```
100 A = 2.0
    B1 = Y
    S = Y
    A1 = Y ** 2 + 1.0
    A2 = Y * (A1 + 2.0)
    B2 = A1 + 1.0
    T = A2 / B2
140 A = A + 1.0
    A0 = A1
    A1 = A2
    A2 = Y * A1 + A * A0
    B0 = B1
    B1 = B2
    B2 = Y * B1 + A * B0
    R = S
    S = T
    T = A2 / B2
    IF (T - R .GT. TOL .OR. T - S .GT. TOL) GOTO 140
    FUNC = T
    IF (SGN .LT. 0.0) FUNC =
   *  T / (2.0 * FPI * EXP(0.5 * Y ** 2) * T - 1.0)
    RETURN
    END
```

Algorithm AS 139

Maximum Likelihood Estimation in a Linear Model from Confined and Censored Normal Data

By M. S. WOLYNETZ

Statistical Research Service, Agriculture Canada

Keywords: NORMAL DISTRIBUTION; REGRESSION; MAXIMUM LIKELIHOOD; CENSORED OBSERVATIONS; CONFINED OBSERVATIONS; INCOMPLETE OBSERVATIONS

LANGUAGE

ISO Fortran

DESCRIPTION AND PURPOSE

Dempster *et al.* (1977) proposed an iterative method, called the *EM* algorithm, for obtaining the maximum likelihood estimates from incomplete data. The procedure consists of alternately estimating the incomplete observations from the current parameter estimates and estimating the parameters from the actual and estimated observations.

Previously, Sampford and Taylor (1959) used this same procedure for finding the maximum likelihood estimates of the location parameters and the scale parameter in a two-factor factorial experiment when the data contain observations censored on the right.

This version of the algorithm explicitly extends their procedure to permit observations to be censored on the left (i.e. only upper bounds for some observations are known); to permit observations to be confined between finite limits (i.e. only finite lower and upper bounds are known for some observations); to handle any fixed effects design matrix $X_{n\times m}$ of rank $m<n$, where n is the number of observations (i.e. any experiment in which there is one homogeneous variance component σ^2).

Numerical Method and Theory

In a related paper (Wolynetz, 1979), it was assumed that each observation, before censoring or confining, arose from a $N(\mu, \sigma^2)$ distribution. Here, suppose that the independent observations, before censoring or confining, arise from several Normal distributions with possibly different means but common variance, σ^2. More specifically, let the mean of the ith observation y_i be μ_i and suppose that, prior to censoring or confining,

$$E(y_i) = \mu_i = \sum_{j=1}^{m} x_{ij}\,\alpha_j, \quad i = 1, 2, \ldots, n,$$

where n is the number of observations. In matrix notation $E(\mathbf{Y}) = \boldsymbol{\mu} = X\boldsymbol{\alpha}$ where $\mathbf{Y}$ is the vector of observations $(y_1, y_2, \ldots, y_n)'$, $\boldsymbol{\mu}$ is the vector of means $(\mu_1, \ldots, \mu_n)'$, $\boldsymbol{\alpha}$ is the vector of location parameters $(\alpha_1, \ldots, \alpha_m)'$ and X is an $n \times m$ matrix with entries x_{ij}. In this parameterization, it is assumed that the matrix $X_{n\times m}$ is of rank m $(m < n)$.

Partition the set $\{i \mid i = 1, \ldots, n\}$ into the four sets A, B, C and D defined by Wolynetz (1979). Letting $h_i = (L_i - \mu_i)/\sigma$ and $H_i = (U_i - \mu_i)/\sigma$ {see Wolynetz (1979) for definitions of L_i, U_i, w_i, r, and the functions $Q(x)$, $Q(x, y)$, $S(x)$, $S_1(x, y)$, $S_2(x, y)$, $T(x)$, $T_1(x, y)$, $T_2(x, y)$, $T_3(x, y)$}, the log-likelihood function of α and σ is

$$l(\boldsymbol{\alpha}, \sigma) = -r \log \sigma - \tfrac{1}{2}\sum_{A}\{(y_i - \mu_i)/\sigma\}^2 + \sum_{B} \log Q(-H_i) + \sum_{C} \log Q(h_i) + \sum_{D} \log Q(h_i, H_i).$$

The normal equations, evaluated at the maximum likelihood estimates, $\hat{\boldsymbol{\alpha}}$ and $\hat{\sigma}$, are

$$\left.\frac{\partial l(\boldsymbol{\alpha}, \sigma)}{\partial \alpha_k}\right|_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}, \sigma=\hat{\sigma}} = \hat{\sigma}^{-2}\sum_{A}(y_i - \hat{\mu}_i)\,x_{ik} - \hat{\sigma}^{-1}\sum_{B} S(-\hat{H}_i)\,x_{ik} + \hat{\sigma}^{-1}\sum_{C} S(\hat{h}_i)\,x_{ik} + \hat{\sigma}^{-1}\sum_{D} S_1(\hat{h}_i, \hat{H}_i)\,x_{ik} = 0, \tag{1}$$

where $k = 1, 2, \ldots, m$; and

$$\left.\frac{\partial l(\boldsymbol{\alpha}, \sigma)}{\partial \sigma}\right|_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}, \sigma=\hat{\sigma}} = -r\hat{\sigma}^{-1} + \hat{\sigma}^{-3}\sum_{A}(y_i - \hat{\mu}_i)^2 - \hat{\sigma}^{-1}\sum_{B} H_i\,S(-\hat{H}_i) + \hat{\sigma}^{-1}\sum_{C} \hat{h}_i\,S(\hat{h}_i) - \sigma^{-1}\sum_{D} S_2(\hat{h}_i, \hat{H}_i) = 0. \tag{2}$$

Using equation (3) in Wolynetz (1979), the system of equations in (1) can be rewritten

$$\sum_{i=1}^{n}\left(\hat{w}_i - \sum_{i=1}^{m}\hat{\alpha}_j x_{ij}\right) x_{ik} = 0, \quad k = 1, 2, \ldots, m,$$

or, in matrix notation,

$$(X'X)\,\hat{\boldsymbol{\alpha}} = X'\hat{\mathbf{w}},$$

where $\hat{\mathbf{w}}$ is the vector $(\hat{w}_1, \hat{w}_2, \ldots, \hat{w}_n)'$. Since X is assumed to be of rank m,

$$\hat{\boldsymbol{\alpha}} = (X'X)^{-1}X'\hat{\mathbf{w}}. \tag{3}$$

As in Wolynetz (1979),

$$\hat{\sigma}^2 = \sum_{i=1}^{n}(\hat{w}_i - \hat{\mu}_i)^2 \Big/ \{r + \sum_{B} T(-\hat{H}_i) + \sum_{C} T(\hat{h}_i) + \sum_{D} T_1(\hat{h}_i, \hat{H}_i)\}. \tag{4}$$

The iterative procedure for finding $\hat{\boldsymbol{\alpha}}$ and $\hat{\sigma}$ consists of alternately using equation (3) of Wolynetz (1979) to estimate $\{\hat{w}_i\}$ for specified values of $\hat{\boldsymbol{\alpha}}$ and $\hat{\sigma}$ and then using (3) and (4) to estimate $\hat{\boldsymbol{\alpha}}$ and $\hat{\sigma}$ from the current values of $\{\hat{w}_i\}$.

The existence of missing values poses no problem. Since the formulas given previously are suitable for any linear model, they can be applied to the original data set with the missing values omitted. On the other hand, since the matrix $X'X$ is inverted during the process of finding the maximum likelihood estimates, it is often more convenient and the results more precise if this matrix is diagonal. For factorial designs in which all of the parameters are estimable from the original data, $X'X$ can often be diagonalized by augmenting X with one row for each missing observation. The values of $\hat{\mathbf{w}}$, corresponding to these additional rows, are the maximum likelihood estimates of the missing values. Several methods by which a missing value can be specified are (i) confining the observation to be between $-\infty$ and ∞ (that is $L_i = -\infty$, $U_i = \infty$); (ii) censoring the observation to be less than ∞ (that is, $L_i = \infty$, $U_i = \infty$); (iii) censoring the observation to be greater than $-\infty$ (that is $L_i = -\infty$, $U_i = -\infty$).

Using the matrix of second partial derivatives of $l(\boldsymbol{\alpha}, \sigma)$, an estimate of the variance–covariance matrix can be obtained. If this estimate is denoted by $V(\hat{\boldsymbol{\alpha}}, \hat{\sigma})$, the elements in $G(\hat{\boldsymbol{\alpha}}, \hat{\sigma}) = V^{-1}(\hat{\boldsymbol{\alpha}}, \hat{\sigma})$ are

$$g_{kj} = g_{jk} = \hat{\sigma}^{-2}\{\sum_A x_{ij} x_{ik} + \sum_B x_{ij} x_{ik} T(-\hat{H}_i) + \sum_C x_{ij} x_{ik} T(\hat{h}_i)$$
$$+ \sum_D x_{ij} x_{ik} T_1(\hat{h}_i, \hat{H}_i)\} \quad (k = 1, 2, \ldots, m;\ j = 1, 2, \ldots, m);$$
$$g_{m+1,j} = g_{j,m+1} = \hat{\sigma}^{-2}[\sum_A \{(y_i - \hat{\mu}_i)/\hat{\sigma}\} x_{ij} + \sum_B \hat{H}_i T(-\hat{H}_i) x_{ij}$$
$$+ \sum_C \hat{h}_i T(\hat{h}_i) x_{ij} - \sum_D [T_3(\hat{h}_i, \hat{H}_i) + S_1(\hat{h}_i, \hat{H}_i)] x_{ij}] \quad (j = 1, 2, \ldots, m);$$
$$g_{m+1,m+1} = \hat{\sigma}^{-2}[r + \sum_A \{(y_i - \hat{\mu}_i)/\hat{\sigma}\}^2 + \sum_B \hat{H}_i^2 T(-\hat{H}_i) + \sum_C \hat{h}_i^2 T(\hat{h}_i) - \sum_D T_2(\hat{h}_i, \hat{H}_i)].$$

Structure

SUBROUTINE REGRES (N, Y1, Y2, P, MPLONE, X, ROWX, COLX, W, LENW, VCOV, WORK, LENWRK, ALPHA, TOL, MAXITS, IFAULT)

Formal parameters

N	Integer	input:	the number of observations n
Y1, *Y2*, *P*	Real array (*N*), Real array (*N*), Real array (*N*)	input:	if $P(i) = 0$, the ith observation is completely specified in $Y1(i)$; if $P(i) = -1$, the ith observation is censored on the left at $Y1(i)$; if $P(i) = 1$, the ith observation is censored on the right at $Y1(i)$; if $P(i) = 2$, the ith observation is confined between the two finite limits $Y1(i)$ and $Y2(i)$
		output:	if $P(i) = 2$ and $\lvert Y1(i) - Y2(i)\rvert < \lvert Y1(i)\rvert . QLIMIT$, the value of $P(i)$ is set to 0; otherwise the value of $P(i)$ is not changed
MPLONE	Integer	input:	the total number of parameters to be estimated (i.e. $m+1$)
X	Real array (*ROWX, COLX*)	input:	the design matrix $X(i, j)$ contains the coefficient of the jth location parameter for the ith observation
ROWX	Integer	input:	the number of rows of X (the program expects $ROWX \geqslant n$)
COLX	Integer	input:	the number of columns of X (the program expects $COLX \geqslant m$)
W	Real array (*LENW*)	work:	
LENW	Integer	input:	the value of *LENW* must be at least $m+n$
VCOV	Real array (*LENWRK*)	output:	if the procedure converged to the maximum likelihood estimates, the first $(m+1) \times (m+1)$

			positions contain an estimate of the variance–covariance matrix of these estimates (see also *IFAULT* conditions -5 and -6)
WORK	Real array (*LENWRK*)	work:	
LENWRK	Integer	input:	the value of *LENWRK* must be at least $m \times n$
ALPHA	Real array (*MPLONE*)	input:	if *ALPHA* (*MPLONE*) $\leqslant 0{\cdot}0$, the subroutine calculates initial parameter estimates; if *ALPHA*(*MPLONE*) $> 0{\cdot}0$, it contains the initial estimate of σ and *ALPHA*(j) contains an initial estimate of the jth location parameter for $j = 1, 2, \ldots, m$
		output:	the most recent parameter estimates before exit from the subroutine
TOL	Real array (*MPLONE*)	input:	convergence to the maximum likelihood parameter estimates has occurred when the absolute value of the difference between consecutive estimates of the jth parameter is less than *TOL*(j) for $j = 1, 2, \ldots, m+1$
MAXITS	Integer	input:	the maximum number of iterations allowed
IFAULT	Integer	output:	failure indicator

Failure Indications

Value of *IFAULT*	*Meaning*
-1	maximum number of iterations reached and convergence has not been obtained
-2	for a confined observation, $Y1(i) > Y2(i)$
-3	at some iteration, for a confined observation $\lvert\Phi\{(Y1(i)-\mu_i)/\sigma\}-\Phi\{(Y2(i)-\mu_i)/\sigma\}\rvert < QLIMIT$, where Φ is the cumulative normal probability function and $\mu_i = \Sigma\, x_{ij}\,\alpha_j$ (summing over j from 1 to m) and σ are the current parameter estimates: when this condition is encountered, it is usually during the first iteration when the calling program has provided initial parameter estimates; the problem usually can be overcome by resubmitting the data but allowing the subroutine to calculate starting parameter estimates
-4	number of completely specified plus confined observations is less than $m+1$
-5	the matrix $X'X$ is not positive definite, as determined by subroutine *SYMINV*, a matrix inversion procedure (Healy, 1968b); the values of *NULLTY* and *IFAULT*, returned by *SYMINV*, are placed in the first two positions of the array *VCOV* before returning to the calling program
-6	the estimate of the variance–covariance matrix is not positive definite, as determined by subroutine *SYMINV* (Healy, 1968b); the values of *NULLTY* and *IFAULT*, returned by *SYMINV*, are placed in the first two positions of the array *VCOV* before returning to the calling program
-7	*ROWX* is less than n
-8	*COLX* is less than m

-9	*LENW* is less than $m+n$
-10	*LENWRK* is less than $m \times n$
>0	number of iterations needed for convergence

Auxiliary algorithm

The subroutine *REGRES* calls the subroutine *RMILLS(X, F, RLIMIT)*, as described by Wolynetz (1979), which is a modification of AS 17 (Swan 1969b).

The subroutine *REGRES* also calls subroutine *SYMINV(A, N, C, W, NULLTY, NA, NC, NW, IFAULT)*. This routine is almost the same as AS 7 (Healy, 1968b). To conform to ISO Fortran, the variables *NA*, *NC* and *NW* have been added to the argument list and are used to dimension the arrays *A*, *C* and *W*, respectively.

Subroutine *SYMINV* calls the subroutine *CHOL(A, N, U, NULLTY, NA, NU, IFAULT)*. The latter routine differs from AS 6 (Healy, 1968a) in that to conform to ISO Fortran, the variables *NA* and *NU* have been added to the argument list and are used to dimension to arrays *A* and *U*, respectively.

Subroutine *REGRES* also calls the subroutine *UNPACK(X, N, LENX)*. This subroutine expands a symmetric matrix of order N, stored in lower triangular form in the first $N(N+1)/2$ positions of the real array X into a matrix, using the first N^2 positions. Although not tested in subroutine *UNPACK*, *LENX*, the length of array X, must be at least N^2. (The argument passed by *REGRES* to *UNPACK* satisfies this condition.)

Constants

The constant *QLIMIT* (see condition under which *IFAULT* is set to -3 and also description of argument P) has been set to 10^{-5}. The constant *RLIMIT* {third argument for subroutine *RMILLS*, see section on "Auxiliary algorithms" and Wolynetz (1979)} has also been set to 10^{-5}.

Timing

The computer times required to analyse data from several design matrix configurations are shown in Table 1. Some general patterns were observed among these results. Within a row the computer time increased between 52 and 88 per cent when the cut-off value decreased from 1·281 to 0·525 and between 55 and 78 per cent when the cut-off value decreased from 0·525 to 0·0; however, the relative change decreased as the number of parameters increased. For $n = 32$ at all levels of censoring, the addition of one extra parameter increased the computer time by between 18 and 39 per cent. Doubling the number of location parameters from three to six increased the computer time by approximately 120, 90 and 70 per cent for $U_i = 1{\cdot}281$, $0{\cdot}525$, $0{\cdot}0$ respectively, for both $n = 32$ and $n = 64$. Doubling the sample size from $n = 32$ to $n = 64$ increased the computer time by between 77 and 87 per cent. Finally, for given m and n, no reliable difference in computer time was noticed between when the matrix $X'X$ is diagonal and when it is not.

Accuracy

This version of the algorithm was tested on a 32-bit machine. The maximum likelihood estimates of some of the test cases in Table 1 were evaluated using a single precision version of Powell's method (Powell, 1964) in order to verify both the correctness of the program and to assess the numerical accuracy. For the scale parameter, there was always agreement to at least three significant figures; for the location parameters, there was agreement to at least three significant figures for that parameter with the largest absolute value and to at least the corresponding digit for the other parameters. An earlier and less general version of this procedure was run on a different 32-bit machine with the arithmetic being done in double precision. Better agreement was obtained.

TABLE 1

Typical computer times†

Design	Characterization of $X'X$	Sample size n	Number of location parameters m	Censoring scheme‡		
				1·281 (0·1)	0·525 (0·3)	0·0 (0·5)
16 reps of 2^1	diag.	32	2	0·026	0·049	0·087
8 reps of 2^2	diag.	32	3	0·036	0·061	0·108
4 reps of 2^3	diag.	32	4	0·050	0·078	0·129
2 reps of 2^4	diag.	32	5	0·061	0·096	0·157
1 rep of 2^5	diag.	32	6	0·077	0·117	0·185
16 reps of 2^2	diag.	64	3	0·065	0·111	0·193
2 reps of 2^5	diag.	64	6	0·144	0·211	0·328
8 reps of 4^1§	block diag.	32	4	0·046	0·074	0·143¶
8 reps of 4‖	diag.	32	4	0·048	0·076	0·147¶
Polynomial	non-orthog.	32	4	0·045	0·076	0·138

† All calculations performed on an IBM 370/168 computer operating under OS/VS2. The program was compiled using the Fortran G1 compiler. The time shown represent the total computer time in minutes to find the maximum likelihood estimates of the parameters for each of the 100 samples of size n. All simulations involving samples of size 32 based on the same generated values. Because of varying demands on the system, times were expected to vary within approximately 0·005 min.

‡ All observations were Type I censored on the right at the value shown. The number in parentheses is the probability that a $N(0, 1)$ variate is greater than the cut-off value.

§ Parameterized as a factorial experiment with one factor (i.e. means for four groups parameterized as $\alpha_1+\alpha_2$, $\alpha_1+\alpha_3$, $\alpha_1+\alpha_4$ and $\alpha_1-\alpha_2-\alpha_3-\alpha_4$).

‖ Means parameterized as α_1, α_2, α_3 and α_4.

¶ In three of the 100 simulated cases, convergence was not obtained because all eight observations within a group were censored (i.e. all generated values were greater than 0·0); hence the computer time is somewhat higher than the other cases with $m = 4$ because program continued until *MAXITS* iterations (set at 100 in these studies) were executed.

Better precision can be obtained by declaring the accumulating variables such as *TEMP*, *SUM2*, *YMEAN* to be *DOUBLE PRECISION* (and making any other necessary changes such as replacing the call to *SORT* and *EXP* by *DSQRT* and *DEXP* as some compilers require). When the data contain confined observations, the accuracy also depends upon the precision of the basic external function *EXP*.

Several test cases were run in which the observations were permuted; the estimates agreed to seven significant digits. Other test cases were reparameterized and rerun (for example, the second and third last rows in Table 1). The estimates of the scale parameter agreed to at least five significant digits and the estimates of the equivalent location parameters with the largest absolute value agreed to at least five significant digits.

RELATED ALGORITHMS

If $m = 1$, either AS 16 (Swan, 1969a) or the algorithm given by Wolynetz (1979) could be used to find the maximum likelihood estimates.

ACKNOWLEDGEMENTS

This work was partially supported by the National Research Council of Canada, the Province of Ontario Scholarship Fund and the University of Waterloo.

References

DEMPSTER, A. P., LAIRD, N. M. and RUBIN, D. B. (1977). Maximum likelihood from incomplete data via the *EM* algorithm (with Discussion). *J. R. Statist. Soc.* B, **39**, 1–38.

HEALY, M. J. R. (1968a). Algorithm AS 6. Triangular decomposition of a symmetric matrix. *Appl. Statist.*, **17**, 195–197.

—— (1968b). Algorithm AS 7. Inversion of a positive semi-definite symmetric matrix. *Appl. Statist.*, **17**, 198, 196, 199 (p. 196 published out of sequence, See *Appl. Statist.*, **18**, 118).

POWELL, M. J. D. (1964). An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer J.*, **7**, 155–162.

SAMPFORD, M. R. and J. TAYLOR (1959). Censored observations in randomized block experiments. *J. R. Statist. Soc.* B, **21**, 214–237.

SWAN, A. V. (1969a). Algorithm AS 16. Maximum likelihood estimation from grouped and censored Normal data. *Appl. Statist.*, **18**, 110–114.

—— (1969b). Algorithm AS 17. The reciprocal of Mills's ratio. *Appl. Statist.*, **18**, 115–116.

WOLYNETZ, M. S. (1979). AS 138. Maximum likelihood estimation from confined and censored Normal data. *Appl. Statist.*, **28**, 185–195.

```
      SUBROUTINE REGRES(N, Y1, Y2, P, MPLONE, X, ROWX, COLX, W, LENW,
     *  VCOV, WORK, LENWRK, ALPHA, TOL, MAXITS, IFAULT)
C
C        ALGORITHM AS 139  APPL. STATIST. (1979) VOL.28, NO.2
C
C        COMPUTE MAXIMUM LIKELIHOOD ESTIMATES
C        FROM A LINEAR MODEL WITH NORMAL HETEROGENEOUS VARIANCE.
C        THE DESIGN MATRIX MUST BE NON-SINGULAR.  THE DEPENDENT
C        VARIABLE MAY INCLUDE OBSERVATIONS CENSORED IN EITHER TAIL
C        AND/OR OBSERVATIONS CONFINED BETWEEN FINITE LIMITS.
C
C
      INTEGER ROWX, COLX, P(N)
      DIMENSION X(ROWX, COLX), TOL(MPLONE), Y1(N), Y2(N), ALPHA(MPLONE)
      DIMENSION VCOV(LENWRK), WORK(LENWRK), W(LENW)
      DATA QLIMIT /0.00001/, RLIMIT /0.00001/
      DATA C /0.39894228/
C
C        CHECK ARRAY SIZES, ETC
C
      IFAULT = -7
      IF (ROWX .LT. N) RETURN
      IFAULT = -8
      IF (COLX .LT. M) RETURN
      IFAULT = -9
      IF (LENW .LT. (M + N)) RETURN
      IFAULT = -10
      IF (LENWRK .LT. (M * N)) RETURN
C
C        INITIALIZATION
C
      M = MPLONE - 1
C
C        COMPUTE X'X IN LOWER TRIANGULAR FORM
C
      II = 0
      DO 53 I = 1, M
      DO 50 J = 1, I
      TEMP = 0.0
      DO 40 K = 1, N
   40 TEMP = TEMP + X(K, I) * X(K, J)
      II = II + 1
      VCOV(II) = TEMP
   50 CONTINUE
   53 CONTINUE
      CALL SYMINV(VCOV, M, WORK, W, NUL, LENWRK, LENWRK, LENW, IFAULT)
      IF (IFAULT .NE. 0) GOTO 60
      IF (NUL .EQ. 0) GOTO 70
   60 VCOV(2) = IFAULT
      VCOV(1) = NUL
      IFAULT = -5
      RETURN
```

```
C         MATRIX NON-SINGULAR AND INVERSE OBTAINED
C         COMPUTE (X'X)INVERSE * X
C         FOLLOWING SCHEME USED TO REDUCE NUMBER OF STORAGE ARRAYS
C         NEEDED, EXPAND FROM TRIANGULAR TO SQUARE MATRIX
C
   70 CALL UNPACK(WORK, M, LENWRK)
C
C         DO MULTIPLICATION - ONE ROW AT A TIME - STARTING
C         WITH THE LAST ONE
C
      JJ = N * M
      II = M * M
      DO 220 I = 1, M
      II = II - M
      DO 200 J = 1, N
      TEMP = 0.0
      DO 170 K = 1, M
      IIK = II + K
      TEMP = TEMP + WORK(IIK) * X(J, K)
  170 CONTINUE
      W(J) = TEMP
  200 CONTINUE
      DO 210 J = 1, N
      IJ = N + 1 - J
      WORK(JJ) = W(IJ)
      JJ = JJ - 1
  210 CONTINUE
  220 CONTINUE
C
      XSIG = ALPHA(MPLONE)
      IF (XSIG .GT. 0.0) GOTO 500
C
C         NO ACCEPTABLE INITIAL VALUE FOR SIGMA HAS BEEN INPUT,
C         OBTAIN INITIAL ESTIMATES FROM EXACTLY SPECIFIED
C         OBSERVATIONS ONLY (ALTHOUGH MATRIX BASED ON ALL
C         OBSERVATIONS) AND CONFINED OBSERVATIONS
C
      II = -N
      DO 300 I = 1, M
      II = II + N
      TEMP = 0.0
      DO 280 J = 1, N
      IIJ = II + J
      IPT = P(J)
      IF (IPT .EQ. 0) GOTO 270
      IF (IPT .EQ. 2) TEMP = TEMP + WORK(IIJ) * (Y1(J) + Y2(J)) * 0.5
      GOTO 280
  270 TEMP = TEMP + WORK(IIJ) * Y1(J)
  280 CONTINUE
      ALPHA(I) = TEMP
  300 CONTINUE
C
C         CALCULATE INITIAL ESTIMATE OF SIGMA
C
      SUM2 = 0.0
      TEMP = 0.0
      DO 350 I = 1, N
      IPT = P(I)
      IF (IABS(IPT) .EQ. 1) GOTO 350
      DEMP = Y1(I)
      IF (IPT .EQ. 2) DEMP = (DEMP + Y2(I)) * 0.5
      DO 320 J = 1, M
  320 DEMP = DEMP - ALPHA(J) * X(I, J)
      SUM2 = SUM2 + DEMP ** 2
      TEMP = TEMP + 1.0
  350 CONTINUE
      XSIG = SQRT(SUM2 / TEMP)
C
C         COMPUTE SOME CONSTANTS NEEDED THROUGHOUT
C
  500 R = 0.0
      R2 = 0.0
```

```
      IFAULT = -2
      DO 600 I = 1, N
      IPT = P(I)
      IF (IPT .EQ. 0) GOTO 550
      IF (IPT .EQ. 2 .AND. ABS(Y1(I) - Y2(I)) .LE.
     *  QLIMIT * ABS(Y1(I))) GOTO 540
      IF (IPT .NE. 2) GOTO 600
      R2 = R2 + 1.0
      IF (Y1(I) .LT. Y2(I)) GOTO 600
      RETURN
  540 P(I) = 0
  550 R = R + 1.0
      W(I) = Y1(I)
  600 CONTINUE
      I = R + R2 + 0.01
      IFAULT = -4
      IF (I .LT. MPLONE) RETURN
      IFAULT = 0
C
C        START OF ITERATION PROCEDURE
C
  620 TD = R
      SUM2 = 0.0
C
C        COMPLETE W-VECTOR
C
      DO 1000 I = 1, N
      IPT = P(I)
      YMEAN = 0.0
      DO 650 J = 1, M
  650 YMEAN = YMEAN + ALPHA(J) * X(I, J)
      IF (IPT .EQ. 0) GOTO 990
C
C        OBSERVATION NOT EXACTLY SPECIFIED
C
      TEMP = (Y1(I) - YMEAN) / XSIG
      IF (IPT - 1) 750, 700, 800
C
C        OBSERVATION CENSORED FROM ABOVE - LOWER BOUND KNOWN
C
  700 CALL RMILLS(TEMP, F, RLIMIT)
      W(I) = YMEAN + XSIG * F
      TD = TD + F * (F - TEMP)
      GOTO 990
C
C        OBSERVATION CENSORED FROM BELOW - UPPER BOUND KNOWN
C
  750 CALL RMILLS(-TEMP, F, RLIMIT)
      W(I) = YMEAN - XSIG * F
      TD = TD + F * (F + TEMP)
      GOTO 990
C
C        OBSERVATION CONFINED TO LIE BETWEEN TWO FINITE LIMITS
C
  800 YN = EXP(-0.5 * TEMP ** 2) * C
      CALL RMILLS(TEMP, F, RLIMIT)
      YQ = YN / F
      TMPU = (Y2(I) - YMEAN) / XSIG
      YNU = EXP(-0.5 * TMPU ** 2) * C
      CALL RMILLS(TMPU, FU, RLIMIT)
      YQU = YNU / FU
      TINT = YQ - YQU
      IF (TINT .GE. QLIMIT) GOTO 820
C
C        AFTER STANDARDIZING, UPPER AND LOWER LIMITS RESULT IN
C        SAME PROBABILITY INTEGRAL
C
      IFAULT = -3
      RETURN
  820 A = (YN - YNU) / TINT
      W(I) = YMEAN + XSIG * A
      TD = TD + (A ** 2 + (TMPU * YNU - TEMP * YN) / TINT)
```

```
C         CALCULATE RESIDUAL SUM OF SQUARES
C
  990 SUM2 = SUM2 + (W(I) - YMEAN) ** 2
 1000 CONTINUE
C
C         UPDATE PARAMETER ESTIMATES - STORE IN END OF W-VECTOR
C
      JJ = -N
      DO 1200 J = 1, M
      JJ = JJ + N
      TEMP = 0.0
      DO 1100 I = 1, N
      JJI = JJ + I
      TEMP = TEMP + WORK(JJI) * W(I)
 1100 CONTINUE
      NJ = N + J
      W(NJ) = TEMP
 1200 CONTINUE
      NJ = N + MPLONE
      W(NJ) = SQRT(SUM2 / TD)
C
C         TEST FOR CONVERGENCE
C
      DO 1300 J = 1, MPLONE
      NJ = N + J
      IF (ABS(ALPHA(J) - W(NJ)) .GE. TOL(J)) GOTO 1400
 1300 CONTINUE
C
C         IF WE REACH HERE, CONVERGENCE OBTAINED
C
      IJ = IFAULT
      IFAULT = -1
C
C         UPDATE VALUES
C
 1400 DO 1450 J = 1, MPLONE
      NJ = N + J
      ALPHA(J) = W(NJ)
 1450 CONTINUE
      XSIG = ALPHA(MPLONE)
      IFAULT = IFAULT + 1
      IF (IFAULT .EQ. 0) GOTO 1600
      IF (IFAULT .LE. MAXITS) GOTO 620
      IFAULT = -1
      RETURN
C
C         CONVERGENCE OBTAINED - COMPUTE VARIANCE-COVARIANCE
C         MATRIX, INITIALIZE WORK ARRAY
C
 1600 II = MPLONE * (MPLONE + 1) / 2
      DO 1650 I = 1, II
 1650 WORK(I) = 0.0
      DO 2500 I = 1, N
      IPT = P(I)
      YS = Y1(I)
      DO 1680 J = 1, M
 1680 YS = YS - ALPHA(J) * X(I, J)
      YS = YS / XSIG
      JJ = 0
      IF (IPT .NE. 0) GOTO 1900
C
C         EXACTLY SPECIFIED OBSERVATION
C
      DO 1750 K = 1, M
      DO 1720 J = 1, K
      JJ = JJ + 1
      WORK(JJ) = WORK(JJ) + X(I, K) * X(I, J)
 1720 CONTINUE
      KK = II + 1 - K
      WORK(KK) = WORK(KK) + YS * X(I, K)
 1750 CONTINUE
      WORK(II) = WORK(II) + 1.0 + YS ** 2
```

```
      GOTO 2500
 1900 IF (IPT - 1) 2100, 2000, 2300
C
C         OBSERVATION CENSORED FROM ABOVE - LOWER BOUND KNOWN
C
 2000 CALL RMILLS(YS, F, RLIMIT)
      TEMP = F * (F - YS)
      GOTO 2150
C
C         OBSERVATION CENSORED FROM BELOW - UPPER BOUND KNOWN
C
 2100 CALL RMILLS(-YS, F, RLIMIT)
      TEMP = F * (F + YS)
C
C         ROUTINE FOR CENSORED OBSERVATIONS
C
 2150 DO 2190 K = 1, M
      DO 2170 J = 1, K
      JJ = JJ + 1
      WORK(JJ) = WORK(JJ) + X(I, J) * X(I, K) * TEMP
 2170 CONTINUE
      KK = II + 1 - K
      WORK(KK) = WORK(KK) + YS * X(I, K) * TEMP
 2190 CONTINUE
      WORK(II) = WORK(II) + YS ** 2 * TEMP
      GOTO 2500
C
C         OBSERVATION CONFINED BETWEEN TWO FINITE LIMITS
C
 2300 YN = EXP(-0.5 * YS ** 2) * C
      CALL RMILLS(YS, F, RLIMIT)
      YQ = YN / F
      YSU = YS + (Y2(I) - Y1(I)) / XSIG
      CALL RMILLS(YSU, FU, RLIMIT)
      YNU = EXP(-0.5 * YSU ** 2) * C
      YQU = YNU / FU
      TINT = YQ - YQU
      A = (YN - YNU) / TINT
      B = (YNU * YSU - YN * YS) / TINT
      TEMP = A ** 2 + B
      DO 2350 K = 1, M
      DO 2330 J = 1, K
      JJ = JJ + 1
      WORK(JJ) = WORK(JJ) + X(I, J) * X(I, K) * TEMP
 2330 CONTINUE
      TEMP = (YS ** 2 * YN - YSU ** 2 * YNU) / TINT
      KK = II + 1 - K
      WORK(KK) = WORK(KK) - (TEMP + A * B) * X(I, K)
 2350 CONTINUE
      TEMP = (YS ** 3 * YN - YSU ** 3 * YNU) / TINT
      WORK(II) = WORK(II) - TEMP + B ** 2
 2500 CONTINUE
C
C         INVERT THE MATRIX
C
      CALL SYMINV(WORK, MPLONE, VCOV, W, NUL, LENWRK,
     *  LENWRK, LENW, IFAULT)
      IF (IFAULT .EQ. 0 .AND. NUL .EQ. 0) GOTO 2550
      VCOV(2) = IFAULT
      VCOV(1) = NUL
      IFAULT = -6
      RETURN
C
C         RESTORE ITERATION COUNTER
C
 2550 IFAULT = IJ
C
C         MULTIPLY BY SIGMA-SQUARED
C
      TEMP = XSIG ** 2
      DO 2580 I = 1, II
 2580 VCOV(I) = VCOV(I) * TEMP
```

```
C          UNPACK THE MATRIX
C
      CALL UNPACK(VCOV, MPLONE, LENWRK)
      RETURN
      END
C
      SUBROUTINE UNPACK(X, N, LENX)
C
C          ALGORITHM AS 139.1  APPL. STATIST. (1979) VOL.28, NO.2
C
C          THIS SUBROUTINE EXPANDS A SYMMETRIC MATRIX STORED IN LOWER
C          TRIANGULAR FORM IN THE FIRST N*(N+1)/2 POSITIONS OF X
C          INTO A MATRIX USING THE FIRST N*N POSITIONS
C
C          LENX = THE LENGTH OF VECTOR X - MUST BE NOT LESS THAN N*N
C
      DIMENSION X(LENX)
      NSQ = N * N
      II = NSQ
      JJ = N * (N + 1) / 2
C
C          STORE LAST ROW
C
      DO 10 I = 1, N
      X(II) = X(JJ)
      II = II - 1
      JJ = JJ - 1
   10 CONTINUE
      DO 80 I = 2, N
C
C          OBTAIN UPPER PART OF MATRIX FROM PART ALREADY SHIFTED
C
      IJ = I - 1
      KK = NSQ + 1 - I
      DO 50 J = 1, IJ
      X(II) = X(KK)
      II = II - 1
      KK = KK - N
   50 CONTINUE
C
C          OBTAIN LOWER PART OF MATRIX FROM
C          ORIGINAL TRIANGULAR STORAGE
C
      IJ = N - IJ
      DO 70 J = 1, IJ
      X(II) = X(JJ)
      II = II - 1
      JJ = JJ - 1
   70 CONTINUE
   80 CONTINUE
      RETURN
      END
```

Algorithm AS 140

Clustering the Nodes of a Directed Graph

By GARY W. OEHLERT†

Yale University

Keywords: CLUSTERING; DIRECTED GRAPH; MAXIMUM LIKELIHOOD; TRANSFER ALGORITHM

LANGUAGE

ISO Fortran

† Development of this algorithm was partially supported by National Science Foundation Grant DCR75–08374.