NAG®

GENSTAT

Newsletter

Issue No. 28

Editors

P W Lane
AFRC Institute of Arable Crops Research
Rothamsted Experimental Station
HARPENDEN
Hertfordshire
United Kingdom AL5 2JQ

G W Morgan
NAG Ltd
Wilkinson House
Jordan Hill Road
OXFORD
United Kingdom OX2 8DR

Printed and Produced by NAG®
NAG is a registered trademark of:
The Numerical Algorithms Group Ltd
The Numerical Algorithms Group Inc
The Numerical Algorithms Group (Deutschland) GmbH
GLIM is a trademark of the Royal Statistical Society

ISSN 0269-0764

Please note that the cover of this Newsletter has been adapted by kind permission of Oxford University Press, from the cover of the Genstat 5 Reference Manual.

Genstat Newsletter

Issue No. 28

NP2369 November 1992

Contents

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

Editorial

This edition of the Newsletter contains a number of articles arising from the Seventh Genstat Conference at Papendal. First there is a report on the many activities that took place during the conference. This is followed by an exposition of different ways of using the **FITNONLINEAR** directive. Thefirst article from the conference is about the application of **FITNONLINEAR** to piecewise linear regression. There is then an introduction to the new facilities in Release 3 for fitting smoothing splines, providing the analysis of generalized additive models. Two new procedures, now available in Library 2[3], are described next: the first carries out calculations for modelling the development of temperature-dependent processes, and the second tests fixed effects in an unbalanced mixed model. The topic of smoothing is continued in two more articles, applied in the problem of fitting mixtures of Normal distributions and in the diagrammatic representation of two-dimensional data. Finally, there is an article giving advice on using Genstat on a PC.

While you now have a chance to examine in detail some of the new ideas given at the last Genstat Conference, we would like to remind you that the Second Australasian Genstat Conference is about to take place in Rotorua during 14–15 December and that the Eighth Genstat Conference, which will take place in Canterbury, England, 19–23 July 1993, is now being planned. Further details are enclosed with this Newsletter.

Erratum: Procedure for conditional logistic analysis, CONDML

There is an error in the procedure CONDML on Page 9 of Newsletter 26, in the article by John Thompson on "Conditional Logistic Regression in Genstat". The 12th line of the procedure (the contents of the second **FOR** loop) should read

```
CALCULATE i = i + 1 : & j = CF$[i] : & a = a + j*X[i]
```

In the printed version, the last **i** was omitted: this causes no problem if the parameter **NP** is 1, but the procedure fails if **NP** is greater than 1.

The editors apologize for this error, introduced during publishing.

Seventh Genstat Conference, Papendal (Arnhem) 23–27 September 1991

J Withagen
Centre for Agrobiological Research (CABO-DLO)
Wageningen
The Netherlands

The papers and presentations shown at the Seventh Genstat Conference once more demonstrated the flexibility of Genstat. The wide, and still increasing, range of possible analyses in Genstat and the ability to add new statistical analyses using the procedure facilities or by linking Fortran programs to it, makes Genstat a very powerful system. From the constantly growing number of procedures in the Genstat Procedure Library we learn that many users work with these facilities and build procedures for use in special areas of work.

At the Conference, new facilities in Release 3 were described and demonstrated. New directives were added and several directives have extra options or parameters providing new interfaces to more statistics. Also the user interface is improved. Roger Payne told us that Release 3 will have a graphical user interface to any Genstat command, and there will also be a 'spreadsheet' for entering and editing data. Especially for interactive users, this will be a welcome improvement of the user interface.

Many talks and posters were about theory and application of statistical techniques from different areas of work. They concerned mainly regression analysis, generalized linear models, the use of **REML**, and nonlinear regression. Also, many new procedures were described for a wide range of purposes such as fitting special models, analysis of repeated measurement experiments, least-median-squares regression and visualizing three-dimensional biplots.

A very interesting extension to Genstat's regression section was described by Trevor Hastie from AT&T Bell Labs: "Generalized Additive Models" (GAMs) can be fitted using the **MODEL** and **FIT** directives. A statement of the form

```
FIT SMOOTH(variate; df)
```

fits smooth curves from linear (df=1) to a rather flexible curve (df=number of unique values of the variate). (Ed: the name **SMOOTH** has been replaced by **SSPLINE** in Release 3.1, or S for short.) The current state of the implementation in Release 3 was demonstrated later by Peter Lane. I believe this new facility will prove to be a very useful extension to what Genstat already has to offer. Another extension to the regression section in Release 3 will be the possibility to fit an ordinal response model with the standard directives; up to Release 2 this can be done only using the procedure **ORDINALLOGISTIC**.

The subject of teaching Genstat was again one of the topics of the conference. Gillian Arnold of the Long Ashton Research Station described how they use the interactive package Instat for teaching basic statistics. Later on, a conversion course from Instat to Genstat is offered to the students who want this. She talked about the experiences in teaching this course for three years. Jan Oude Voshaar of the Agricultural Mathematics Group in Wageningen described how he uses Genstat for teaching statistics to Dutch agricultural researchers in post-academic courses.

On Tuesday we had a special session. Since we were staying at the National Sport Centre Papendal, Mr J A Vos, connected to this centre, gave a lecture on Statistics and Sport. After dinner we were invited to go to the Sports Hall to do some training with or without supervision. Some of us even came away with a graph to prove how hard we had worked.

On Wednesday afternoon a coach took us to the National Park 'De Hoge Veluwe' where we could choose from several activities such as visiting the Kroeller–Mueller Museum with its art gallery (Vincent van Gogh and others), the sculpture garden and special expositions, or a cycling tour in the nature reserve. A number of participants, myself included, chose the cycling tour; the weather, however, did not cooperate and we got wet through. Looking at the grasses and trees, I do believe they welcomed the rain a lot more than we did. But back at the museum restaurant our cycling tour made the coffee taste even better.

The conference dinner on Thursday night was on a boat going from Arnhem to Nijmegen on the rivers Rhine and Waal. While enjoying the meal, we had good views of the city of Nijmegen.

For their excellent local organization, thanks are due to Paul Goedhart of the Agricultural Mathematics group (GLW-DLO) and Valentijn van den Berg of the Institute of Agricultural Engineering (IMAG-DLO) at Wageningen.

Using the FITNONLINEAR Directive

Ruth Butler
Department of Agricultural Sciences
University of Bristol
AFRC Institute of Arable Crops Research
Long Ashton Research Station
Bristol
United Kingdom BS18 9AF

1. Introduction

FITNONLINEAR is a very powerful command which can be used for a wide range of applications. This great flexibility can make the command difficult to use in that there are several ways in which the **FITNONLINEAR** directive and associated **MODEL**, **RCYCLE** and **EXPRESSION** directives can be set up; it is not always easy to see exactly how to produce the required analysis. There are other problems in using **FITNONLINEAR**: in nearly all cases, good initial parameter values must be found because otherwise the iterative process may fail. Some solutions to these problems are discussed below.

There are three main uses for **FITNONLINEAR**: function minimization, fitting of empirical distributions to frequency data, and the fitting of nonlinear models. Several types of nonlinear model can be fitted: those which include linear parameters, and those which do not; and also those for which linear or nonlinear parameters vary with the levels of a factor. The type of model that is fitted depends on the settings of the parameters and options in the **FITNONLINEAR** and **MODEL** directives as well as the specification of the **EXPRESSION** to be used. Each type of analysis has a standard format for the directives, and once it is understood which format is needed **FITNONLINEAR** becomes very much easier to use. These basic formats are outlined below, with examples to illustrate their use. All references are to the Genstat 5 Reference Manual unless otherwise stated.

2. Function minimization

The general format is as follows (reference: Section 8.6.4, p385):

```
EXPRESSION e; VALUE=!e(f = F(theta))
MODEL [FUNCTION=f]
RCYCLE theta; INITIAL=??
FITNONLINEAR [CALC=e]
```

A scalar value **f**, which is the result of calculating function **F** with parameters theta, can be minimized with respect to theta. The function, defined with an **EXPRESSION** directive, is passed to **FITNONLINEAR** using its **CALCULATION** option, and the parameters theta are indicated using the **RCYCLE** directive. In this case there is no response variate, so the **MODEL** statement has no parameter; instead Genstat is informed of the name of the scalar to be minimized using the **FUNCTION** option of **MODEL**. The result of the function must be a scalar, but the function itself may include variates or factors as well as the scalar parameters to be estimated.

For example, the following statements find the value of x that minimizes $f = x^2$.

```
EXPRESSION e; VALUE=e!(f = x**2)
MODEL [FUNCTION=f]
RCYCLE x; INITIAL=0.5
FITNONLINEAR [CALC=e]

***** Results of optimization *****

*** Minimum function value: ***

  0.10381E-20


*** Estimates of parameters ***
                          sq. root of
                 estimate   2nd derivs
x                   0.000        1.000
```

Note: The estimated minimum value of the function f is not exactly 0, because the process has converged within the tolerances of the convergence process (p358).

The manual gives another example (8.6.4a on p385), and more information is given in the article by Lane (1991).

3. Fitting empirical distributions

The general format is as follows (reference: 8.6.3, p384):

```
EXPRESSION e; VALUE=!e(c = DIFF(f(x; theta))*num)
MODEL [DISTRIBUTION=multinomial] count
RCYCLE theta; INITIAL=??
FITNONLINEAR [CALC=e]
```

There are many experiments where the aim is to estimate the distribution (with parameters theta) of a continuous variable. For example, the length of 100 seeds may be found, and the distribution of these lengths estimated. Instead of recording each individual length, the numbers of seeds falling into each of several length intervals (eg 0mm to 1mm, 1mm to 2mm etc) should be recorded (individual lengths could be tabulated to give a variate of counts). These counts are used as the response variate to estimate the parameters of the distribution of the lengths. The variate x contains the interval limits in ascending order, and an expression is set up using x to define the cumulative distribution. This distribution must then be differenced to give the predicted proportion of seeds falling in the intervals defined by x, and then multiplied by the total number (num) of seeds to give the predicted counts.

It is sometimes necessary to make sure that the last count is 0 (to allow proper definition of the distribution) so the last interval of x must reflect this. Also, since the **DIFFERENCE** function is used in the expression, the first fitted value will be missing; thus the first value of x is really the lower value of the first interval, so the count for this should be a missing value.

The **DISTRIBUTION** option of **MODEL** is set to 'multinomial' to indicate that a distribution is being fitted; it does not indicate that the distribution of the counts is multinomial! There is no parameter for the **FITNONLINEAR** statement, because the calculated variate **f** is to be fitted directly to the counts. More information about this type of distribution fitting can be found in Brain and Butler (1988); see also Butler and Brain (1991).

For example, say x is the length of a seed in mm, the distribution to be fitted is Normal, and the data are as follows:

length	0	<1	<2	<3	<4	<5	<6	<7	<8	<9	<10	<20	>20
count	*	1	3	4	4	7	8	6	6	4	1	5	0

The counts are for **length** $\geq x_{i-1}$ and **length** $< x_i$; for example, there are four seeds ≥ 3 and < 4mm. The parameters of the distribution to be estimated are **'mean'** and **'SD'**.

```
VARIATE [VALUE=0...10,20,50] length
VARIATE count; VALUE=!(*,1,3,4,4,7,8,6,6,4,1,5,0)
CALC num = SUM(count)
MODEL [DISTRIBUTION=multinomial] count; FITTED=f
EXPRESSION e; VALUE=!e(f = DIFF(NORMAL((length-mean)/SD))*num)
RCYCLE mean,SD; INITIAL=5.5,3
FITNONLINEAR [CALC=e]
```

This gives the following output.

```
***** Nonlinear regression analysis *****

 Response variate: count
     Distribution: Multinomial
```

```
*** Summary of analysis ***
    Dispersion parameter is 1

                                      mean  deviance
             d.f.     deviance     deviance     ratio
Regression      2            *            *
Residual        9        3.190       0.3545
Total          11            *            *


*** Estimates of parameters ***

                 estimate        s.e.
mean                5.699       0.492
SD                  2.999       0.412
*MESSAGE: s.e.s are based on dispersion parameter with value 1
```

Note that only the residual deviance is given, which is a measure of the goodness-of-fit of the distribution, rather than an estimate of random variation or variation left unexplained, as in regression.

The example in the Manual (p384, 8.6.3b) shows an alternative method for fitting an empirical distribution. The new **DISTRIBUTION** directive in Release 3 of Genstat 5 will give a simpler method for fitting distributions.

4. Fitting nonlinear models

Unlike the other two uses of **FITNONLINEAR**, the fitting of nonlinear models is the exact analogue of a linear fitting process, in that there is a response variate Y to be described by an explanatory variate (or variates) X. The difference is that Y is described by a nonlinear function of X so that, as well as the linear parameters (constant A and slope B in our examples below), there are also nonlinear parameters N:

$$Y = A + B.f(X; N)$$

The **MODEL**, **FITNONLINEAR**, **RCYCLE**, and **EXPRESSION** statements needed to fit such a nonlinear model depend on whether the parameters vary with the levels of a factor, and whether the constant and slope parameters (A and B) are to be included.

The basic format is as follows:

```
MODEL Y
EXPRESSION e; VALUE=!e(f = f(X; N))
RCYCLE N; INITIAL=??
FITNONLINEAR [CALC=e] f
```

This fits a basic model where there is no factor involved. It is important that initial values of the parameters are supplied using the **RCYCLE** directive, since the iterative search will usually fail without them. Some methods for choosing initial parameter values are described in Section 3. If the parameter estimates are displayed using the **RDISPLAY** directive or the **PRINT** option of **FITNONLINEAR**, then the constant parameter A will be labelled 'Constant' and the slope parameter B will be labelled by the parameter of **FITNONLINEAR** (**f** in the example).

Below are shown the variations to this basic format needed to allow parameters to vary between the levels of a factor **F** with **nlevF** levels, or to omit either A or B.

4.1. No constant ($A = 0$)

The model is: $y = B.f(x; N)$

```
MODEL Y
EXPRESSION e; VALUE=!e(f = f(X; N))
RCYCLE N; INITIAL=??
FITNONLINEAR [CALC=e; CONSTANT=omit] f
```

If A is to be fixed at a value other than 0, then subtract this value from the data, and fit the model as above.

4.2. Constant A varies with the levels of F

The model is: $y = A_i + B.f(x;N)$

This model can be fitted in two ways: the first method gives the constant for the first level of F, and then the differences between this constant and those for the other levels of F; the second gives each constant.

```
MODEL Y
EXPRESSION e; VALUE=!e(f = f(X; N))
RCYCLE N; INITIAL=??
FITNONLINEAR [CALC=e] F+f
  "this gives A1, A2-A1, A3-A1 etc labelled F 1, F 2, F 3 etc"
```

```
MODEL Y
EXPRESSION e; VALUE=!e(f = f(X; N))
RCYCLE N; INITIAL=??
FITNONLINEAR [CALC=e; CONSTANT=omit] F+f
  "this give parameters A1, A2, A3 etc labelled F 1, F 2, F 3 etc"
```

4.3. No slope (B is a fixed value k, A is omitted)

The model is: $y = k.f(X;N)$

```
MODEL Y; FITTED=f
EXPRESSION e; VALUE=!e(f = k*f(X; N))
RCYCLE N; INITIAL=??
FITNONLINEAR [CALC=e]
```

Note: The parameter of **FITNONLINEAR** is not set because no linear parameters are being estimated (8.6.3, p382).

4.4. No slope (B fixed with value k), A is estimated

The model is: $y = A + k.f(X;N)$

There is no straight-forward method of fitting this model. A needs to be specified in the expression and is treated as a nonlinear parameter. Therefore, this method does not use linear estimation for the constant, which can make estimation slow.

```
MODEL Y; Fitted=f
EXPRESSION e; VALUE=!e(f = A+k.f(X; N))
RCYCLE A,N; INITIAL=??
FITNONLINEAR [CALC=e]
```

Note: The parameter of **FITNONLINEAR** is not set. In Release 3, this model will be able to be fitted by using the **OFFSET** option of **MODEL**.

4.5. Slopes (B) vary with the levels of F

The model is: $Y = A + B_i.f(X;N)$

```
MODEL Y
POINTER [NVALUES=nlevF] f
EXPRESSION e; VALUE=!e(f[] = (F.EQ.1...nlevF)*f(x; N))
RCYCLE N; INITIAL=??
FITNONLINEAR [CALC=e] f[]
```

A dummy variate `f[i]` of 1s and 0s must be set up for each factor level `i`. These are multiplied by the nonlinear function to give a set of dummy variates, effectively giving a separate regression for each level of the factor. In output, the slope parameter for level `i` is labelled using the name of the relevant dummy variate (`f[i]` in this case).

4.6. Nonlinear parameter(s) vary with F

The model is: $Y = A + B.f(X; N_i)$

```
MODEL Y; fitted=f
POINTER [NVALUES=nlevF] N,tempN
EXPRESSION e[1]; VALUE=!e(tempN[] = (F.EQ.1...nlevF)*N[])
&          e[2]; VALUE=!e(sumN = VSUM(tempN))
&          e[3]; VALUE=!e(f = f(x; sumN))
RCYCLE N[]; INITIAL=??
FITNONLINEAR [CALC=e]
```

The structures called `tempN[]` are dummy variates used to form `sumN` which is a variate containing different `N[i]` for each level of the factor, so that the nonlinear function uses a different `N[i]` according to the factor level that is applicable for each unit of the response variate.

4.7. All parameters vary with the levels of F.

The model is: $Y = A_i + B_i.f(X; N_i)$

The best method of fitting this is to restrict `Y` to each level of `F` in turn, and estimate the model separately. This is easily done in a `FOR` loop:

```
MODEL Y
RCYCLE N; INITIAL=??
FOR i=1...nlevF
  RESTRICT Y; F.EQ.i
  FITNONLINEAR [CALC=e] f
ENDFOR
```

If the nonlinear parameters are very different for each level of `F`, then obviously the loop must include the `RCYCLE` statement with the different initial parameter values included as parameters of the FOR statement.

```
MODEL Y
FOR i=1...nlevF; initial=??...??
  RESTRICT Y; F.EQ.i
  RCYCLE N; INITIAL=#initial
  FITNONLINEAR [CALC=e] f
ENDFOR
```

The procedure `FITPARALLEL` in the Genstat Procedure Library fits this model, pooling the results over all factor levels, and also the separate constants, separate slopes and single curve models. The results are combined in order to mimic `FITCURVE` used sequentially to produce a 'parallel curve' analysis, but for non-standard curves.

4.8. Examples

All the examples below fit variants of the following logistic function of log-concentration (Figure 1):

$$y = A + C/(1 + \text{Conc} * \exp(-M)^B)$$

This function is equivalent to using `FITCURVE [CURVE=logistic] logConc`, but the parameterization above allows zero concentrations to be included. This cannot be done with `FITCURVE`, since log(0) is not defined. Also, some of the models fitted below cannot be fitted with `FITCURVE`, even if zero concentration is not included. The variate `Y` stores the dry weight of weed, and `Conc` is the concentration of a herbicide used to control them. The treatment factor `GroTemp` is the temperature at which the plants were grown, with three levels.

```
UNIT [24]
VARIATE [VALUE=3(0,0.01,0.03,0.1,0.3,1,3,10)] Conc
FACTOR [LEVELS=3; LABELS=!t(lo,med,hi); VALUES=(1...3)8] GroTemp
READ [PRINT=data,summary] y
 11.338      12.550      12.279
 10.766      12.804      11.543
 10.725      12.590      11.621
  9.556      10.575      11.695
  4.293       5.357      10.274
```

```
    1.933       2.705       5.590
    1.440       2.466       2.513
    1.481       2.238       1.854 :
```

The following statements fit the model with $A = 0$, allowing C to vary between the levels of F; B and M are common (Figure 2).

```
MODEL y
POINTER [NVALUES=3] f
EXPRESSION e; VALUE=!e(f[] = (GroTemp.EQ.1...3)/(1+(Conc*EXP(-M))**-B))
RCYCLE B,M; INITIAL=-2,-1
FITNONLINEAR [CALC=e; CONSTANT=omit] f[]
```

The output is as follows.

```
***** Nonlinear regression analysis *****

 Response variate: y
     Fitted terms: f[1], f[2], f[3]


*** Summary of analysis ***

             d.f.         s.s.         m.s.      v.r.
Regression      5      1785.13      357.026    203.21
Residual       19        33.38        1.757
Total          24      1818.51       75.771

Percentage variance accounted for 91.3


*** Estimates of parameters ***

                 estimate         s.e.
B                  -0.908        0.153
M                  -0.793        0.226
* Linear
f[1]                10.94
f[2]                12.68
f[3]                13.28
```

The following statement lets A, B and M vary with the levels of F, but C is constant between levels (Figure 3).

```
MODEL y
POINTER [NVALUES=3] M,B,tempM,tempB
EXPRESSION e[1]; VALUE=!e(tempB[],tempM[] = (GroTemp.EQ.1...3)*B[],M[])
&          e[2]; VALUE=!e(sumB,sumM = VSUM(tempB,tempM))
&          e[3]; VALUE=!e(f = 1/(1+(Conc*exp(-sumM))**-sumB))
RCYCLE B[],M[]; INITIAL=3(-0.9,-0.8)
FITNONLINEAR [CALC=e; CONSTANT=omit] GroTemp+f
```

The output is as follows.

```
***** Nonlinear regression analysis *****

 Response variate: y
     Fitted terms: f, GroTemp


*** Summary of analysis ***

             d.f.         s.s.         m.s.      v.r.
Regression     10      1817.400    181.73996   2287.29
Residual       14         1.112      0.07946
Total          24      1818.512     75.77134

Percentage variance accounted for 99.6
```

```
*** Estimates of parameters ***

                 estimate        s.e.
B[1]               -2.018       0.195
B[2]               -2.209       0.211
B[3]               -1.867       0.180
M[1]              -1.5784      0.0634
M[2]              -1.6398      0.0609
M[3]              -0.2773      0.0693
* Linear
f                   9.986
GroTemp lo          1.256
GroTemp med         2.543
GroTemp hi          1.834
```

The following statements fit the model with A and C omitted (fixed at 0 and cw), lets M vary with the levels of `GroTemp` but keeps B constant between levels (Figure 4).

```
SCALAR cw; VALUE=12
MODEL y; fitt=f
POINTER [NVALUES=3] M,tempM
EXPRESSION e[1]; VALUE=!e(tempM[] = (GroTemp.EQ.1...3)*M[])
&          e[2]; VALUE=!e(sumM = VSUM(tempM))
&          e[3]; VALUE=!e(f = cw/(1+(Conc*exp(-sumM))**-B))
RCYCLE B,M[]; INITIAL=-2,-1.6,-1.6,-0.3
FITNONLINEAR [CALC=e; CONSTANT=omit]
```

The output is as follows.

```
***** Nonlinear regression analysis *****

 Response variate: y


*** Summary of analysis ***

             d.f.         s.s.         m.s.      v.r.
Regression      4      1799.79     449.9464    480.54
Residual       20        18.73       0.9363
Total          24      1818.51      75.7713

Percentage variance accounted for 95.4

* MESSAGE: The following units have large standardized residuals:
                  23        2.02


*** Estimates of parameters ***

                 estimate        s.e.
B                  -1.125       0.130
M[1]               -1.465       0.199
M[2]               -1.003       0.200
M[3]                0.044       0.200
```

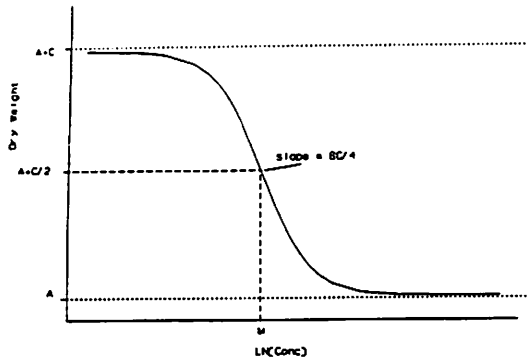


Figure 1

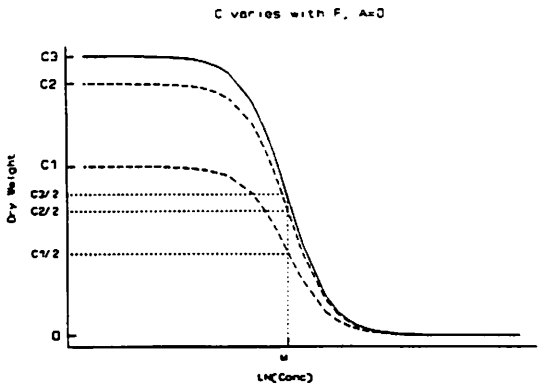


Figure 2

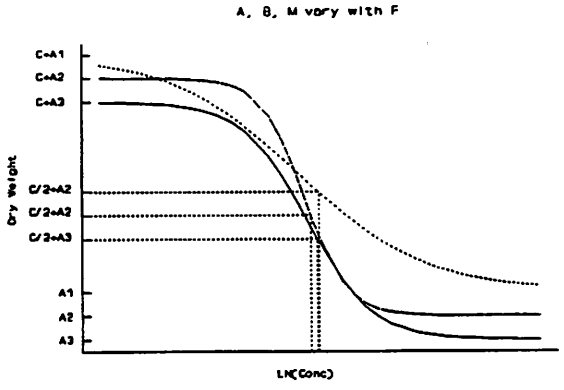


Figure 3

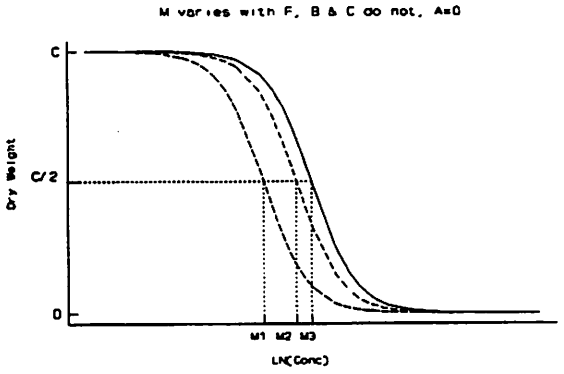


Figure 4

5. Using the RFUNCTION directive

This directive was introduced in Genstat 5, Release 2 and is described in the Reference Manual Supplement 8.4 p64. **RFUNCTION** can be used to calculate functions of the parameters estimated by **FITNONLINEAR** and their standard errors. Parameters that are explicitly mentioned in the expressions can be used in the expression for **RFUNCTION** as they stand, but the constant and slope parameters are more difficult. These can be included in the expression by using the name that appears in the output from **PRINT=estimates**, but encased in single quotes.

The examples below are based on finding the $LD10$, the concentration giving $y = A+0.9C$, for the logistic in the first two examples of 3.7, and also the value of y at the $LD50$ ($y = A + 0.5C$). A third example is given for these two parameters when the model fitted was a logistic with no factor involved and both A and C estimated. Note that the **SELINEAR** option of **FITNONLINEAR** must be set to yes if standard errors are required for functions involving linear parameters, as $Y50$ does.

$$LD10 = \exp(M + \log(9)/B)$$
$$Y50 = A + C/2$$

5.1. A=0, B, M common, C varies with the levels of GroTemp (see first fitted model)

```
   EXPRESSION l10; !e(LD10 = exp(M+log(9)/B))
   RFUNCTION [CALC=l10; PRINT=estimates,se] LD10

***** Estimates of functions of parameters *****

*** Estimates and standard errors ***

                 estimate        s.e.
LD10               0.0402      0.0214

   POINTER [NVALUES=3] Y50
   EXPRESSION y50; VALUE=!e(Y50[] = 'f[1]','f[2]','f[3]'/2)
   RFUNCTION [CALC=y50; PRINT=estimates,se] Y50[]

***** Estimates of functions of parameters *****

*** Estimates and standard errors ***

                 estimate        s.e.
Y50[1]              5.469       0.384
Y50[2]              6.339       0.400
Y50[3]              6.639       0.406
```

5.2. A, B, M vary with the levels of GroTemp, C does not (see second fitted model)

```
   POINTER [NVALUES=3] LD10
   EXPRESSION l10; VALUE=!e(LD10[] = exp(M[]+log(9)/B[]) )
   RFUNCTION [CALC=l10; PRINT=estimates,se] LD10[]

***** Estimates of functions of parameters *****

*** Estimates and standard errors ***

                 estimate        s.e.
LD10[1]           0.06934     0.00878
LD10[2]           0.07171     0.00839
LD10[3]            0.2335      0.0319

   POINTER [NVALUES=3] Y50
   EXPRESSION y50; VALUE=!e(Y50[]='GroTemp lo','GroTemp med','GroTemp hi'+'f'/2)
   RFUNCTION [CALC=y50; PRINT=estimates,se] Y50[]

***** Estimates of functions of parameters *****

*** Estimates and standard errors ***

                 estimate        s.e.
Y50[1]              6.250       0.122
Y50[2]              7.536       0.120
Y50[3]              6.827       0.129
```

5.3. All parameters estimated with none varying with GroTemp

```
   MODEL y
   EXPRESSION e; VALUE=!e(f = 1/(1+(Conc*exp(-M))**-B))
   RCYCLE B,M; INITIAL=-2,-1
   FITNONLINEAR [PRINT=estimates; CALC=e; SELINEAR=yes] f

*** Estimates of parameters ***

                 estimate        s.e.
B                  -1.573       0.491
M                  -1.226       0.213
* Linear
f                  10.032       0.887
Constant            1.915       0.641

   EXPRESSION l10; VALUE=!e(LD10 = EXP(M+log(9)/B))
```

```
   RFUNCTION [CALC=l10; PRINT=estimates,se] LD10

***** Estimates of functions of parameters *****

*** Estimates and standard errors ***

                 estimate        s.e.
LD10               0.0726      0.0332

   EXPRESSION y50; VALUE=!e(Y50 = 'Constant'+'f'/2)
   RFUNCTION [CALC=y50; PRINT=estimates,se] Y50

***** Estimates of functions of parameters *****

*** Estimates and standard errors ***

                 estimate        s.e.
Y50                 6.931       0.368
```

6. Choosing initial parameter values

For many models, the choice of initial parameter values is crucial to the success of the fitting process. Some methods are given here to help make this choice.

6.1. Plot the response variate against the explanatory variate(s)

For example, a graph of the data for the logistic model used above will give fairly obvious initial values for M, and B can be estimated by finding the slope at M.

6.2. Fit an approximate model using FITCURVE

Sometimes a nonlinear model is similar, but not identical, to one that can be fitted using **FITCURVE**. The logistic model above can be fitted using **FITCURVE [curve=logistic]** with explanatory variate **logConc**, replacing the value for concentration zero with a large negative number in **logConc**. The results of this can be used by **RCYCLE**.

6.3. Linearize the data

Some models can be linearized if approximate values for the linear parameters are known. For the logistic model above, if A and C are guessed, then:

$$ly = \log\left(\frac{C + A - Y}{Y - A}\right) = BM = B\log(\text{conc})$$

ly can be calculated and then linearly regressed on **logConc** using **FIT**; the resulting parameter estimates can be used to find initial values for B and M to use with **RCYCLE**.

7. Problems with the iterative process

If several attempts have been tried using different initial values, and the iterative process has failed to converge each time, a solution can sometimes be found if the optimization method is changed. The standard method is the Gauss–Newton, but the Newton–Raphson method will often converge when this fails. The method can be changed using the **RCYCLE** directive:

```
   RCYCLE [METHOD=newton] B,M; INITIAL=??
```

Alternatively, some parameters can be held constant, whilst the rest are estimated. This is useful if parameters are highly correlated, or if there are many parameters to be estimated relative to the amount of data. It can be done by setting the **STEPLENGTH** parameter of **RCYCLE** to zero for those to be held constant:

```
   RCYCLE B,M; INITIAL=1,0.5; STEPLENGTH=0,*
```

If the fitting process converges, then the parameter estimates obtained can be used as initial values in a new **RCYCLE** statement, either setting the steplengths for other parameters to zero (to hold these constant), or setting all steplengths to * (to estimate all parameters).

In the above process, it is useful to note that all the nonlinear parameters are set up by **RCYCLE** to be scalars, and that at the end of the fitting process they contain the final estimates for the parameters. If the same parameters are used in a later **FITNONLINEAR** statement, without reinitializing their values in an **RCYCLE** statement, then the previously estimated values are used as initial values.

```
RCYCLE B,M; INITIAL=1,*; step=*,0
```

Sometimes the iterative process crashes completely, producing a 'Fatal Fortran' error. This is not (usually) due to a program fault, but due to calculations attempted by Genstat. Changing the initial values or the iterative method can solve this problem; alternatively, the expression can be re-parameterized, for instance replacing a parameter that must be positive by exp(parameter). For example, replace

$$y = 1/(1 + \exp(-b * (x - m)))$$

by

$$y = 1/(1 + \exp(-\exp(lb) * (x - m))).$$

8. Problems with functions used in expressions

In the fitting process, Genstat can produce values that are too large or too small for the functions (eg **LOG**, **EXP**) used in the expression to deal with. This usually causes a large number of warnings of the type:

'Invalid value for function...'

These warnings can be prevented, and the iterative process kept 'on track' if logicals are included in the expression to prevent invalid values being produced.

For example, the largest value that the **EXP** function can take is machine dependent, on a PC the value is 87. If the expression included the calculation is

```
EXP(M)
```

then the argument for **EXP** can be prevented from exceeding 87 by replacing the calculation with:

```
EXP( 87*(M.GT.87)+M*(M.LE.87) )
```

9. References

Brain P and Butler R (1988) Cumulative Count Data *Genstat Newsletter* (22) 38–47.

Butler R and Brain P (1991) CUMDISTRIBUTION *Genstat 5 Procedure Library Manual, Release 2[2]* NAG, Oxford.

Lane P W (1991) Minimization of a function *Genstat Newsletter* (27) 36–38.

Payne R W *et al* (1987) *Genstat 5 Reference Manual* Oxford University Press, Oxford.

Payne R W *et al* (1990) *Genstat 5 Release 2 Reference Manual Supplement* NAG, Oxford.

Piecewise Linear Regression With Genstat

J B van Biezen
Institute for Forestry and Nature Research (IBN-DLO)
PO Box 9102
6800 HB Arnhem
The Netherlands

A piecewise linear regression is a model for the relationship between two variables that is made up of line segments. Fitting such a model is a "normal" regression analysis if the positions, on the axis of the explanatory variable, of the intersection points of the segments are known and fixed before fitting. When they are not fixed one cannot fit a piecewise regression by means of **FIT** statements alone. A first approach for the estimation of the position of intersection points is to use one or more loops of X-values and **FIT** and **RKEEP** statements. A second more exact fit can then be obtained using **RCYCLE** and **FITNONLINEAR**. Asymptotic standard errors of the positions, and correlation coefficients between them, can also be estimated. An asymptotic confidence ellipse of an intersection point can also be calculated.

It is not easy to write a general Genstat procedure for fitting piecewise linear regression models as tuning of the fitting process is necessary. The output below shows the fitting of a model with three segments, the intermediate segment being constrained to be horizontal. It should be fairly easy to modify the program to work with other models, such as two segments, with or without a constraint that one of them should be horizontal.

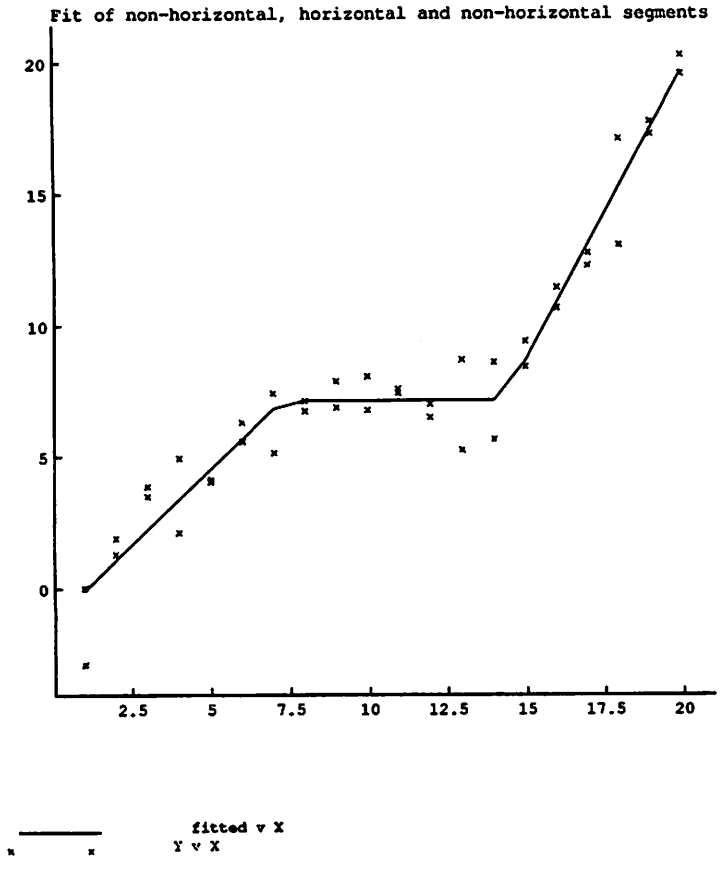


Figure 1
Fit of non-horizontal, horizontal and non-horizontal segments

Because the statistical aspects of piecewise linear regression models are easily understood, these models can be used to introduce the **RCYCLE** and **FITNONLINEAR** directives in Genstat courses. The program shown below has been used in this way, though for the purpose of this article the original program has been rewritten somewhat and simulated observations are used. The original version of this program was used to examine real observations made by a research worker.

The model can be extended in several ways. It is possible to include a horizontal line in the model; further, it is possible to fix the Y value of the horizontal line. Estimating such a value is another possibility. Also, extension to a generalized linear model is possible.

The program makes use of the **SORT** directive to divide the set of X values into groups for the initial search. The search is time-consuming, running through many possibilities to find a good and reliable starting point for the final fit. Estimates of standard errors and correlation coefficients are calculated. However it is doubtful whether these estimates are good ones; they give rough indications of the precision of each estimate and the relationship between these estimates. It is possible to add a sixth step, using the **RFUNCTION** directive to get standard errors and correlation coefficients of functions of the primary, estimated parameters.

It is possible to use Newton–Raphson, Gauss–Newton or Fletcher–Powell algorithms in Step 4. If one algorithm does not converge, one of the others may do so.

Note that the graph that is drawn does not display the model completely correctly. This is because the line is drawn by joining up fitted values rather than by calculating points from the fitted equation.

Appendix: Output from the Genstat Program

```
  1  JOB 'Fit a horizontal segment between two non-horizontal segments'
  2  " Simulate Y values instead of real observed values:"
  3  CALCULATE X,Y = !(2(1...20)),!(2(1...6,8(7),9,11...19))
  4  & Y = Y+NED(URAND(368412;40))
  5
  6  " STEP 1: group X values and calculate the number of different X values"
  7  SORT [INDEX=X; GROUPS=help]
  8  CALCULATE nn= NLEVELS(help)
  9
 10  " STEP 2: calculate an initial residual sum of squares"
 11  MODEL Y; FITTED=fitted
 12  FIT [PRINT=*] X
 13  RKEEP Y; DEVIANCE=ss
 14
 15  " STEP 3: rough first fit"
 16  TABULATE [CLASS=help] X; MEAN=tmx
 17  SCALAR mx[1...nn]; VALUE=#tmx
 18  FOR I=1...nn
 19    CALCULATE X1 = (X-mx[I])*(help<I)
 20    FOR J=I...nn
 21      CALCULATE X2 = (X-mx[J])*(help>J)
 22      FIT [PRINT=*; NOMESSAGE=aliasing] X1,X2
 23      RKEEP Y; DEVIANCE=ss2
 24      IF ss2<ss
 25        SCALAR ss,left,right; VALUE=ss2,mx[I,J]
 26      ENDIF
 27    ENDFOR
 28  ENDFOR
 29  " Output of first rough search:"
 30  CALCULATE X1 = (X-left)*(X<left)
 31  & X2 = (X-right)*(X>right)
 32  FIT X1,X2

32.....................................................................

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

 Response variate: Y
     Fitted terms: Constant, X1, X2
```

```
*** Summary of analysis ***

             d.f.         s.s.         m.s.      v.r.
Regression      2      1002.58      501.288    410.44
Residual       37        45.19        1.221
Total          39      1047.77       26.866

Percentage variance accounted for 95.5

* MESSAGE: The following units have large standardized residuals:
                    1       -2.79

* MESSAGE: The following units have high leverage:
                    1       0.205
                    2       0.205
                   39       0.205
                   40       0.205


*** Estimates of regression coefficients ***

                estimate         s.e.         t
Constant           7.010        0.243     28.80
X1                1.1906       0.0993     11.99
X2                2.0683       0.0993     20.82

  33
  34  " STEP 4: Gauss-Newton fit"
  35  EXPRESSION calc3[1]; VALUE=!e(lcoef = (X-left)*(X<left))
  36  & calc3[2]; VALUE=!e(rcoef = (X-right)*(X>right))
  37  RCYCLE [METHOD=gaussnewton] left,right
  38  FITNONLINEAR [PRINT=model,summary,estimates,correlations; \
  39    CALCULATION=calc3; SELINEAR=yes] lcoef,rcoef

39.....................................................................


***** Nonlinear regression analysis *****

 Response variate: Y


*** Summary of analysis ***

             d.f.         s.s.         m.s.      v.r.
Regression      4      1003.65      250.913    199.07
Residual       35        44.11        1.260
Total          39      1047.77       26.866

Percentage variance accounted for 95.3

* MESSAGE: The following units have large standardized residuals:
                    1       -2.51


*** Estimates of parameters ***

                estimate         s.e.
left               7.271        0.564
right             14.310        0.342
* Linear
lcoef              1.151        0.150
rcoef              2.193        0.190
Constant           7.152        0.300
```

```
*** Correlations ***

estimate      ref    correlations

left            1    1.000
right           2    0.185  1.000
lcoef           3   -0.756  0.000  1.000
rcoef           4    0.000  0.808  0.000  1.000
Constant        5    0.462  0.400  0.000  0.000  1.000
                        1      2      3      4      5
```

```
 40
 41  IF right<left
 42  " STEP 5: extra fit if there is not a horizontal segment"
 43    CALCULATE limit = (left+right)/2
 44    RCYCLE limit
 45    EXPRESSION calc2[1]; VALUE=!e(lcoef = (X-limit)*(X<limit))
 46    & calc2[2]; VALUE=!e(rcoef = (X-limit)*(X>limit))
 47    FITNONLINEAR [PRINT=model,summary,estimates,correlations; \
 48      CALCULATION=calc2; SELINEAR=yes] lcoef,rcoef
 49    DGRAPH [TITLE=title] fitted,Y; X; PEN=1,2
 50  ENDIF
 51
 52  TEXT title; VALUES=\
 53    'Fit of non-horizontal, horizontal and non-horizontal segments'
 54  PEN 1,2; METHOD=line,point; SYMBOL=0,1
 55  DGRAPH [TITLE=title] fitted,Y; X; PEN=1,2
 55
 56  STOP
```

Generalized Additive Models in Genstat: flexible nonparametric modelling extensions

Trevor J Hastie
AT&T Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974, USA
Peter W Lane
Statistics Department, AFRC Institute of Arable Crops Research, Rothamsted Experimental Station, Harpenden, Herts, United Kingdom, AL5 2JQ
and
Rob J Tibshirani
Department of Preventive Medicine and Biostatistics, and Department of Statistics, University of Toronto, Ontario, Canada

1. Additive models

An additive model is a multivariate regression model that is composed of a sum of lower dimensional components. Each component relates the response variable to some of the explanatory variables — usually just one — and the form of the relationship is general. So, an additive model whose components each involve just one explanatory variable can be represented by

$$Y = f_1(x_1) + f_2(x_2) \ldots + e$$

where $f_i(\ )$ are arbitrary functions. These models are described in detail in Hastie and Tibshirani (1990).

The usual multiple linear regression model is a simple example of an additive model:

$$Y = \alpha + \beta_1 x_1 + \beta_2 x_2 \ldots + e$$

where the individual components are just linear effects of each explanatory variable. Polynomial regression models are also additive; for example

$$Y = \alpha + \{\beta_{11} x_1 + \beta_{12} x_1^2\} + \{\beta_{21} x_2 + \beta_{22} x_2^2 + \beta_{23} x_2^3\} + e$$

where each component is a polynomial function of an explanatory variable.

More importantly, the components may be non-parametric, typically modelling the relationship between the response and an explanatory by some smoothing device. A simple model involving one explanatory model is

$$Y = f(x) + e$$

where $f(\ )$ is a smooth function produced by a scatterplot smoother. Estimates of f can vary between the simple running mean, or a function more expensive to calculate, such as a kernel smooth or a smoothing spline. The individual components of a model need not be all parametric, or all non-parametric; semi-parametric models can also be considered, such as

$$Y = \alpha + \beta_1 x_1 + \beta_2 x_2 + f_3(x_3) + e$$

This extension to include general functions of explanatory variables provides the data analyst with a powerful tool to explore data. Non-parametric components can be included initially in a model to suggest a suitable parametric form for the component. Alternatively, they can be included permanently to adjust for the effect of explanatory variables for which an explicit parametric form is of no interest, or for which a smooth data-generated effect is preferred to an imposed parametric form.

2. Generalized additive models

The additive structure described above can be embedded in a richer class of models than just those of multiple linear regression. In conjunction with the link function and exponential-family distribution of generalized linear models, an additive model is more suitably called a "generalized additive model". This of course includes additive models as a subclass, just as linear regression models are a subclass of generalized linear models. Generalized additive models can also be based on quasi-likelihood, rather than on likelihood for explicit distributions from the exponential family.

For example, a logistic regression model can be modified to include smooth contributions from explanatory variables.

Y is distributed as Binomial$(1, p(X))$

$$\text{logit}(p(X)) = \log(p(X)/(1-p(X)) = \alpha + f_1(x_1) + f_2(x_2)...$$

Other examples where the extension can be made include:

- Log-linear models
- Transformation models, using the ACE algorithm, Breiman and Friedman (1985)
  $g(Y) = \alpha + f_1(x_1) + f_2(x_2)... + e$
- The Cox model for censored survival data
  $l(X,t) = l_0(t)\exp(f_1(x_1) + f_2(x_2)...)$
- Resistant additive models, using tapered likelihoods
- Semiparametric additive models for designed experiments
- Additive decomposition of time series

3. Fitting additive models

Additive models can be fitted by straightforward application of an iterative technique to solve a suitable set of estimating equations. For a model with p explanatory variables, the equations have the form

$$\begin{array}{lllll} f_1(x_1) & = S_1(y - & f_2(x_2) - & f_3(x_3) - & \ldots - f_p(x_p)) \\ f_2(x_2) & = S_2(y - f_1(x_1) - & & f_3(x_3) - & \ldots - f_p(x_p)) \\ & \ldots & & & \\ f_p(x_p) & = S_p(y - f_1(x_1) - & f_2(x_2) - & & \ldots - f_{p-1}(x_{p-1})) \end{array}$$

where S_j represent either linear regression operators, or univariate regression smoothers such as kernel smoothers or smoothing splines.

The iterative technique known as Gauss–Seidel iteration, or "back-fitting" solves these by successively carrying out the operations for each equation, refining the estimates for one component at a time; convergence is checked by monitoring the change in the estimates. This can be set up as a simple loop around existing code for fitting a linear regression, together with algorithms to carry out whatever smoothing operations are required.

An appealing non-parametric smoother for the effect of a single explanatory variable is the smoothing spline. This is a cubic spline with knots at the unique values of the explanatory variable, and with constraints to ensure continuity and smoothness (continuity of gradient). This spline function arises as the solution of the penalized least-squares problem: to minimize the function

$$\sum_{i=1}^{n}\{y_i - f(x_i)\}^2 + \lambda \int \{f''(t)\}^2 dt$$

for some values of the Lagrange multipliers λ_j. Thus, the smoothing spline minimizes a criterion which is the sum of squared deviations from the fitted values, plus a penalty for non-smoothness. The values of the multipliers (known as the smoothing parameters) allow the smooth function to vary between the extremes of being very "wiggly" and passing through all the data points, and of being very smooth — in fact, being a straight line.

4. Implementation in Genstat

The regression section is being extended in Release 3 to allow smooth terms in model formulae. So it will be possible to give statements of the form

```
FIT SSPLINE(age)
```

or

```
ADD number + SSPLINE(start; 3)
```

The **SSPLINE** function (or **S** for short) specifies that the effect of an explanatory variate should be represented by a smoothing spline. By default, four (effective) degrees of freedom will be assigned to the smoothing operation, including the linear component. The second argument of the function can be set to specify any other number of degrees of freedom — from 1, equivalent to a linear effect, up to the number of distinct values of the variate. One uses these effective degrees of freedom as an alternative and more intuitive way of specifying the amount of smoothing λ_j, and they are based on the expected residual sum-of-squares for a null model (see Hastie and Tibshirani (1990) for more details).

Initially, the function may be used only for the effects of variates; but smooth effects for interactions between smoothed variates and factors may also be added. It is also intended to allow smooth terms in nonlinear models with Normal likelihood.

This extension has been integrated as far as possible with the existing regression facilities without disturbing them. Thus there is no other syntax addition or change required to fit generalized additive models. The maximum number of iterations made in backfitting can be controlled by specifying the **MAXCYCLE** option of **RCYCLE**: this same limit is applied both to the backfitting algorithm and to the iterative weighted algorithm used for generalized linear models. After the fit, the nonlinear components of each smoothed term is available in variates attached to the regression save structure; a change will be made to the **RKEEP** directive to allow access to these structures in the same way as to fitted values and residuals.

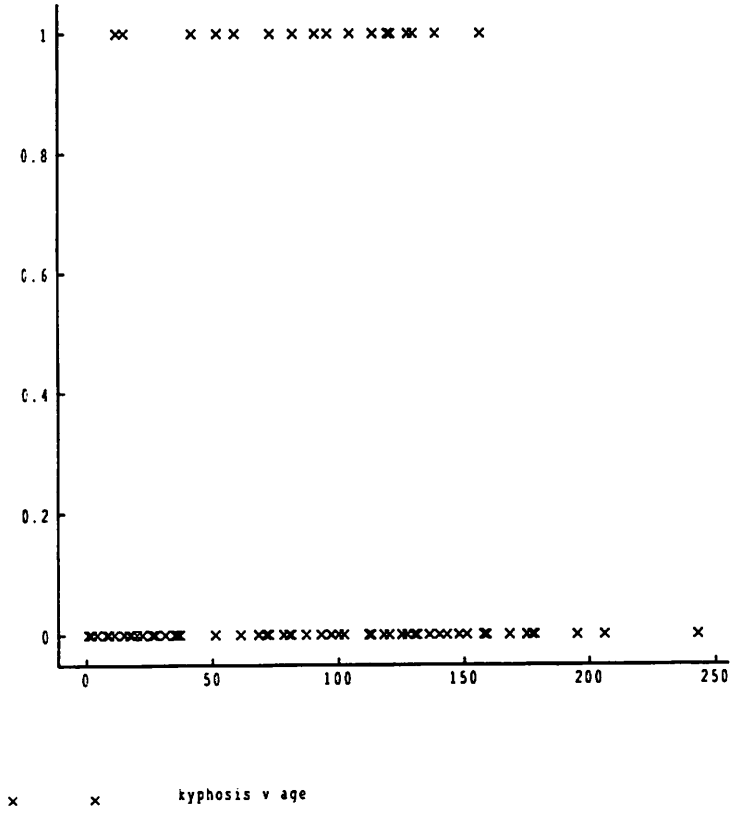


Figure 1
Plot of kyphosis vs age

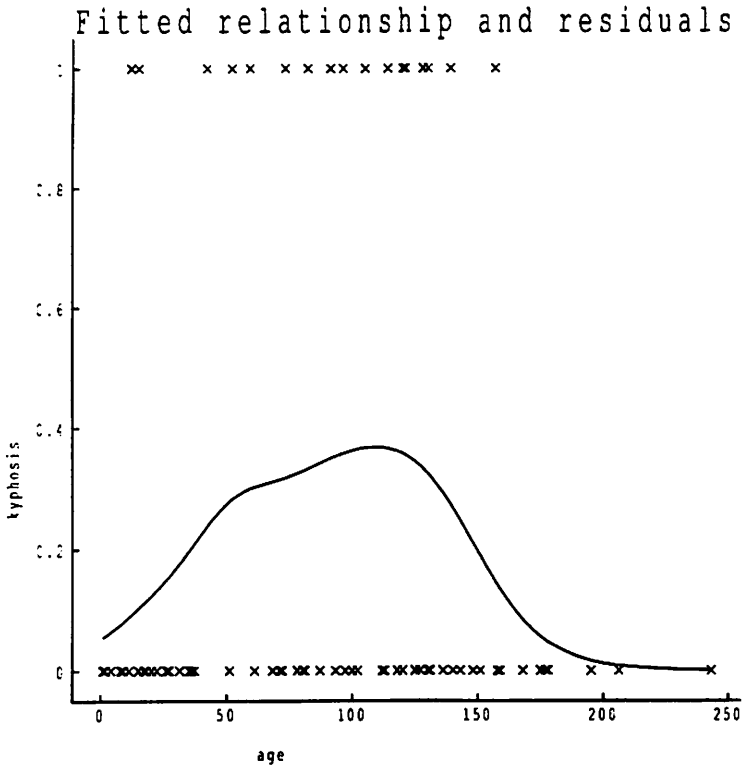


Figure 2
Plot from **RGRAPH**

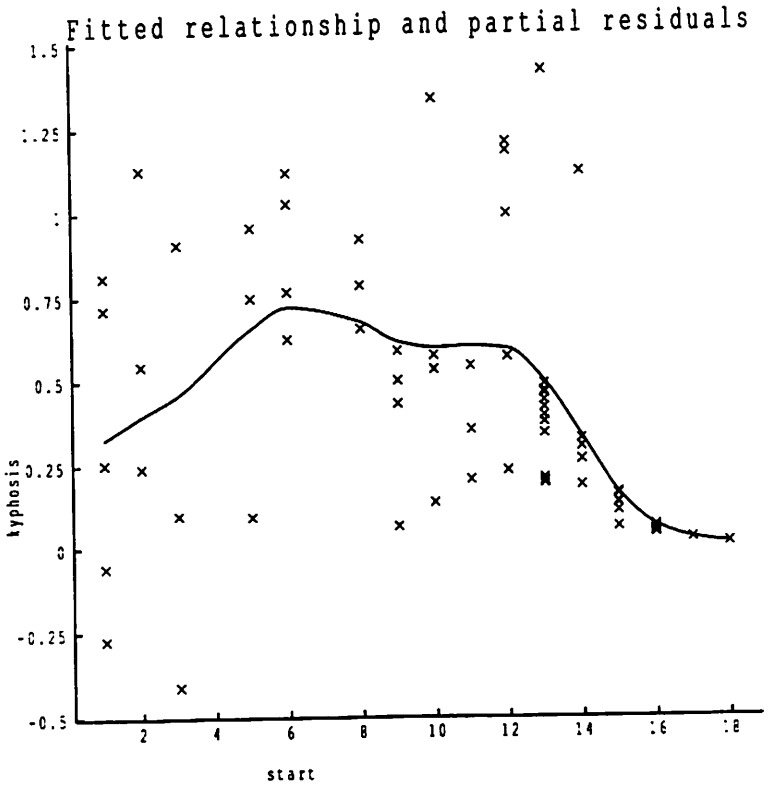


Figure 3
Plot from **RGRAPH**

5. Example

The following example illustrates the fitting of a series of generalized additive models to binomial data. The pictures are produced by a test version of procedure **RGRAPH**: a simpler version of this procedure is in Procedure Library 2[3], and this will be extended for Release 3.

Some other new features of Release 3 are also evident. Further checks are made on fitted models, and messages printed if the residuals appear to have a distinctly non-random pattern. However, with a binary response variable as in this example, it is not at all surprising that such patterns appear.

```
Genstat 5  Release 3.0  (Vax/VMS5)                  2-MAR-1992 10:37:48.13
Copyright 1990, Lawes Agricultural Trust (Rothamsted Experimental Station)

   1  " Data on 83 patients undergoing corrective spinal surgery;
  -2    determine risk factors for kyphosis (forward flexion of the spine).
  -3    Data from Hastie & Tibshirani p301."
   4
   5  " Open data file, and procedure library file containing test version
  -6    of procedure RGRAPH."
   7  OPEN 'kyphosis.dat','test.glb'; CHANNEL=2,1; FILE=input,procedure
   8  READ [CHANNEL=2; SETNVALUES=yes] unit,kyphosis,age,number,start

    Identifier   Minimum      Mean   Maximum    Values   Missing
          unit      1.00     42.00     83.00        83         0
      kyphosis    0.0000    0.2169    1.0000        83         0    Skew
           age      1.00     84.71    243.00        83         0
        number     2.000     4.217    14.000        83         0    Skew
         start      1.00     11.34     18.00        83         0

   9  DGRAPH kyphosis; age
  10
  11  " Choose binomial distribution (logit link by default);
 -12    note that NBINOMIAL can be set to a scalar in Release 3."
  13  MODEL [DISTRIBUTION=binomial] kyphosis; NBINOMIAL=1
  14  " Specify explanatories, and that smooth effects may be fitted."
  15  TERMS SSPLINE(age,start)+number
  16  " Fit smooth effect of age. "
  17  FIT SSPLINE(age)

17.....................................................................


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

 Response variate: kyphosis
  Binomial totals: 1
     Distribution: Binomial
    Link function: Logit
     Fitted terms: Constant + age
        Functions: SSPLINE(age; 4)


*** Summary of analysis ***
    Dispersion parameter is assumed fixed, with value 1

                                      mean  deviance
             d.f.      deviance    deviance     ratio
Regression      4          9.31      2.3265      2.34
Residual       78         77.50      0.9936
Total          82         86.80      1.0586

Change         -4         -9.31      2.3265      2.34

* MESSAGE: The error variance does not appear to be constant:
           large responses are more variable than small responses
```

```
*** Estimates of regression coefficients ***

                             estimate         s.e.         t
Constant                       -1.617        0.620     -2.61
age                           0.00591      0.00611      0.97
* MESSAGE: s.e.s are based on dispersion parameter with value 1

  18
  19  " Display fitted model with procedure RGRAPH."
  20  RGRAPH [GRAPHICS=high] age
  21
  22  " Add linear effect of number."
  23  ADD number

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


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

 Response variate: kyphosis
  Binomial totals: 1
     Distribution: Binomial
    Link function: Logit
     Fitted terms: Constant + age + number
        Functions: SSPLINE(age; 4)

*** Summary of analysis ***
    Dispersion parameter is assumed fixed, with value 1

                                        mean  deviance
              d.f.     deviance     deviance     ratio
Regression       5        23.37       4.6750      5.68
Residual        77        63.43       0.8238
Total           82        86.80       1.0586

Change          -1       -14.07      14.0688     17.08

* MESSAGE: The error variance does not appear to be constant:
           intermediate responses are more variable than small or large responses
* MESSAGE: The following units have high leverage:
                  15       0.289
                  28       0.223
                  45       0.190


*** Estimates of regression coefficients ***

                             estimate         s.e.         t
Constant                       -4.51         1.31     -3.46
age                           0.00940      0.00767      1.23
number                          0.583        0.189      3.08
* MESSAGE: s.e.s are based on dispersion parameter with value 1

  24
  25  " Add smooth effect of start, specifying explicitly 4 d.f."
  26  ADD SSPLINE(start; 4)

26.....................................................................


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

 Response variate: kyphosis
  Binomial totals: 1
     Distribution: Binomial
    Link function: Logit
     Fitted terms: Constant + age + number + start
        Functions: SSPLINE(age; 4)
                   SSPLINE(start; 4)
```

```
*** Summary of analysis ***
    Dispersion parameter is assumed fixed, with value 1

                                        mean  deviance
             d.f.     deviance     deviance     ratio
Regression      9        39.95       4.4387      6.92
Residual       73        46.86       0.6419
Total          82        86.80       1.0586

Change         -4       -16.57       4.1433      6.46

* MESSAGE: The residuals do not appear to be random;
           for example, fitted values in the range 0.00 to 0.07
           are consistently larger  than observed values
           and fitted values in the range 0.61 to 0.85
           are consistently smaller than observed values
* MESSAGE: The error variance does not appear to be constant:
           intermediate responses are more variable than small or large responses
* MESSAGE: The following units have high leverage:
                 15        0.59
                 28        0.46


*** Estimates of regression coefficients ***

                             estimate         s.e.         t
Constant                        -2.29         1.53     -1.49
age                           0.01278      0.00858      1.49
number                          0.410        0.196      2.09
start                         -0.1673       0.0780     -2.14
* MESSAGE: s.e.s are based on dispersion parameter with value 1

  27  RDISPLAY [accumulated]

27.....................................................................


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

*** Accumulated analysis of deviance ***

Change                                                 mean  deviance
                                d.f.     deviance     deviance     ratio
+ age                              4       9.3060       2.3265      3.62
+ number                           1      14.0688      14.0688     21.92
+ start                            4      16.5734       4.1433      6.46
Residual                          73      46.8556       0.6419

Total                             82      86.8038       1.0586

  28
  29  " Display smoothing spline for start alone."
  30  RGRAPH [GRAPHICS=high] start
  31
  32  STOP
```

6. References

Breiman L and Friedman J H (1985) Estimating optimal transformations for multiple regression and correlation *J. Amer. Statist. Assoc.* **80** 580–619.

Hastie T J and Tibshirani R J (1990) *Generalized Additive Models* Chapman and Hall, London.

Modelling the Development of Temperature-dependent Processes

R J Reader and K Phelps
Biometrics Department
Horticulture Research International
Wellesbourne
Warwick
United Kingdom CV35 9EF

1. Introduction

The development of many biological organisms can often be regarded as an ordered sequence of processes which take place over a period of time, punctuated by more-or-less discrete events (Thornley and Johnson 1990). Examples of these processes are seed germination, larval development and spore production for plants, insects and fungi respectively.

Much modelling work on development is based on the idea that it is possible to associate a scalar variable h with each phase, and to use the value of h to denote progression through that phase. If we assume that the development of the process proceeds at a rate k (day^{-1}) then

$$h = \int_0^t k\,dt \tag{1}$$

where t is the time variable (day) and time $t = 0$ is the beginning of the developmental phase under consideration.

It is more usual to assume that the developmental rate k, and hence the variable h, depend directly upon the environmental variables. In our case we assume that they depend solely upon temperature so that

$$k = r(T) \tag{2}$$

and temperature varies with time so that

$$T = T(t) \tag{3}$$

and hence

$$h = \int_0^t r(T(t))\,dt. \tag{4}$$

The temperature-sum hypothesis has been widely used for many years. In this it is assumed that the rate depends on temperature according to

$$r(T) = \begin{cases} k_0(T - T_0) & T \geq T_0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where k_0 is a constant and T_0 is a base temperature below which development ceases. The quantity h can then be obtained by integration, but it is more usual to approximate it using summation because temperature data are available as daily maxima and minima. Some authors use mean temperature for each day to complete the summation, but this is highly inaccurate if temperatures lie either side of the threshold. More often the concept of day-degrees (Meteorological Office, 1946) is used. This assumes that temperatures each day follow a sine wave with period one day where the maximum and minimum define the amplitude and mean. The properties of the sine curve make it convenient to calculate the temperature-sum by hand.

Equation (5) is not appropriate for some processes because the rate/temperature relationships are nonlinear or non-monotonic. It is also well known that the sine wave does not describe the diurnal temperature variation particularly well (Reicosky *et al* 1989). Although the day-degree approximations are accurate enough to be useful in many applications, we have found several examples of processes where a more sophisticated approach is needed.

In most cases we are interested simply in the day on which a process was completed rather than the exact time at which it was completed. We therefore define the change in development on day i as being

$$\Delta h_i = \int_0^1 r(T_i(t))\,dt \tag{6}$$

and the total development is then

$$h_i = \sum_{j=0}^{i} \int_0^1 r(T_j(t))\,dt \tag{7}$$

where $T_j(t)$ relates temperature, T, to time, t, on day j.

2. A Genstat procedure to calculate development

We have written a Genstat procedure, **HEATUNITS**, (Reader, Sutherland and Phelps 1991), also available in Fortran, to allow for any rate function that can be described by a linear spline and a selection of relationships for diurnal temperature variation discussed by Reicosky *et al* (1989). The possible diurnal temperature variation functions are as follows, where X and N denote maximum and minimum temperatures, S and R denote sunset and sunrise times (in days) and subscripts p and n denote previous and next days:

Sawtooth

$$T(t) = \begin{cases} x_p - \frac{(9+24t)(X_p-N)}{14} & 0 < t \le \frac{5}{24} \\ N + \frac{(24t-5)(X-N)}{10} & \frac{5}{24} < t \le \frac{15}{24} \\ X - \frac{(24t-15)(X-N_n)}{14} & \frac{15}{24} < t \le 1 \end{cases}$$

Cosine

$$T(t) = \begin{cases} \frac{X_p+N}{2} + \frac{X_p-N}{2}\cos\left(\frac{\pi(24t+10)}{(10+24R)}\right) & 0 < t \le R \\ \frac{X+N}{2} - \frac{X-N}{2}\cos\left(\frac{24\pi(t-R)}{914-24R)}\right) & R < t \le \frac{14}{24} \\ \frac{X+N_n}{2} + \frac{X-N_n}{2}\cos\left(\frac{\pi(24t-14)}{(10+24R_n)}\right) & \frac{14}{24} < t \le 1 \end{cases}$$

LinSine

$$T(t) = \begin{cases} T(S_p) - \frac{(T(S_p)-N)(t+1-S_p)}{(1-S_p+R+\frac{2}{24})} & 0 < t \le (R+\frac{2}{24}) \\ N + (X-N)\sin\left(\frac{\pi(t-R-\frac{2}{24})}{(S-R)}\right) & (R+\frac{2}{24}) < t \le S \\ T(S) - \frac{(T(S)-N_n)(t-S)}{(1-S+R_n+\frac{2}{24})} & S < t \le 1 \end{cases}$$

ExpSine

$$T(t) = \begin{cases} \frac{T(S_p)\left(\exp\left(\frac{-b(1-S_p+t)}{Z_1}\right)-\exp\left(\frac{-b(Z_1+c)}{Z_1}\right)\right)+N\left(1-\exp\left(\frac{-b(1-S_p+t)}{Z_1}\right)\right)}{1-\exp\left(\frac{-b(Z_1+c)}{Z_1}\right)} & 0 < t \le (R+c) \\ N + (X-N)\sin\left(\frac{\pi(t-R-c)}{Y+2a}\right) & (R+c) < t \le S \\ \frac{T(S)\left(\exp\left(\frac{-b(t-S)}{Z_2}\right)-\exp\left(\frac{-b(Z_2+c)}{Z_2}\right)\right)+N_n\left(1-\exp\left(\frac{-b(t-S)}{Z_2}\right)\right)}{1-\exp\left(\frac{-b(Z_2+c)}{Z_2}\right)} & S < t \le 1 \end{cases}$$

where for the ExpSine method $Z_1 = 1 - S_p + R$, $Z_2 = 1 - S + R_n$, $Y = S - R$ and the parameters a, b, c control the period of the sine wave, the rate of decay of the exponential and the offset of the sine wave respectively. This is a slight modification of that used by Reicosky *et al* (1989) to constrain it to pass through the next day's minimum.

Our procedure calculates the contribution for each day as the analytic solution of using the maximum and minimum temperatures for days $i-1$, i, $i+1$.

Integration of these functions was approached as follows. First we defined a general linear spline rate function

$$r(T) = \begin{cases} 0 & T < T_0 \\ r_{j-1} + \frac{(T-T_{j-1})(r_j-r_{j-1})}{T_j-T_{j-1}} & T_{j-1} \le T \le T_j \quad j = 1 \ldots n \\ 0 & T > T_n \end{cases} \qquad (8)$$

in terms of its $n+1$ cardinal temperatures, $T_0 \ldots T_n$, and the rate, $r_0 \ldots r_n$, at those temperatures (Figure 1); this allowed Equation (6) to be integrated directly.

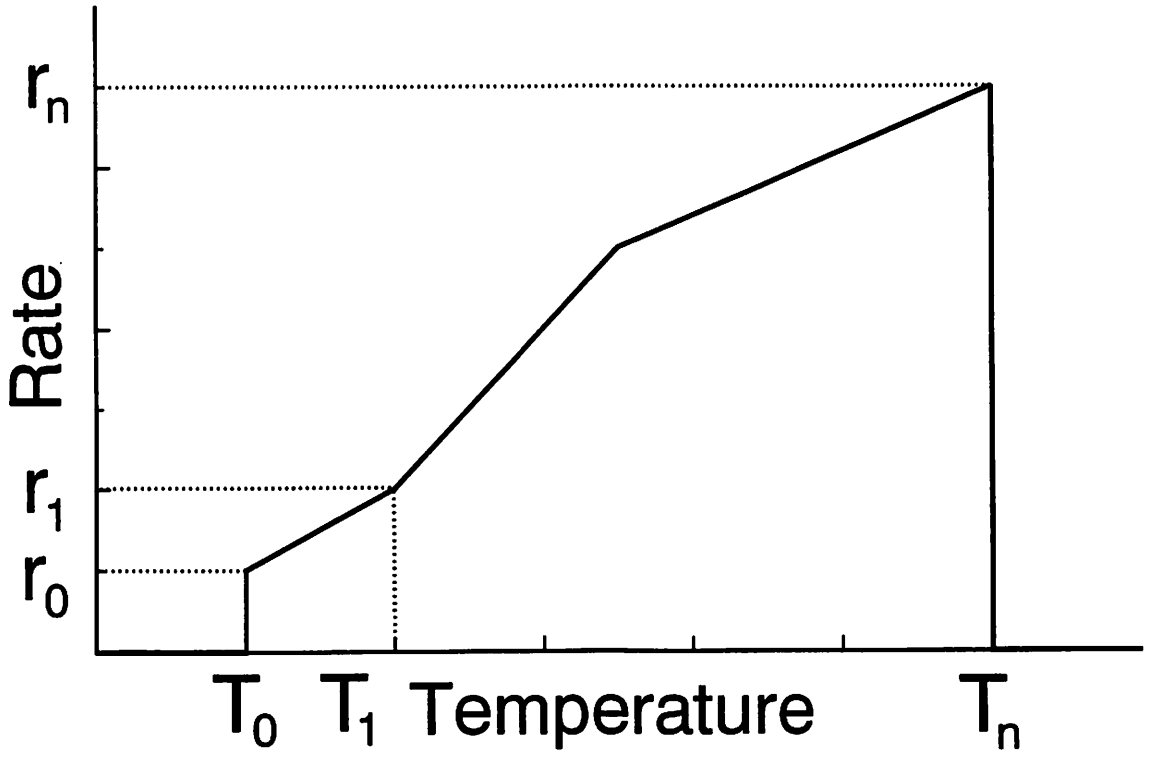


Figure 1

Example of rate/temperature relationship approximated by a linear spline. $T_0 \ldots T_n$ are cardinal temperatures and $r_0 \ldots r_n$ are the corresponding rates.

Next the integral for each day is broken down into three or four sections depending on the temperature–time relationship chosen (Figure 2). The endpoints of the sections are defined by the start and end of each day, the times of the maximum and minimum temperature and, for the ExpSine and LinSine method, the time at which the sinusoidal function ends. The contribution from all sections is then given by

$$\sum_{\text{section}=1}^{3 \text{ or } 4} \sum_{j=1}^{n} \int_{u_j}^{v_j} r(T(t))\, dt \tag{9}$$

where u_j is the earliest time that the temperature falls within the interval defined by the cardinal temperatures and v_j is the latest time. The earlier time limit, u_j, is defined by the first occurrence of cardinal temperature T_{j-1} when the temperature is increasing and T_{n-j+1} when the temperature is decreasing and the later time limit, v_j, is defined by T_j when the temperature is increasing and T_{n-j} when the temperature is decreasing.

3. Example of calculation of u_j and v_j

The middle part of the Sawtooth relationship results in the following definitions for u_j and v_j in terms of X, N and T_{n-j+1}

$$U_j \begin{cases} \frac{5}{24} & T_{j-1} \le N \\ \frac{5}{24} + \frac{10(T_{j-1}-N)}{24(X-N)} & N < T_{j-1} \le X \\ \frac{5}{24} & T_{j-1} > X \end{cases} \tag{10}$$

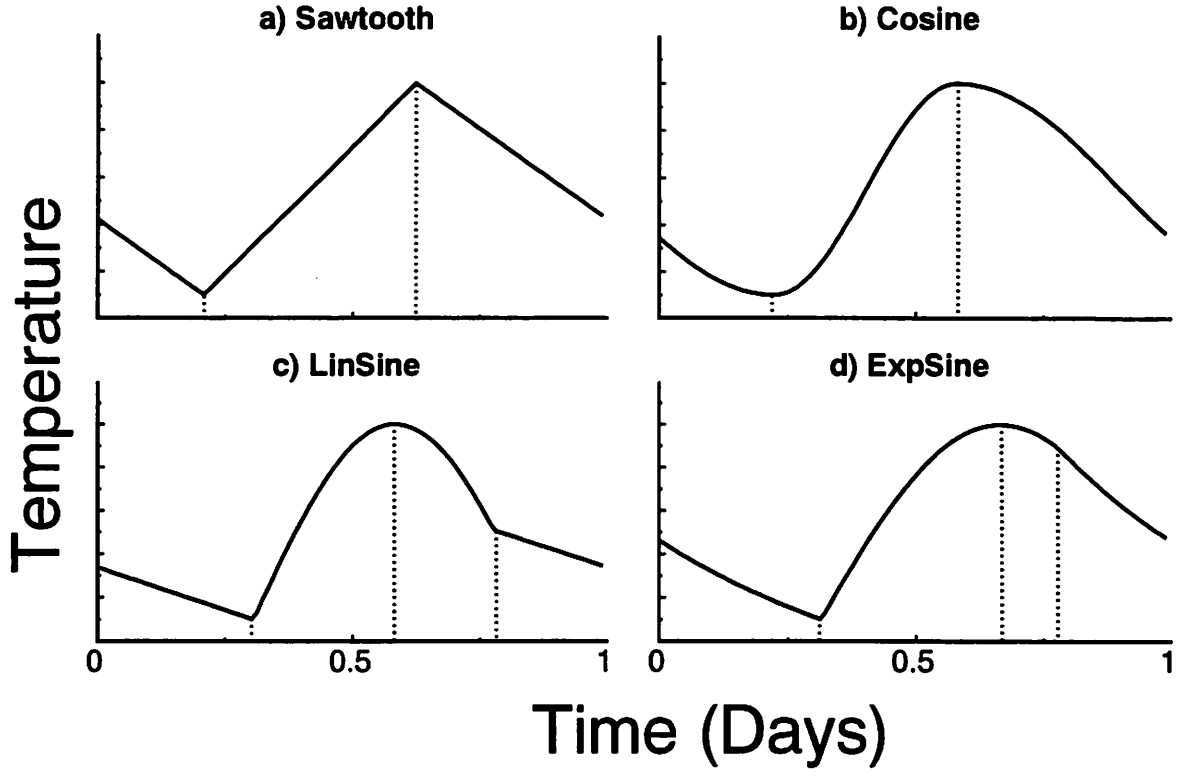


Figure 2

Temperature–time relationships. Vertical lines represent boundaries for integration.

$$V_j \begin{cases} \frac{5}{24} & T_{n-j+1} \leq N \\ \frac{5}{24} + \frac{10(T_{n-j+1}-N)}{24(X-N)} & N < T_{n-j+1} \leq X \\ \frac{15}{24} & T_{n-j+1} > X \end{cases} \qquad (11)$$

Full details of the calculation of the limits for all temperature–time relationships and sections of the day are not reproduced here but copies are available from the authors.

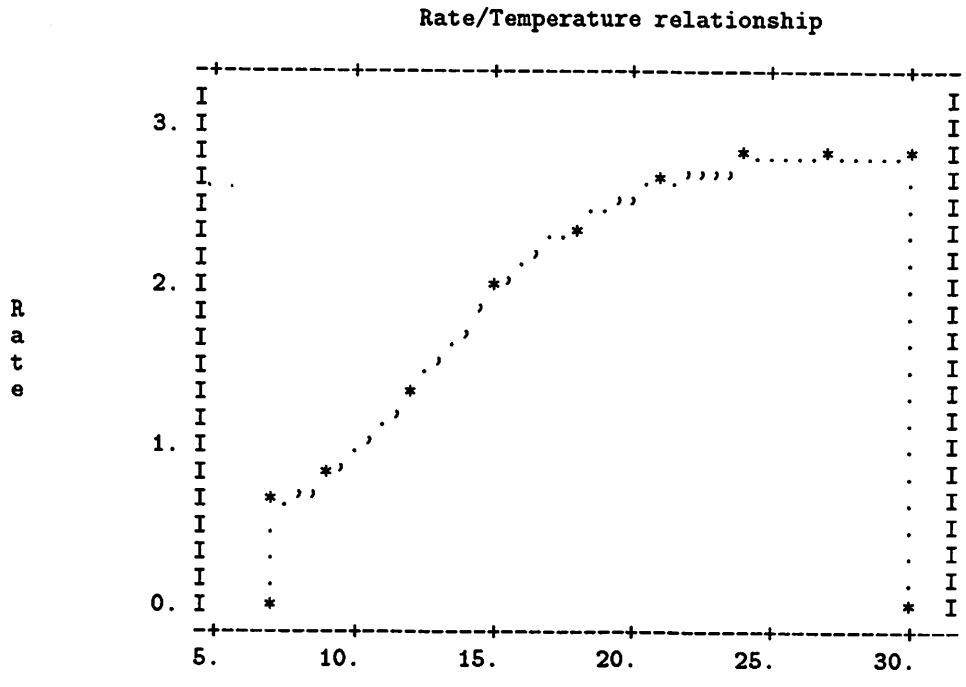
4. Example of use of HEATUNITS

This example, using data taken from Collier *et al* (1991), demonstrates the use of **HEATUNITS** with a nonlinear rate–temperature relationship approximated by a linear spline. This is then used to show the difference in time to 50% emergence of the cabbage root fly that would be expected if there was a 1–3°C increase in temperatures due to global warming.

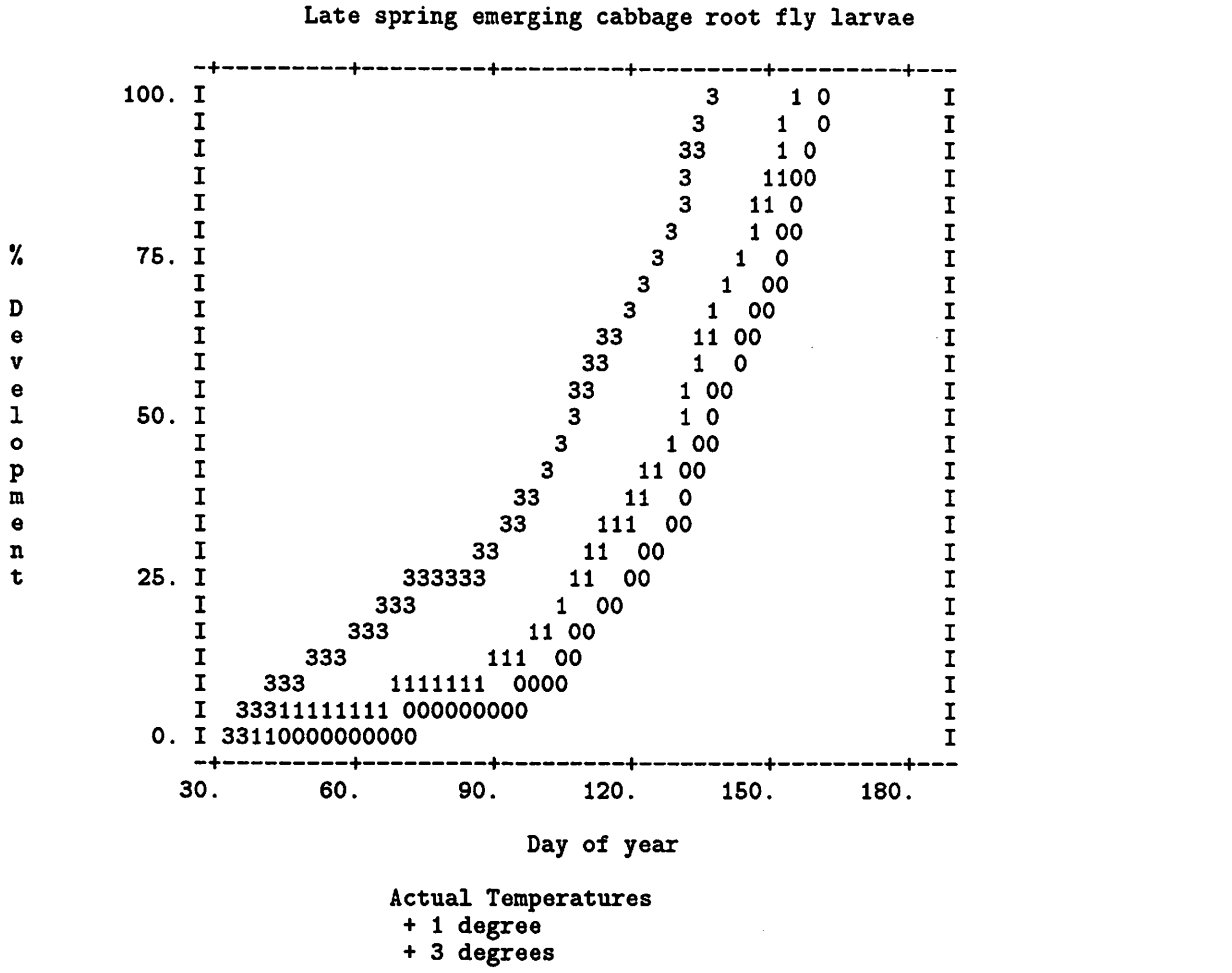
```
  6  SCALAR StartDate;  VALUE=32
  7  VARIATE [NVALUES=150] SoilMax[0, 1, 3], SoilMin[0, 1, 3]
  8  "Typical soil maxima and minima"
  9  READ SoilMax[0], SoilMin[0]

   Identifier   Minimum      Mean   Maximum    Values   Missing
   SoilMax[0]      3.62     12.29     28.80       150         0
   SoilMin[0]     0.589     7.652    17.543       150         0
 41  VARIATE Params, Dayno, Temperature; !(2, 1.75, 3.5), \
 42    !(32...181), !(7,9,12...30)
 43
 44  "Approximate nonlinear late spring emergence rate
-45  equation by a linear spline"
 46  CALCULATE CRF = 0.592 + 2.2525 * EXP(-EXP(-0.274 * \
 47    (Temperature - 12.49)))
```

```
 48  GRAPH [NROWS=20; TITLE=' Rate/Temperature relationship'; \
 49    YTITLE='Rate'; XTITLE='Temperature'] 2(!(0,#CRF,0)); \
 50    !(7,#Temperature,30); METHOD=line, point
```




```
 51  "Show effect of 1 or 3 degree rise in temperature on time
-52  of emergence of cabbage root fly"
 53  CALCULATE SoilMax[1, 3], SoilMin[1, 3] = \
 54    2(SoilMax[0], SoilMin[0]) + 1, 3
 55
 56  HEATUNITS [LATITUDE=52.205; METHOD=expsine; RATE=CRF; \
 57    TEMPERATURE=Temperature; PARAMETERS=Params] \
 58    MINTEMPERATURE=SoilMin[0]; MAXTEMPERATURE=SoilMax[0]; \
 59    FIRSTDAY=StartDate; HEATUNITS=Heatunits[0]
 60  HEATUNITS [LATITUDE=52.205; METHOD=expsine; RATE=CRF; \
 61    TEMPERATURE=Temperature; PARAMETERS=Params] \
 62    MINTEMPERATURE=SoilMin[1]; MAXTEMPERATURE=SoilMax[1]; \
 63    FIRSTDAY=StartDate; HEATUNITS=Heatunits[1]
 64  HEATUNITS [LATITUDE=52.205; METHOD=expsine; RATE=CRF; \
 65    TEMPERATURE=Temperature; PARAMETERS=Params] \
 66    MINTEMPERATURE=SoilMin[3]; MAXTEMPERATURE=SoilMax[3]; \
 67    FIRSTDAY=StartDate; HEATUNITS=Heatunits[3]
 68
 69  "Restrict out missing values at start and end of variate \
-70   before cumulating"
 71  RESTRICT Heatunits[]; Heatunits[] .NE. CONSTANTS('*')
 72  CALCULATE CHeatunits[0, 1, 3] = CUM(Heatunits[])
 73  RESTRICT Heatunits[], CHeatunits[]
 74  RESTRICT CHeatunits[]; CHeatunits[].LE.100
 75
 76  GRAPH [NROWS=25; YLOWER=0; YUPPER=100; \
 77    YTITLE='% Development'; XTITLE='Day of year'; \
 78    TITLE='Late spring emerging cabbage root fly larvae'] \
 79    CHeatunits[]; Dayno; METHOD=line; SYMBOLS='0', '1', '3'; \
 80    DESCRIBE='Actual Temperatures', ' + 1 degree', ' + 3 degrees'
```



5. References

Collier R H, Finch S , Phelps K and Thompson A R (1991) Possible impact of global warming on cabbage root fly (Delia radicum) activity in the UK *Ann. Appl. Biol.* **118** 261–271.

Meteorological Office (1946) Tables for the evaluation of daily values of accumulated temperature above and below 42°F from daily values of maximum and minimum temperature *Meteorological Office Leaflet No. 10* Meteorological Office, London Road, Bracknell, Berkshire, UK.

Reader R J, Sutherland R A and Phelps K (1992) Procedure HEATUNITS *Genstat 5 Procedure Library.* Manual Release 2[3].

Reicosky D C, Winkelman L J, Baker J M and Baker D G (1989) Accuracy of hourly air temperatures calculated from daily minima and maxima *Agricultural and Forest Meteorology* **46** 193–209.

Thornley J H M and Johnson I R (1990) *Plant and Crop Modelling - A Mathematical Approach to Plant and Crop Physiology* Oxford Science Publications, 166–193.

A Genstat Procedure for Testing Main Effects and Interactions in an Unbalanced Mixed Model

Willem Buist
Research Institute for Animal Production 'Schoonoord' (IVO-DLO) PO Box 501, 3700 AM, Zeist, The Netherlands
and
Bas Engel
Agricultural Mathematics Group (GLW-DLO) PO Box 100, 6700 AC, Wageningen, The Netherlands

1. Introduction

One of the major extensions of Genstat 5 in Release 2.1 is the analysis of an unbalanced mixed model. Variance components are estimated by restricted (or residual) maximum likelihood (REML), fixed effects are estimated by generalized least squares (GLS) and random effects are predicted by regression (BLUP). For details and references see Engel (1990). Facilities for testing treatment effects (or fixed effects) are restricted to paired comparisons between means corresponding to levels of a factor or to a combination of factors. Therefore, the procedure **VWALD** has been written in order to produce Wald tests for main effects and interactions. These tests may be used in the same way as F-tests in balanced **ANOVA** models.

2. The Wald test

Suppose that we have the following mixed model:

$$y = Xa + Z_1 b_1 + \ldots + Z_c b_c + e,$$

where a is a vector of fixed effects, $b_1 \ldots b_c$ are vectors of random effects and e is a vector of residual error terms. The elements of $b_1 \ldots b_c$ and e are assumed to be independently Normally distributed with zero means and variances $\sigma_1^2 \ldots \sigma_c^2$ and σ^2 respectively.

A simple example is a split-plot model where $c = 1$, b_1 represents variation between whole-plots with variance σ_1^2, and e represents variation between split-plots within whole-plots with residual variance σ^2.

Suppose that $a = (a_1, a_2)'$ and the hypothesis of interest is $H_0 : a_1 = 0$. Sub-vector a_1 may for instance represent the set of highest order interactions in the model.

The Wald test is based on approximate Normality of the estimators for the fixed effects:

$$\hat{a} = (\hat{a}_1, \hat{a}_2)' \sim N((a_1, a_2)', \hat{V}),$$

where $\hat{V}$ is the estimated covariance matrix

$$\hat{V} = \begin{pmatrix} \hat{V}_{11} & \hat{V}_{12} \\ \hat{V}_{21} & \hat{V}_{22} \end{pmatrix}.$$

The hypothesis H_0 may be tested by referring the statistic

$$t = a_1' \hat{V}_{11}^{-1} a_1$$

to a χ^2-distribution with p_1 degrees of freedom, where p_1 is the number of free parameters associated with H_0; i.e. for a full rank design matrix X, p_1 is the number of elements of a_1.

For the special case where a_1 represents a single contrast, i.e. $p_1 = 1$, the statistic t is simply the square of the standardized estimate $\hat{a}_1$ / s.e.($\hat{a}_1$), and may be compared with $(1.96)2 = 3.84 = X^2_{1;0.95}$.

The Normal approximation for a single contrast may be replaced by an approximate t-test (Giesbrecht and Burns 1985), accounting for the variability in the REML estimates of the variance components which are used to evaluate the standard error s.e.(a_1). For $p_1 > 1$ a similar procedure, for instance replacing the χ^2-distribution by an appropriate F-distribution, is not known, except for some special cases. In the procedure VWALD the χ^2-distribution is used in all cases.

3. Example

As an illustration the procedure will be applied to a reproductive study of rats (Dempster *et al* 1984).

Weights of offspring (pups) of rats are measured. The mothers (dams) have been randomly allocated to three treatment groups: control, low dose and high dose of an experimental compound. The weights may depend on litter size and sex. Litter sizes vary from 2 to 18 pups with an average of 11.9.

Treatment	No of litters	No of pups		Mean Weight	
		Male	Female	Male	Female
Control	10	77	54	6.47	6.12
Low	10	61	65	6.03	5.84
High	7	33	32	5.92	5.85

Table 1
Numbers of pups and mean weight

There are two sources of variation: variation between litters, represented by the dams, comprising both genetic and common environment effects, and residual variation within litters. The mixed model is:
weight $= \mu +$ dose $+ \beta *$ (littersize $- 12) + \epsilon_{\text{dam}} +$ sex $+$ dose.sex $+ \epsilon_{\text{residual}}$

```
UNITS [NVALUES=322]
FACTOR [LEVELS=322] Pup
GENERATE Pup
FACTOR [LEVELS=27] Dam
FACTOR [LEVELS=2; LABELS=!T('Male   ','Female')] Sex
FACTOR [LEVELS=3; LABELS=!T('Control ','Lowdose ','Highdose')] Dose
OPEN NAME='DEMPSTER.DAT'; CHANNEL=2
READ [CHANNEL=2; LAYOUT=fixed; FORMAT=!(8,3,3,-1,6,5,*)] \
  Dose,Dam,Littersz,Sex,Weight; FREPRESENT=labels,(*)2,labels,*
VCOMPONENT [FIXED=Dose*Sex+Littersz; ABSORB=Dam] RANDOM=Dam+Pup
REML Weight
VWALD [FIXED=Dose*Sex+Littersz]
```

We start by inspecting the highest order interaction: sex.dose. The Wald test indicates that this interaction is not important and it is consequently dropped from the model.

```
Fixed Model : Constant +

      (Dose * Sex) + Littersz

W a l d   t e s t s

CHIsq.    df.   Prob.
 0.931      2   0.628  Dose . Sex
```

By default procedure VWALD produces only a test on the highest-order interaction(s); tests for lower-order interactions or main effects are optional. Note that the latter tests are difficult to interpret because main effects or lower-order interactions will be tested in the presence of higher-order interactions. All test results are reproduced below.

```
VWALD [ALLEFFECTS=yes; FIXED=Dose*Sex+Littersz]

CHIsq.    df.   Prob.
 0.931      2   0.628  Dose . Sex
23.669      2   0.000  Dose
31.671      1   0.000  Sex
46.855      1   0.000  Littersz
```

For the reduced model, tests for main effects are produced. Observe that there are (at least) two possibilities here: the components of variance may be fixed to their estimates obtained from the largest model fitted or they may be estimated again for the reduced model. The latter approach is followed in this example.

```
VCOMPONENT [FIXED=Dose+Sex+Littersz; ABSORB=Dam] RANDOM=Dam+Pup
REML Weight
VWALD [FIXED=Dose+Sex+Littersz]

            CHIsq.    df.   Prob.
            23.184      2   0.000   Dose
            57.182      1   0.000   Sex
            47.109      1   0.000   Littersz
```

For the effects of dose and littersize, which are both related to the dams, differences with the optional test results as obtained before are small. For sex, which relates to the individual pups, the test statistic is almost doubled relative to the optional result.

4. Procedure VWALD

The procedure VWALD is in Procedure Library 2[3] (Buist and Engel 1992). At Release 3 of Genstat the Wald test will become available within the REML directives, possibly together with a modified likelihood-ratio test.

```
PROCEDURE 'VWALD'
"-------------------------------------------------------------------------
 Procedure for testing main effects and interactions in an unbalanced mixed
 model (the Wald test) after using REML.

 The Wald test is based on approximate Normality of the estimators for the
 fixed effects. The test uses a chi-square distribution.

 By default VWALD produces only a test on the highest order interaction(s);
 tests for lower order interactions or main effects are optional.

    VWALD expects REML to have been used earlier in the program.
    There is
            no check for REML being used.
            no check for the fixed model definition in VCOMPONENTS.

    You may use the procedure remlWALD for controlling VWALD.

 -------------------------------------------------------------------------
"
OPTION 'FIXED',      " (formula) treatment formula for the design"\
       'ALLEFFECTS', " (string)  yes=tests for lower order interactions
                                    or main effects ;
                       DEFAULT=NO : only highest order interactions"\
       'FACTORIAL'   " (scalar)  limit on the number of factors or variates
                                 in each fixed term ; DEFAULT=3";\
       SET     =yes,(no)2;\
       MODE    =f,t,v;\
       DEFAULT =*,!T(no),3;\
       TYPE    ='formula','text','scalar'
"-------------------------------------------------------------------------
Break the formula for fixed-effects (FIXED) up into separate formulae
(one for each term)
 -------------------------------------------------------------------------
"
FCLASSSIFICATION [NTERM=Nt] #FIXED
FCLASSIFICATION #FIXED ; CLASS=Pt[1...Nt] ; OUTTERM=Ft[1...Nt]
FOR PT=Pt[1...Nt] ; NTlev=Nlev[1...Nt] ; df=Df[1...Nt] ; inter=Nint[1...Nt]
  CALC NTlev,df=1
  GETATTRIBUTE [ATTRIBUTE=type] #PT ; SAVE=Ptype  "Get type of term S"
  IF Ptype[] == 4
    CALC inter=1                                  "Type 4 : Covariate"
  ELSIF Ptype[] == 2
    CALC inter=NVALUES(PT)                        "Type 2 : Factor"
    IF inter <= FACTORIAL
      FOR i=1...inter
        CALC Nl=NLEVELS(PT[i])            "inter =1 : main-effect"
         &   NTlev=NTlev*Nl               "      =2 : interaction (a.b)"
         &   df=df*(Nl-1)                 "      =3 : interaction (a.b.c)"
      ENDFOR
```

```
    ELSE
      CALC inter,NTlev=0,0
    ENDIF
  ELSE
    PRINT 'Incorrect definition FIXED-effects'
    EXIT [CONTROL=Procedure]                    "Wrong type"
  ENDIF
ENDFOR
CALC NTfix=VSUM(Nlev)                           "Total number of levels"
CALC maxINT=VMAX(Nint)                          "Highest order interaction"
"
 ------------------------------------------------------------------------
  Store FULL variance/covariance matrix into VC.
  Constant term in model ?
  Wald tests for all fixed terms ?
 ------------------------------------------------------------------------
"
VKEEP [FULLVCOV=VC]

GETATTRIBUTE [ATTRIBUTE=row] VC ; Prow
CASE begin=!t('Constant') .in. Prow[]
  PRINT [IPRINT=*] !T('Fixed Model :'),!T('Constant +'),FIXED
  CALC NTfix=NTfix+1
ELSE
  PRINT [IPRINT=*] !T('Fixed Model :'),FIXED
ENDCASE

SCALAR CHIsq,pCHI,nb,ne,v[1...NTfix],minINT ; VALUE=(0)4,(0)#NTfix,1
POINTER PvFIXED ; VALUES=!P(v[1...NTfix])
SYMMETRIC [ROWS=PvFIXED] Vfixed
EQUATE VC ; Vfixed
DELETE [REDEFINE=yes] VC

CASE ALLEFFECTS .in. !t('Y','y','yes','YES')
  CALC minINT=1
ELSE
  CALC minINT=maxINT
ENDCASE
"
 ------------------------------------------------------------------------
   VWALD tests
"
PRINT [SERIAL=yes] !T('W a l d   t e s t s'),\
                   !T('  CHIsq.    df.  Prob.') ; FIELDWIDTH=20,22
FOR kINT=maxINT...minINT
  CALC ne=begin                                 "Skip CONSTANT"
  FOR FT=Ft[1...Nt] ; NTlev=Nlev[1...Nt] ; df=Df[1...Nt] ; inter=Nint[1...Nt]
      CALC nb=ne+1
      & ne=nb+NTlev-1
      IF inter == kINT
                                                "Current effect is m, say"
          VKEEP TERMS=#FT; EFFECT=ef_B          "Save table of effects"
          MATRIX [ROWS=NTlev; COLUMNS=1] b      "and change type from"
          EQUATE ef_B ; b                       "table to matrix."

          POINTER PvarB ; VALUES=!P(v[nb...ne])  "Extract part from"
          SYMM [ROWS=PvarB] varB                "Variance/Covar-matrix"
          CALC varB=SUBMAT(Vfixed)              "corresponding to m"
                                                "CHIsq="
          & CHIsq=LTPROD(b; PROD(INV(varB); b))  "  TRANSP(b)"
          & pCHI=1-CHISQ(CHIsq; df)             "       *INVERSE(var)*b"
          PRINT [SQUASH=yes; SERIAL=yes; IPRINT=*;\
            ORIENT=across] CHIsq,df,pCHI,' ',FT ;\
            SKIP=(1)5 ; FIELDWIDTH=8,6,7,1,10 ;\
            DECIMALS=3,0,3,(0)2
          DELETE [REDEFINE=yes] PvarB,varB,b,ef_B
      ENDIF
  ENDFOR
ENDFOR
ENDPROCEDURE
```

5. Procedure remlWALD

An alternative way for calling VWALD is via the procedure remlWALD. Instead of

```
VCOMPONENT [FIXED=Dose+Sex+Littersz; ABSORB=Dam] RANDOM=Dam+Pup
REML Weight
VWALD [FIXED=Dose+Sex+Littersz]
```

you may use

```
remlWALD [FIXED=Dose+Sex+Littersz; ABSORB=Dam; RANDOM=Dam+Pup] Weight
```

The procedure remlWALD is available, by e-mail or on floppy disk, from the first author.

```
PROCEDURE 'remlWALD'
"-------------------------------------------------------------------------

        Procedure for controlling the VWALD procedure

    directives used :
             VCOMPONENTS for model definition
             REML for unbalanced mixed model analysis
             VWALD for tests for interactions or main effects (Wald tests)
-------------------------------------------------------------------------
"
OPTION 'FIXED' ,     " (formula) treatment formula for the design        "\
       'RANDOM',     " (formula) block formula for the design            "\
       'ABSORB',     " (factor)  the absorbing factor                    "\
       'CONSTANT',   " (string)  how to treat the constant term
                                      (estimate, omit) ; def=e          "\
       'MVINCLUDE',  " (string)  include units with missing values ;
                                                              def=no"\
       'ALLEFFECTS', " (string)  yes=tests for lower order interactions
                                     or main effects  ; def=no          "\
       'MAXCYCLE',   " (scalar)  limit on the number of iterations;
                                                              def=10"\
       'FACTORIAL',  " (scalar)  limit on the number of factors or
                            covariates in each fixed term ; default=3   "\
       'INITIAL',    " (scalars) initial values for each var.-component  "\
       'CONSTRAINTS'," (strings) how to constrain each var.-component    "\
       'TOLERANCES'  " (variate) tolerances for matrix inversion         ";\
       SET     =yes,(no)10;\
       MODE    =(f)2,p,(t)3,(v)3,t,p;\
       DEFAULT =(*)3,!T(e),!T(no),!T(no),10,3,1,!T(p),*;\
       TYPE    =('formula')2,'factor',('text')3,('scalar')3,'text','variate'

PARAMETER 'Y',       " (pointer) the variate to be analysed " \
          'WEIGHTS'  " (variate) a weight for each unit in the analysis";\
       SET     =yes,no;\
       MODE    =p,p;\
       TYPE    ='variate','variate'
"-------------------------------------------------------------------------"
GET [ENVIRONMENT=Env]
SET [OUTPRINT=dots]
PAGE
PRINT [IPRINT=*] !t(' Response variate ='),!p(Y)
"-------------------------------------------------------------------------
  VCOMPONENTS , REML and VWALD
"
IF .NOT. UNSET(RANDOM)
  IF .NOT. UNSET(ABSORB)
    VCOMPONENTS [FIXED=#FIXED; ABSORB=ABSORB; CONSTANT=#CONSTANT] \
      RANDOM=#RANDOM ; INITIAL=#INITIAL ; CONSTRAINTS=#CONSTRAINTS
  ELSE
    VCOMPONENTS [FIXED=#FIXED; CONSTANT=#CONSTANT] \
      RANDOM=#RANDOM ; INITIAL=#INITIAL ; CONSTRAINTS=#CONSTRAINTS
  ENDIF
ELSE
  IF .NOT. UNSET(ABSORB)
    VCOMPONENTS [FIXED=#FIXED; ABSORB=ABSORB; CONSTANT=#CONSTANT] \
      INITIAL=#INITIAL ; CONSTRAINTS=#CONSTRAINTS
  ELSE
```

```
    VCOMPONENTS [FIXED=#FIXED; CONSTANT=#CONSTANT] \
      INITIAL=#INITIAL ; CONSTRAINTS=#CONSTRAINTS
  ENDIF
ENDIF

IF UNSET(TOLERANCE)
  IF UNSET(WEIGHTS)
    REML [PRINT=monitor,components; MVINCLUDE=#MVINCLUDE; \
      FACTORIAL=FACTORIAL; MAXCYCLE=MAXCYCLE] Y
  ELSE
    REML [PRINT=monitor,components; MVINCLUDE=#MVINCLUDE; \
      FACTORIAL=FACTORIAL; MAXCYCLE=MAXCYCLE; WEIGHTS=WEIGHTS] Y
  ENDIF
ELSE
  IF UNSET(WEIGHTS)
    REML [PRINT=monitor,components; MVINCLUDE=#MVINCLUDE; \
      FACTORIAL=FACTORIAL; MAXCYCLE=MAXCYCLE; TOLERANCE=TOLERANCE] Y
  ELSE
    REML [PRINT=monitor,components; MVINCLUDE=#MVINCLUDE; \
      FACTORIAL=FACTORIAL; MAXCYCLE=MAXCYCLE; WEIGHTS=WEIGHTS; \
      TOLERANCE=TOLERANCE] Y
  ENDIF
ENDIF

VWALD [ALLEFFECTS=#ALLEFFECTS; FACTORIAL=FACTORIAL] #FIXED

SET [OUTPRINT=#Env['outprint']]

ENDPROCEDURE
```

References

Buist W and Engel B (1992) Procedure VWALD Genstat 5 *Procedure Library Manual Release 2[3].*

Dempster A P, Selwyn M R, Patel C M and Roth A J (1984) Statistical and computational aspects of mixed model analysis *Appl. Statist.* **33** 203–214.

Engel B (1990) The analysis of unbalanced linear models with variance components *Statistica Neerlandica* **44** 195–219.

Giesbrecht F G and Burns J C (1985) Two-stage analysis based on a mixed model: large sample asymptotic theory and small-sample simulation results *Biometrics* **41** 477–486.

Finite Mixture Distributions

C J Whitaker
Centre for Applied Statistics
School of Mathematics
University College of North Wales
Dean Street
Bangor
Gwynedd
United Kingdom LL57 1UT

1. Introduction

Mixtures of distributions have been fitted to data for over 100 years. The typical problem is that observations are available on units each of which may belong to one of a number of populations; the aim is to predict which population each unit belongs to, using only the available observations. Statistical analysis of such data usually requires numerical methods, because explicit estimators of the model parameters do not usually exist. In the method that follows only univariate observations will be used.

2. Mixtures of Normal distributions

Consider the following probability model for data. The observations $x_i, i = 1...n$, are independent and identically distributed observations from a super population G which is a mixture of K populations $G_1...G_K$ in some proportions $\pi_1...\pi_K$. The density function for x_i can be written as

$$f(x_i) = \pi_1 f_1(x_1) + ... + \pi_k f_k(x_i)$$

where $f_j(x_i)$ is the probability density function corresponding to population G_j. Here the density function will be taken to have the specific form

$$f_j(x_i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x_i-\mu_i)^2/2\sigma^2}.$$

Aitken and Tunnicliffe Wilson (1980) showed how the maximum-likelihood estimates of the parameters μ_j, σ_j and π_j, $j = 1...K$, are the solution of the likelihood equations

$$\hat{\pi}_j = \sum_i \hat{P}(j/x_i)\Big/n \tag{1}$$

$$\hat{\mu}_j = \sum_i x_i \hat{P}(j/x_i)\Big/\sum_i \hat{P}(j/x_i) \tag{2}$$

$$\hat{\sigma}_j^2 = \sum_i (x_i - \mu_j)^2 \hat{P}(j/x_i)\Big/\sum_i \hat{P}(j/x_i) \tag{3}$$

where $\hat{P}(j/x_i) = \hat{\pi}_j f_j(x_i)/\sum_j \hat{\pi}_j f_j(xi)$ and $\hat{P}(j/x_i)$ are the ML estimators of the posterior probability that the ith observation comes from the jth population. This forms a simple iteration scheme. From the $\hat{P}(j/x_i)$ the estimates of π_j, μ_j and σ_j^2 can be calculated easily; then from the estimates of π_j, μ_j and σ_j^2 the estimates of $\hat{P}(j/x_i)$ can be calculated. This has been shown to be an example of the EM algorithm (Dempster *et al* 1977). Initial values for $\hat{P}(j/x_i)$ can be made by splitting the range of the x_i into j non-overlapping intervals and assigning the value 1 to $\hat{P}(j/x_i)$ if x_i falls in the jth interval and 0 otherwise. If the populations are reasonably distinct then convergence is acceptably quick. This algorithm has been implemented in a Genstat procedure, with a limit of up to three populations, and the variances can be optionally constrained to be the same for each population. In this case equation (3) is replaced by

$$\hat{\sigma}^2 = \sum_j \sum_i (x_i - \mu_j)^2 \hat{P}(j/x_i)/n.$$

The observations are stored in a variate and supplied using the DATA parameter. The number of populations is also needed, supplied using the NMIXTURE parameter. If only one population is being fitted then this is all that is required to use the procedure.

For the more usual situation where a mixture of two or three populations are to be fitted, the INITIAL parameter needs to be set. Having 'eyeballed' the distribution of the observations, a 'guesstimate' is made as to where the modes of the distribution are. Then the INITIAL parameter can be estimated as halfway between the modes.

The remaining PROB parameter allows the estimated posterior probability ($\hat{P}(j/i)$) to be saved in a pointer if desired.

The options allow the iterations to be monitored (TRACE), whether you wish to have equal or unequal variance estimates (EQUAL), how many iterations should be used (ITERATIONS), and how to stop the iteration scheme (ACCURACY). By default the iterations will stop when the change in deviance (= $-2\times$log-likelihood) is less than 0.001 multiplied by the deviance.

The output consists of the estimated means and variances of the model together with the deviance. The difference in deviances between fitting, say, one and two populations can be used as an indication of which model is to be preferred, but the large sample χ^2 test with d.f. defined as the difference in the number of parameters is not valid. Aitken and Rubin (1985) describe an approach to estimation as well as testing of finite mixture models.

The procedure MIXTURE checks the options and parameters for consistency and accuracy and then uses a second procedure MIXITER which performs the numerical calculations.

3. Visual assessment of the mixture

In the practical situation where a theory suggests that some data may have arisen from a mixture of two or three populations, a natural first step is to draw a graph to assess visually the potential validity of the theory. The obvious candidate is a histogram. However if the data do not show clear multimodality then the choice of origin and number of classes can lead to different visual impressions which leads even experienced data analysts astray.

Kernel estimates of the density function, as suggested by Silverman (1986), can be used to present and explore data while overcoming some of the problems associated with the histogram. Basically this technique produces a smoothed version of the histogram. One advantage this technique has is that the smooth curve produced leaves the eye to concentrate on the overall shape, in contrast to a histogram where haphazard peaks and troughs can distract the eye. The estimate of the density function $\hat{f}(x)$ with kernel K for data $x_1 \ldots x_n$ is defined as

$$\hat{f}(x) = (1/nh) \sum_i K\left((x - x_i)/h\right)$$

where h is the window width or smoothing parameter. If the kernel K is a probability density function then $\hat{f}$ will be a probability density. A kernel that is often used in a variety of situations is the Normal or Gaussian kernel (see Silverman (1986) for details), as it performs reasonably well under a wide range of circumstances. Of course the window width, h, needs to be determined and does affect the visual impression of the resulting density. A value for h can be chosen automatically from the data to provide reasonable density estimates for a wide range of types of distribution for the observed data. Silverman gives the value as

$$h = 0.9\,(\min(\text{sd},\ \text{interquartile range}/1.34))\,n^{-1/5}.$$

A Genstat procedure has been written which uses this as the default window width but which allows the user to set an alternative. The procedure works by finding the minimum and maximum values for the data and then extending the plotting range by three times the window width. The multiplier three is used because the density function of a standard Normal distribution is essentially zero beyond three standard deviations from the mean. By default, 100 grid points are then equipositioned over the plotting range. For each grid point the densities from the Normal distributions placed on each data point are summed to give the density estimate of the observations. Then a smooth curve is plotted through these density estimates.

A different approach for graphical representation of data is to use a quantile quantile (Q–Q) plot. This plots the quantiles of the data against the quantiles of a Normal distribution. If the data are Normally distributed, a straight line should be observed. If the data are from a mixture of two Normals, two straight lines joined by an S-shaped curve should be observed. With a mixture of three Normals, a double S-shaped curve should be observed. This approach works well when there is relatively large separation between the populations, but with relatively little separation, information about the possible mixtures occurs mainly in the middle of the plot.

To try to overcome this, Fowlkes (1979) suggested a variation of the percentile–percentile (P–P) plot. Taking the quantiles of the data to be $x_{(1)} < x_{(2)} < \ldots < x_{(n)}$ and letting $p_i = (i - \frac{1}{2})/n$ be the sample probabilities, then plot

$$\Phi((x_{(i)} - \bar{x})/s) - p_i \quad \text{against } (x_{(i)} - \bar{x})/s$$

where $\bar{x}$ and s are the mean and standard deviation of the data and Φ is the standard Normal distribution function. Fowlkes termed this a $\Phi - p$ versus Q plot. When there is only a single Normal distribution, the plot should be a horizontal straight line. When there is a mixture of Normals, there should be a pronounced peak or trough in the plot, or both, depending on the proportions of the mixtures. Its use was suggested by the observation that P–P plots are sensitive to departures from the assumption of there being only one population in the middle values.

Two procedures are shown which use these techniques. **DENSITY** will produce density estimates and requires only a variate containing the data. A **WINDOW** parameter allows the user to specify a value for h, as defined above; by default, the value suggested by Silverman is used. One hundred points throughout the range of the observations are used for plotting although **NGRIDPTS** can be changed. The procedure **PLOTMIXTURE** will produce the Q–Q and $\Phi - p$ versus Q plots; it requires just a variate containing the data to be supplied. Shown below are the use of these procedures for simulated data using the data generation facilities of Genstat.

The Genstat output showing the density estimates does show the expected shape for one or two Normal distributions. However the mixture of three Normals does not clearly show the third peak: only a flattening of the curve is seen. The Normal plot for a single Normal does look reasonably straight, and the plot for two Normals does suggest an S-shaped curve. However the plot for three Normals does not look like a double S-shaped curve: a single S-shaped curve is all that is suggested. The $\Phi - p$ versus Q plot for a single Normal does produce a moderately flat line, but only when compared to the plot for the mixture of two Normals where the trough is much more pronounced. However the plot for three Normals is not markedly different from the mixture of two Normals.

This emphasises the problem with mixtures of Normal distributions: their identification is difficult, and even if you can be fairly confident that a single Normal is an inappropriate model for the data it is not always clear which model is most appropriate. This problem is possibly most clearly demonstrated by the observation that a mixture of Normals with decreasing proportions as the mean increases often looks like a lognormal distribution. Considering only the Normal plots shown here, it would be reasonable to conclude that a lognormal distribution is possibly an acceptable model.

4. Conclusion

Mixture models have a long history and no doubt will continue to be used as models for data, but the identification of the models can be problematic in practice. I suggest that mixtures of Normals with both relatively large and small separation be simulated and then all three types of plot examined before using these techniques in the analysis of your own data.

5. References

Aitken M and Rubin D B (1985) Estimation and hypothesis testing in finite mixture models. *J.R. Statist. Soc. B* **47** (1) 67–75.

Aitken M and Tunnicliffe Wilson G (1980) Mixture models, outliers and the EM algorithm. *Technometrics* **22** (3) 325–331.

Fowlkes E B (1979) Some methods for studying the mixture of two normal (lognormal) distributions. *J. Amer. Statist Ass.* **74** 561–575.

Silverman B (1986) *Density estimation for statistics and data analysis.* New York: Chapman and Hall.

Appendix 1: The procedures

[Editors' note: The procedures DENSITY and PLOTMIXTURE have been modified to produce high resolution graphics; comments give lineprinter version]

```
PROCEDURE 'MIXITER'
OPTION NAME='STRACE','SEQUAL','ITERATION','ACCURACY' ; MODE=P
PARAMETER NAME='DATA','PROB','NMIXTURE'; MODE=P
SCALAR N,NMISS,COUNTER,DEVIANCE,PREVIOUS,ACC
SCALAR P[1...NMIXTURE],MU[1...NMIXTURE],SIGMA2_[1...NMIXTURE]
VARIATE F[1...NMIXTURE],VDUM[1...NMIXTURE],SUMF
CALC ROOT2PI = SQRT(2*3.1415926536)
CALC COUNTER = 0
CALC N = NVAL(DATA) - ( NMISS = NMV(DATA) )
FOR  [NTIMES=ITERATION]
     CALC P[] = SUM( PROB[] ) / N
     CALC MU[] = SUM( DATA*PROB[] ) / ( SUM( PROB[] ))
     CALC SIGMA2_[] = SUM( ((DATA-MU[])**2)*PROB[] )
     IF SEQUAL .EQ. 1
          CALC SIGMA2_[] = SIGMA2_[] / N
          CALC SIGMA2 = VSUM(SIGMA2_)
          CALC F[] = (1/(ROOT2PI*SQRT(SIGMA2)))*EXP(-((DATA-MU[])**2) \
            /(2*SIGMA2))
     ELSE
          CALC SIGMA2_[] = SIGMA2_[] / ( SUM( PROB[] ))
          CALC F[] = (1/(ROOT2PI*SQRT(SIGMA2_[])))*EXP(-((DATA-MU[])**2) \
            /(2*SIGMA2_[]))
     ENDIF
     CALC VDUM[] = P[]*F[]
     CALC SUMF = VSUM( VDUM )
     CALC DEVIANCE = -2*SUM( LOG( ROOT2PI*SUMF ) )
     CALC PROB[] = ( P[]*F[] ) / SUMF
     IF STRACE .EQ.1
          IF SEQUAL .EQ. 1
               PRINT MU[],SIGMA2,P[],DEVIANCE
          ELSE
               PRINT MU[],SIGMA2_[],P[],DEVIANCE
          ENDIF
     ENDIF
     CALC ACC = DEVIANCE*ACCURACY
     EXIT ABS(PREVIOUS - DEVIANCE) .LT. ACC
     CALC PREVIOUS = DEVIANCE
     CALC COUNTER = COUNTER + 1
     IF COUNTER .EQ. ITERATION
          PRINT '***** FAILED TO CONVERGE'
     ENDIF
ENDFOR
IF STRACE .EQ.0
     IF SEQUAL .EQ. 1
          PRINT MU[],SIGMA2,P[],DEVIANCE
     ELSE
          PRINT MU[],SIGMA2_[],P[],DEVIANCE
     ENDIF
ENDIF
ENDPROCEDURE

PROCEDURE 'MIXTURE'
OPTION NAME='TRACE','EQUAL','ITERATIONS','ACCURACY'; \
  MODE=T,T,P,P; DEFAULT='NO','YES',20,0.001
PARAMETER NAME='DATA','NMIXTURE','INITIAL','PROB'; MODE=P
FOR SETTING=TRACE,EQUAL; SSET=STRACE,SEQUAL
     IF SETTING .IN. !T(YES,YEs,Yes,yes,YE,Ye,yE,ye,Y,y)
          CALC SSET = 1
     ELSIF SETTING .IN. !T(NO,No,nO,no,N,n)
          CALC SSET = 0
     ELSE
          PRINT '***** INVALID OPTION SETTINGS FOR TRACE AND/OR EQUAL'
          EXIT [CONTROL = P]
     ENDIF
ENDFOR
```

```
IF ITERATIONS .LT. 2
     PRINT '***** ITERATIONS MUST BE AT LEAST 2'
     EXIT [CONTROL = P]
ENDIF
IF ACCURACY .LE. 0
     PRINT '***** ACCURACY MUST BE A POSITIVE VALUE'
     EXIT [CONTROL = P]
ENDIF
IF UNSET(DATA)
     PRINT '***** DATA PARAMETER MUST BE SUPPLIED'
     EXIT [CONTROL = P]
ELSE
     GETATTRIBUTE [TYPE] DATA ; TYPEDATA
     IF TYPEDATA[1] .NE. 4
          PRINT '***** DATA MUST BE A VARIATE'
          EXIT [CONTROL = P]
     ENDIF
ENDIF
IF (NMIXTURE .LT. 1) .OR. (NMIXTURE .GT. 3)
     PRINT '***** ONLY 1, 2 OR 3 MIXTURES CAN BE FITTED'
     EXIT [CONTROL = P]
ENDIF
IF UNSET(INITIAL)
     CALC NINITIAL = 0
ELSE
     CALC NINITIAL = NVALUES(INITIAL)
ENDIF
CALC SETTINGS =  ( (NMIXTURE .EQ. 1) .AND. (NINITIAL .EQ. 0) ) \
  .OR. ( (NMIXTURE .EQ. 2) .AND. (NINITIAL .EQ. 1) ) \
  .OR. ( (NMIXTURE .EQ. 3) .AND. (NINITIAL .EQ. 2) )
IF SETTINGS .EQ. 0
     PRINT '***** INCOMPATIBLE NMIXTURE AND INITIAL PARAMETERS'
     EXIT [CONTROL = P]
ENDIF
IF UNSET(PROB)
     POINTER UPROB
     ASSIGN UPROB ; PROB
ENDIF
IF NMIXTURE .EQ. 1
     SCALAR MU,SIGMA2,DEVIANCE
     CALC MU = MEAN(DATA)
     CALC SIGMA2 = VARIANCE(DATA)
     CALC DEVIANCE = -2*LLNORMAL(DATA;MU;SIGMA2)
     PRINT MU,SIGMA2,DEVIANCE
ELSIF NMIXTURE .EQ. 2
     CALC PROB[1] = DATA .LE. INITIAL
     CALC PROB[2] = DATA .GT. INITIAL
     MIXITER [STRACE; SEQUAL; ITERATION; ACCURACY] DATA; PROB; NMIXTURE
ELSE
     CALC PROB[1] = DATA .LE. MIN(INITIAL)
     CALC PROB[2] = ( DATA .GT. MIN(INITIAL) ) .AND. ( DATA .LE. MAX(INITIAL) )
     CALC PROB[3] = DATA .GT. MAX(INITIAL)
     MIXITER [STRACE; SEQUAL; ITERATION; ACCURACY] DATA; PROB; NMIXTURE
ENDIF
ENDPROCEDURE
PROCEDURE 'DENSITY'
OPTION NAME='NGRIDPTS'; MODE=P; DEFAULT=100
PARAMETER NAME='DATA','WINDOW'; MODE=P
SCALAR A,B,DELTA,ADELTA,N,NG1,NMISS,H
CALC N = (NOBS = NVALUES(DATA)) - NMV(DATA)
IF UNSET(WINDOW)
     CALC S[1] = SQRT( VARIANCE(DATA) )
     QUANTILE [PRINT=*; PROP=!(0.25,0.75)] DATA; H_SPREAD
     CALC S[2] = ( MAX(H_SPREAD) - MIN(H_SPREAD) ) / 1.349
     CALC H = 0.9*VMINIMA( S ) / N**(0.2)
ELSE
     CALC H = WINDOW
ENDIF
CALC A = MIN(DATA) - 3*H
CALC B = MAX(DATA) + 3*H
```

```
CALC NG1 = NGRIDPTS - 1
CALC DELTA = (B-A) / NG1
CALC ADELTA = A + DELTA
VARIATE [VALUES=A,ADELTA...B] XVALUES
VARIATE [NVALUES=NGRIDPTS] FK
FOR  I=1...NGRIDPTS
     CALC DUM = EXP( -(((DATA-ELEM(XVALUES;I))/H)**2) / 2 ) \
       / SQRT(2*3.1415926536)
     CALC ELEM(FK;I) = SUM( DUM ) / (N*H)
ENDFOR
PRINT '***** WINDOW = ',H
"
For lineprinter graphics use the following line:
GRAPH [TITLE='density plot'] FK; XVALUES; METHOD=CURVE
else for high resolution graphics use:
"
PEN 1; LINESTYLE=1; METHOD = open; SYMBOLS = 0
DGRAPH [TITLE='density plot'] FK; XVALUES; PEN = 1
ENDPROCEDURE

PROCEDURE 'PLOTMIXTURE'
PARAMETER NAME='DATA'
CALC NMV = NVAL(DATA) - ( NOBS = NOBSERVATIONS(DATA) )
VARIATE QUANT,YY; VALUES=!(#NMV(*),(1...NOBS)),DATA
SORT YY
CALC PI = (QUANT-0.375) / (NOBS+0.25)
CALC QUANT = NED(PI)
CALC STANYY = (YY - MEAN(YY))/SQRT(VAR(YY))
CALC PHI_P = NORMAL( STANYY ) - PI
"
For lineprinter graphics use the following lines:
GRAPH [TITLE='normal plot'] YY; QUANT; METHOD=C
GRAPH [TITLE='Phi - p versus Q plot'; YLOWER=-0.1; YUPPER=0.1] \
  PHI_P; STANYY; METHOD=CURVE
else for high resolution graphics use:
"
PEN 1; LINESTYLE=1; METHOD = open; SYMBOLS = 0
DGRAPH [TITLE='normal plot'] YY; QUANT; PEN = 1
AXES 1 ; YLOWER=-0.1; YUPPER=0.1
DGRAPH [TITLE='Phi - p versus Q plot'] PHI_P; STANYY; PEN = 1
AXES 1 ; YLOWER=*; YUPPER=*
ENDPROCEDURE
```

Appendix 2: An example

The following Genstat directives produced the plots shown below.

```
"
generate 500 observations from a normal distribution with mean = 10
and sd = 3
"
VARIATE [NVALUES=500] ONE,TWO,THREE
GRNORMAL [ 500 ; 10  ; 3 ; SEED = 140586] ONE
DHISTOGRAM ONE
DENSITY ONE
PLOTMIXTURE ONE

"
generate 300 observations from a normal distribution with mean = 7
and sd = 3 and 200 observations from a normal distribution with mean = 16
and sd = 3
"

GRNORMAL [ 300 ; 7 ; 3 ] X[1]
&        [ 200 ; 16 ; 3 ] X[2]
EQUATE X ; TWO
DHISTOGRAM TWO
DENSITY  TWO
PLOTMIXTURE  TWO
```

```
"
generate 250, 150 and 100 observations from normal distributions with
mean = 7, 16, 24 and sd = 3, 3 and 3
"
GRNORMAL [ 250 ; 7 ; 3 ] Y[1]
&        [ 150 ; 16 ; 3 ] Y[2]
&        [ 100 ; 24 ; 3 ] Y[3]
EQUATE Y ; THREE
DHISTOGRAM THREE
DENSITY  THREE
PLOTMIXTURE  THREE
```

Appendix 3: The plots

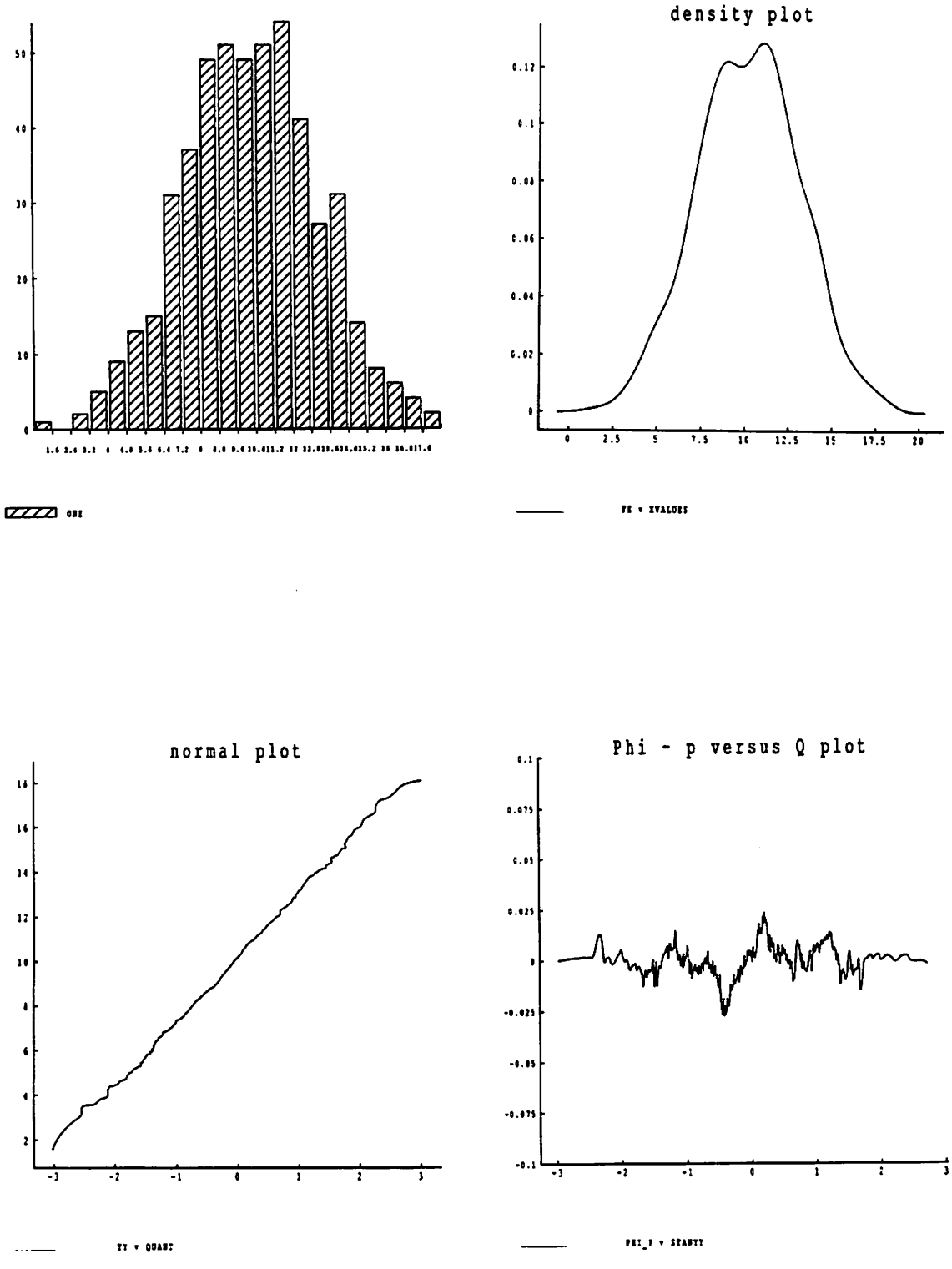
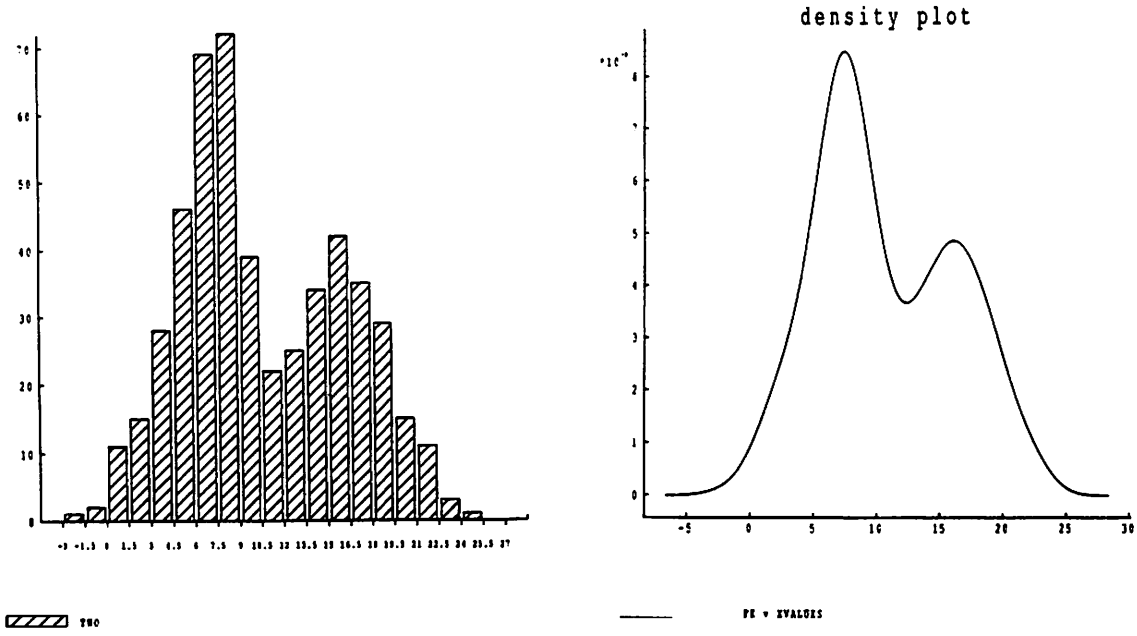


Figure 1
Plots for single Normal



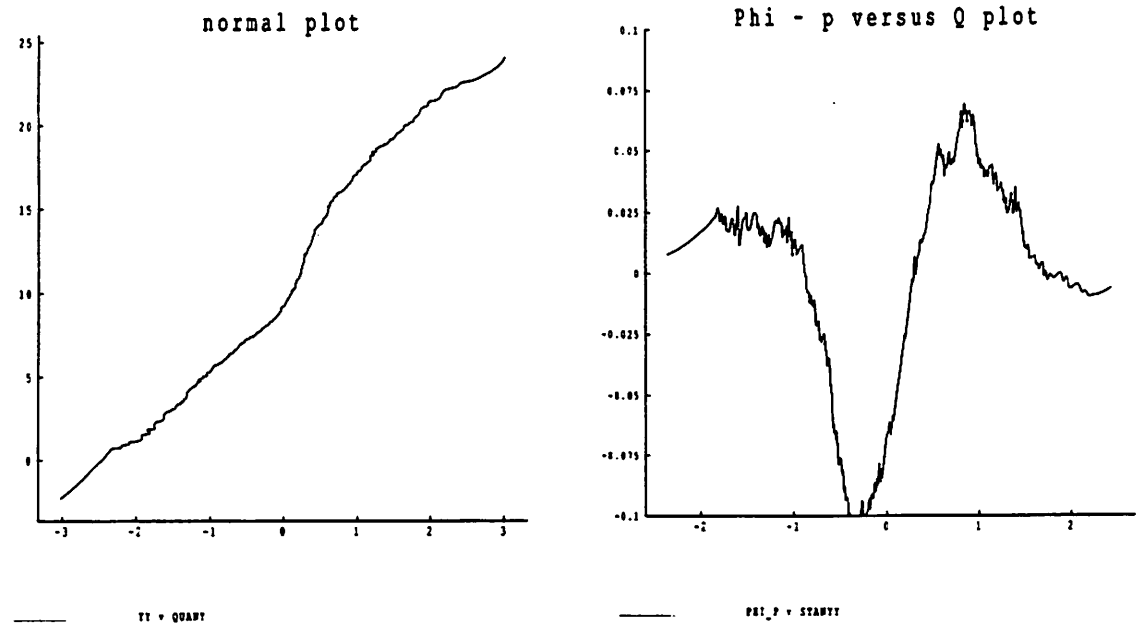


Figure 2
Plots for two Normal mixture

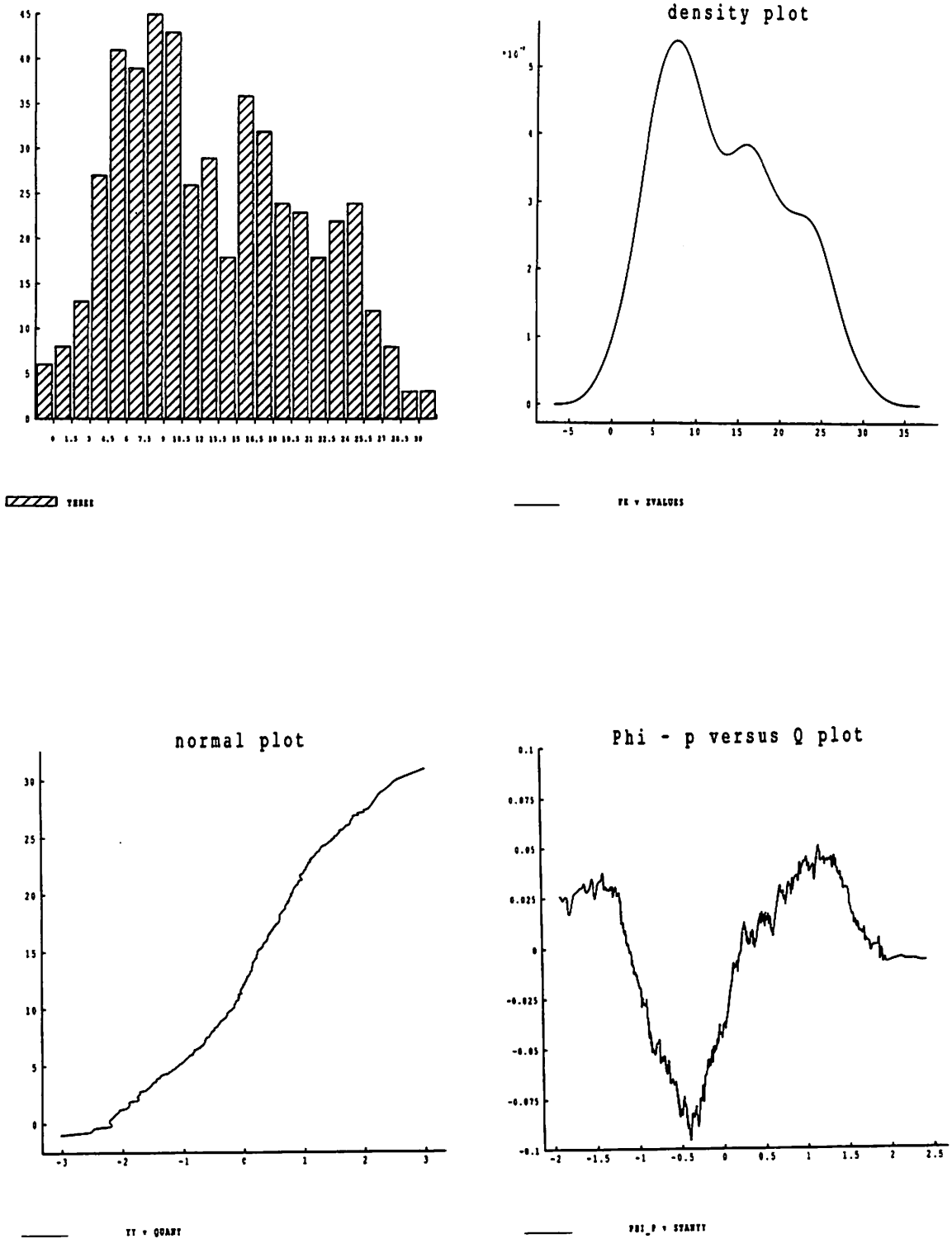


Figure 3
Plots for three Normal mixture

Patterns of Variability on Terraces in Nepal

Rosie Poultney
AFRC Institute of Arable Crops Research
Rothamsted Experimental Station
Harpenden, Herts, United Kingdom, AL5 2JQ
email: POULTNEYR@UK.AC.AFRC.RESA

In large areas of Nepal, terracing is necessary to provide agricultural land. The conditions on adjacent terraces can be quite different, especially in the steeper upland areas; as a result, field experiments on Nepalese terraces are subject to high levels of variability. Terraces can vary in size, shape, orientation, altitude, gradient, farming practices, soil type, moisture level, age, etc. In addition, within each terrace there may be differences in soil type, width, orientation, manure application, moisture, gradient and so on.

For an initial investigation of the patterns of variability between and within terraces, data were collected from six terraces. Maize and millet were intercropped on each terrace and the two crops harvested in $1m^2$ plots. The terraces were of different sizes, and, in addition, the width varied within terrace. The assumption was made that the front edge of the terrace was straight, and the data were stored as a rectangular matrix with the first column being the front edge. For the narrow sections of the terrace, the final columns were filled in with missing values.

Figure 1a shows a shaded plot of the matrix of millet yields from one terrace, with the darker squares indicating higher values. The squares on the right-hand side that are blank indicate the rear of the terrace, and not areas of very low yield. The code to produce the shaded graph is shown below; it is simple and could easily be made into a procedure.

Figure 1b shows the same data after smoothing by taking a weighted average of each value and the surrounding eight points. The code for the procedure SMOOTH which was used to do this is given below.

The shading tends to vary from one plotting device to another. These particular brush settings were very good on a Digital LJ250 inkjet plotter, but were less useful on an Apple Laserwriter II, as there was negligible difference between the two darkest shadings. However the brush settings are very easy to alter.

I am grateful to Simon Harding for his help in writing these programs. This work was carried out under funding by the UK Overseas Development Administration.

Appendix 1. Program to draw a shaded plot

```
" The data file contains the number of rows (n) and columns (p)
  of the data matrix, a general title and the data matrix."
OPEN 'original.dat'; CHANNEL=2
SCALAR n,p
TEXT [NVALUES=1] title
READ [CHANNEL=2] n,p
& title
MATRIX [ROWS=n; COLUMNS=p] mat
READ [CHANNEL=2] mat
CLOSE 2

" Define coordinates for boxes."
CALC np = n*p
POINTER [NVALUES=np] boxx,boxy
CALC boxx[] = !(0,0,1,1) + (1...p)#n - 1
CALC boxy[] = !(0,1,1,0) + #p(1...n) - 1

" Set the thresholds for the shadings, and use them to define which plot
  is to be shaded with which density."
VARIATE [VALUES=125,150...275] thresh
VARIATE vmat; VALUES=mat
SORT [INDEX=vmat; GROUP=gpen; LIMIT=thresh]
CALC vgpen = MVREPLACE(gpen; 9)

" Specify axes, define pens, set the frame size and draw the picture."
AXES 3; YMARK=!(1...n); XMARK=!(1...p); YLOWER=0; YUPPER=n; \
   XLOWER=0; XUPPER=p; STYLE=grid
```

```
PEN 1...9; COLOUR=8(1),0; METHOD=fill; BRUSH=1,9,10...13,15,16,1; SYMBOL=0
CALC xup = p/n
FRAME 3; YLOWER=0; YUPPER=1; XLOWER=0; XUPPER=xup
DGRAPH [WINDOW=3; KEY=0] boxy[]; boxx[]; PEN=#vgpen

" The statements can be repeated in a FOR loop to draw the smoothed version;
  further statements are needed to draw the key."
```

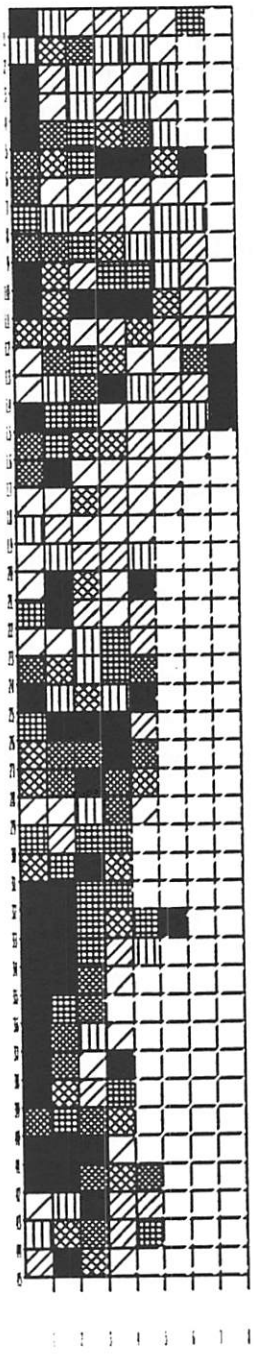


Figure 1a

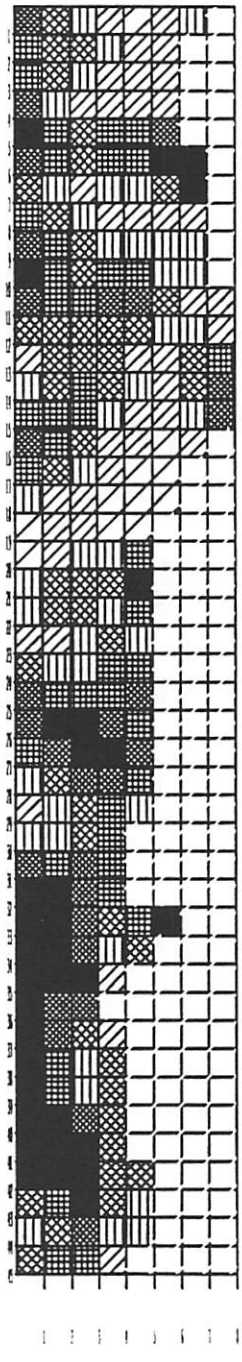
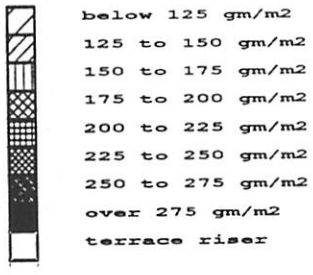


Figure 1b



Appendix 2. Smoothing procedure

```
PROCEDURE 'SMOOTH'
" Procedure SMOOTH calculates a weighted average of each square and the
  surrounding eight squares.  It has one option, WEIGHT, which by default
  is a variate of length 9 with all values set equal to 1, i.e. equal
  weighting.  The position of the entries of WEIGHT relative to the centre
  square 5, is as follows
                          1          2          3
                          4          5          6
                          7          8          9"
OPTION 'WEIGHT'; SET=no; TYPE='variate'; MODE=v; NVAL=9; DEFAULT=!(9(1))
PARAMETER 'IN','OUT'; SET=yes; TYPE='matrix'; DECLARED=y,n; PRESENT=y,n; \
   COMPATIBLE=!T(rows,columns)
" The procedure has two parameters, IN and OUT, which are respectively the
  original and smoothed data."
  CALC n = NROW(IN)
  CALC p = NCOL(IN)
  CALC nv = n*p
  MATRIX [ROWS=n; COLUMNS=p] OUT,X[1...9],N,Q
  VARIATE vin; VALUES=IN
  CALC shifts = p*!(3(1,0,-1)) + !((1,0,-1)3)
  CALC tmp[1...9] = SHIFT(vin; #shifts)
  EQUATE OLD=tmp[]; NEW=X[]
  DELETE tmp[],shifts,vin
  SCALAR mv
  CALC X[1,2,3]$[1;*],X[7,8,9]$[4;*],X[1,4,7]$[*;1],X[3,6,9]$[*;3] = mv
  CALC N,Q,OUT = 0
  CALC 9(N) = N+(X[].ne.mv)*#WEIGHT
  CALC 9(OUT) = OUT+MVREPLACE(X[1...9]; 0)*#WEIGHT
  CALC Q = Q + (IN.ne.mv)
  CALC Q = Q/Q
  CALC OUT = OUT/N * Q
ENDPROCEDURE
```

Customizing the Computing Environment on a 386/486 Machine for Genstat

Simon Heisterkamp
National Institute of Public Health and Environmental Protection (RIVM)
Antonie van Leeuwenhoeklaan 9
3720 BA Bilthoven
The Netherlands

1. Introduction

Running Genstat on personal computers has several advantages as compared to Vax or MicroVax machines. As the computer is dedicated to the user, your program will always run at the same speed, though admittedly in some cases this means the same slow speed. I personally prefer the use of the 386/486 version, because most of the time the turn-around is much faster than with the MicroVax. Another advantage is that you may configure Genstat according to your personal needs—for example, by deleting directives from the bootstrap file or by extensions in Fortran—without being bothered by the system manager. The disadvantage is of course that your DOS-machine can only do one thing at a time, and editing and running a Genstat program at the same time is not possible. In this article I make some suggestions for customizing your machine to get the most out of it.

2. Space requirements

Genstat needs at least 2 Mbytes of RAM memory. As Genstat runs within a special environment, DBOS, which uses its own mapping of the RAM memory, you cannot use any other extended-memory program nor a cache-program. For example, using `HIMEM.SYS` with DOS 5.0 causes a hang-up right after booting. Some users have reported to me an endless booting of the system when using other extended-memory managers. [Editor's Note: this problem has been overcome in DBOS Version 2.60 which is supplied with Genstat Release 2.2.] On the other hand, the main program of Genstat uses only a few kbytes of memory, so memory-resident programs such as the Norton Commander, the Dosshell of DOS 5.0, or a useful tool like Dosedit, and even large Local Area Network programs can run while using Genstat. However, if Genstat runs out of workspace, you can try the job again, albeit with a substantial loss of speed, using a type of disk memory called Virtual Memory Swap Space. Details about swap space are given on Pages 5–6 of the Release 1.3 Installers' Note (see also pages 6–7 of the Release 2.2 Installers' Note). Although creating a special disk partition is more efficient, it can only be done on a empty machine, so in most cases there is no alternative to creating a swap-file.

You should reconfigure the environment of DBOS with the command `CONFIGDB`, following the instructions on Pages 6–7 of the Installers' Note. Make sure that you create the directory where the swap file should reside before you start the reconfiguration.

```
MKDIR swap
```

After that, give the command

```
CONFIGDB
```

You will be asked a number of questions, which can be answered by the suggested default by just hitting the return key until meeting the question `"Paging to disk enabled(y/n) : N"`. Answer this by typing `Y`. The next question asks whether you want Disk or a Partition: answer `D` (for Disk). Next you are asked for the path of the swap file; you need to specify the path and the name of the swap file: e.g. `c:\swap\sweep`. Next you are asked for the amount of swap space required; you need to answer in units of bytes: so 14000000 will give you about 14 Mbyte of diskspace. After confirming your changes the program will then start to format the file: this will take some time! Clearly, despite the name Virtual Swap Space, the amount of space you specify has to be actually available on the hard disk! Rebooting the machine will give you the new options.

Note that actual use of the `SWAP` file may slow Genstat down dramatically. The `S` option in the command line, for extra workspace ($S = 1, 2,$ etc) should be used only when necessary, because if the actual RAM memory is not sufficient to load all the arrays used by Genstat, it will instantly use the `SWAP` file, even if your program would fit. The `S` option tells the operating system to declare a fixed amount of workspace; this static use of memory is due to the use of Fortran as programming language.

3. Using the SUSPEND directive

When using Genstat interactively, the **SUSPEND** directive is very useful. Suppose you prepare a Genstat program in the file **myprog.gen**. This should end with the statement **RETURN**, and it is usually preferable not to start with a **JOB** statement. Then you can execute the statements in the program from within an interactive session as follows:

```
OPEN 'myprog.gen'; CHANNEL=2
INPUT [REWIND=yes] 2
```

If an error occurs, then recover as follows:

```
CLOSE 2
SUSPEND
   { back to dos }
> {editor} myprog.gen
   { make changes, save and quit the editor }
> EXIT
   { back to Genstat }
INPUT [REWIND=yes] 2
   { etc }
```

Note that when using a utility such as Dosedit you can recall all the commands by using the arrow keys, so it would not be necessary to retype the last command. That is why the option **REWIND=yes** was used (otherwise unnecessarily) the first time. [Editor's note: command line recall is included in Release 2.2, so Dosedit is not required.]

Another potential problem with this style of working is that the statements in the pre-prepared program may fail when run for a second time because of the work that was done when they ran the first time. This is particularly likely if you open auxiliary files and do not explicitly close them. However, with a little care in programming, this problem can be avoided.

SUSPEND is also useful when using external programs to communicate with Genstat using files. For example, the statement

```
SUSPEND [SYSTEM='MYEXT < myinp.dat > myoutp.dat']
```

causes DOS to run the program **MYEXT.EXE** with input **MYINP.DAT** (which may just have been created by Genstat) and output **MYOUTP.DAT**. In the next statement, you can open **MYOUTP.DAT** and read the contents. The applications are numerous; however, if you try this in the standard configuration of Genstat, it will fail. To be fair, you are warned about this on Page 8 of the Installers' Note, and instructions are given to allow **SUSPEND** to work. You can make the change by typing a **COMSPACE** command at the DOS command prompt. But to make the change permanent, edit the file **G5SETUP.BAT** in the GENSTAT subdirectory so that the line

```
COMSPACE D30000
```

becomes

```
COMSPACE D300000
```

This will reserve 300kbytes for external programs to run. It depends on your particular program and your particular memory-resident programs how much space in low-memory is needed and/or is free.

4. Extending Genstat

Editor's note: the following section refers to Release 1.3 only; for instructions for Release 2.2 see the Installers' Note. Release 2.2 has space for almost 4 million real data values.

Sometimes you may want to extend Genstat either by defining your own directives or subroutines for **FITNONLINEAR**, or by just claiming more workspace than is allowed by the default configuration even when using the option **S=4**. For the extension with **OWN**, you should read Chapter 12 of the Manual. The creation of a larger Genstat is much simpler to accomplish.

Step 1
Get the correct Salford compiler. The version of the compiler is extremely important, because it must correspond with the version used for compiling Genstat for the 386/486 machine. Unfortunately this was not indicated on the Genstat Release 1.3 we used. The proper version for Release 1.3 of Genstat is Version 1.66 of the Salford compiler; for Release 2.2 it is Version 2.60.

Step 2
Check whether NAG has shipped the graphical routines to you. Our version of Release 1.3 did not contain these, but NAG supplied us with them immediately we asked. The source files are named `GIA.FOR` and the corresponding object files are `GIA.OBJ`.

Step 3
Make a copy of the file `MN.FOR`, from the directory `GENSTAT\ SETUP`, calling it `MN.OLD`, for example. Then edit `MN.FOR` as indicated in the Installers' Note on Page 10. This involves finding the line containing the word `"PARAMETER"`, ignoring any lines that have been commented out with a `"C"` in the first column,

```
PARAMETER (MLRDAT=262144,MRDAT=2*MLRDAT,MIDAT=MRDAT,MLDAT=MRDAT)
```

and increasing the first number. For example, you could multiply the amount of space by 10:

```
PARAMETER (MLRDAT=2621440,MRDAT=2*MLRDAT,MIDAT=MRDAT,MLDAT=MRDAT)
```

Also another statement has to be changed:

```
MAXDAT=655360
```

This number has also been multiplied by 10 in this example. Note that this version of Genstat will allow you 2.6 million real data values! The SWAP space needed with this version, assuming 6 Mbyte RAM memory is about 14Mb.

Step 4
Compile the modified files and link the program. This can be done with a batch file; if you call it `G5COMP.BAT`, then the whole operation can be done by typing:

```
G5COMP mn
```

The batch file `G5COMP.BAT` reads:

```
ECHO Relink Genstat: supply the name of a subroutine
KILL_DBOS
CD c:\ftn77
DBOS
CD c:\genstat\setup
FTN77 %1 /intl /nocheck > error.lst
MORE < error.lst
ECHO Do you want to continue with linking?
PAUSE
LINK77 c:\genstat\setup\g5relink.177
DBOS
```

The Salford compiler is supposed to be in the directory `C:\ FTN77` with the name `FTN77`. An error listing of the compiler will appear in the file `ERROR.LST`. Hopefully there will be no errors, and you can continue with linking. Note that the command `KILL_DBOS` is really needed, as otherwise you will not be allowed to recompile Genstat by DBOS! In the process of linking, you may be alarmed by the screenfulls of warnings, minutes long, about the commons that are misspecified. You need not be concerned: your program will be correct. Even if you do not have the graphical routines from NAG, the new version of Genstat will run, except for the high-resolution graphics. Be aware that neither is the old version of Genstat destroyed, nor is the new version accessed when you type GENSTAT! In fact you have now two different versions of the Genstat run-time library: `GENNEW.LIB` and `GENLIB.LIB` in the `GENSTAT\ LIB` directory.

Step 5
To invoke either of two versions you can use two batch-files:

Normal mode:

```
ECHO Change to Genstat normal size mode
COPY c:\genstat\ftn77\librarie.old c:\genstat\ftn77\librarie.dir
KILL_DBOS
DBOS
ECHO The standard library will now be used
```

Large, or even Huge, mode:

```
ECHO Change to Genstat large size mode
COPY c:\genstat\ftn77\librarie.new c:\genstat\ftn77\librarie.dir
KILL_DBOS
c:\ftn77\DBOS
ECHO The non-standard library will now be used
```

The file `librarie.new` has only one line, `C:\GENSTAT\LIB\GENNEW.LIB`. Also the file `librarie.old` has similar contents, `C:\GENSTAT\LIB\GENLIB.LIB`. Each of the batch files simply kills DBOS, copies a file to `librarie.dir`, which tells the system what version of Genstat to use, and restarts DBOS again. Note that it is the version of DBOS from the directory `FTN77`, not the one supplied with the Genstat disks, because that one allows you only the use of Genstat as it was shipped to you.

Lastly, another warning about the use of this huge Genstat version. Using the `S = 4` option will always start the swapping, even before you type some Genstat statements. The advantage of such a version as compared to the Vax is clear: on a Vax it is impossible to have such a large memory, at least on our machine. We tried the same on a Sun 4 workstation; here the huge version of Genstat worked, but you can't use more than one such huge Genstat on one workstation at the same time. Who said that the sky was the limit?