0; window=1] price; age
  20  &       [window=2] price; years
  21  model price; res=res; fit=fit
  22  terms age,years
  23  fit [fprob=y] age, years


23.............................................................


***** Regression Analysis *****

 Response variate: price
     Fitted terms: Constant, age, years


*** Summary of analysis ***

              d.f.         s.s.         m.s.      v.r.  F pr.
Regression       2       239.12       119.56      2.50  0.286
Residual         2        95.68        47.84
Total            4       334.80        83.70

Percentage variance accounted for 42.8
Standard error of observations is estimated to be 6.92


*** Estimates of parameters ***

                  estimate         s.e.      t(2)
Constant              33.1         10.5      3.15
age                  -0.19         1.11     -0.17
years                10.72         9.73      1.10

  24  print price,fit,res,age,years; dec=0,1,4,0,0

       price         fit          res          age        years
          50        43.6       1.2466            1            1
          40        42.8      -0.7276            5            1
          52        53.6      -0.5074            5            2
          47        52.6      -0.9145           10            2
          65        61.4       1.3306           20            3
 25  rkeep inverse=inv
 26  print inv

                       inv

   Constant          2.308
        age          0.157        0.026
      years         -1.884       -0.204        1.978

                  Constant          age        years
```





VARIATES VERSUS TEXTS VERSUS FACTORS

Consider the following set of data that is a set of counts from a small experiment: the numbers of one type bacteria found in samples from two crops.

Numbers	Crop
18	pea
117	pea
21	cereal
7	pea
176	cereal
85	cereal
244	cereal
4	pea
55	cereal
8	pea

Notice that the numbers of bacteria are positive integers and that the types of crop are the two words pea and cereal. An issue to be considered is what sort of data structure do we want to end up with — variates, texts, factors? The numbers of bacteria is straightforward — they would go into a variate as they are arbitrary numbers. However, the type of crop is different as it has only two values and they are words. We could:

1. decide on some codes — 1 = pea and 2 = cereal — and enter these into a variate;
2. enter the words into a text variate; or
3. enter either codes or words into a factor.

Factor allows one to specify a limited set of values that the variable can take and is treated specially in analyses by Genstat. It is most likely that the type of crop would be entered into a factor.

Factor data structures

A factor is a vector that has only a limited set of possible values; the attributes of a factor, as used in Genstat, are:

- the number of possible values (= the number of levels)
- the numeric values that are associated with the different possible values (= the levels)
- the textual values that are associated with the different possible values (= the labels)
- the number of elements in the factor's vector (= the number of values)
- the particular values that each element of the vector takes (= the levels taken by the elements = the values)

In Genstat, factors are declared using the factor statement or are created using the spreadsheet commands. Not all properties of a factor have to be declared. The minimum properties that can be declared for a factor are its name and its number of levels. In this case Genstat will assume that the levels of the factor are the integers from 1 up to the number of levels. The number of values in the factor's vector will be assumed to be equal to the number of rows in the spreadsheet or the number of units declared in a previous UNITS statement. However, you can change the values of the levels from the default 1 to the number of levels to being some other set of numeric values. In addition, or alternatively, you can set labels that are text values describing the levels of the factor.

Once you have set up a factor, you will need to set the factor values. The factor values can be set by entering them in manually into a spreadsheet, setting them in a FACTOR statement or by using *Stats > Design > Generate Factors in Standard Order* or the *Spread > Column > Fill* menu commands. The menu commands save a lot of keystrokes but are not as simple as just entering each value, either manually or in a factor statement.

If you are entering the factor values manually into a spreadsheet, you will have to enter the labels (or enough letters so that a unique label can be assigned) if these have been set. If labels have not been set, you enter the levels values — either the values that you have set or, if you didn't set them, the default integers from 1 to the number of levels.

The GENERATE statement or the *Stats > Design > Generate Factors in Standard Order* command generates factors in standard order by cycling the levels if the factors in the defined order; the values of the last factor in the list are cycled fastest, the second last factor in the list next fastest and so on so that the first factor in the list is cycled slowest.

Using Generate or *Stats > Design > Generate Factors in Standard Order* to set factor values

We generate 3 factors of which A has 2 levels with the default values of 1 and 2, B has 3 levels set to the values 4, 1 and 2 and C has 4 levels with the default values of 1–4.

```
FACTOR [NVALUES=24; LEVELS=2] A
&      [            LEVELS=!(4,1,2)] B
&      [            LEVELS=4] C
GENERATE A,B,C
```

gives A, B and C the values

```
A: 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2
B: 4 4 4 4 1 1 1 1 2 2 2 2 4 4 4 4 1 1 1 1 2 2 2 2
C: 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4
```

Placing a number or scalar in the parameter list has the same effect as if a factor with the same number of levels had been listed. Thus to generate values only for A and C can be done using the following statement:

```
    GENERATE A,3,C
```

With *Stats > Design > Generate Factors in Standard Order* have to include a dummy factor that you delete afterwards.

Note that the number of values for the factors must be the same for all factors in the list and must be such that all cycles of all factors are complete. This would not happen if C had only 3 levels as complete cycles would be obtained in 18 values and only one-and-one-third of a cycle by 24 values.

The following statement sets up the factor Gender with 2 levels whose values are the default values of 1 and 2 and with levels' labels of male and female.

```
    FACTOR [LAB=!T(male,female)] Gender; !T(4(male,female))
```

Finally, the following statement sets up the factor Rate with 3 levels whose values are 0, 2.5 and 5 and with levels' labels.

```
    FACTOR [LEV=!(0,2.5,5); LAB=!T(none,standard,double)] Rate; \
              VALUES=!T(none,double,standard,double,none,standard)
```

SUMMARY OF RELEVANT COMMANDS

Commands in Genstat are of two types: **directives** which are the basic commands that have been programmed using a computer programming language, actually Fortran; **procedures** are the commands that have been programmed in the Genstat language.

Data structure commands

Data structures store the information on which a Genstat program operates. Structures, can be defined, or declared, by a Genstat statement known as a declaration. The directive for declaring each type of structure has the same name as the structure, for example SCALAR to declare a scalar (or single-valued numerical structure). and so on. These are some of the directives, with details of their corresponding data structures:

FACTOR	series of group allocations (using a predefined set of numbers or strings to indicate the groups).
MATRIX	rectangular matrix.
SCALAR	single number.
TABLE	table (to store tabular summaries like means, totals, etc.).
TEXT	series of character strings or lines of text.
VARIATE	series of numbers.

Control structure commands

JOB	Starts a Genstat job.
STOP	Ends a Genstat program.

Input and output commands

CLOSE	Closes files, freeing the channels to which they are attached.
OPEN	Opens files and connects them to Genstat input/output channels.
PRINT	Prints data in tabular format in an output file, unformatted file or text.
READ	Reads data from an input file, an unformatted file or a text.
FILEREAD	Reads data from a file, assumed to be in a rectangular array.

Graphics commands

BARCHART	Plots a bar chart.
BOXPLOT	Draws box-and-whisker diagrams (schematic plots).
DBARCHART	Plots barcharts for one or two-way tables.
DGRAPH	Produces scatter plots and line graphs.
DHISTOGRAM	Plots histograms.
DOTPLOT	Produces a dot-plot.
DSCATTER	Produces a scatter-plot matrix.
DSHADE	Produces a pictorial representation of a data matrix.
GRAPH	Produces scatter and line graphs.

RUGPLOT	Draws "rugplots" to display the distribution of one or more samples.
STEM	Produces a simple stem-and-leaf chart.

Backing store

CATALOGUE	Displays the contents of a backing-store file.
RETRIEVE	Retrieves data structures from a backing-store file.

Calculation and Manipulation

CALCULATE	Perform arithmetical and logical calculations.
DELETE	Deletes the attributes and values of data structures.
EQUATE	Transfers data between structures of different sizes or types (but the same modes i.e. numerical or text) or where transfer is not from single structure to single structure.
GENERATE	Generates values of factors in systematic order or as defined by a design key, or forms values of pseudo-factors.
RANDOMIZE	Randomizes the elements of a factor or variate.
RESTRICT	Defines a restricted set of units of vectors for subsequent statements.
TABULATE	Forms summary tables from the values of variate.
UNITS	Defines the default length or labelling for vectors defined subsequently in the job.

Regression analysis

ADD	Adds extra terms to a linear, generalized linear or nonlinear model.
DROP	Drops terms from a linear, generalized linear or nonlinear model.
FIT	Fits a linear or generalized linear regression model.
MODEL	Defines the response variate(s) and the type of model to be fitted for linear, generalized linear and nonlinear regression models.
PREDICT	Forms predictions from a linear or generalized linear model.
RDISPLAY	Displays the fit of a linear, generalized linear or nonlinear model.
TERMS	Specifies a maximal model, containing all terms to be used in subsequent linear, generalized linear and nonlinear models.

Design and analysis of balanced experiments

Analyzing

A2PLOT	Plots effects from two-level designs with robust s.e. estimates.
ADISPLAY	Displays further output from analyses produced by ANOVA.
AGRAPH	Plots one- or two-way tables of means from ANOVA.
AKEEP	Copies information from an ANOVA analysis into Genstat data structures.
ANOVA	Analyses y-variates by analysis of variance according to the model defined by earlier BLOCKSTRUCTURE, COVARIATE and TREATMENTSTRUCTURE directives.
APLOT	Plots residuals from an ANOVA analysis.

APOLYNOMIAL	forms the equation for a polynomial contrast fitted by ANOVA
BLOCKSTRUCTURE	Defines the blocking structure of the design.
COVARIATE	Specifies covariates for use in subsequent ANOVA directives.
DAPLOT	Plots residuals from ANOVA in high-resolution, with interactive identification of outliers.
NLCONTRASTS	Fits non-linear contrasts to quantitative factors in ANOVA.
TREATMENTSTRUCTURE	Specifies the treatment terms to be fitted by subsequent ANOVA directives.
ASTATUS	Provides information about the settings of ANOVA models and variates.

Designing

GENERATE	Generates values of factors in systematic order or as defined by a design key, or forms values of pseudo-factors.
DESIGN	Acts as a menu-driven interface to the Genstat design system, providing a convenient way of selecting and generating various types of factorial designs.
AFORMS	Prints data forms for an experimental design.
ARANDOMIZE	Randomizes and prints an experimental design.
DDESIGN	Plots the plan of an experimental design.
FACPRODUCT	Forms a factor with a level for every combination of other factors.
PDESIGN	Prints or stores treatment combinations tabulated by the block factors.
RANDOMIZE	Puts units of vectors into random order, or randomizes units of an experimental design.
REPLICATION	Calculates the replication necessary to detect a treatment effect.