

## HEIGHT GROWTH IN RADIATA PINE

### I. GENERAL MODEL AND ESTIMATION PROCEDURE

#### PROGRESS REPORT

##### 1. OVERVIEW.

The model consists of a differential equation for height growth, plus environmental variation and measurement errors. The differential equation is equivalent to the Bertalanffy-Richards model. The parameters can be taken as being the same for all the plots, or as varying between plots (e.g. functions of site).

The parameters of the model are estimated by maximum likelihood (any of them can also be fixed at given values in particular versions of the model). The likelihood function is obtained by analytical integration of the model, and maximized over the parameters using a numerical optimization algorithm (actually the negative logarithm of the likelihood is minimized).

The optimization is carried out with a version of Newton's method. Advantage is taken of the special structure of the Hessian matrix for reducing the storage requirements and arranging the computations sequentially. The method provides with estimates for the estimation errors, and statistics that can be used for goodness of fit comparisons and hypothesis testing.

Several other refinements are used for improving the efficiency and reliability of the optimization procedure, such as Cholesky's triangular factorization of matrices and modifications if the Hessian becomes non-positive-definite. Recurrence relationships are used for computing values of the likelihood function and its derivatives.

The estimation procedure has been coded in FORTRAN as a group of subroutines in order to facilitate modifications and maintenance. Any versions of the model differing which parameters are taken as fixed, common for all plots or varying between plots, can be implemented by small changes in the basic programs. The subroutines are currently in the debugging and testing stage.

Data from Kaingaroa Forest will be used for the first application. Initially data from non-frost sites will be used for exploring the relationships between parameters, and later ways of handling the effect of frosts will be studied.

##### 2. MODEL.

There are important advantages in developing a model for the growth rate as a function of size, instead of the usual approach of fitting directly the size-age relationship (Hotelling 1927, Bailey and Clutter 1974). The model used is based on the assumption that there is some power transformation of the top height for which the evolution over time can be well approximated by a linear differential equation:

$$\frac{dH^c}{dt} = \alpha + \beta H^c,$$

or

$$\frac{dH^c}{dt} = b(a^c - H^c), \quad (1)$$

where  $a$  is an upper asymptote for the height  $H$ .

This equation is equivalent to a growth model proposed by von Bertalanffy (1949, 1957) and extensively analyzed by Richards (1959):

$$\begin{aligned} cH^{c-1} \frac{dH}{dt} &= b(a^c - H^c) \\ \frac{dH}{dt} &= \frac{ba^c}{c} H^{1-c} - \frac{b}{c} H \\ \frac{dH}{dt} &= \eta H^m - \kappa H. \end{aligned} \quad (2)$$

This model is very flexible, containing as special cases several well-known growth functions such as the logistic, Mitscherlich, exponential, and Gompertz curves (Richards 1959). Lately it has been frequently used for Site Index curves, including those developed for Radiata Pine in New Zealand by Goulding and Elliott and by Burkhart.

The stochastic component of the model is a compromise between realism and mathematical tractability. It is felt that even a grossly oversimplified model should provide with better estimates than the indiscriminate use of least-squares procedures. Two sources of variation are recognized. One is the environment-caused variation of the growth rate, represented by a Brownian motion perturbation added in equation (1). This means essentially that the variation or error in  $H^c$  accumulated over a time interval is normally distributed with mean zero and variance increasing with the interval length, and that errors for non-overlapping intervals are independent. The other source of variation is the measurement error for top height, which is approximated by a normal random variable added to  $H^c$ .

The formulas take a somewhat simpler form using instead of  $H^c$  the transformation

$$x = \frac{1}{c} \left[ \left( \frac{H}{a} \right)^c - 1 \right].$$

Equation (1) is then simply

$$\frac{dx}{dt} = -bx.$$

This transformation also has the property, defining  $x = \ln(H/a)$  for  $c=0$ , of continuity at  $c=0$  as a function of  $H$  and  $c$  (Box and Cox 1964). Denoting by  $h_i$  the top height measurements at ages  $t_i$ ,  $i=1,2,\dots,n$ , for a given plot, the model is then:

$$x = \frac{1}{c} \left[ \left( \frac{H}{a} \right)^c - 1 \right] \quad (3a)$$

$$y_i = \frac{1}{c} \left[ \left( \frac{h_i}{a} \right)^c - 1 \right]; \quad i=1,\dots,n \quad (3b)$$

$$\frac{dx}{dt} = -bx + \sigma(t)\dot{w}(t) \quad (3c)$$

$$x(t_0) = x_0 =$$

$$x(t_0) = x_0 = \left[ \frac{1}{c} \left[ \left( \frac{H}{a} \right)^c - 1 \right] \right] \quad (3d)$$

$$y_i = x(t_i) + \varepsilon_i ; i=1, \dots, n \quad (3e)$$

$$w(t) = \text{standard Brownian motion process (Erickson 1971)} \quad (3f)$$

$$\sigma(t) = \sigma \text{ for } t \geq t_1 \quad (3g)$$

$$\begin{aligned} \varepsilon_i &\sim N(0, \sigma_m^2) ; i=1, \dots, n \\ \text{cov}(\varepsilon_i, \varepsilon_j) &= 0 ; i \neq j \end{aligned} \quad (3h)$$

Notice in (3g) that the variance of the environmental error term in (3c) is assumed constant beyond the age of the first measurement. It is left free before that age to account for the probably different circumstances in the initial few years following establishment.

Integrating equation (3c) and using (3e) we get

$$y_i = e^{-b(t_i - t_0)} x_0 + \delta_i + \varepsilon_i , \quad (4)$$

where

$$\delta_i = \int_{t_0}^{t_i} e^{-b(t_i - s)} \sigma(s) dw(s)$$

is a normal random variable with zero mean (Erickson 1971). The covariance between  $\delta_i$  and  $\delta_j$ ,  $i \leq j$ , is

$$\text{cov}(\delta_i, \delta_j) = E(\delta_i \delta_j) = \int_{t_0}^{t_1} e^{-b(t_i + t_j - 2s)} \sigma^2(s) ds .$$

Defining

$$\sigma_1^2 = \frac{\int_{t_0}^{t_1} e^{2bs} \sigma^2(s) ds}{\int_{t_0}^{t_1} e^{2bs} ds} , \quad (5)$$

a weighted average of  $\sigma^2(t)$  between  $t_0$  and  $t_1$ , and using (3g),

$$\begin{aligned} \text{cov}(\delta_i, \delta_j) &= \frac{1}{2b} e^{-b(t_i + t_j)} \left[ \sigma_1^2 (e^{2bt_1} - e^{2bt_0}) \right. \\ &\quad \left. + \sigma^2 (e^{2bt_i} - e^{2bt_1}) \right] , i \leq j . \end{aligned} \quad (6)$$

This model describes the data from any plot. We shall assume that measurements from different plots are statistically independent. We are thus neglecting the effects of the correlation which certainly exists between the height increments of different plots in a same year.

A model including these correlations would be too complicated at this stage, and must be left as a matter for future research.

The general model contains eight parameters for each plot:  $a, b, c, \sigma^2, \sigma_1^2, \sigma_m^2, t_0$ , and  $H_0$ . For any particular version of the model, some of them will be considered as fixed at given values, some will have the same value for all the plots ("global" parameters), and others will be allowed to differ from plot to plot ("local" parameters). Functional relationships between some parameters may also be imposed.

### 3. LIKELIHOOD FUNCTION.

For the estimation of parameters and other statistical inference purposes, we need the likelihood function, that is, the probability density function of the observations, considered as a function of the parameters\*

Because of the assumption of independence between plots, the likelihood is given by the product of the density functions of the height measurements in the  $N$  plots:

$$L = \prod_{k=1}^N f_k(h_{1k}, \dots, h_{n_k k}) , \quad (7)$$

where  $h_{ik}$  is the  $i$ -th height measurement in plot  $k$ . Now dropping the indices  $k$ , the probability density function for the  $h_i$ 's in a given plot can be obtained from the p.d.f. for the  $y_i$ 's as

$$f(h_1, \dots, h_n) = f_Y(y_1, \dots, y_n) J , \quad (8)$$

where  $J$  is the Jacobian of the transformation (3b):

$$J = \left| \frac{\partial y_i}{\partial h_j} \right| = a^{-cn} \left( \prod_{i=1}^n h_i \right)^{c-1} . \quad (9)$$

From (4) we know that  $f_Y(y_1, \dots, y_n)$  is a multivariate normal p.d.f. with means

$$e^{-b(t_i - t_0)} x_0 ,$$

and covariances given by (3h) and (6).

Some simplification is possible by introducing the new variables

$$z_i = y_i - e^{-b(t_i - t_{i-1})} y_{i-1} ; i=1, \dots, n , \quad (10)$$

with  $y_0 = x_0$ . Using (4) we find

\* Some authors define the likelihood function only up to an arbitrary multiplicative constant.

$$z_i = \int_{t_{i-1}}^{t_i} e^{-b(t_i-s)} \sigma(s) dw(s) + \varepsilon_i - e^{-b(t_i-t_{i-1})} \varepsilon_{i-1},$$

where  $\varepsilon_0=0$ , and from here,

$$E(z_i) = 0 ; i=1, \dots, n, \quad (11)$$

and

$$\text{cov}(z_i, z_j) = \begin{cases} (1-e^{-2b\zeta_i}) \frac{\sigma_i^2}{2b} + (1+e^{-2b\zeta_i}) \sigma_m^2 ; i=j \\ -e^{-b|t_i-t_j|} \sigma_m^2 ; |i-j|=1 \\ 0, \text{ otherwise,} \end{cases} \quad (12)$$

where  $\zeta_i = t_i - t_{i-1}$ , and  $\sigma_i = \sigma$  for  $i > 1$ . The Jacobian of the transformation (10) is one, so that

$$f_{\underline{y}}(y_1, \dots, y_n) = (2\pi)^{-\frac{1}{2}n} |\underline{C}|^{-\frac{1}{2}} \exp(-\frac{1}{2} \underline{z}' \underline{C}^{-1} \underline{z}), \quad (13)$$

where  $\underline{z}' = (z_1, \dots, z_n)$ , and  $\underline{C}$  is the covariance matrix with elements  $c_{ij} = \text{cov}(z_i, z_j)$  given by (12). The fact that the matrix  $\underline{C}$  is tridiagonal will be used in Section 4.4 to obtain recurrence relationships for computing the likelihood and its derivatives.

It will be convenient to work with  $-2$  times the log-likelihood:

$$\begin{aligned} -2 \ln L = & \sum \left[ n \ln 2\pi + \ln |\underline{C}| + \underline{z}' \underline{C}^{-1} \underline{z} + 2cn \ln a \right. \\ & \left. - 2(c-1) \sum_{i=1}^n \ln h_i \right]. \end{aligned} \quad (14)$$

The sum is over the  $N$  plots.

#### 4. PARAMETER ESTIMATION.

Parameters shall be estimated by the method of maximum likelihood. The maximum likelihood estimates are the values of the parameters for which the likelihood function is a maximum, for the given data. The maximum of the likelihood, or, equivalently, the minimum of (14), can be found using numerical optimization techniques (Chambers 1973). The value of the likelihood function and its shape at the maximum can be used to make inferences about estimation errors and adequacy of different hypothesis, as discussed below.

Classical inference procedures for maximum likelihood estimates are based on asymptotic properties, true when the sample size tends to infinite. Under fairly general conditions the maximum likelihood estimator is consistent, asymptotically efficient, and asymptotically normally distributed with covariance matrix that can be estimated

from the matrix of second derivatives of the log-likelihood. In addition, the asymptotic Chi-squared distribution of  $-2$  times the logarithm of likelihood ratios is used for likelihood ratio tests of hypothesis. However, the standard conditions for proving these results do not apply to the present model, and it is not clear yet to what extent they may be true in this case. For inference on the global parameters when the number of plots is large, we have an example of a class of problems studied, among others, by Kiefer and Wolfowitz 1956, Kalbleisch and Sprott 1970, and Andersen 1970 (see also Cox and Hinkley 1974, pp 292, 298). In this situation what I have called global and local parameters are called structural and incidental parameters, respectively. For inference about the local parameters it would be necessary to consider asymptotic properties as the number of observations in a plot increases. Then, except for the added complication of the presence of the global parameters, we have the case of dependent observations treated in Cox and Hinkley 1974 pp 293, 299, Weiss 1971, and Crowder 1976. Anyhow, asymptotic results are not likely to be very useful for the local parameters because of the small number of observations per plot. In summary, it seems desirable to study these problems a little further. It may be also possible to obtain information about estimation errors, still within the classical framework, by using simulation or jackknifing techniques (Miller 1974, Bissel and Ferguson 1975).

On the other hand, if one accepts the ideas of Likelihood Inference, it is possible to make use of the maximum likelihood estimates, likelihood ratios and second derivatives of the log-likelihood in a very direct way (Barnett 1973 sec.8.2, Edwards 1972). Even if some of the ideas of Likelihood Inference are highly controversial (the same is true with classical inference, see Barnett 1973), it seems clear that its techniques provide at least with useful qualitative guides for choosing between hypothesis and for making other inferences.

A useful property of maximum likelihood estimators which is valid in any case is the invariance principle: if  $\hat{\theta}$  is a maximum likelihood estimate of  $\theta$ , then  $g(\hat{\theta})$  is a M.L.E. of  $g(\theta)$ , for any function  $g$  of  $\theta$ . This is a nice property since in the applications we will be usually interested in different more or less complicated functions of the parameters, more than in the parameters themselves.

#### 4.1 Minimization procedure.

A variation of Newton's method will be used to find the minimum of (14). The fundamentals of the method are explained first, and then some possible alternatives are discussed.

Newton's method for finding the minimum of a function  $F(\theta)$  of vector  $\theta$  is based on approximating the function by the first three terms of its Taylor series expansion around an initial estimate  $\theta_0$ :

$$F(\theta) \approx F(\theta_0) + g'(\theta - \theta_0) + \frac{1}{2}(\theta - \theta_0)' H (\theta - \theta_0), \quad (15)$$

where  $\underline{g}$  is the gradient

$$\underline{g}' = \left( \frac{\partial F}{\partial \theta_1}, \dots, \frac{\partial F}{\partial \theta_p} \right),$$

and  $H$  is the Hessian matrix

$$H = \begin{bmatrix} \frac{\partial^2 F}{\partial \theta_1^2} & \dots & \frac{\partial^2 F}{\partial \theta_1 \partial \theta_p} \\ \vdots & & \vdots \\ \frac{\partial^2 F}{\partial \theta_p \partial \theta_1} & \dots & \frac{\partial^2 F}{\partial \theta_p^2} \end{bmatrix},$$

both evaluated at  $\underline{\theta} = \underline{\theta}_0$ . If (15) were exactly true, the optimum  $\underline{\theta}$  could be obtained immediately by putting the derivative of (15) with respect to  $\underline{\theta}$  equal to zero:

$$\begin{aligned} \underline{g}' + H(\underline{\theta} - \underline{\theta}_0) &= 0 \\ \underline{\theta} &= \underline{\theta}_0 - H^{-1} \underline{g}. \end{aligned} \quad (16)$$

Since (15) is only an approximation, the value given by (16) will not in general be the optimum, but usually will be an improvement over  $\underline{\theta}_0$ . The procedure can be repeated using the new value as  $\underline{\theta}_0$ , until no further improvement is possible. The repeated application of (16) is the basic Newton method.

Two problems may cause failure to converge to the minimum, especially if the initial estimate is poor:

a) If at a given  $\underline{\theta}_0$  the Hessian is not positive-definite,  $F$  may not decrease in the direction given by  $-H^{-1} \underline{g}$ . The solution is to substitute temporarily for  $H^{-1}$  some other positive-definite matrix. This is discussed in 4.3.

b) Even if  $H$  is positive-definite, taking a step  $-H^{-1} \underline{g}$  may overshoot the minimum in that direction and fail to decrease the value of  $F$ . The modification used here will be to use, instead of (16),

$$\underline{\theta} = \underline{\theta}_0 - \alpha H^{-1} \underline{g}. \quad (17)$$

For each iteration  $\alpha$  is initially 1, and if it fails in reducing  $F$  it is halved until  $F(\underline{\theta}) < F(\underline{\theta}_0)$ .

Notice that in our problem of minimizing (14) the number  $p$  of components of  $\underline{\theta}$  is very large: the number of global parameters plus the number of local parameters times the number of plots. In general, it is then out of question handling the  $p \times p$  matrix  $H$  or  $H^{-1}$  in core memory. However, it is possible to take advantage of the special structure of  $H$  in this case for arranging the computation so that the plots can be processed one at a time in each

iteration. This is discussed in 4.2.

Newton's method is just one of many numerical optimization procedures (Murray 1972a, Jacoby et al. 1972, Chambers 1973; Fletcher 1972 reviews available computer programs). In general, this is the best method in terms of reliability of convergency, but has the disadvantage of requiring the computation of second derivatives. More popular are the so-called quasi-Newton methods. In these an approximation to  $H^{-1}$  or to  $H$  is updated at each iteration according to information obtained on that iteration. Usually the value of  $\alpha$  in (17) is selected by a unidimensional minimization of  $F(\theta_0 - \alpha H^{-1}g)$  with the estimated  $H$ .

Some versions use finite differences approximations for the gradient, so that only functional values need to be provided. It may be possible to modify some of these methods for making use of the method in 4.2 for handling  $H$ , but likely this would require a considerable programming and testing effort.

Conjugate gradients methods (Fletcher and Reeves 1964, Powell 1964) are generally slower than quasi-Newton methods, but do not use the Hessian matrix. Powell's method do not require derivatives, but there seem to be some doubts about its performance for large number of variables. Because of the absence of the problems associated with handling  $H$ , and of being relatively simple to program, these methods seem potentially useful for problems with global and local parameters like the present one. On the other hand, the value of the Hessian at the optimum may be useful for statistical inference purposes (section 4). A FORTRAN version of the ALGOL procedure in Fletcher and Reeves (1964) is available in IBM's Scientific Subroutine Package.

The other commonly used method is the simplex method of Nelder and Mead. This is a direct search method requiring only functional values, but its performance when the number of dimensions in which the search is conducted is large seems to be poor.

Initially it was planned to use nested optimization as suggested by Ross (1970), and the coding of a procedure for that was initiated. This approach consists in fixing the global parameters and minimizing over the local parameters (this can be done separately for each plot), and then minimizing over the global parameters with the local ones fixed at the values just obtained. The process is repeated with the new values until convergency is attained. Thus the number of variables in each minimization is small and a quasi-Newton method can be used. However, it can be seen that this procedure can converge extremely slowly, since it has characteristics similar to those of the sectioning or relaxation method (Wilde and Beightler 1967, p.296) which involves altering only one variable at a time. The direct approach should have a much better performance.

Finally, it may be mentioned that it is possible to avoid the programming of derivative computations leaving this work to the computer. This can be achieved by using symbolic manipulation languages which produce computer code (e.g. Tobey 1966), or using Wengert's technique



for producing directly the numerical values of the derivatives (Wengert 1964, Wilkins 1964, Lesk 1967). These techniques may be useful for incorporating them in general purpose estimation programs, and as a debugging aid.

#### 4.2 Sequential arrangement of computations.

The application of (17) can be described in a slightly different way as follows. At each iteration we compute a vector  $\underline{d}$  defining the direction of movement by solving

$$H\underline{d} = -\underline{g} , \quad (18)$$

and then we move by a multiple  $\alpha$  of this vector:

$$\underline{\theta} = \underline{\theta}_0 + \alpha \underline{d} .$$

← In general, the size of  $H$  does not allow the direct solution of (18) by manipulating  $H$  in core memory.

Consider the partition of the vector of parameters

$$\underline{\theta} = \begin{bmatrix} \underline{\theta}_0 \\ \underline{\theta}_1 \\ \vdots \\ \underline{\theta}_N \end{bmatrix} ,$$

where  $\underline{\theta}_0$  is the vector of global parameters and  $\underline{\theta}_k$  is the vector of local parameters for plot  $k$ ,  $k=1, \dots, N$  (note the change of notation:  $\underline{\theta}_0 \neq \underline{\theta}_0$  !). Consider also  $\underline{d}$ , the gradient and the Hessian partitioned in the same way:

$$\underline{d} = \begin{bmatrix} \underline{d}_0 \\ \vdots \\ \underline{d}_N \end{bmatrix} , \quad \underline{g} = \begin{bmatrix} \underline{g}_0 \\ \vdots \\ \underline{g}_N \end{bmatrix} , \quad H = \begin{bmatrix} H_{00} & \dots & H_{0N} \\ \vdots & & \vdots \\ H_{N0} & \dots & H_{NN} \end{bmatrix} .$$

Our objective function (14) is a sum of functions of the form

$$F(\underline{\theta}) = \sum_{k=1}^N F_k(\underline{\theta}_0, \underline{\theta}_k) , \quad (19)$$

where  $F_k$  represents the expression in brackets on the right hand side of (14). Then

$$\begin{aligned} \underline{g}_0 &= \frac{\partial F}{\partial \underline{\theta}_0} = \sum_{k=1}^N \frac{\partial F_k}{\partial \underline{\theta}_0} = \sum_{k=1}^N \underline{g}_{k0} \\ \underline{g}_1 &= \frac{\partial F}{\partial \underline{\theta}_1} = \frac{\partial F_1}{\partial \underline{\theta}_1} = \underline{g}_{11} , \quad i=1, \dots, N \\ H_{00} &= \frac{\partial^2 F}{\partial \underline{\theta}_0^2} = \sum_{k=1}^N \frac{\partial^2 F_k}{\partial \underline{\theta}_0^2} = \sum_{k=1}^N H_{k00} \end{aligned}$$

$$H_{0i} = H'_{i0} = \frac{\partial^2 F}{\partial \underline{\theta}_0 \partial \underline{\theta}_i} = \frac{\partial^2 F_i}{\partial \underline{\theta}_0 \partial \underline{\theta}_i} = H_{i0i}, \quad i=1, \dots, N$$

$$H_{ii} = \frac{\partial^2 F}{\partial \underline{\theta}_i^2} = \frac{\partial^2 F_i}{\partial \underline{\theta}_i^2} = H_{iii}, \quad i=1, \dots, N$$

$$H_{ij} = \frac{\partial^2 F}{\partial \underline{\theta}_i \partial \underline{\theta}_j} = 0, \quad i \neq j, \quad i, j=1, \dots, N.$$

Using then the fact that  $H$  is "quasi-block-diagonal", with  $H_{ij}=0$  for  $i \neq j$  and  $i, j > 0$ , (18) can be written as

$$\begin{cases} H_{00} \underline{\delta}_0 + \sum_{k=1}^N H_{0k} \underline{\delta}_k = -\underline{g}_0 & (20a) \\ H_{k0} \underline{\delta}_0 + H_{kk} \underline{\delta}_k = -\underline{g}_k, \quad k=1, \dots, N. & (20b) \end{cases}$$

From (20b)

$$\underline{\delta}_k = -H_{kk}^{-1}(\underline{g}_k + H_{k0} \underline{\delta}_0)$$

and, substituting into (20a) and rearranging,

$$\begin{cases} (H_{00} - \sum_{k=1}^N H_{0k} H_{kk}^{-1} H_{k0}) \underline{\delta}_0 = \sum_{k=1}^N H_{0k} H_{kk}^{-1} \underline{g}_k - \underline{g}_0 \\ H_{kk} \underline{\delta}_k = -\underline{g}_k - H_{k0} \underline{\delta}_0, \quad k=1, \dots, N. \end{cases}$$

In terms of the gradients and Hessians of the  $F_k$ 's,

$$\begin{cases} \left[ \sum_{k=1}^N (H_{k00} - H_{k0k} H_{kkk}^{-1} H_{kk0}) \right] \underline{\delta}_0 = \sum_{k=1}^N (H_{k0k} H_{kkk}^{-1} \underline{g}_{kk} - \underline{g}_{k0}) & (21a) \\ H_{kkk} \underline{\delta}_k = -\underline{g}_{kk} - H_{kk0} \underline{\delta}_0, \quad k=1, \dots, N. & (21b) \end{cases}$$

These equations can be used to perform each iteration as follows. Solve (21a) for  $\underline{\delta}_0$ . Then, for each plot, solve (21b) for  $\underline{\delta}_k$ , compute  $F_k$ , its gradient and its Hessian at the new  $\underline{\theta}_0$  and  $\underline{\theta}_k$ , and accumulate the quantities in the sums in (21a) to be used in the next iteration.  $F_k$  is also accumulated to obtain the new  $F$  according to (19). The new values of  $\underline{\theta}_k$ ,  $\underline{g}_{kk}$ ,  $H_{kkk}$ , and  $H_{kk0}$  are stored together with the data from plot  $k$  in auxiliary memory for their use in the next iteration.  $\underline{\delta}_k$  is also saved for cases in which it is necessary to move back by halving  $\alpha$ .

Further details on the implementation of this procedure are given in 4.3 and in the Appendix.

This approach can be extended to more than two levels of parameters, e.g. country-regions-plots.

### 4.3 Use of Cholesky factorization.

An efficient way of solving (21) is using Cholesky factorization (Martin et al. 1971). In addition, this makes possible to use a simple technique for handling the problem of non-positive-definite Hessians mentioned in p.7.

If  $A$  is a positive-definite symmetric matrix it can be factorized in the form

$$A = LL' , \quad (22)$$

where  $L$  is a lower-triangular matrix. This is the Cholesky decomposition or factorization of  $A$ . The elements of  $L$  can be determined row by row or column by column by equating corresponding elements in (22).

This factorization can be used for computing the solution of the set of equations

$$Ax = b$$

if  $A$  is symmetric and positive-definite. Using (22),

$$LL'x = b.$$

Let  $L'x = y$ . The solution is obtained in two steps. First solve  $Ly = b$  for  $y = L^{-1}b$ . Then solve  $L'x = y$  for  $x = L'^{-1}y$ .  $L^{-1}b$  and  $L'^{-1}y$  are easy to compute because of the triangularity of  $L$ .

Using these results we can factorize

$$H_{kkk} = L_k L_k' , \quad (23)$$

with  $L_k$  lower-triangular. Then (21) can be written as

$$\left\{ \begin{array}{l} \left[ \sum_{k=1}^N (H_{k00} - M_k' M_k) \right] \delta_0 = \sum_{k=1}^N (M_k' v_k - g_{k0}) \end{array} \right. \quad (24a)$$

$$\left\{ \begin{array}{l} \delta_k = -L_k'^{-1} (v_k - M_k \delta_0), k=1, \dots, N, \end{array} \right. \quad (24b)$$

where  $M_k = L_k'^{-1} H_{kk0}$ ,  $v_k = L_k'^{-1} g_{kk}$ . Equation (24a) is also solved using Cholesky factorization.  $L_k$ ,  $M_k$  and  $v_k$  are saved on auxiliary memory, instead of  $H_{kkk}$ ,  $H_{kk0}$  and  $g_{kk}$  as indicated in 4.2.

As mentioned in 4.1, if the Hessian is not positive-definite it is desirable to replace it by some positive-definite matrix to ensure a descent direction. If  $H$  is not positive-definite, a negative value (or 0, if  $H$  is singular) will be found for a diagonal element of  $L$  in the process of computing the factorization for some  $H_{kkk}$  or for the matrix in (24a), making impossible to continue with the procedure (see Martin et al. 1971). A simple way of obtaining a modified positive-definite matrix is to substitute its absolute value for the negative value found, or 1 for a 0, and carry on with the procedure. This is similar to the method proposed by Matthews and

Davies (1971), with the difference that they use Crout's factorization instead of Cholesky's. Crout factorization does not take advantage of the symmetry of the matrix, taking about twice as much computer time. No particular good properties can be claimed for the matrix thus obtained, besides being positive-definite; the justification is, as Matthews and Davies pointed out, that its implementation is simple and that it essentially requires no extra computation.

It was found later that Murray (1972b) proposed another method, more elaborate, for modifying the matrix, also based on Cholesky decomposition. He criticizes Matthews and Davies procedure as being numerically unstable. It seems feasible to adapt Murray's approach for use in the present problem, though it would require reprogramming our Cholesky factorization subroutine for performing it column by column instead of row by row. Murray's approach shall be tried if the simpler one proves unsatisfactory.

#### 4.4 Derivatives.

For each plot we need to compute  $F_k$ , its gradient and its Hessian matrix with respect to  $\underline{\theta}_0$  and  $\underline{\theta}_k$ . From (14),  $F_k$  is, again omitting the indices  $k$ ,

$$F = n \ln 2\pi + \ln |C| + \underline{z}' C^{-1} \underline{z} - 2 \ln J. \quad (25)$$

Expressions for the first and second derivatives can be obtained using matrix derivatives (Dwyer 1967, Neudecker 1969). Using subindices to denote derivatives with respect to parameters  $\theta$  and  $\eta$ , we get

$$F_{\theta} = \text{tr}(C^{-1} C_{\theta}) - \underline{z}' C^{-1} C_{\theta} C^{-1} \underline{z} + 2 \underline{z}' C^{-1} \underline{z}_{\theta} - 2(\ln J)_{\theta} \quad (26)$$

$$\begin{aligned} F_{\theta\eta} = & \text{tr}(C^{-1} C_{\theta\eta}) - \text{tr}(C^{-1} C_{\theta} C^{-1} C_{\eta}) - \underline{z}' C^{-1} C_{\theta\eta} C^{-1} \underline{z} \\ & + 2 \underline{z}' C^{-1} C_{\theta} C^{-1} C_{\eta} C^{-1} \underline{z} - 2 \underline{z}' C^{-1} C_{\theta} C^{-1} \underline{z}_{\eta} \\ & - 2 \underline{z}' C^{-1} C_{\eta} C^{-1} \underline{z}_{\theta} + 2 \underline{z}'_{\theta} C^{-1} \underline{z}_{\eta} + 2 \underline{z}' C^{-1} \underline{z}_{\theta\eta} \\ & - 2(\ln J)_{\theta\eta}. \end{aligned} \quad (27)$$

However, a more efficient computational procedure was found, so that these equations were used only for checking the numerical results.

The matrix  $C$  is tridiagonal:

$$C = \begin{bmatrix} q_1 & p_2 & 0 & \cdot & \cdot & \cdot & 0 \\ p_2 & q_2 & p_3 & \cdot & \cdot & \cdot & 0 \\ 0 & p_3 & q_3 & \cdot & \cdot & \cdot & 0 \\ \vdots & \vdots & \vdots & & & & \vdots \\ 0 & 0 & 0 & \cdot & \cdot & \cdot & q_n \end{bmatrix},$$

with  $p_i = \text{cov}(z_i, z_{i-1})$  and  $q_i = \text{cov}(z_i, z_i)$  given by (12). It has a Cholesky factorization  $C = LL'$  with  $L$  of the form

$$L = \begin{bmatrix} s_1 & 0 & 0 & \dots & 0 \\ r_2 & s_2 & 0 & \dots & 0 \\ 0 & r_3 & s_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & s_n \end{bmatrix},$$

with  $s_i > 0$  for all  $i$ .

The determinant in (25) can be written

$$|C| = |LL'| = |L|^2 = \left( \prod_{i=1}^n s_i \right)^2,$$

and

$$\ln |C| = 2 \sum_{i=1}^n \ln s_i.$$

Also

$$\underline{z}' C^{-1} \underline{z} = \underline{z}' (LL')^{-1} \underline{z} = (L^{-1} \underline{z})' (L^{-1} \underline{z}).$$

Let  $L^{-1} \underline{z} = \underline{u}$ . Then

$$\underline{z}' C^{-1} \underline{z} = \sum_{i=1}^n u_i^2.$$

Substituting in (25) we get

$$F = n \ln 2\pi - 2 \ln J + \sum_{i=1}^n (\ln s_i^2 + u_i^2). \quad (28)$$

It can be shown that the  $s_i$  and  $u_i$  can be obtained recursively from the elements of  $C$  by

$$\begin{cases} r_1 = 0; r_i = \frac{p_i}{s_{i-1}}, \quad i=2, \dots, n \end{cases} \quad (29a)$$

$$\begin{cases} s_i = \sqrt{q_i - r_i^2}, \quad i=1, \dots, n \end{cases} \quad (29b)$$

$$\begin{cases} u_i = \frac{1}{s_i} (z_i - r_i u_{i-1}), \quad i=1, \dots, n \end{cases} \quad (29c)$$

For the first derivatives, from (28),

$$F_\theta = -2(\ln J)_\theta + \sum_{i=1}^n [(\ln s_i^2)_\theta + u_i(2u_i)_\theta], \quad (30)$$

and from (29), after some algebra,

$$(2r_i)_\theta = r_i [(\ln p_i^2)_\theta - (\ln s_{i-1}^2)_\theta] \quad (31a)$$

$$(\ln s_i$$

$$(\ln s_i^2)_\theta = \frac{1}{s_i^2} [q_{i\theta} - r_i(2r_i)_\theta] \quad (31b)$$

$$(2u_i)_\theta = \frac{1}{s_i} [2z_{i\theta} - (2r_i)_\theta u_{i-1} - r_i(2u_{i-1})_\theta] - u_i(\ln s_i^2)_\theta \quad (31c)$$

Analogously, for the second derivatives,

$$F_{\theta\eta} = -2(\ln J)_{\theta\eta} + \sum_{i=1}^n [(\ln s_i^2)_{\theta\eta} + u_i(2u_i)_{\theta\eta} + \frac{1}{2}(2u_i)_\theta(2u_i)_\eta] \quad (32)$$

$$(2r_i)_{\theta\eta} = r_i [(\ln p_i^2)_{\theta\eta} - (\ln s_{i-1}^2)_{\theta\eta}] + \frac{(2r_i)_\theta(2r_i)_\eta}{2r_i}, \text{ if } r_i \neq 0$$

$$= 0, \text{ otherwise} \quad (33a)$$

$$(\ln s_i^2)_{\theta\eta} = \frac{1}{s_i^2} [q_{i\theta\eta} - \frac{1}{2}(2r_i)_\theta(2r_i)_\eta - r_i(2r_i)_{\theta\eta}] - (\ln s_i^2)_\theta(\ln s_i^2)_\eta \quad (33b)$$

$$(2u_i)_{\theta\eta} = \frac{1}{s_i} [2z_{i\theta\eta} - (2r_i)_{\theta\eta} u_{i-1} - \frac{1}{2}(2r_i)_\theta(2u_{i-1})_\eta - \frac{1}{2}(2r_i)_\eta(2u_{i-1})_\theta - r_i(2u_{i-1})_{\theta\eta}] - \frac{1}{2} [(2u_i)_\theta(\ln s_i^2)_\eta + (2u_i)_\eta(\ln s_i^2)_\theta + u_i(\ln s_i^2)_{\theta\eta}] - u_i(\ln s_i^2)_{\theta\eta} \quad (33c)$$

A subroutine based on these formulae has been written and tested. Results were checked against those obtained with (25)-(27) for a case with  $n=2$ , and against finite difference approximations as suggested by Murray (1972c), for several other sets of data.

## 5. DATA.

Data from Kaingaroa Forest shall be used for testing and for the first application of the model and estimation procedure. About 300 permanent plots have been selected, eliminating plots from natural-regenerated stands, fertilizer trials, and plots with other conditions that could produce abnormal height growth. The plots were classified in frost-prone sites and frost-free sites.

The fact that not all measurements are made at the same time of the year makes desirable some kind of correction of the ages. The use of the fraction of year corresponding to the date of measurement is unsatisfactory due to the seasonal characteristics of the growth. A simple alternative is to use fictitious time fractions obtained by reading backwards on a graph of accumulated relative yearly growth against time of year. This has the effect of linearizing the growth within a year, as a function of the corrected age. Data for average height growth provided by J. Beekhuis was used to obtain the following corrections, to be added to the age in years (reference date is 1st July):

January	-0.2	July	0
February	-0.1	August	0
March	-0.1	September	+0.1
April	0	October	+0.3
May	0	November	+0.5
June	0	December	+0.6

It may be better to eliminate measurements for which the correction would be high. These figures are based on rather few data, and it is intended to see if more data can be found. In any case, it does not seem worthwhile to use more elaborate procedures for this correction.

## 6. THE ROAD AHEAD.

The subroutines for computing the derivatives and for the Cholesky factorization and solution of linear equations are finished and tested. The subroutine for the minimization procedure is in the debugging stage. A small master program, which essentially handles the input-output and calls the minimization subroutine, needs to be improved. It is also necessary to extract and validate the required data from the Permanent Sample Plot System.

If the system works, the patterns of height growth shall be explored first using the frost-free site data. Initially, a model with  $H_0$  fixed,  $c$  and  $\sigma_m$  global,  $a, b, \sigma_1$  and  $\sigma$  local, and  $t_0$  either fixed, global or local, may be fitted. This would help to decide if the usual practice of assuming  $t_0=0$  for  $H_0=0$  is adequate, or if some other origin may be preferred.

The next step would be to study the relationship between  $a$  and  $b$ , by plotting the values obtained against each other and against site index. Many different assumptions have been made in the past about these coefficients for developing site index equations, a common one being that  $a$  is a constant independent of the site index. These and

other hypothesis can be tested graphically or by fitting the appropriate versions of the model. The computed relative likelihoods and Hessians can be used with the principles of Likelihood Inference to evaluate different hypothesis. An impossibility of replacing  $a$  and  $b$  by a single local parameter would show that the conventional concept of a unidimensional site index is inadequate.

Having a satisfactory model for frost-free sites, we can see to what extent the effect of frost on the early years growth can be handled by shifting the origin.

Finally, given a model it is easy to develop site index estimators based on any number of height-age measurements. Several height-age pairs can be obtained by making use of the branch whorls and bud scars. These can provide site index estimates more precise and less affected by establishment practices and conditions than the usual ones based only on the current height and age. It may also be possible to account, to some extent, for fertilization effects.

#### REFERENCES

- Andersen, E.B. (1970). Asymptotic properties of conditional maximum-likelihood estimators. J.R.Statist.Soc.B, 32, 283-301.
- Bailey, R.L. and Clutter, J.L. (1974). Base-age invariant polymorphic site curves. For.Sci., 20, 155-159.
- Barnett, V.P. (1973). Comparative statistical inference. John Wiley & Sons, London.
- Bertalanffy, L.von (1949). Problems of organic growth. Nature, 163, 156-158.
- Bertalanffy, L.von (1957). Quantitative laws in metabolism and growth. The Quart.Rev.Biol., 32, 217-231.
- Bissel, A.F. and Ferguson, R.A. (1975). The jackknife - toy, tool or two-edged weapon?. The Statistician, 24, 79-100.
- Box, G.E.P. and Cox, D.R. (1964). An analysis of transformations. J.R.Statist.Soc.B, 26, 211-252.
- Chambers, J.M. (1973). Fitting nonlinear models: numerical techniques. Biometrika, 60, 1-13.
- Cox, D.R. and HINKLEY, D.V. (1974). Theoretical Statistics. Chapman and Hall, London.
- Crowder, M.J. (1976). Maximum likelihood estimation for dependent observations. J.R.Statist.Soc.B, 38, 45-53.
- Dwyer, P.S. (1967). Some applications of matrix derivatives in multivariate analysis. J.Am.Statist.Ass., 62, 607-625.
- Edwards, A.W.F. (1972). Likelihood. Cambridge Univ.Press.
- Erickson, R.V. (1971). Constant coefficient linear differential equations driven by white noise. Ann. Math.Statist., 42, 820-823.
- Fletcher, R. (1972). A survey of algorithms for unconstrained optimization. In: Murray, W. (1972a).
- Fletcher, R. and Reeves, C.M. (1964). Function minimization by conjugate gradients. Computer J., 7, 149-154.
- Hotelling, H. (1927). Differential equations subject to error, and population estimates. J.Am.Statist.Ass., 22, 283-314.
- Jacoby, S.L.S., Kowalik, J.S. and Pizzo, J.T. (1972). Iterative methods for nonlinear optimization problems. Prentice-Hall, Inc., Englewood Cliffs, N.J.



- Kalbfleisch, J.D. and Sprott, D.A. (1970). Application of likelihood methods to models involving large numbers of prof parameters. (with discussion). J.R. Statist. Soc. B, 32, 175-208.
- Kiefer, J. and Wolfowitz, J. (1955). Consistency of the maximum likelihood estimator in the presence of infinitely many incidental parameters. Ann. Math. Statist. 27, 887-906.
- Lesk, A.M. (1967). Dynamic computation of derivatives. Comm. A.C.M., 10, 571-572.
- Martin, R.S., Peters, G. and Wilkinson, J.H. (1971). Symmetric decomposition of a positive definite matrix. In: Wilkinson, J.H. and Reinsch, C. Linear Algebra - Handbook for Automatic Computation, Volume II. Springer-Verlag, Berlin. (Prepublished in Numer. Math., 7, 362-383, 1965).
- Mathews, A. and Davies, D. (1971). A comparison of modified Newton methods for unconstrained optimisation. Computer J., 14, 293-294.
- Miller, R.G. (1974). The jackknife - a review. Biometrika, 61, 1-15.
- Murray, W. (1972a)<sup>(Ed.)</sup>. Numerical methods for unconstrained optimization. Academic Press, London.
- Murray, W. (1972b). Second derivative methods. In Murray (1972a).
- Murray, W. (1972c). Failure, the causes and cures. In Murray (1972a).
- Neudecker, H. (1969). Some theorems on matrix differentiation with special reference to Kronecker matrix products. J. Am. Statist. Ass., 64, 953-963.
- Powell, M.J.D. (1964). An efficient method of finding the minimum of a function of several variables without calculating derivatives. Computer J., 7, 155-162.
- Richards, F.J. (1959). A flexible growth function for empirical use. J. Expt. Botany, 10, 290-300.
- Ross, G.J.S. (1970). The efficient use of function minimization in non-linear maximum-likelihood estimation. Appl. Statist., 19, 205-221.
- Tobey, R.G. (1966). Eliminating monotonous mathematics with FORMAC. Comm. A.C.M., 10, 742-751.
- Weiss, L. (1971). Asymptotic properties of maximum likelihood estimators in some nonstandard cases. J. Am. Statist. Ass., 66, 345-350.
- Wengert, R.E. (1964). A simple automatic derivative evaluation program. Comm. A.C.M., 7, 463-464.
- Wilde, D.J. and Beightler, C.S. (1967). Foundations of Optimization. Prentice-Hall, Inc., Englewood Cliffs, N.J.
- Wilkins, (1964). Investigation of a new analytical method for numerical derivative evaluation. Comm. A.C.M., 7, 465-471.

Oscar García

2nd June 1977

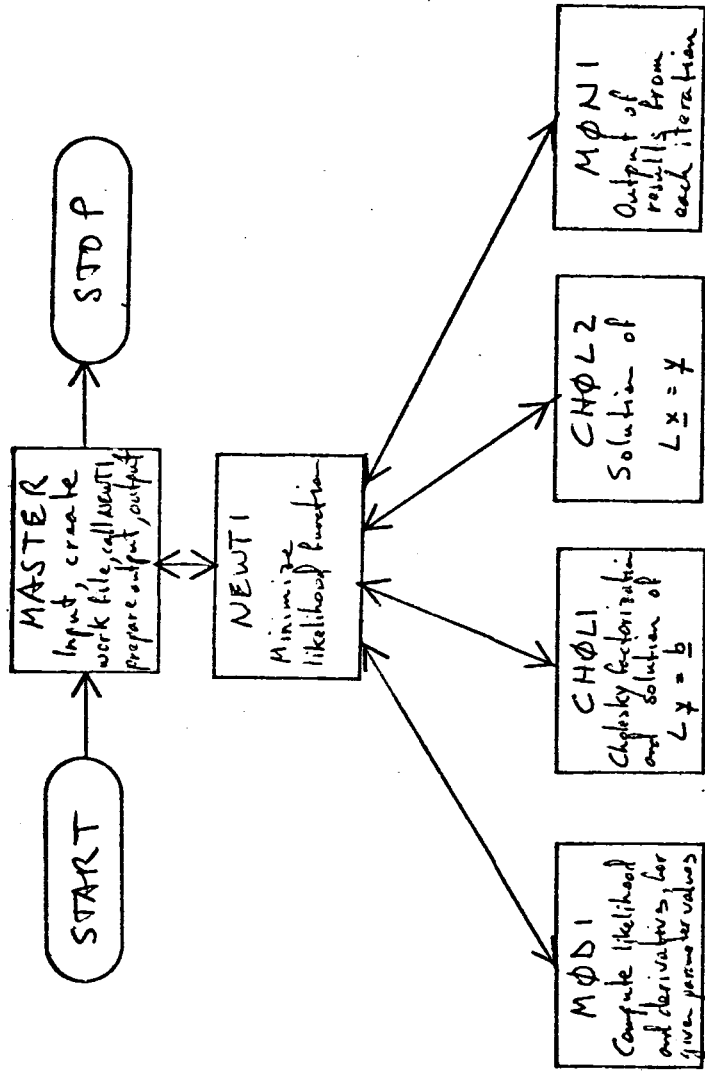
## APPENDIX

### PROGRAM DOCUMENTATION

The program and subroutines described here are under testing, and subject to modification. A final version of the MASTER program will recompute the Hessian from the Cholesky factors left by subroutine NEWT.1 in the workfile, or perhaps parts of the inverse Hessian which may be more useful for inference purposes. It will also save and print the results.

Currently there seem to be problems with non-positive-definite matrices, resulting in too large a  $\underline{S}$ , producing overflows. It has been decided to re-write subroutines CHOL1 and CHOL2 using a variant of Murphy's method mentioned in 4.3.

The estimation procedure has been programmed as a master program and 5 subroutines, as follows:



#### MASTER Program (Temporary version for testing)

Sets initial parameter estimates, reads height and age data from cards, creates disc work file, and calls NEWT.1.

In version 1, ~~the parameter~~ structure is:

Fixed:  $N_0 = 0$

Global:  $c, \sigma_m^2$

Local:  $a, b, \sigma^2, \sigma_1^2, t_0$

The minimizing uses  $\ln a, \sigma_m, \sigma$  and  $\sigma_1$ .

instead of  $a, \sigma_m^2, \sigma^2$  and  $\sigma^2$ , in order to ensure positive values for these parameters.

Program variable	Equivalent in report
H(I)	$h_1, h_2, \dots, h_n$
T(I)	$t_1, t_2, \dots, t_n$
X2(I)	dummy
C	c
W	$\sigma_m$
X1(I)	dummy
NPLPTS	N (number of pts)
NEXT	second number in work file
A	$\ln a$
B	b
C	c
V	$\sigma$
VI	$\sigma^2$
W	$\sigma_m$
TD	$t_0$
N	n (number of measurements - p.d.)

# Subroutine NEWT1

## Arguments:

NFILE	File number
NPLPTS	N, number of pts
MODEL	Subroutine for likelihood and derivatives
MOUTITR	Subroutine for output from each iteration
MO	Number of global parameters
MI	Number of local parameters
EPS	Tolerance for convergence test
ITMAX	Limit on number of iterations

This subroutine can be modified for other versions of the model by changing the array dimensions in the DIMENSION and COMMON statements, as indicated in the initial comment statements. The additional notation in there is ~~index of variables~~ <sup>index of variables</sup> in work file

RECORD LENGTH	$MO = (MO+1)/2$ (element - lower triangle of $MO \times MO$ matrix)
LTO	$MI = (MI+1)/2$
LT1	length of H(I) and T(I) in MASTER program.
NMAX	

See flowchart in Figure 1.

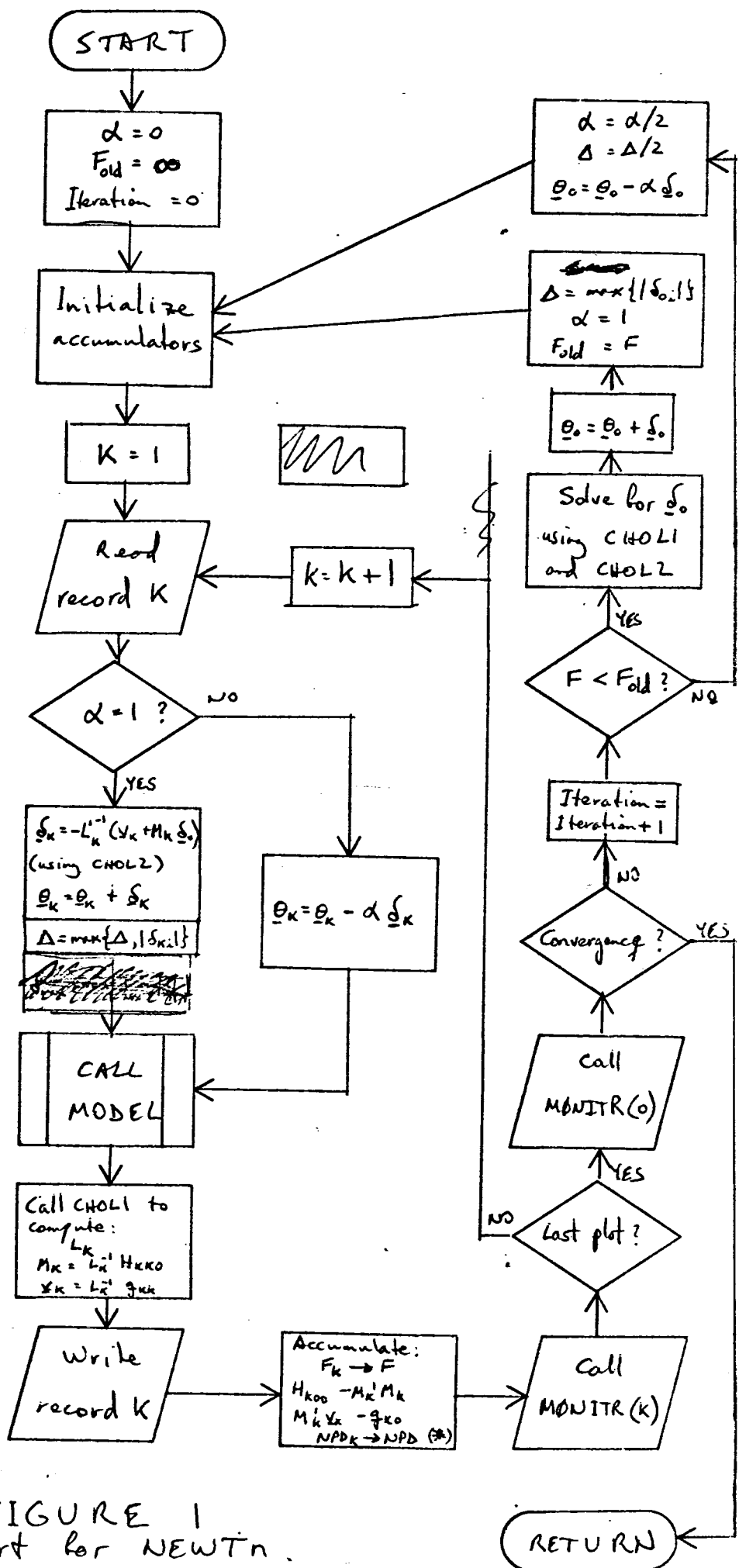


FIGURE 1  
Flowchart for NEWTON.

(\*)  $NPD_k = 0$  if  $H_{k00}$  is positive-definite, 1 otherwise

Program variable	Equivalent in report
V ( )	$\sum_{k=1}^N (M_k^i y_k - g_{k0})$
SO ( )	$\underline{\delta}_0$
GIHIO ( )	$[GI; HIO] = [g_{kk}; H_{kko}], [y_k; M_k]$
RECORD ( )	Content of record - work file = content of REC command block
THETAO ( )	$\underline{\theta}_0$
F	F
ITER	iteration number
NPDO	0 if matrix is (2N) is posid-def., 1 otherwise
NPD	<del>Sum of NPD I</del>
SHAX	$\max_{k, i} \{  S_{k, i}  \}$
ALPHA	$\alpha$
HOO ( )	$\sum_{k=1}^N (H_{kko} - M_k^i M_k)$ (lower triangle, stored by rows)
THETA1 ( )	$\underline{\theta}_k$
FI	$F_k$
NPDI	0 if $H_{kkk}$ is positive-definite, 1 otherwise
GI ( )	$g_{kk}, y_k$
HIO ( )	$H_{kko}, M_k$
HII ( )	$H_{kkk}, L_k$ (lower triangle, stored by rows)
SI ( )	$\underline{\delta}_k$
NI	$n_k, n$
HT ( )	$h_1, \dots, h_n$
T ( )	$t_1, \dots, t_n$

Program variable	Equivalent in report
GOI ( )	$g_{k0}$
HOOI ( )	$H_{kko}$ (lower triangle, stored by rows)
FOLD	Value of F in previous iteration
I	k (job, record number)

Notice EQUIVALENCE statement for equating RECORD with the content of REC COMMON and with a record of the workfile.

Note: ~~storage~~ storage by rows of lower triangle of matrices will be changed to storage by columns ~~of~~  $CNPL1$  all CHOLZ are rewritten.

### Subroutine MOD1

For given values of the parameters, this subroutine evaluates F and its first and second derivatives according to (28)-(33). The different versions of the basic model are implemented by using an appropriate common statement for matching the variables with the corresponding elements in COMMON - NEWTON. The notation is:

Program variable

Equivalent in report

ALN

ln a

b

b

c

c

VRT

$\sigma$

WRT

$\sigma_m$

VIRT

$\sigma_i$

TO

$t_0$

HO

$H_0$

N

n

HT

$h_1, \dots, h_n$

T

$t_1, \dots, t_n$

X1, X2

dummy

F

$F_K, F$

GO

$F_\theta$

HON

$F_{\theta\eta}$

The correspondence between  $\theta$  and  $\eta$  and the ~~relationship~~ parameters is:

$\frac{\partial \eta}{\partial \theta}$

A

parameter

ln a

B

b

C

c

V

$\sigma$

$\sigma_m$

$\sigma_i$

$t_0$

For a given version, it is necessary also to set the values of the fixed parameters and to eliminate any segmentally computed derivatives with respect to fixed parameters. Other variables used in the program are

Program variable

a

$\sigma^2$

$\sigma_m^2$

$\sigma_i^2$

i

$Y_i$

$Y_{i-1}$

$t_i - t_{i-1}$

$\epsilon_i$

$u_i$

$u_{i-1}$

$r_i$

$S_i$

$S_i^2$

$Y_{i0}$

W

VI

TO

A

V

W

VI

I

Y

YOLD

DT

U

UOLD

R

S

SZ

Y0

$$Y\theta\phi_{LD}$$

$$Y\theta\eta$$

$$Y\theta\eta\phi_{LD}$$

$$R\theta$$

$$S\theta$$

$$U\theta$$

$$U\theta\phi_{LD}$$

$$R\theta\eta$$

$$S\theta\eta$$

$$U\theta\eta$$

$$EX$$

$$Y_{i-1,\theta}$$

$$Y_{i,\theta\eta}$$

$$Y_{i-1,\theta\eta}$$

$$(Zr_i)_{\theta}$$

$$(\ln s_i^2)_{\theta}$$

$$(Zu_i)_{\theta}$$

$$(Zu_{i-1})_{\theta}$$

$$(Zr_i)_{\theta\eta}$$

$$(\ln s_i^2)_{\theta\eta}$$

$$(Zu_i)_{\theta\eta}$$

$$e^{-b(t_i - t_{i-1})}, e^{-bz_i}$$

The general form of a segment for computing the derivatives with respect to  $\theta$ , using (30)-(33) is:

$$R\theta = R * ([\ln p_i^2]_{\theta} - S\theta)$$

$$S\theta = (q_{i0} - R * R\theta) / S2$$

$$U\theta\phi_{LD} = U\theta$$

$$Y\theta\phi_{LD} = Y\theta$$

$$Y\theta = Y_{i0}$$

$$U\theta = (2 * (Y\theta - [e^{-bz_i}]_{\theta} * Y\phi_{LD} - EX * Y\theta\phi_{LD})$$

$$- R\theta * U\phi_{LD} - R * U\theta) / S - U * S\theta$$

$$G\theta = G\theta + [\text{argument of } \theta \text{ in case of } \theta = c] + S\theta + U * U\theta$$

$$(-2\ln u)_{\theta}$$

$$R\theta\theta = 0.$$

$$\text{IF (R.NE.O.) } R\theta\theta = R * ([\ln p_i^2]_{\theta\theta} - S\theta\theta) + 0.5 * R\theta * R\theta / R$$

$$S\theta\theta = (q_{i00} - 0.5 * R\theta * R\theta - R * R\theta\theta) / S2 - S\theta * S\theta$$

$$Y\theta\theta\phi_{LD} = Y\theta\theta$$

$$Y\theta\theta = Y_{i00}$$

$$U\theta\theta = (2 * (Y\theta\theta - [e^{-bz_i}]_{\theta\theta} * Y\phi_{LD} - 2 * [e^{-bz_i}]_{\theta} * Y\theta\phi_{LD})$$

$$- EX * Y\theta\theta\phi_{LD}) - R\theta\theta * U\phi_{LD} - R\theta * U\theta\phi_{LD}$$

$$- R * U\theta\theta) / S - U\theta * S\theta - 0.5 * U * S\theta * S\theta$$

$$- U * S\theta\theta$$

$$H\theta\theta = H\theta\theta + S\theta\theta + U * U\theta\theta + 0.5 * U\theta * U\theta * U\theta$$

Not all these derivatives are necessarily needed for every  $\theta$ , so that there are some expressions which are considerably simplified in each particular case. For the cross derivatives we have:

$$R\theta\eta = 0.$$

$$\text{IF (R.NE.O.) } R\theta\eta = R * ([\ln p_i^2]_{\theta\eta} - S\theta\eta) + 0.5 * R\theta * R\eta / R$$

$$S\theta\eta = (q_{i0\eta} - 0.5 * R\theta * R\eta - R * R\theta\eta) / S2 - S\theta * S\eta$$

$$Y\theta\eta\phi_{LD} = Y\theta\eta$$

$$\begin{aligned}
 Y_{\theta\eta} &= Y_{i\theta\eta} \\
 U_{\theta\eta} &= (2 \cdot (Y_{\theta\eta} - [e^{-b_2 z_i}]_{\theta\eta} \cdot Y_{\phi\Delta\Delta} - [e^{-b_2 z_i}]_{\theta} \cdot Y_{\eta\phi\Delta\Delta} - [e^{-b_2 z_i}]_{\eta} \cdot Y_{\theta\phi\Delta\Delta} - EX \cdot Y_{\theta\eta\phi\Delta\Delta}) \\
 &\quad - R_{\theta\eta} \cdot U_{\phi\Delta\Delta} - 0.5 \cdot (R_{\theta} \cdot U_{\eta\phi\Delta\Delta} \\
 &\quad + R_{\eta} \cdot U_{\theta\phi\Delta\Delta}) - R \cdot U_{\theta\eta} / S - 0.5 \cdot (U_{\theta} \cdot S_{\eta} \\
 &\quad + U_{\eta} \cdot S_{\theta} + U \cdot S_{\theta} \cdot S_{\eta}) - U \cdot S_{\theta\eta}
 \end{aligned}$$

$$H_{\theta\eta} = H_{\theta\eta} + S_{\theta\eta} + U \cdot U_{\theta\eta} + 0.5 \cdot U_{\theta} \cdot U_{\eta}$$

Again, simplification is possible.  
The following table indicate existence of derivatives w.r.t variables:

$[A_{\theta\eta}]_{\theta}$	$q_{i\theta}$	$R_{\theta}$	$S_{\theta}$	$Y_{i\theta}$	$Y_{\theta}$	$[e^{-b_2 z_i}]_{\theta}$	$[A_{\theta\eta}]_{\eta}$	$q_{i\eta}$	$R_{\eta}$	$S_{\eta}$	$Y_{i\eta}$	$Y_{\eta}$	$[e^{-b_2 z_i}]_{\eta}$	$\theta$
$\checkmark$	$\checkmark$	$\checkmark$		$\checkmark$	$\checkmark$						$\checkmark$	$\checkmark$		A
		$\checkmark$				$\checkmark$		$\checkmark$	$\checkmark$				$\checkmark$	B
				$\checkmark$	$\checkmark$						$\checkmark$			C
	$\checkmark$	$\checkmark$	$\checkmark$					$\checkmark$	$\checkmark$	$\checkmark$				V
$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$			$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$			W
	$\checkmark^*$	$\checkmark$						$\checkmark^*$	$\checkmark$	$\checkmark$				VI
	$\checkmark^*$	$\checkmark$	$\checkmark$					$\checkmark^*$	$\checkmark$	$\checkmark$				TD

\* - only for  $i \neq 1$   
# - only for  $i > 1$

$\theta$	$\eta$	$[A_{\theta\eta}]_{\theta\eta}$	$q_{i\theta\eta}$	$R_{\theta\eta}$	$S_{\theta\eta}$	$Y_{i\theta\eta}$	$Y_{\theta\eta}$	$[e^{-b_2 z_i}]_{\theta\eta}$
A	B							
A	C						$\checkmark$	
A	V							
A	W							
A	VI							
A	TD							
B	C		$\checkmark$	$\checkmark$	$\checkmark$			
B	V <sup>+</sup>		$\checkmark$	$\checkmark$	$\checkmark$			
B	W		$\checkmark$	$\checkmark$	$\checkmark$			
B	VI		$\checkmark^*$	$\checkmark$	$\checkmark$			
B	TD		$\checkmark^*$	$\checkmark$	$\checkmark$			$\checkmark^*$
C	V							
C	W							
C	VI							
C	TD							
V <sup>+</sup>	W			$\checkmark$	$\checkmark$			
V <sup>+</sup>	VI			$\checkmark$	$\checkmark$			
V <sup>+</sup>	TD			$\checkmark$	$\checkmark$			
W	VI		$\checkmark^*$	$\checkmark$	$\checkmark$			
W	TD		$\checkmark^*$	$\checkmark$	$\checkmark$			
VI	TD			$\checkmark$	$\checkmark$			



The derivations of ~~the~~ -2nd are constants, except for  $(-2nd)_c$ , and are included when the GO and HOG are initialized.

#### Subroutines CHDL1 and CHDL2

Subroutine CHDL1 performs the Cholesky factorization  $A = LL^T$  of a symmetric positive-definite matrix  $A$ , and solves  $LY = B$  for  $Y$ .

$A$  is a  $N \times N$  matrix, given by its lower triangular part stored by rows in vector  $A$  of length  $LT = N \cdot (N+1)/2$ .  $B$  is a  $N \times M$  matrix. On exit from CHDL1,  $L$  is overwritten on  $A$  and  $Y$  is overwritten on  $B$ .

If  $A$  is not positive-definite then  $L$  is computed as described in 4.3, and NPD is set equal to 1. Otherwise NPD is 0. In any case, the reciprocals of the diagonal elements of  $L$  are stored instead of the elements themselves.

The subroutine follows closely the ALGOL procedure cholset 2 and first pos

of cholset 2 in Martin et al (1971).

Subroutine CHDL2 solves  $AX = B$  for  $X$  where  $A$  is lower triangular  $N \times N$  stored by rows in a vector of length  $LT$ . (It is a Cholesky factor  $L$  produced by CHDL1 in this case), and  $B$  is  $N \times M$ . The result  $X$  is overwritten on  $B$ .  $A$  remains unmodified.

These two subroutines will be rewritten ~~to~~ for using Murphy's method for dealing with non-positive-definite matrices mentioned in 4.3. Triangular matrices will be stored by columns instead of by rows.

#### Subroutine MONI

At each iteration of NEWT1, this subroutine prints information for monitoring the progress of the minimization procedure. If the argument  $I = k \neq 0$ , it prints results for

plot k for the current iteration  
if  $I=0$ , it prints global results  
for the iteration.

In a final version of the  
subroutine, to be used in production  
runs, results would be printed only  
for a subset of the plots, i.e.,  
only ~~for~~ if k is a multiple of  
some given number.

Different versions of this  
subroutine will be necessary for  
different versions of the logistic  
model.

PROGRAM

LISTING

---

O. García

17/6/77.

LIST  
 SEND TO (ICLA-DEFAULT.FORTSEMICOMP)  
 WORK(COR WORKFILE(1))  
 DUMP ON (PROGRAM NNNN)

PROGRAM(NAME)  
 INPUT1=CRQ  
 OUTPUT2=LPO  
 CREATE8=EDD/DIRECT(OSCAWORK(0))/1024  
 TRACE 2  
 END

MASTER MAIN  
 DIMENSION H(10),T(10),X2(37)  
 COMMON C,W,X1(9)  
 EXTERNAL MOD1,MON1  
 DEFINE FILE 8(10,63,U,NEXT)  
 NPLOTS=0  
 NEXT=1  
 A=4.  
 B=.07  
 C=.5  
 V,V1,W=1.  
 TO=0.  
 DO 1,I=1,37  
 1 X2(I)=0.  
 10 READ (1,100) N,(H(I),T(I),I=1,N)  
 IF(N.EQ.0) GO TO 20  
 NPLOTS=NPLOTS+1  
 WRITE(8,NEXT) A,B,V,V1,TO,X2,N,H,T  
 GO TO 10  
 20 WRITE(2,200) NPLOTS  
 CALL NEWT1(8,NPLOTS,MOD1,MON1,2,5,1.E-6,50)  
 100 FORMAT(I2,2OF3.1)  
 200 FORMAT(1H1,20X,44HFIRST TEST OF ESTIMATION PROCEDURE - MODEL 1//  
 + 17H NUMBER OF PLOTS:,I3///)  
 STOP  
 END

END OF SEGMENT, LENGTH 154, NAME MAIN

SUBROUTINE MON1(I)  
 COMMON C,W,F,ITER,NPDO,NPD,SMAX,ALPHA  
 + /REC/ A,B,V,V1,TO,FI,X1,GA,GR,GV,GV1,GTO  
 + /AUX/ GC,GW  
 IF(I.EQ.0) GO TO 10  
 WRITE(2,100) I,FI,A,B,V,V1,TO  
 RETURN  
 10 WRITE(2,200) ITER,F,C,W,NPD,NPDO,SMAX,ALPHA  
 RETURN  
 100 FORMAT(I4,6G14.5)  
 200 FORMAT(1HG/6H ITER:,I3,10H FUNCT. =,G14.5,5H C=,G14.5,  
 + 11H SQRT(W)=,G14.5/11X,4HNPD=,I3,8H NPDO=,I2,8H SMAX=,  
 + G14.5,9H ALPHA=,G14.7//16H PLOT FUNCTION,8X,5H LN(A),11X,

+ 1HB,10X,7HSQRT(V),6X,8HSQRT(V1),9X,2HTO)  
END

END OF SEGMENT, LENGTH 73, NAME MON1

```

SUBROUTINE NEWT1(NFILE,NPLOTS,MODEL,MONITR,M0,MI,EPS,ITMAX)
C  SUBSTITUTE THE APPROPRIATE CONSTANTS INSIDE PARENTHESIS IN THE
C  FOLLOWING COMMON AND DIMENSION STATEMENTS:
C    DIMENSION V(M0),SQ(M0,1),GIHIO(MI,M0+1),RECORD(RECORD LENGTH)
C    COMMON THETA0(M0),F,ITER,NPDO,NPD,SMAX,ALPHA,H00(LT0)
C    + /REC/ THETA1(MI),FI,NPDI,GI(MI),HIO(MI,M0),HII(LTI),
C    + SI(MI,1),NI,HT(NMAX),T(NMAX) /AUX/ GOI(M0),HOOI(LT0)
C    DIMENSION V(2),SQ(2,1),GIHIO(5,3),RECORD(63)
C    COMMON THETA0(2),F,ITER,NPDO,NPD,SMAX,ALPHA,H00(3)
C    + /REC/ THETA1(5),FI,NPDI,GI(5),HIO(5,2),HII(15),
C    + SI(5,1),NI,HT(10),T(10) /AUX/ GOI(2),HOOI(3)
EQUIVALENCE (RECORD(1),THETA1(1)),(GIHIO(1),GI(1))
ALPHA=0.
FOLD=1.E75
ITER=0
LT0=M0*(M0+1)/2
LTI=MI*(MI+1)/2
C  INITIALIZE ACCUMULATORS
100 F=0.
    NPDI=0
    DO 110,I=1,M0
110 V(I)=0.
    DO 120,I=1,LT0
120 H00(I)=0.
C  PROCESSING OF PLOTS BEGINS
DO 200,I=1,NPLOTS
    READ(NFILE,I) RECORD
    IF(ALPHA.EQ.1.0) GO TO 140
C  MOVE BACK
    DO 130,J=1,MI
130 THETA1(J)=THETA1(J)-ALPHA*SI(J,1)
    GO TO 170
C  COMPUTE NEW DIRECTION,AND MOVE
140 DO 150,J=1,MI
    SI(J,1)=-GI(J)
    DO 150,K=1,M0
150 SI(J,1)=SI(J,1)-HIO(J,K)*SQ(K,1)
    CALL CHOL2(MI,LTI,1,HII,SI)
    DO 160,J=1,MI
    THETA1(J)=THETA1(J)+SI(J,1)
160 IF(SMAX.LT.ABS(SI(J,1))) SMAX=ABS(SI(J,1))
C  UPDATE
170 CALL MODEL
    CALL CHOL1(MI,LTI,M0+1,HII,GIHIO,NPDI)
    WRITE(NFILE,I) RECORD
    F=F+FI
    NPD=NPDI+NPDI
    JK=0
    DO 190,J=1,M0
    V(J)=V(J)-GOI(J)
    DO 180,K=1,MI
180 V(J)=V(J)+HIO(K,J)*GI(K)
    DO 190,K=1,J
    JK=JK+1
    H00(JK)=H00(JK)+HOOI(JK)
    DO 190,L=1,MI
190 H00(JK)=H00(JK)-HIO(L,J)*HIO(L,K)
```

```

CALL MONITR(I)
200 CONTINUE
C ALL PLOTS PROCESSED
  FIND(M,FILE*1)
  CALL MONITR(O)
C CONVERGENCY TEST
  IF(SMAX.LT.EPS.AND.ABS(FOLD-F).LT.EPS.OR.ITER.GE.ITMAX) RETURN
C NEW ITERATION BEGINS
  ITER=ITER+1
  IF(F.LT.FOLD) GO TO 220
C NO DECREASE IN OBJECTIVE FUNCTION. MOVE BACK
  ALPHA=0.5*ALPHA
  SMAX=0.5*SMAX
  DO 210 J=1,M0
210 THETA(J)=THETA(J)-ALPHA*SO(J,1)
  GO TO 100
C COMPUTE NEW DIRECTION, AND MOVE
220 DO 225,J=1,M0
225 SO(J,1)=V(J)
  CALL CHOL1(M0,LTO,1,H00,SO,NPD0)
  CALL CHOL2(M0,LTO,1,H00,SO)
  SMAX=0.
  DO 230 J=1,M0
  THETA(J)=THETA(J)+SO(J,1)
230 IF(SMAX.LT.ABS(SO(J,1))) SMAX=ABS(SO(J,1))
  ALPHA=1.
  FOLD=F
  GO TO 100
END

```

END OF SEGMENT, LENGTH 754, NAME NEWT1

```

SUBROUTINE CHOL1(N,LT,M,A,B,NPD)
  DIMENSION A(LT),B(N,M)
  INTEGER P,Q,R
C TRIANGULARIZATION
  NPD=0
  P=0
  DO 10,I=1,N
  Q=P+1
  R=0
  DO 10,J=1,I
  X=A(P+1)
  IF(Q.GT.P) GO TO 15
  DO 20,K=Q,P
  R=R+1
20 X=X-A(K)*A(R)
15 R=R+1
  P=P+1
  IF(I.NE.J) GO TO 30
  IF(X.GT.U.) GO TO 40
  NPD=1
  X=ABS(X)
  IF(X.EQ.0.) X=1.
40 A(P)=1./SQRT(X)
  GO TO 10
30 A(P)=X*A(R)
10 CONTINUE
C SOLUTION OF L*V=B
  DO 50,J=1,M
  P=1
  DO 50,I=1,N

```

```

Q=I-1
X=B(I,J)
IF(Q.EQ.0) GO TO 55
DO 60,K=1,Q
X=X-A(P)*B(K,J)
60 P=P+1
55 B(I,J)=X*A(P)
50 P=P+1
RETURN
END

```

END OF SEGMENT, LENGTH 322, NAME CHOL1

```

SUBROUTINE CHOL2(N,LT,M,A,B)
DIMENSION A(LT),B(N,M)
DO 10,J=1,M
B(N,J)=B(N,J)*A(LT)
IF(N.EQ.1) GO TO 10
LO=LT
DO 10,I=1,N-1
LO=LO-1
L=L0
X=B(N-I,J)
DO 20,K=1,I
X=X-A(L)*B(N-K+1,J)
20 L=L-N+K
B(N-I,J)=X*A(L)
10 CONTINUE
RETURN
END

```

END OF SEGMENT, LENGTH 185, NAME CHOL2

```

SUBROUTINE MOD1
C HQ FIXED; C,W GLOBAL; A,B,V,V1,TO LOCAL
C INSERT HERE THE APPROPRIATE COMMON STATEMENT FOR MATCHING
C VARIABLES WITH THOSE IN BLANK, REC AND AUX COMMON AREAS
COMMON C,WRT /REC/ ALN,B,VRT,V1RT,TO,F,X1,GA,GP,GV,GV1,GTO,
+ HAC,HAU,HBC,HBW,HCV,HVW,HCV1,HVW1,HCTO,HUTO,HAA,HAB,HBS,
+ HAV,HBV,HVV,HAV1,HV1,HV1V1,HATC,HBT0,HVTO,HV1TO,
+ HTOTO,X2(5),N,HT(10),T(10) /AUX/ GC,GW,HCC,HCW,HWW
C FIXED PARAMETERS:
HQ=0.
C INITIAL VALUES
U,R,SP,SV,SW,SV1,STO,UA,UB,UC,UV,UW,UV1,UTO=0.
UAA,UAB,UAC,UAV,UAW,URB,UBC,URV,URW,UCC,UCV,UCW,UVV,UVW,UWW=0.
UAV1,UBV1,UCV1,UWV1,UV1V1,UATO,UBTO,UCTO,UWTO,UV1TO,UTOTO=0.
GB,GC,GV,GW,GV1,GTO=0.
HAA,HAB,HAV,HAU,HBB,HBC,HRV,HPW,HCC,HCV,HCW,HVV,HVW,HWW=0.
HAV1,HBV1,HCV1,HV1,HV1V1,HATC,HBT0,HCTO,HVTO,HUTO=0.
HV1TO,HTOTO=0.
SAV,SPV,SCV,SVV,SVW,SV1,SVTO=0.
A=EXP(ALN)
V=VRT*VRT
W=WRT*WRT
V1=V1RT*V1RT
VNE1=V
V=V1
Y=((HQ/A)+*C-1.)/C

```

```

DT=T(1)-T0
F=N*(1.837877066+2.*ALN)
YA=-1.-C*Y
GA=2.*N*C
YAA=-C*YA
IF(H0.EQ.0.) GO TO 55
YC=((C*Y+1.)*ALOG(H0/A)-Y)/C
YCC=((C*YC+Y)*(ALOG(H0/A)-2./C)+2.*Y/C)/C
GO TO 66
55 YC=1./C/C
YCC=-2.*YC/C
66 CONTINUE
YAC=-C*YC-Y
HAC=2.*N
C LOOP
DO 10 I=1,N
C AUX. VARIABLES FOR OBS. I
YOLD=Y
HS=HT(I)/A
ALOGHS=ALOG(HS)
HSC=HS*C
Y=(HSC-1.)/C
IF(I.NE.1) DT=T(I)-T(I-1)
EX=EXP(-P*DT)
EX2=EX*EX
C LIKELIHOOD COMPONENTS
UOLD=U
IF(I.NE.1) R=(-EX*W)/S
S2=(1.-EX2)*V/(2.*B)+(1.+EX2)*W-R*R
S=SQRT(S2)
U=(Y-EX*YOLD-R*U)/S
F=F+2.*(1.-C)*ALOGHS+ALOG(S2)+U*U
C DERIVATIVES WITH RESPECT TO A
UAOLD=UA
YAOLD=YA
YA=-HSC
UA=(2.*(YA-EX*YAOLD)-R*UA)/S
GA=GA+U*UA
UAAOLD=UAA
YAAOLD=YAA
YAA=C*HSC
UAA=(2.*(YAA-EX*YAAOLD)-R*UAA)/S
HAA=HAA+U*UAA+0.5*UA*UA
C DERIVATIVES W.R. TO B
RB=R*(-2.*DT-SB)
SB=((DT*EX2-0.5*(1.-EX2)/B)*V/B-2.*DT*EX2*W-R*RB)/S2
UBOLD=UB
UB=(2.*DT*EX*YOLD-RB*UOLD-R*UB)/S-U*SB
GB=GB+SB+U*UB
RBB=0.
IF(R.NE.0.) RRB=-R*SB+0.5*RB*RB/R
SRB=(2.*DT*EX2*(2.*DT*W-(DT*V+V/B)/B)+(1.-EX2)*V/(B*B*B)-.5*RB*RB
+ -R*RB)/S2-SR*SB
URB=(-2.*DT*DT*EX*YOLD-RB*UOLD-RB*UBOLD-R*URB)/S-UB*SB-0.5*U*SB
+ *SB-U*SRB
HBB=HBB+SRB+U*URB+0.5*UB*UB
C DERIVATIVES W.R. TO C
UCOLD=UC
YCOLD=YC
YC=(HSC*ALOGHS-Y)/C
UC=(2.*(YC-EX*YCOLD)-P*UC)/S
GC=GC-2.*ALOGHS+U*UC
YCCOLD=YCC
YCC=(HSC*ALOGHS*(ALOGHS-2./C)+2.*Y/C)/C
UCC=(2.*(YCC-EX*YCCOLD)-R*UCC)/S

```

6

```

HCC=HCC+U*UCC+0.5*UC*UC
C DERIVATIVES W.R. TO W
RW=R*(4./WRT-SW)
SW=(2.*WRT*(1.+EX2)-R*RW)/S2
UWOLD=UW
UW=(-RW*UOLD-R*UW)/S-U*SW
GW=GW+SW+U*UW
RWW=0.
IF(R.NE.0.) RWW=R*(-4./W-SWW)+0.5*RW*RW/R
SWW=(2.*(1.+EX2)-0.5*RW*RW-R*RWW)/S2-SW*SW
UWW=(-RWW*UOLD-R*UWOLD-R*UWW)/S-U*SW-0.5*U*SW*SW-U*SWW
HWW=HWW+SWW+U*UWW+0.5*UW*UW

C DERIVATIVES W.R. TO V1
RV1=-R*SV1
SV1=-R*RV1/S2
IF(I.EQ.1) SV1=SV1+V1RT*(1.-EX2)/(B*S2)
UV1OLD=UV1
UV1=(-RV1*UOLD-R*UV1)/S-U*SV1
GV1=GV1+SV1+U*UV1
RV1V1=0.
IF(R.NE.0.) RV1V1=-R*SV1V1+0.5*RV1*RV1/R
SV1V1=(-0.5*RV1*RV1-R*RV1V1)/S2-SV1*SV1
IF(I.EQ.1) SV1V1=SV1V1+(1.-EX2)/(P*S2)
UV1V1=(-RV1V1*UOLD-RV1*UV1OLD-R*UV1V1)/S-UV1*SV1-0.5*U*SV1*
+ SV1-U*SV1V1
HV1V1=HV1V1+SV1V1+U*UV1V1+0.5*UV1*UV1

C DERIVATIVES W.R. TO TO
RTO=-R*STO
STO=-R*RTO/S2
IF(I.EQ.1) STO=STO+EX2*(2.*B*W-V)/S2
UTOOLD=UTO
UTO=(-RTO*UOLD-R*UTO)/S-U*STO
IF(I.EQ.1) UTO=UTO-2.*R*EX*YOLD/S
GTO=GTO+STO+U*UTO
RTOTO=0.
IF(R.NE.0.) RTOTO=-R*STOTO+0.5*RTO*RTO/R
STOTO=(-0.5*RTO*RTO-R*RTOTO)/S2-STO*STO
IF(I.EQ.1) STOTO=STOTO+2.*B*(2.*B*W-V)*EX2/S2
UTOTO=(-RTOTO*UOLD-RTO*UTOOLD-R*UTOTO)/S-UTO*STO-0.5*U*STO*STO
+ -U*STOTO
IF(I.EQ.1) UTOTO=UTOTO-2.*B*R*EX*YOLD/S
HTOTO=HTOTO+STOTO+U*UTOTO+0.5*UTO*UTO

C BY-PASS V FOR QHS.1
IF(I.EQ.1) GO TO 11

C DERIVATIVES W.R. TO V
RV=-R*SV
SV=(VRT*(1.-EX2)/B-R*RV)/S2
UVOLD=UV
UV=(-RV*UOLD-R*UV)/S-U*SV
GV=GV+SV+U*UV
RVV=0.
IF(R.NE.0.) RVV=-R*SVV+0.5*RV*RV/R
SVV=((1.-EX2)/B-0.5*RV*RV-R*RVV)/S2-SV*SV
UVV=(-RVV*UOLD-RV*UVOLD-R*UVV)/S-UV*SV-0.5*U*SV*SV-U*SVV
HVV=HVV+SVV+U*UVV+0.5*UV*UV

C CROSS-DERIVATIVE A,V
UAV=(-0.5*RV*UAOLD-R*UAV)/S-0.5*UA*SV
HAV=HAV+U*UAV+0.5*UA*UV

C CROSS-DERIVATIVE B,V
RBV=0.
IF(R.NE.0.) RBV=-R*SBV+0.5*RB*RV/R
SBV=((2.*DT*EX2-(1.-EX2)/B)*VRT/B-0.5*RB*RV-R*RBV)/S2-SB*SV
UBV=(-RBV*UOLD-0.5*(PB*UVOLD+RV*UBOLD)-R*UBV)/S-0.5*(UB*SV+UV*SB
+ U*SB*SV)-U*SBV
HBV=HBV+SBV+U*UBV+0.5*UB*UV

```



```

C CROSS-DERIVATIVE C,V
UCV=(-0.5*RV*UCOLD-R*UCV)/S-0.5*UC*SV
HCV=HCV+U*UCV+0.5*UC*UV
C CROSS-DERIVATIVE V,W
RVW=0.
IF(R.NE.0.) RVW=-R*SVW+0.5*RV*RW/R
SVW=(-0.5*RV*RW-R*RVW)/S2-SV*SW
UVW=(-RVW*UCOLD-0.5*(PV*UWOLD+RW*UVOLD)-R*UVW)/S-0.5*(UV*SW+UW*SV
+U*SV*SW)-U*SVW
HVV=HVV+SVW+U*UVW+0.5*UV*UW
C CROSS-DERIVATIVE V,V1
RVV1=0.
IF(R.NE.0.) RVV1=-R*SVV1+0.5*RV*RV1/R
SVV1=(-0.5*RV*RV1-R*RVV1)/S2-SV*SV1
UVV1=(-RVV1*UCOLD-0.5*(RV*UV1OLD+RV1*UVOLD)-R*UVV1)/S-
+0.5*(UV*SV1+UV1*SV+U*SV*SV1)-U*SVV1
HVV1=HVV1+SVV1+U*UVV1+0.5*UV*UV1
C CROSS-DERIVATIVE V,TO
RVTO=0.
IF(R.NE.0.) RVTO=-R*SVTO+0.5*PV*RT0/R
SVTO=(-0.5*RV*RT0-R*RVTO)/S2-SV*ST0
UVTO=(-RVTO*UCOLD-0.5*(RV*UT0OLD+RT0*UVOLD)-R*UVTO)/S-
+0.5*(UV*ST0+UT0*SV+U*SV*ST0)-U*SVTO
HVVTO=HVVTO+SVTO+U*UVTO+0.5*UV*UTO
11 CONTINUE
C CROSS-DERIVATIVE A,B
UAB=(2.*DT*EX*YAOLD-0.5*RB*UAOLD-R*UAB)/S-0.5*UA*SB
HAB=HAB+U*UAB+0.5*UA*UB
C CROSS-DERIVATIVE A,C
YACOLD=YAC
YAC=-HSC*ALOGHS
UAC=(2.*(YAC-EX*YACOLD)-R*UAC)/S
HAC=HAC+U*UAC+0.5*UA*UC
C CROSS-DERIVATIVE A,W
UAW=(-0.5*RW*UAOLD-R*UAW)/S-0.5*UA*SW
HAW=HAW+U*UAW+0.5*UA*UW
C CROSS-DERIVATIVE A,V1
UAV1=(-0.5*RV1*UAOLD-R*UAV1)/S-0.5*UA*SV1
HAV1=HAV1+U*UAV1+0.5*UA*UV1
C CROSS-DERIVATIVE A,TO
UATO=(-0.5*RT0*UAOLD-R*UATO)/S-0.5*UA*ST0
IF(I.EQ.1) UATO=UATO-2.*B*EX*YAOLD
HATO=HATO+U*UATO+0.5*UA*UTO
C CROSS-DERIVATIVE P,C
UPC=(2.*DT*EX*YCOLD-0.5*RP*UCOLD-R*UPC)/S-0.5*UC*SB
HPC=HPC+U*UPC+0.5*UB*UC
C CROSS-DERIVATIVE B,W
RBW=0.
IF(R.NE.0.) RBW=-R*SBW+0.5*RB*RW/R
SBW=(-4.*DT*EX2*WPT-0.5*RP*PW-R*RBW)/S2-SB*SW
UBW=(-RBW*UCOLD-0.5*(RB*UWOLD+RW*UBOLD)-R*UBW)/S-0.5*(UB*SW+UW*SB
+U*SB*SW)-U*SBW
HVB=HVB+SBW+U*UBW+0.5*UB*UW
C CROSS-DERIVATIVE B,V1
RBV1=0.
IF(R.NE.0.) RBV1=-R*SBV1+0.5*RB*RV1/R
SBV1=(-0.5*RB*RV1-R*RBV1)/S2-SB*SV1
IF(I.EQ.1) SBV1=SBV1+(2.*DT*EX2-(1.-EX2)/R)*V1RT/(B*S2)
UBV1=(-RBV1*UCOLD-0.5*(RB*UV1OLD+RV1*UBOLD)-R*UBV1)/S-0.5*(UB*
+SV1+UV1*SB+U*SB*SV1)-U*SBV1
HBV1=HBV1+SBV1+U*UBV1+0.5*UB*UV1
C CROSS-DERIVATIVE B,TO
RBT0=0.
IF(R.NE.0.) RBT0=-R*SBT0+0.5*RB*RT0/R
SBT0=(-0.5*RB*RT0-R*RBT0)/S2-SB*ST0

```

```

IF(I.EQ.1) SRT0=SRT0+2.*EX2*(W-DT*(2.*R+W-V))/S2
UBT0=(-RBT0+UOLD-0.5*(RB*UTOOLD+RTO*UBOLD)-R*URT0)/S-
+ 0.5*(UB*STO+UTO*SR+U*SB*STO)-U*SRT0
IF(I.EQ.1) UBT0=UBT0-2.*(1.-B*DT)*EX*YOLD/S
HRT0=HBT0+SRT0+U*URT0+0.5*UP*UTO

```

C CROSS-DERIVATIVE C,W

```

UCW=(-0.5*RW*UCOLD-R*UCW)/S-0.5*UC*SW
HCW=HCW+U*UCW+0.5*UC*UW

```

C CROSS-DERIVATIVE C,V1

```

UCV1=(-0.5*RV1*UCOLD-R*UCV1)/S-0.5*UC*SV1
HCV1=HCV1+U*UCV1+0.5*UC*UV1

```

C CROSS-DERIVATIVE C,TO

```

UCT0=(-0.5*PT0*UCOLD-R*UCT0)/S-0.5*UC*STO
IF(I.EQ.1) UCT0=UCT0-2.*R*EX*YOLD/S
HCT0=HCT0+U*UCT0+0.5*UC*UTO

```

C CROSS-DERIVATIVE W,V1

```

RWV1=0.
IF(R.NE.0.) RWV1=-R*SWV1+0.5*RV1*RW/R
SWV1=(-0.5*RV1*RW-R*RWV1)/S2-SV1*SW
UWV1=(-RWV1*UOLD-0.5*(RV1*UWOLD+RW*UV1OLD)-R*UWV1)/S-
+ 0.5*(UV1*SW+UW*SV1+U*SV1*SW)-U*SWV1
HWV1=HWV1+SWV1+U*UWV1+0.5*UV1*UW

```

C CROSS-DERIVATIVE W,TO

```

RWT0=0.
IF(R.NE.0.) RWT0=-R*SWT0+0.5*RW*RT0/R
SWT0=(-0.5*RW*RT0-R*RWT0)/S2-SW*STO
IF(I.EQ.1) SWT0=SWT0+4.*B*EX2*WRT/S2
UWT0=(-RWT0*UOLD-0.5*(RW*UTOOLD+RTO*UWOLD)-R*UWT0)/S-0.5*(UW*STO
+ UTO*SW+U*SW*STO)-U*SWT0
HWT0=HWT0+SWT0+U*UWT0+0.5*UW*UTO

```

C CROSS-DERIVATIVE V1,TO

```

RV1TO=0.
IF(R.NE.0.) RV1TO=-R*SV1TO+0.5*RV1*RT0/R
SV1TO=(-0.5*RV1*RT0-R*RV1TO)/S2-SV1*STO
IF(I.EQ.1) SV1TO=SV1TO-2.*EX2*V1RT/S2
UV1TO=(-RV1TO*UOLD-0.5*(RV1*UTOOLD+RTO*UV1OLD)-R*UV1TO)/S-
+ 0.5*(UV1*STO+UTO*SV1+U*SV1*STO)-U*SV1TO
HV1TO=HV1TO+SV1TO+U*UV1TO+0.5*UV1*UTO

```

V=VNE1

10 CONTINUE

RETURN

END

END OF SEGMENT, LENGTH 3047, NAME MOD1

FINISH

END OF COMPILATION - NO ERRORS

S/C SUBFILE: 75 BUCKETS USED