AN ALGORITHM FOR THE COMPUTATION OF GENERALIZED LIKELIHOOD OR SELF-CRITIC. (U) RENSSELAER POLYTECHNIC INST TROY NY SCHOOL OF MANAGEMENT T A DELAHANTY ET AL.
Rensselaer Polytechnic Institute
School of Management

Working Paper No. 37-810-P26
AN ALGORITHM FOR THE COMPUTATION OF GENERALIZED LIKELIHOOD OR SELF-CRITICAL ESTIMATORS FOR BINARY DATA

by

T. A. Delahanty
and
A. S. Paulson

School of Management
Rensselaer Polytechnic Institute
Troy, NY 12180-3590
(518)266-6586

The view, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

Note: This paper is not to be quoted without the permission of the author(s).
For a list of Working Papers available from the School of Management, please contact Sheila Chao, School of Management.
An Algorithm for the Computation of Generalized Likelihood or Self-Critical Estimators for Binary Data

by

T.A. Delehanty*

and

A.S. Paulson*
Rensselaer Polytechnic Institute

Documentation for Self-Critical Binary Program

1. Purpose
2. Specification (FORTRAN)
3. Description
4. Numerical Method
5. Parameters
6. Error Indicators and Warnings
7. Auxiliary Routines
8. References
9. Storage
10. Precision, Machine Dependent Constants
11. Further Comments
12. Example
1. **Purpose**

Subroutine BINARY is an implementation of the self-critical estimation procedure of Paulson, Presser and Lawrence (1963). The logistic, Gaussian or Type I extreme value distribution may be selected as tolerance distribution. Estimates are expressed in location-scale form on entry and exit, but results may be printed out in regression form, location-scale form, or in both forms (see Section 3 for a description of the two parametrizations). It is possible to hold location parameters constant during the estimation procedure. Estimation is accomplished by a Newton-Raphson method.

2. **Specification (FORTRAN)**

SUBROUTINE BINARY (N, IX, X, IA, NPAR, ISUB, ISTART, IDEF, IDIST, C, RELTOL, ABSTOL, MAXIT, IPRINT, IFLAG, BETA, XLOGL, ICOV, COV, LMEM, MEMORY, IFAULT)

3. **Description**

The routine is applicable to the following modeling situation: For \( i = 1, \ldots, n \) (\( n \) is the sample size), let \( v_i \) be the stress variable, \( a_i \) a zero-one indicator of withstand or failure, and \( X_i = (x_{i1}, \ldots, x_{ip})^T \) a column vector of covariates. A constant is incorporated as a covariate identically equal to unity. The case \( p = 0 \) is possible, but should be rare.

The tolerance distribution and density, \( F(v_i) \) and \( f(v_i) \) respectively, depend on the covariates, a scale parameter \( \sigma \), and a vector of location parameters \( \beta = (\beta_1, \ldots, \beta_p)^T \) as follows, where
\[ u_i = \frac{(v_i - \beta'X_i)}{\sigma}, \]

- logistic, \[ F(v_i) = \frac{\exp(u_i)}{1+\exp(u_i)}, \]
\[ f(v_i) = \frac{1}{\sigma} \frac{\exp(u_i)}{(1+\exp(u_i))^2}; \]

- Gaussian, \[ F(v_i) = \Phi(u_i) \]
\[ (\Phi \text{ is the standard Normal distribution function}), \]
\[ f(v_i) = \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{u_i^2}{2}\right); \]

- extreme value, \[ F(v_i) = 1 - \exp(-\exp(u_i)), \]
\[ f(v_i) = \frac{1}{\sigma} \exp(u_i - \exp(u_i)). \]

In principle, it is possible for \( \sigma \) to be negative, but this contradicts the physical notion of a stress variable.

In the method of maximum likelihood, parameters \( \theta \) are estimated by maximizing the log likelihood,
\[ L(\theta) = \sum_{i=1}^{n} \left[ a_i \log F(v_i) + (1-a_i) \log S(v_i) \right], \]
where \( S = 1 - F \). The self-critical procedure depends on a user-specified quantity \( c \), and estimates \( \theta \) by solving the system
\[ \sum_{i=1}^{n} f^c(v_i) \left[ a_i \frac{\partial}{\partial \theta} \log F(v_i) + (1-a_i) \frac{\partial}{\partial \theta} \log S(v_i) \right] = 0. \]

When \( c = 0 \), maximum likelihood estimates are obtained. As \( c \) increases from zero, increasingly robust estimators result. The sensitivity of parameter estimates to departures from model assumptions can be examined by starting with \( c = 0 \) and refitting the model for increasing values of \( c \). Negative values of \( c \) seem less useful.
4. **Numerical Method**

For the purpose of estimation, the routine (internally) reparameterizes the problem in a "regression" form. In this parameterization, \( P(\cdot) \) depends on

\[
v_i = \alpha v_i + \mu' X_i
\]

instead of

\[
u_i = \frac{v_i - \beta' X_i}{\sigma}.
\]

The reparameterization offers two advantages:

1) Computation of the necessary partial derivatives is simple;

2) For maximum likelihood estimation with a logistic tolerance distribution, it can be shown that the regression parameterization results in a concave maximization problem. We anticipate that it will have fairly good properties in the more complicated cases.

It has a disadvantage in terms of potential ill-conditioning of the Hessian (or Jacobian) matrix, so that the input data should be sensibly scaled (see Section 11). Parameters are assumed to be expressed in the easier to understand location-scale form on entry, and are transformed back to location-scale form on exit, so the user need not worry about details of reparameterization.

Let

\[
S_{\theta_1} = f^C(v_1)[a_1 \frac{\partial}{\partial \theta} \log P(v_1) + (1 - a_1) \frac{\partial}{\partial \theta} \log S(v_1)].
\]

The gradient used in Newton-Raphson iteration has \( \theta \)th component

\[
\theta = n^{-1} \sum_{i=1}^{n} S_{\theta_1}, \text{ and the Hessian has } (\theta, \theta') \text{ component } S_{\theta \theta'} = n^{-1} \sum_{i=1}^{n} \frac{\partial}{\partial \theta} \frac{\partial}{\partial \theta'} S_{\theta_1},
\]

(the scaling by \( n^{-1} \) should be helpful when \( n \) is large).
When \( c = 0 \), the estimated asymptotic covariance matrix of the estimators is
\[ n^{-1}(-H)^{-1}, \]
while when \( c \neq 0 \) it is \( n^{-1}H^{-1}VH^{-1} \), where \( V \) has \((\theta, \theta')\) component
\[ V_{\theta\theta} = n^{-1} \sum_{i=1}^{n} S_{\theta i} S_{\theta i}' \]
The estimated asymptotic covariance matrix is expressed in location-scale form by pre- and post-multiplying it by the Jacobian of the underlying parameter transformation.

Newton-Raphson iteration converges when
\[ \max_i \left| \frac{c(k) - c(k-1)}{c(k-1)} \right| < \text{ABSTOL} \]
and
\[ \max_i \left| \frac{c(k) - c(k-1)}{c(k-1)} \right| < \text{RELTOL}, \]
where superscripts represent iteration numbers and \( \text{ABSTOL} \) and \( \text{RELTOL} \) are user-supplied. The user also specifies a maximum allowable number \( \text{MAXIT} \) of iterations.

Solution of linear equations and matrix inversion is accomplished by subroutines \textsc{decomp} and \textsc{solve}, taken from Forsythe, Malcolm and Moler (1977). These routines are of high numerical quality, and provide an estimate of the condition number of the input matrix, which is often useful. In the interest of portability, iterative improvement is not used.

5. Parameters

5.1 Input Parameters

\[ N \quad \text{INTEGER} \]
Sample size. Unchanged on exit.

\[ IX \quad \text{INTEGER} \]
Row dimension of data matrix \( X \). Unchanged on exit.
X - REAL array of DIMENSION (IX, q), where q > N. Data matrix containing the dependent (stress) variable and all covariates. Each column corresponds to one observation. If parameters are held fixed, values of the dependent variable will be changed, but their input values restored. Unchanged on exit.

IA - INTEGER array of DIMENSION N. Withstand or failure is indicated for observation I according as IA(I) = 0 or IA(I) ≠ 0. Unchanged on exit.

NPAR - INTEGER. Number of parameters in the model. Unchanged on exit.

ISUB - INTEGER array of DIMENSION (NPAR). Indicates rows of data matrix X corresponding to parameters (the scale parameter is taken to correspond to the dependent variable). For instance, if ISUB(3) = 5, the third parameter corresponds to the fifth row of X. Unchanged on exit.

ISTART - INTEGER array of DIMENSION (NPAR). Indicates the status of parameters in the model, and whether a starting value is to be supplied. If ISTART(I) is equal to 0, BETA(I) is to be estimated, and its input value is to be disregarded; 1, BETA(I) is to be estimated, and its input value is to be used as starting value; 2, BETA(I) is to be held constant at its input value. (See Section 11.2 for comments on starting values; the scale of parameter is not allowed to be held constant.) Unchanged on exit.

IDEP - INTEGER. Row of dependent (stress) variable in X. Unchanged on exit.

IDIST - INTEGER. Indicates the tolerance distribution desired. If IDIST equals 1, logistic distribution will be used; 2, Gaussian distribution will be used; 3, extreme value distribution will be used. Unchanged on exit.

RELTOl - DOUBLE PRECISION.
Relative convergence tolerance for Newton-Raphson iteration
(see Section 4).
Unchanged on exit.

MAXIT - INTEGER.
Maximum allowable number of Newton-Raphson iteration.
Unchanged on exit.

IPRINT - INTEGER.
Output unit number. If IPRINT \leq 0, no output will be produced.
If IPRINT > 0, standard output will be produced on logical
output unit IPRINT.
Unchanged on exit.

IFLAG - INTEGER.
Indicator for form of output.
If IFLAG < 0, the standard output summary will be in regression
form. If IFLAG = 0, output summaries will be produced for
both regression and location-scale parameterizations. If
IFLAG > 0, the standard output summary will be in location-
scale form. (See Section 11.3).
Unchanged on exit.

5.2 Input/Output (and associated dimension) Parameters.

BETA - DOUBLE PRECISION array of DIMENSION (MPAR).
On entry, contains starting values as specified by ISTART.
On exit, contains parameter estimates, in location scale form.

XLOGL - DOUBLE PRECISION.
On exit, contains the log likelihood if c = 0 (maximum
likelihood estimation), and zero otherwise.

ICOV - INTEGER.
Row dimension for COV.
Unchanged on exit.

COV - DOUBLE PRECISION array of DIMENSION (ICOV, q), where q \geq MPAR.
On exit, estimated asymptotic covariance information for the
parameters. The diagonal contains standard errors, the strict
lower triangle correlations, and the strict upper triangle
covariances. If a parameter is held constant, all correspond-
ing entries are zero. The covariance matrix will be for the
location-scale parameterization rules IFLAG < 0, when it will
be in regression form. See Section 11.3.
5.3 Workspace (and associated dimension) parameters

**LMEM** - INTEGER.
Length of work array MEMORY, as declared in calling program unit.
**LMEM** > **NPAR** + 4* **NACT***(1+**NACT**) + 2*max(**NPIX**, 2* **NACT**), where **NACT** is the number of parameters estimated and **NPIX** = **NPAR**-**NACT** is the number of parameters held constant.
Unchanged on exit.

**MEMORY** - INTEGER array of DIMENSION (**LMEM**).
Used as workspace.

5.4 Diagnostic Parameter

**IFAULT** - INTEGER.
Unless the routine finds an error or gives a warning, **IFAULT** will be 0 on exit. See Section 6.

6. Error Indications and Warnings

Errors or warnings specified by the routine:

**IFAULT** < 0 IF **IFAULT** = -I, I > 0, then an arithmetic exception was about to occur on observation I, while computing partial derivatives. This failure should be rare. If the data have been sensibly scaled, and the starting values are not too bad (see Sections 11.1-11.2), the probable cause is a data error. The routine stops as soon as the error is detected, and output parameter values are not of interest.

**IFAULT** > 2 Insufficient workspace. This failure will occur if, on entry, N < 1, **NPAR** < 1, IX < **NPAR**, ICOV < **NPAR**, RELTOL < 0, ABSTOL < 0, MAXIT < 1, IDIST < 1, IDIST > 3, IFLAG ≤ 0 and IPRINT ≤ 0 (see Section 11.3), ISTART(I) < 0 or ISTART(I) > 2 for some I, ISTART(I) = 2 for all I, ISUB(I) ≠ IDEP for all I, or ISUB(I) = IDEP and ISTART(I) = 2 for some I. (The last restriction means that the scale parameter cannot be held constant.) The routine stops without doing any calculations.

**IFAULT** = 3 The Hessian matrix has become numerically singular during Newton-Raphson iteration. The routine transforms estimates to location-scale form, restores values of the dependent variable if parameters were held fixed, and stops. Output parameter values are not generally of interest.
The Newton-Raphson iteration did not converge to within RELTOL and ABSTOL in the specified MAXIT iterations. The routine transforms estimates to location-scale form, restores values of the dependent variable if parameters were held fixed, and stops. Output parameter values are not generally of interest.

7. Auxiliary Routines

SUBROUTINE PREPAR(IX, N, X, NPAR, BETA, ISUB, ISTART, NACT, IACT, IMAP, PAR, NFIX, IFIXED, IDEP, WORK)

Prepares for iteration by setting up some indexing and working arrays, subtracting the effects of fixed parameters from the dependent variable, and setting initial active parameter values in regression form.

SUBROUTINE NEWTON(C, MAXIT, RELTOL, ABSTOL, DIFFER, XLOGL, NACT, IACT, PAR, IPRIVOT, HESS, HESFAC, N, IX, X, IA, IPRINT, LWORK, WORK, IFAULT)

Carries out the Newton-Raphson iteration.

SUBROUTINE VARIAB(C, XLOGL, REGRES, DIFFER, VROUT, NACT, IACT, IMAP, PAR, IPRIVOT, HESS, HESFAC, ICOV, NPAR, COV, DET, N, IX, X, IA, WORK)

Computes the estimated asymptotic covariance matrix, transforms it to location-scale form if requested, and computes correlations and standard errors.

SUBROUTINE EXPOST(REGRES, IDEP, NPAR, BETA, ISUB, ISTART, NACT, PAR, IMAP, N, IX, X, NFIX, IFIXED, WORK)

Sets the output vector BETA, transforms to location-scale form if requested. If location-scale form is requested and parameters were held fixed, restore initial values of dependent variable.

SUBROUTINE RESULT(C, REGRES, IPRINT, IDIST, N, IDEP, NACT, NFIX, XLOGL, NPAR, BETA, ISUB, ISTART, ICOV, COV, DET)

Produces a standard output summary on logical output unit IPRINT.

The following routines are specific to particular tolerance distributions, and are declared EXTERNAL in BINARY. The first three, prefixed D, are passed to NEWTON and VARIAB. The next three, prefixed V, are passed to VARIAB.
SUBROUTINE DLOGBN(C, OBJECT, NACT, IACT, PAR, GRAD, HESS, N, IX, X, IA, IFault)
SUBROUTINE DGAUBN (       "       )
SUBROUTINE DEXUBN (       "       )

Compute gradient and Hessian for logistic, Gaussian and extreme value models, resp.

SUBROUTINE VLOGBN(C, NACT, IACT, PAR, V, N, IX, X, IA)
SUBROUTINE VGGAUBN(       "       )
SUBROUTINE VXGUBN(       "       )

Compute V factor of estimated asymptotic covariance matrix for logistic, Gaussian and extreme value models, resp.

The following procedures perform general numerical tasks. They have been included in the interest of portability, although equivalents exist on many computer systems.

SUBROUTINE DECOMP(NDIN, N, A, UL, COND, IPUT, WORK)
SUBROUTINE SOLVE(NDIN, N, A, B, X, IPUT)

DECOMP decomposes a matrix into LU factors and estimates its condition. SOLVE solves a linear system, using the results of DECOMP. These routines are in Forsythe, Malcolm and Moler (1977), but the present versions include an extra argument so that A and B need not be overwritten.

SUBROUTINE DNXMLT(A, IA, N1, B, IB, N2, C, IC, N3, WORK, LWORK, IFLAG, IFault)

Double precision matrix multiplication -
A(N1 x N3) = B(N1 x N2) * C(N2 x N3), where B is overwritten if IFLAG < 0 and C is overwritten if IFLAG > 0.

DOUBLE PRECISION FUNCTION ALNORM (X, UPPER)
Algorithm as 66 (Hill, 1973) to compute tail areas of the standard Normal curve.

DOUBLE PRECISION FUNCTION RMILLS(X)
Computes Z(x)/Q(x), the reciprocal of Mills' ratio, where x is a standard Normal variate. This procedure is based on procedures by Hill (1973) and Adams (1969).
8. References


9. Storage

There are no internally declared arrays.

10. Precision, Machine Dependent Constants

The routine was developed on an IBM computers. The data matrix is single precision to conserve storage. To convert to single precision, take the following steps, with the exception noted below:

1) Change all DOUBLE PRECISION declarations to REAL;

2) Replace references to double precision FORTRAN library functions with single precision versions, e.g., EXP replaces DEXP;

3) Replace double precision constants by their single precision versions, e.g., 1.0 replaces 1.0D0.

Note: In routines PREPAR and EXPOST, the dependent variable is adjusted for the effect of fixed parameters. These adjustments must be calculated in double precision.
to avoid a loss of significant digits in X which could affect subsequent computations.

Procedures DLOGBN, VLOGBN, DGAUBN, VQAUBN, DEXUBN, VEXUBN, DECOMP, ALNORM and RMILLS use machine-dependent constants whose values are set in DATA statements. These constants may have to be altered for some computers. They are pointed out by comments in the program units.

11. Further Comments

11.1 Scaling, conditioning of Hessian matrix

The user should be aware of potential problems of ill-conditioning. Because a regression-type parameterization is used, calculation of the Hessian matrix involves operations similar to the formation of $X^TX$, where $X$ is the data matrix. Unfortunately, the resulting Hessian is often rather ill-conditioned. If IPRINT $> 0$, an estimate of the Hessian's condition number is printed out at each iteration. Roughly speaking, a condition number in excess of $10^7$ is worrisome, although condition numbers in excess of $10^{10}$ can be tolerated when working in double precision on an IBM 3081. The user can and should avoid potentially excessive ill-conditioning by scaling the data matrix $X$ before calling the routine.

Although it is not known how to optimally scale a problem, it seems that a nearly ideal scaling will be achieved if nonconstant variables are transformed to the range $[-1, 1]$, centered at zero. However, such precise scaling is often tedious. It seems most important to "equilibrate" the data matrix so that all variables have roughly the same (moderate) magnitude. Such equilibration is often simply accomplished by dividing by suitable powers of 10, e.g., expressing voltages in megavolts instead of kilovolts. Centering the variables will further reduce the condition number. One often centers covariates anyway, so that the "intercept"
parameter has a clear interpretation.

11.2. Starting values

Since Newton-Raphson procedure is employed, starting values are important. If no starting values are supplied by the user (ISTART(I) = 0 for all I), the routine will use unity for the scale parameter and zero for all location parameters. These starting values will not be acceptable unless the dependent variable has been centered and scaled. It is recommended that the user employ the mean and standard deviation of the dependent variable as starting values for the "intercept" parameter (if any) and scale parameter, respectively. If the model is sequentially refit with different values of $c$, it is recommended that estimates from the most recent call be used as starting values for the next call.

11.3 Output Flags

Because the routine will generally be called sequentially with different values of $c$, it is most convenient to always express entry and exit parameter values in the same form. The location-scale form is used, because that form affords the clearest interpretation. An inconsistency arises of IFLAG < 0, for then the output estimates are in location-scale form, but the covariance matrix is in regression form. Thus, if the user wants to make separate use of the output parameters of BINARY, the routine should be called with IFLAG > 0.

It is recommended that the routine be called with IPRINT > 0. If IFLAG < 0, the requirement IPRINT > 0 is enforced. (When IFLAG < 0, the requirement is in keeping with the inconsistency mentioned above.) The standard output summary should
be sufficient for most applications. For the maximum flexibility in output and
interpretation of results, call BINARY with IPRINT > 0, IFLAG = 0.

12. Example

The following program illustrates the use of BINARY, and the standard output
produced when IPRINT > 0 and IFLAG = 0.
C******************************************************************************
C          (OPTIONALLY ROBUST) BINARY DATA ESTIMATION ROUTINE
C          LOGISTIC, EXTREME VALUE, OR GAUSSIAN TOLERANCE DISTRIBUTION
C          CAN BE USED WHEN PARAMETERS ON INPUT/OUTPUT ARE IN LOCATION-
C          SCALE FORM, BUT ESTIMATION IS CARRIED OUT USING A
C          REGRESSION-TYPE REPARAMETERIZATION. IF PRINTOUT IS DESIRED
C          (IPRINT GT 0), RESULTS CAN BE PRINTED OUT IN REGRESSION
C          FORM, LOCATION-SCALE FORM, OR BOTH. ARGUMENT IFLAG
C          SPECIFIES THE OUTPUT DESIRED.
C          FAILURE CODES -
C          IFAULT = 1 - INPUT ERROR
C          IFAULT = 2 - INSUFFICIENT WORKSPACE SUPPLIED
C          IFAULT = 3 - HESSIAN MATRIX IS NUMERICALLY SINGULAR
C          IFAULT = 4 - NO CONVERGENCE IN MAXIT ITERATIONS
C          IFAULT = -1 - AN EXCEPTION WAS ABOUT TO OCCUR WHILE
C          PROCESSING THE I TH OBSERVATION
C******************************************************************************

SUBROUTINE BINARYN, IX, X, IA, NPAR, ISUB, ISTART, IDEF, IDIST,
   C RELTOL, ABSTOL, MAXIT, IPRINT, IFLAG, BETA, XLOGL,
   I COV, IC(NPAR)
   C
   C EXTERNAL PROCEDURES FOR PARTICULAR TOLERANCE DISTRIBUTIONS
   C EXTERNAL DLOGBN, DGAUBN, DEXVBK, VLGBN, VGBWBN, VEXVB
   C
   INTEGER N, IA, I(N), NPAR, ISUB(NPAR), ISTART(NPAR), IDEF, IDIST,
   C MAXIT, IPRINT, IFLAG, IC(NPAR), LMEM, MEMORY (LMEM), IFAULT
   C
   REAL X(I,N), N
   C
   DOUBLE PRECISION C, RELTOL, ABSTOL, BETA(NPAR), XLOGL,
   C IC(NPAR)
   C
   LOCAL SCALARS
   LOGICAL REGRES
   INTEGER NACT, NFIX, INO, INO1, MACT, MFIX, MMAP, MIPVT, NPAR,
   C NHES, NHFAC, NHWORK
   C
   DOUBLE PRECISION DET, ZERO
   C
   CONSTANTS
   DATA ZERO /0.000/
   C
   C CHECK FOR INPUT ERRORS
   IFAULT = 1
   IF ( IN LT 1 OR NPAR LT 1 OR IX LT NPAR OR IC(LT NPAR) ) RETURN
   IF ( (RELTOL LE ZERO OR ABSTOL LE ZERO) OR (MAXIT LT 1) ) RETURN
   IF ( (IDIST LT 1 OR IDIST GT 3) ) RETURN
   IF ( (IFLAG LT 0 AND IPRINT LT 0) ) RETURN
   C
   C MEMORY MANAGEMENT, POSSIBLE RELATED INPUT ERRORS, CHECK
   C ISUB(*), ISTART(*) VECTORS, COUNT ACTIVE PARAMETERS.
   C NOTE SCALE PARAMETER CANNOT BE FIXED.
   NACT = 0
   INO = 0
   DO 10 1 = 1, NPAR
   INO = ISUB(1)
   IF ( (INO LT 1 OR INO GT IX) ) RETURN
   IF ( INO EQ IDEF ) INO = I
   ISTART(1)
   IF ( (INO LT 0 OR INO GT 2) ) RETURN
   IF ( INO NE 2 ) NACT = NACT + 1
   CONTINUE
   IF ( (NACT EQ 0 OR INO EQ 0) ) RETURN
   IF ( (IFLAG EQ 1 OR INO EQ 1) ) RETURN
   BINA0001
   BINA0002
   BINA0003
   BINA0004
   BINA0005
   BINA0006
   BINA0007
   BINA0008
   BINA0009
   BINA010
   BINA011
   BINA012
   BINA013
   BINA014
   BINA015
   BINA016
   BINA017
   BINA018
   BINA019
   BINA020
   BINA021
   BINA022
   BINA023
   BINA024
   BINA025
   BINA026
   BINA027
   BINA028
   BINA029
   BINA030
   BINA031
   BINA032
   BINA033
   BINA034
   BINA035
   BINA036
   BINA037
   BINA038
   BINA039
   BINA040
   BINA041
   BINA042
   BINA043
   BINA044
   BINA045
   BINA046
   BINA047
   BINA048
   BINA049
   BINA050
   BINA051
   BINA052
   BINA053
   BINA054
   BINA055
   BINA056
   BINA057
   BINA058
   BINA059
   BINA060
NFIX = NPAR - NACT

CHECK IF WORKSPACE SIZE IS ADEQUATE, ALLOCATE IT

IF AULT = 2
IND = 2 * NACT
IND1 = IND * NACT
IF (LWEN LT. NPAR + 2*NACT + IND + 2*IND1 + MAX0(2*NFIX,2*IND))
1 RETURN
MIACT = 1
MIFIX = MIACT + NACT
MNAP = MIACT + NPAR
MPVT = MNAP + NACT
MPAR = MPVT + NACT
MHESS = MPAR + IND
MHFAC = MHESS + IND
MNWORK = MHFAC + IND

PREPARE FOR ESTIMATION - SET UP SUBSCRIPT ARRAYS AND
STARTING VALUES FOR ACTIVE PARAMETERS, SUBTRACT EFFECT OF
FIXED PARAMETERS FROM DEPENDENT VARIABLE

CALL PREPAR(I, N, X, NPAR, BETA, ISUB, ISTART, NACT, 
1 MEMORY(MIACT), MEMORY(MNAP), MEMORY(MPAR), NFIX, 
2 MEMORY(MIFIX), IEP, MEMORY(MNWORK))

NEWTON-RAPHSON ITERATION WITH EXTERNAL ROUTINE PASSED FOR
FIRST AND SECOND PARTIALS

IF (IDIST .EQ. 1) CALL NEWTONC, MAXIT, RETOL, ABSTOL, DLOGBN, 
1 XLOGL, NACT, MEMORY(MIACT), MEMORY(MPAR), MEMORY(MIFIX), 
2 MEMORY(MHESS), MEMORY(MHAC), N, IX, X, IA, IPRINT, 2*NACT, 
3 MEMORY(MNWORK), IFAULT
IF (IDIST .EQ. 2) CALL NEWTONC, MAXIT, RETOL, ABSTOL, DGAUBBN, 
1 XLOGL, NACT, MEMORY(MIACT), MEMORY(MPAR), MEMORY(MIFIX), 
2 MEMORY(MHESS), MEMORY(MHAC), N, IX, X, IA, IPRINT, 2*NACT, 
3 MEMORY(MNWORK), IFAULT
IF (IDIST .EQ. 3) CALL NEWTONC, MAXIT, RETOL, ABSTOL, DGEBBN, 
1 XLOGL, NACT, MEMORY(MIACT), MEMORY(MPAR), MEMORY(MIFIX), 
2 MEMORY(MHESS), MEMORY(MHAC), N, IX, X, IA, IPRINT, 2*NACT, 
3 MEMORY(MNWORK), IFAULT

ERROR HANDLING IF NEWTON-RAPHSON PROCEDURE FAILS.
ON ERROR, PRINT OUT MESSAGE IF IPRINT GT 0, SET UP
THE FINAL CALL TO EXPOST)

IF (IFAILT EQ 0) GO TO 30
REGRES = FALSE
IF (IPRINT LE 0) GO TO 50
IF (IFAILT GT 0) GO TO 20
IND = -1
WRITE (IPRINT,60) IND
GO TO 50
20 IFAULT = IFAULT + 1
IF (IFAILT EQ 3) WRITE (IPRINT,70)
IF (IFAILT EQ 4) WRITE (IPRINT,80) MAXIT
GO TO 50

COMPUTE APPROXIMATE ASYMPTOTIC COVARIANCE MATRIX.
REGRES IS A FLAG FOR WHETHER OR NOT TO SET UP OUTPUT IN
REREGRESSION FORM. STATEMENT 30, THE FIRST BRANCH POINT,
SETS UP REGRES FOR FIRST OUTPUT (IF REGRESSION AND
LOCATION-SCALE BOTH REQUESTED, REGRESSION GOES FIRST).
STATEMENT 40, THE SECOND BRANCH POINT, IS FOR REPEAT
CALL (ALWAYS LOCATION-SCALE)
BNAO121
30 REGRES * IFLAG LE 0
BNAO122
40 IF (IDIST EQ 1) CALL VARIAB(C, XLOGL, REGRES, DLOGBN, VLOGBN, BNAO123
1 NACT, MEMORY(MIACT), MEMORY(MIMAP), MEMORY(MPAR), BNAO124
2 MEMORY(MIPVT), MEMORY(MHESS), MEMORY(MHFACE), ICOV, NPAR, COV, BNAO125
3 DET, N, IX, IA, MEMORY(MWORK))
BNAO126
4 IF (IDIST EQ 2) CALL VARIAB(C, XLOGL, REGRES, DGAUBN, VGAUBN, BNAO127
1 NACT, MEMORY(MIACT), MEMORY(MIMAP), MEMORY(MPAR), BNAO128
2 MEMORY(MIPVT), MEMORY(MHESS), MEMORY(MHFACE), ICOV, NPAR, COV, BNAO129
3 DET, N, IX, IA, MEMORY(MWORK))
BNAO130
5 IF (IDIST EQ 3) CALL VARIAB(C, XLOGL, REGRES, DEXVBH, VEXVBH, BNAO131
1 NACT, MEMORY(MIACT), MEMORY(MIMAP), MEMORY(MPAR), BNAO132
2 MEMORY(MIPVT), MEMORY(MHESS), MEMORY(MHFACE), ICOV, NPAR, COV, BNAO133
3 DET, N, IX, IA, MEMORY(MWORK))
BNAO134
C ARRANGE ESTIMATED PARAMETERS IN BETA(*) FOR OUTPUT
BNAO135
C IF LOCATION-SCALE FORM, TRANSFORM PARAMETERS AND POSSIBLY
BNAO136
C RESTORE DEPENDENT VARIABLE TO VALUE ON ENTRY.
BNAO137
50 CALL EXPPOST(REGRES, IDEP, NPAR, BETA, ISTART, NACT, MEMORY(MPAR), BNAO138
1 MEMORY(MIMAP), N, IX, NFIX, MEMORY(MIFIX), MEMORY(MWRFK))
BNAO139
C IF NEWTON-RAPHSON FAILED, GOODBYE.
BNAO140
IF (IFAILT NE 0) RETURN
BNAO141
C PRODUCE STANDARD OUTPUT IF REQUESTED.
BNAO142
IF (IPRINT GT 0) CALL RESULTIC, REGRES, IPRINT, IDIST, N, IDEP, BNAO143
1 NACT, NFIX, XLOGL, NPAR, BETA, ISUB, ISTART, ICOV, COV, DET)
BNAO144
C IF OUTPUT DESIRED IN BOTH REGRRESSION AND LOCATION-SCALE
BNAO145
C FORMS, LOOP BACK, ELSE EXIT
BNAO146
REGRES = NOT REGRES
BNAO147
IF ( NOT REGRES AND IFLAG EQ 0) GO TO 40
BNAO148
C IF ONLY REGRESSION FORM OUTPUT WAS REQUESTED, AN EXTRA
BNAO149
C CALL TO EXPPOST() IS NEEDED TO RESTORE LOCATION-SCALE FORM
BNAO150
C IF (IFLAG LT 0) CALL EXPPOST(REGRES, IDEP, NPAR, BETA, ISTART, BNAO151
1 NACT, MEMORY(MPAR), MEMORY(MIFIX), N, IX, NFIX)
BNAO152
2 MEMORY(MIFIX), MEMORY(MWRFK))
BNAO153
C 60 FORMAT ('OFailure - Exception on Observation No.', 15)
BNAO154
70 FORMAT ('OFailure - Hessian Matrix Is Numerically Singular')
BNAO155
80 FORMAT ('OWarning - No Convergence In', 14, 'Iterations - Possible
BNAO156
1E FAILURE')
BNAO157
RETURN
BNAO158
END
BNAO159
C--------------------------------------------------------------------
PREPO001
C PREPARE FOR ANALYSIS BY SETTING UP THE FOLLOWING ARRAYS -
PREPO002
C IACT(*) - X(*,*) ROWS FOR ACTIVE PARAMETERS
PREPO003
C IFIXED(*) - X(*,*) ROWS FOR FIXED PARAMETERS
PREPO004
C IMAP(*) - POSITIONS IN BETA(*) OF ACTIVE PARAMETERS
PREPO005
C PAR(*) - INITIAL VALUES OF ACTIVE PARAMETERS IN
PREPO006
C REGRESSION FORM
PREPO007
C ALSO SUBTRACT EFFECTS OF FIXED PARAMETERS FROM DEPENDENT
PREPO008
C VARIABLE, USING WORK(*)
PREPO009
C NOTE - DEPENDENT VARIABLE (SCALE) PARAMETER IS ALWAYS
PREPO10
C PLACED FIRST IN PAR(*). FOR CONVENIENCE LATER
PREPO11
C--------------------------------------------------------------------
PREPO12
SUBROUTINE PREPARI(X, N, NPAR, BETA, ISUB, ISTART, NACT, IACT, PREPO13
1 IMAP, PAR, NFIX, IFIXED, IDELETE, WORK)
PREPO14
C ARGUMENTS
PREPO15
INTEGER IX, N, NPAR, ISUB(NPAR), ISTART(NPAR), NACT, IACT(NACT), PREPO016
1 IMAP(NACT), NFIX, IFIXED(I), IDEP
PREPO017
REAL X(IIX,N)
PREPO018
DOUBLE PRECISION BETA(NPAR), PAR(NPAR), WORK(1)
PREPO019
LOCAL SCALARS
PREPO020
INTEGER IND, IND1, LACT, ISCALE
PREPO021
DOUBLE PRECISION TEMP, ZERO, ONE
PREPO022
DATA ZERO, ONE /0.000, 1.000/
PREPO023
C
C SET UP IACT(*), IFIXED(*), IMAP(*)
LACT = 0
PREPO024
IND = 0
PREPO025
ISCALE = 0
PREPO026
DO 20 I = 1, NPAR
PREPO027
IF (ISTART(I) EQ 2) GO TO 10
PREPO028
LACT = LACT + 1
PREPO029
IMAP(LACT) = I
PREPO030
IND1 = ISUB(I)
PREPO031
IACT(LACT) = IND1
PREPO032
IF (IND1 EQ IDEP) ISCALE = 1
PREPO033
GO TO 20
PREPO034
10 IND = IND + 1
PREPO035
IFIXED(IND) = ISUB(I)
PREPO036
WORK(IND) = BETA(I)
PREPO037
20 CONTINUE
C
C SWITCH PLACES SO SCALE IS FIRST ACTIVE PARAMETER
IND = IACT(1)
PREPO038
IACT(1) = IACT(ISCALE)
PREPO039
IACT(ISCALE) = IND
PREPO040
IND = IMAP(1)
PREPO041
IMAP(1) = IMAP(ISCALE)
PREPO042
IMAP(ISCALE) = IND
PREPO043
C
C MAIN LOOP OVER SAMPLE IF PARAMETERS ARE FIXED, TO ADJUST
C DEPENDENT VARIABLE (TO BE RESET ON EXIT FROM BINARY())
IF (NFIX .EQ. 0) GO TO 50
PREPO044
DO 40 I = 1, N
PREPO045
C
C THE FOLLOWING INNER PRODUCT MUST BE ACCUMULATED IN DOUBLE
C PRECISION...
TEMP = DBLE(X(IIDEP,1))
PREPO046
DO 30 J = 1, NFIX
PREPO047
IND = IFIXED(J)
PREPO048
TEMP = TEMP - WORK(J) * X(IND,1)
PREPO049
30 CONTINUE
PREPO050
X(IIDEP,1) = SNGL(TEMP)
PREPO051
C
C 40 CONTINUE
C
C SET STARTING VALUES
50 DO 60 I = 1, NPAR
PREPO052
PAR(I) = ZERO
PREPO053
TEMP = ONE
PREPO054
IND = IMAP(I)
PREPO055
IF (ISTART(IND) EQ 1 AND BETA(IND) NE. ZERO) TEMP = BETA(IND)
PREPO056
PAR(I) = ONE / TEMP
PREPO057
IF (NACT EQ 1) RETURN
PREPO058
DO 70 I = 2, NACT
PREPO059
IND = IMAP(I)
PREPO060
70 RETURN
PREPO061
C
C EXECUTE OUTER PRODUCT
80 DO 90 J = 1, NPAR
PREPO062
PAR(J) = ZERO
PREPO063
IF (ISTART(J) EQ 2) GO TO 10
PREPO064
LACT = LACT + 1
PREPO065
IMAP(LACT) = J
PREPO066
IND1 = ISUB(J)
PREPO067
IACT(LACT) = IND1
PREPO068
IF (IND1 EQ IDEP) ISCALE = 1
PREPO069
GO TO 80
PREPO070
10 IND = IND + 1
PREPO071
IFIXED(IND) = ISUB(J)
PREPO072
WORK(IND) = BETA(J)
PREPO073
20 CONTINUE
C
C EXECUTE INNER PRODUCT
90 DO 100 J = 1, NPAR
PREPO074
PAR(J) = ZERO
PREPO075
IF (ISTART(IND) EQ. 1) PAR(I) = -BETA(IND) / TEMP
70 CONTINUE
C RETURN
END
C***********************************************************************
C NEWTON-RAPHSON ITERATION
C FAILURE CODES
C IFAULT = 1 - INSUFFICIENT WORKSPACE (LESS THAN 2 * NACT LOCATIONS)
C IFAULT = 2 - THE HESSIAN MATRIX IS NUMERICALLY SINGULAR
C IFAULT = 3 - CONVERGENCE HAS NOT OCCURRED IN MAXIT ITNS
C IFAULT = -1 - AN EXCEPTION WAS ABOUT TO OCCUR WHILE
C PROCESSING THE I TH OBSERVATION IN DIFFER()
C***********************************************************************
SUBROUTINE NEWTON(C, MAXIT, RELTOL, ABSTOL, DIFFER, XLOGL, NACT, NEWTON)
1 IACT, PAR, IPIVOT, HESS, HESFAC, N, IX, IA, IPRINT, NEWTON
2 LWORK, WORK, IFAULT)
C ARGUMENTS - DIFFER IS EXTERNAL ROUTINE FOR PARTIALS
INTEGER MAXIT, NACT, IACT(NACT), IPIVOT(NACT), N, IX, IA(N).
1 IPRINT, LWORK, IFAULT
REAL X(N,N)
DOUBLE PRECISION C, RELTOL, ABSTOL, XLOGL, PAR(NACT),
1 HESS(NACT,NACT), HESFAC(NACT,NACT), WORK(LWORK)
C LOCAL SCALARS
INTEGER ITER, IND
DOUBLE PRECISION GNORM, COND, ABERR, RELERR, TEMP, XNEW, ZERO.
1 ONE
C DATA ZERO, ONE /0 000, 1 000/
C IFAULT = 1
C IF (LWORK LT 2*NACT) RETURN
C IFAULT = 0
C IF (COND = 0) RETURN
C IND = NACT + 1
C ITER = 0
C IF (IPRINT GT 0) WRITE (IPRINT,50) C, MAXIT, RELTOL, ABSTOL
C LOOPING POINT FOR ITRATION - THE FIRST NACT LOCATIONS OF
C WORK(*) ARE USED FOR GRADIENT/INCREMENT, WHILE THE NEXT
C NACT LOCATIONS ARE WORKSPACE FOR DECOMP
1 ITER = ITER + 1
C CALL DIFFER(C, XLOGL, NACT, IACT, PAR, WORK(1), HESS, N, IX, X, NEWTON)
1 IA, IFAULT)
C IF (IFault LT 0) RETURN
C FACTOR THE HESSIAN INTO L * U, CHECK IF SINGULAR
C CALL DECOMP(NACT, NACT, HESS, HESFAC, COND, IPIVOT, WORK(IND))
C IF (COND + 1 NE COND) GO TO 20
C IFAULT = 2
C RETURN
C C SOLVE THE SYSTEM HESS * INCREMENT = -GRADIENT.
C OVERWRITING GRADIENT
C GNORM = ZERO
C DO 30 I = 1, NACT
C TEMP = WORK(I)
C WORK(I) = TEMP
C GNORM = GNORM + TEMP*TEMP
C 30 CONTINUE
C GNORM = DSQRT (GNORM),
CALL SOLVE(NACT, NACT, HESFAC, WORK, WORK, IPIVOT)
C
C INCREMENT PARAMETERS, COMPUTE CONVERGENCE CRITERIA
C
RELLR = ZERO
C SBSERR = ZERO
C DO 40 I = 1, NACT
40 TEMP = WORK(I)
ABSERR = OMAX(ABSERR, DBS(TEMP))
XNEW = PAR(I) + TEMP
PAR(I) = XNEW
IF (ONE .NE. ONE) TEMP = TEMP / XNEW
RELLR = OMAX(RELLR, DBS(TEMP))
C
40 CONTINUE
IF (IPRINT GT 0) WRITE (IPRINT,60) ITER, GNORM, RELLR, SBSERR, ICNDO
C
C CHECK CONVERGENCE, SET FAILURE CODE IF NONE
IF (RELLR .LT. RELTOL AND ABSERR .LT. ABSTOL) RETURN
IF (ITER .LT. MAXIT) GO TO 10
IFault = 3
RETURN
C
C
50 FORMAT ('1C MAX ITNS REL TOLER ABS TOLER'/D11.3, 19.
1 2011 2)
60 FORMAT ('D ITN GRAD NORM REL CHANGE ABS CHANGE'/15. 3D12. 2.
1 ESTIMATED CONDITION NUMBER OF HESSIAN IS', D10. 2)
RETURN
END
C
C*****************************************************************************
C COMPUTATIONS FOR VARIABILITY OF THE ESTIMATORS
C ESTIMATE ASYMPTOTIC COVARIANCE MATRIX, TRANSFORM IT FROM
C REGRESSION TO LOCATION-SCALE FORM, COMPUTE ASYMPTOTIC
C CORRELATIONS AND STD. ERRORS
C*****************************************************************************
C SUBROUTINE VARIAB(I, XLOGL, REGRES, DIFFER, VROUT, NACT, IACT,
I
1 IMAP, PAR, IPIVOT, HESS, HESFAC, ICOV, NPAR, COV, DET, VAR)
2 N, IX, X, IA, WORK)
C
C ARGUMENTS - DIFFER AND VROUT ARE SUBROUTINES
C LOGICAL REGRES
C INTEGER NACT, IACT(NACT), IMAP(NACT), IPIVOT(NACT), ICOV, NPAR, N
C REAL IX(N)
C DOUBLE PRECISION C, XLOGL, PAR(NACT), HESS(NACT, NACT),
C HESFAC(NACT, NACT), COV(ICOV, NPAR), DET, WORK(NACT)
C LOCAL SCALARS
C LOCAL FLAG
C INTEGER IND, IND1, IND2
C DOUBLE PRECISION ALPHA, ALPHA2, TEMP, TEMPI, ZERO, ONE, SMALL
C MACHINE-DEPENDENT CONSTANT - SMALL SET SO THAT DEXP(X)
C WILL CAUSE EXCEPTION IF X LT SMALL
C DATA ZERO, ONE, SMALL /O ODD, 1 ODD, -180 ODD/
C
C CALCULATE HESSIAN AT OPTIMAL POINT AND FACTOR II
C THIS MAY BE WASTEFUL IN SOME CASES, BUT IT AVOIDS
C SOME LOGICAL COMPLICATIONS, WORK(*) USED AS SCRATCH
C THE HESSIAN IS ASSUMED NONSINGULAR, AS IT SHOULD BE IF
C THIS POINT IS REACHED FACTORIZATION TO HESFAC(*,*)
C CALL DIFFER(I, XLOGL, NACT, IACT, PAR, WORK, HESS, N, IX, X, IA,
C IND)
C CALL DECOMP(NACT, NACT, HESS, HESFAC, TEMPI, IPIVOT, WORK)
C
C*****************************************************************************
C INVERT NEGATIVE OF HESSIAN, USING IPIVOT(*) AND
C FACTORIZATION IN HESFACT(*,*) PLACE INVERSE IN HESS(*,*)
DO 20 J = 1, NACT
   DO 10 I = 1, NACT
      HESS(I,J) = ZERO
      HESS(I,J) = ONE
   CALL SOLVE(NACT, NACT, HESFACT, HESS(I,J), HESS(I,J), IPIVOT)
20 CONTINUE
C SECTION FOR ROBUST ANALYSIS -
C PLACE PRODUCT (H. INVERSE) * V * (H INVERSE) IN HESS(*,*)
C FIRST COMPUTE V(*), PLACE IN COV(*,*)
C CALL VROUT(C, NACT, IACT, PAR, COV, N, IX, X, IA)
C MULTIPLY HESS * COV, OVERWRITING COV(*,*)
C CALL DMXMLT(COV, NACT, NACT, HESS, NACT, NACT, COV, NACT, NACT,
C            WORK, NACT, 1, IND)
C MULTIPLY COV * HESS, OVERWRITING HESS(*,*)
C CALL DMXMLT(HESS, NACT, NACT, COV, NACT, NACT, HESS, NACT, NACT,
C            WORK, NACT, 1, IND)
C ALL VALUES OF C - IF LOCATION-SCALE FORM, TRANSFORM
30 IF (REGRES) GO TO 80
ALPHA = PAR(I)
ALPHA2 = ALPHA * ALPHA
FLAG = NACT EQ 1
C LEFT MULTIPLY HESS(*,*) BY JACOBIAN, RESULT TO HESFACT(*,*)
DO 50 J = 1, NACT
   TEMP = HESS(I,J) / ALPHA2
   HESFACT(I,J) = TEMP
   IF (FLAG) GO TO 50
   DO 40 I = 2, NACT
      HESFACT(I,J) = PAR(I) * TEMP * HESS(I,J) / ALPHA
30 CONTINUE
C RIGHT MULTIPLY HESFACT(*,*) BY TRANSPOSE OF JACOBIAN.
C RESULT TO HESS(*,*)
DO 70 I = 1, NACT
   TEMP = HESFACT(I,1) / ALPHA2
   HESS(I,1) = TEMP
   IF (FLAG) GO TO 70
   DO 60 J = 2, NACT
      HESS(I,J) = PAR(J) * TEMP * HESFACT(I,J) / ALPHA
70 CONTINUE
C BOTH PARAMETERIZATIONS - DIVIDE COVARIANCE MATRIX BY
C SAMPLE SIZE
80 TEMP = OBLVE(FLOAT(N))
   DO 90 J = 1, NACT
      DO 90 I = 1, J
         HESS(I,J) = HESS(I,J) / TEMP
      HESS(J,J) = HESS(J,J)
90 CONTINUE
C FIND DETERMINANT OF COVARIANCE MATRIX - FACTOR IT.
C RESETTING IPIVOT(*) AND HESFACT(*,*)
CALL DECOMP(NACT, NACT, HESS, HESFAC, TEMP, IPIVOT, WORK)
DET = ZERO
IF (TEMP = ONE EQ TEMP) GO TO 110
IND = IPIVOT(NACT)
DO 100 I = 1, NACT
   TEMP = HESFAC(I, I)
   IF (TEMP LT ZERO) IND = -IND
   DET = DET * DLOG(DABS(TEMP))
100 CONTINUE
C
   TAKE THE NACT ROOT OF DETERMINANT. SET DET TO ZERO
   IF IT UNDERFLOWS OR IS NEGATIVE
   DET = DET / DBLE(FLOAT(NACT))
   IF (DET LT SMALL OR IND LT 0) DET = ZERO
   IF (DET GE SMALL AND IND GT 0) DET = DEXP(DET)
C
   MOVE CONTENTS OF HESS(*) TO UPPER TRIANGLE OF COV(*,*)
110 DO 120 J = 1, NPAR
   DO 120 I = 1, NPAR
   COVL(J, J) = ZERO
   DO 130 J = 1, NACT
      IND = IFAIL(J)
      DO 130 J = 1, NACT
         IND = IFAIL(J)
         COV(J, IND) = COV(IND, J) = HESS(J, J)
130 CONTINUE
C
   PUT SIG ERRORS ON DIAG OF COV(*,*) CORRELATIONS BELOW
   DO 150 I = 1, NPAR
      TEMP = COVL(I, I)
      IF (TEMP EQ ZERO) GO TO 150
      TEMP = DSQRT(TEMP)
      COVL(I, I) = TEMP
      IF (I EQ 1) GO TO 150
      IND = I - 1
      DO 140 J = 1, IND
         TEMP = COVL(J, J)
         IF (TEMP NE ZERO) COVL(J, J) = COVL(J, J) / (TEMP*TEMP)
140 CONTINUE
C
   RETURN
END
C
**********************************************************************************************
C EX POST ADJUSTMENTS BEFORE EXIT
C PUT OPTIMAL PARAMETERS IN BETA(*), IF LOCATION-SCALE
C FORM, TRANSFORM THE PARAMETERS AND RESTORE DEPENDENT
C VARIABLE TO INITIAL VALUES IF PARAMETERS WERE FIXED
C SUBROUTINE EXPOST(REGRES, IDEP, NPAR, BETA, ISTART, NACT, PAR, PARAM, N, IX, NFIX, IFIXED, WORK)
C ARGUMENTS
LOGICAL REGRES
INTEGER IDEP, NPAR, ISTART(NPAR), NACT, NUMACT, N, IX, NFIX, IFIXED
REAL X(IX, N)
DOUBLE PRECISION BETA(NPAR), PAR(NACT), WORK(1)
C LOCAL SCALARS
EX00001
EX00002
EX00003
EX00004
EX00005
EX00006
EX00007
EX00008
EX00009
EX00010
EX00011
EX00012
EX00013
EX00014
EX00015
INTEGER IND
DOUBLE PRECISION TEMP, ONE
DATA ONE /1.0d0/
C
INSERT WORKING PARAMETERS PAR(*) IN BETA(*). IN THE
SUITABLE FORM.
C
TEMP = PAR(I)
IND = IMAP(I)
IF (REGRES) BETA(IND) = TEMP
IF ( NOT REGRES) BETA(IND) = ONE / TEMP
IF (NACT EQ 1) GO TO 20
DO 10 I = 2, NACT
IND = IMAP(I)
IF (REGRES) BETA(IND) = PAR(I)
IF ( NOT REGRES) BETA(IND) = -PAR(I) / TEMP
10 CONTINUE
C
IF LOCATION-SCALE FORM AND PARAMETERS WERE HELD FIXED,
RESTORE DEPENDENT VARIABLE
C
20 IF (REGRES OR NFIX .EQ. 0) RETURN
IND = 0
DO 30 I = 1, NPAR
IF (ISTART(I) NE 2) GO TO 30
IND = IND + 1
WORK(IND) = BETA(I)
30 CONTINUE
DO 50 I = 1, N
C
THE FOLLOWING INNER PRODUCT MUST BE ACCUMULATED IN DOUBLE
PRECISION
C
TEMP = DBLE(X(IDEP,1))
DO 40 J = 1, NFIX
IND = IFIXED(J)
WORK(IND) = TEMP + WORK(J) * X(IND,1)
40 CONTINUE
X(IDEP,1) = SNGL(TEMP)
C
50 CONTINUE
C
RETURN
END
C
**********************************************************************
C
PRINT OUT TYPICAL OUTPUT SUMMARY ON OUTPUT UNIT IPRINT
C
SUBROUTINE RESULTIC, REGRES, IPRINT, IDIST, N, IDEP, NACT, NFIX,
1 XLOGL, NPAR, BETA, ISUB, ISTART, ICNV, COV, DET)
C
ARGUMENTS
LOGICAL REGRES
INTEGER IPRINT, IDIST, N, IDEP, NACT, NFIX, NPAR, ISUB(NPAR),
1 ISTART(NPAR), ICNV
DOUBLE PRECISION C, XLOGL, BETA(NPAR), COV(ICNV,NPAR), DET
C
LOCAL SCALARS
LOGICAL MLE
INTEGER IND
DOUBLE PRECISION ISTAT, ZERO
DATA ZERO /0.0d0/
C
BASE HEADINGS
MLE : C EQ ZERO
IF (MLE) WRITE (IPRINT, 401)
C           FAILURE CODE - IFault = -1 - AN EXCEPTION WAS ABOUT TO OCCUR WHILE PROCESSING THE I TH OBSERVATION
C******************************************************************************
C SUBROUTINE DLOGBN(C, OBJECT, NACT, IACT, PAR, GRAD, HESS, N, IX, IFAULT)
       1      X, IA, IFault)
C******************************************************************************
C ARGUMENTS
C INTEGER NACT, IACT(NACT), N, IX, IA(N), IFault
C REAL X(IX,N)
C DOUBLE PRECISION C, OBJECT, PAR(NACT), GRAD(NACT), HESS(NACT,NACT)
C LOCAL SCALARS
C LOGICAL MLE
C INTEGER IND
C DOUBLE PRECISION EXTRA, F, S, G, FC, EITHER, DDENS, X(j), ZERO,
C ONE, BIG
C MACHINE-DEPENDENT CONSTANT - BIG ROUGHLY CHOSEN SO THAT
C DEXP(X) WILL CAUSE EXCEPTION IF /X/ .GT. BIG
C DATA ZERO, ONE, BIG /0.000, 1.000, 174.000/
C******************************************************************************
C IFault = 0
C MLE = ONE + C * EQ. ONE
C EXTRA = ONE / PAR(1)
C OBJECT = ZERO
C DO 10 J = 1, NACT
C GRAD(J) = ZERO
C DO 10 J = 1, NACT
C HESS(I,J) = ZERO
C 10 CONTINUE
C******************************************************************************
C MAIN LOOP OVER SAMPLE
C******************************************************************************
C DO 90 I = 1, N
C FC = ZERO
C DO 20 J = 1, NACT
C IND = IACT(J)
C FC = FC + PAR(J) * X(IND,1)
C FC = FC / S
C 20 CONTINUE
C IF (DABS(FC) .GT. BIG) GO TO 110
C FC = DEXP(FC)
C S = ONE + FC
C F = FC / S
C S = ONE / S
C G = F * S
C IF (IA(I) .EQ. 0) GO TO 30
C EITHER = S
C X(j) = F
C GO TO 40
C EITHER = -F
C X(j) = S
C 30 IF (MLE) GO TO 50
C FC = G ** C
C GO TO 60
C OBJECT = OBJECT + DLOG(X(j))
C FC = ONE
C 50 OBJECT = OBJECT + DLOG(X(j))
C******************************************************************************
C LOOP TO INCREMENT GRADIENT AND HESSIAN UPPER TRIANGLE
C******************************************************************************
C EITHER = FC * EITHER
G = FC + G
FC = C * EITHER
DO 60 J = 1, NACT
IND = IACT(J)
X(j) = DBLE(X(IND,1))
C******************************************************************************
C******************************************************************************
IF ( NOT MLE) WRITE (IPRINT,500) C
IF (IDIST EQ 1) WRITE (IPRINT,60)
IF (IDIST EQ 2) WRITE (IPRINT,70)
IF (IDIST EQ 3) WRITE (IPRINT,80)
WRITE (IPRINT,90) N, NPAR, NACT, NFIX
IF (REGRES) WRITE (IPRINT,100)
IF ( NOT REGRES) WRITE (IPRINT,110)
C
PARAMETER ESTIMATES
WRITE (IPRINT,120)
DO 20 I = 1, NPAR
   IND = ISUB(I)
   IF (ISTART(I) LT 2) GO TO 10
   WRITE (IPRINT,130) I, IND, BETA(I)
   GO TO 20
TSTAT = BETA(I) / COV(I,1)
1 TSTAT
   IF (IND EQ IREP) WRITE (IPRINT,140) I, IND, BETA(I), COV(I,1), RESUV036
   RESUV038
   IF (IND NE IREP) WRITE (IPRINT,150) I, IND, BETA(I), COV(I,1), RESUV039
   RESUV040
   CONTINUE
   C
ASYMPOTIC QUANTITIES
WRITE (IPRINT,160) XLOG
IF (MLE) WRITE (IPRINT,160) XLOG
IF (DET LE ZERO) WRITE (IPRINT,170)
IF (DET GT ZERO) WRITE (IPRINT,180) NACT, DET
WRITE (IPRINT,190)
DO 30 J = 1, NPAR
30 WRITE (IPRINT,200) (COV(I,J), J=1,1)
C
40 FORMAT (' MAXIMUM LIKELIHOOD BINARY ANALYSIS')
50 FORMAT ('SELF CRITICAL BINARY ANALYSIS. C *', D11.3)
60 FORMAT ('LOGISTIC TOLERANCE DISTRIBUTION')
70 FORMAT ('GAUSSIAN TOLERANCE DISTRIBUTION')
80 FORMAT ('EXTREME VALUE TOLERANCE DISTRIBUTION')
90 FORMAT ('ANALYSIS OF', I6, ' CASES AND', I4, ' VARIABLES', '0', '1', I4, ' PARAMETERS WERE ESTIMATED AND', I4, ' WERE HELD CONSTANT: 21', '0')
100 FORMAT ('OPARAMETERS ARE EXPRESSED IN REGRESSION FORM', '0')
110 FORMAT ('OPARAMETERS ARE EXPRESSED IN LOCATION SCALE FORM', '0')
120 FORMAT ('I VAR', 8X, 'BETA(I)', 6X, 'STD ERR', 8X, 'T STAT', 5X, 'STATUS')
130 FORMAT (2I5, D15.7, 35X, 'FIXED LOCATION')
140 FORMAT (2I5, 3D15.7, 5X, 'ESTIMATED SCALE')
150 FORMAT (2I5, 3D15.7, 5X, 'ESTIMATED LOCATION')
160 FORMAT (2I5, '0', ' THE LOG LIKELIHOOD IS', D15.8)
170 FORMAT (2I5, '0', ' THE ESTIMATED ASYMPTOTIC COVARIANCE', 'MATRIS IS NOT POSTIVE DEFINITE')
180 FORMAT (' OF THE ESTIMATED', 100, ' ROOT OF THE DETERMINANT', 1 OF THE ESTIMATED', 100, ' ASYMPTOTIC COVARIANCE MATRIX IS', 1 Dig 8)
190 FORMAT (' ESTIMATED ASYMPTOTIC STD ERRORS (ON DIAGNOL)', ' CORRELATIONS (BELOW DIAGNOL)', ')
200 FORMAT (10D12.4)
RETURN
END
DDENS = XJI * EITHER
GRAD(J) = GRAD(J) + DDENS
DDENS = XJI * (S - F)
IF (J .EQ. 1) DDENS = DDENS + EXTRA
DDENS = DDENS + FC - G * XJ
DO 70 K = J, NACT
   IND = IACT(K)
   HESS(K,J) = HESS(K,J) + X(IND,1) * DDENS
70 CONTINUE
80 CONTINUE
C C SCALE GRADIENT AND HESSIAN BY SAMPLE SIZE
C EXTRA = DBLE(FLOAT(N))
DO 100 J = 1, NACT
   GRAD(J) = GRAD(J) / EXTRA
DO 100 I = J, NACT
   HESS(I,J) = HESS(I,J) / EXTRA
   HESS(J,I) = HESS(J,I)
100 CONTINUE
IFault = 0
RETURN
C C ERROR EXIT
110 IFault = -1
C C RETURN
END
C C******************************************************************************
C COMPUTE V(**) FACTOR FOR ASYMPTOTIC COVARIANCE MATRIX.
C LOGISTIC BINARY. CALLED ONLY WHEN C .NE. 0
C******************************************************************************
C SUBROUTINE VLOGNBNC(NACT, IACT, PAR, V, N, IX, X, IA)
C ARGUMENTS
C INTEGER NACT, IACT(NACT), N, IX, IA(N)
C REAL X(IX,N)
C DOUBLE PRECISION C, PAR(NACT), V(NACT,NACT)
C LOCAL SCALARS
C INTEGER IND
C DOUBLE PRECISION FC, S, F, EITHER, TEMP, ZERO, ONE, BIG
C MACHINE-DEPENDENT CONSTANT - BIG ROUGHLY CHOSEN SO THAT
C DEXP(X) WILL CAUSE EXCEPTION IF /X/.GT. BIG
C DATA ZERO, ONE, BIG /0 0 0, 1 0 0, 174.000/
C SET UPPER TRIANGLE OF V(**) TO 0
DO 10 J = 1, NACT
   DO 10 I = J, NACT
   V(I,J) = ZERO
10 CONTINUE
C MAIN LOOP OVER SAMPLE
DO 50 I = 1, N
   FC = ZERO
   DO 50 J = 1, NACT
      IND = IACT(J)
      FC = FC + PAR(J) * X(IND,1)
50 CONTINUE
IF (DABS(FC) GE. BIG) GO TO 50
   FC = DEXP(FC)
   S = DNE + FC
   F = FC / S
S = ONE / S
FC = (FC5) ** C
EITHER = S
IF (IA(I) EQ 0) EITHER = F
EITHER = (EITHER+FC) ** 2
C
C INCREMENT TERM OF V(*,*)
DO 40 J = 1, NACT
   IND = IACT(J)
   TEMP = EITHER * X(IND,I)
   DO 30 K = 1, NACT
      IND = IACT(K)
      V(K,J) = V(K,J) + TEMP * X(IND,I)
30  CONTINUE
40  CONTINUE
50 CONTINUE
C
C DIVIDE V(*,*) BY SAMPLE SIZE, FILL OUT
TEMP = DBLE(FLOAT(N))
DO 60 J = 1, NACT
   DO 60 I = 1, NACT
      V(I,J) = V(I,J) / TEMP
60  CONTINUE
60 RETURN
END

*********************************************************************************
C FIRST AND SECOND PARTIAL DERIVATIVES FOR BINARY GAUSSIAN
C REGRESSION PARAMETERIZATION PARTIALS ARE SCALED BY
C SAMPLE SIZE
C FAILURE CODE - IFault = -1: AN EXCEPTION WAS ABOUT TO OCCUR WHILE PROCESSING THE I TH OBSERVATION
C
C SUBROUTINE DGAUBN( C, OBJECT, NACT, IACT, PAR, GRAD, HESS, N, IX, IFAULT)
   I X (IA, IFault)
C ARGUMENTS
INTEGER NACT, IACT(NACT), N, IX, IA(N), IFault
REAL X(IX,N)
DOUBLE PRECISION C, OBJECT, PAR(NACT), GRAD(NACT), HESS(NACT,NACT)
LOCAL SCALARS
LOGICAL MLE
INTEGER IND
DOUBLE PRECISION EXTRA, DOT, DOTMIN, FC, CBY2, RATIO, EITHER,
1 HTERM, XJI, DDEMS, ZERD, ONE, TWO, BIG
C FUNCTIONS CALLED
DOUBLE PRECISION ALNORM, RMILLS
C MACHINE-DEPENDENT CONSTANT - BIG ROUGHLY CHOSEN SO THAT
C DEP(X) WILL CAUSE EXCEPTION IF /X/ GT BIG
C DATA ZERO, ONE, TWO, BIG /0 OOO, 1 OOO, 2 OOO, 174.000/
C
IFault = 0
MLE = ONE + C EQ ONE
EXTRA = ONE / PAR(I)
CBY2 = C / TWO
OBJECT = ZERO
DO 10 J = 1, NACT
   GRAD(J) = ZERO
10  CONTINUE
DO 10 I = 1, NACT
   HESS(I,J) = ZERO
10 CONTINUE
C MAIN LOOP OVER SAMPLE
   DO 90 I = 1, N
      DOT = ZERO
      DO 20 J = 1, NACT
         IND = IACT(J)
         DOT = DOT + PAR(J) * X(IND, I)
      20 CONTINUE
      DOTM = DOT
   30 CONTINUE
      IF (IA(I) .EQ. 0) GO TO 30
      RATIO = RMILLS(DOTM)
      EITHER = RATIO
      GO TO 40
   40 IF (MLE) GO TO 50
      XJ1 = CBY2 * DOT * DOT
   50 IF (ABS(XJ1) .GT. BIG) GO TO 110
      FC = DEXP(-XJ1)
   60 CONTINUE
      EITHER = FC * EITHER
      HTERM = -FC * RATIO * (DOT + RATIO)
      FC = C * EITHER
      DO 80 J = 1, NACT
         IND = IACT(J)
         XJ1 = DBLE(X(IND, I))
      80 CONTINUE
      GRAD(J) = GRAD(J) + XJ1 * EITHER
   90 CONTINUE
      DDENS = XJ1 + DOTM
      IF (J .EQ. 1) DDENS = DDENS + EXTRA
      DDENS = DDENS + FC + XJ1 * HTERM
      DO 100 K = J, NACT
         IND = IACT(K)
         HESS(K, J) = HESS(K, J) + X(IND, I) * DDENS
   100 CONTINUE
C SCALE GRADIENT AND HESSIAN BY SAMPLE SIZE
       EXTRA = DBLE(FLOAT(N))
   110 CONTINUE
       IFault = 0
      RETURN
C ERROR EXIT
C 110 IFault = -1
C      RETURN
END

C*******************************************************************************/
C COMPUTE V(*,*) FACTOR FOR ASYMPOTIC COVARIANCE MATRIX.
C GAUSSIAN BINARY. CALLED ONLY WHEN C_NE_0
C*******************************************************************************/

SUBROUTINE VGAUBN(C, NACT, IACT, PAR, V, N, IX, X, IA)

C ARGUMENTS
INTEGER NACT, IACT(NACT), N, IX, IA(N)
REAL X(IX,N)
DOUBLE PRECISION C, PAR(NACT), V(NACT, NACT)

C LOCAL SCALARS
INTEGER IND
DOUBLE PRECISION F2C, TEMP, EITHER, ZERO, BIG

C FUNCTION CALLED
DOUBLE PRECISION RMILLS

C MACHINE-DEPENDENT CONSTANT - BIG ROUGHLY CHOSEN SO THAT
DEXP(X) WILL CAUSE EXCEPTION IF /X/ GT. BIG
DATA ZERO, BIG /0 ODO, 174.000/

C SET UPPER TRIANGLE OF V(*,*) TO ZERO
DO 10 J = 1, NACT
DO 10 I = J, NACT
10 V(I,J) = ZERO

C MAIN LOOP OVER SAMPLE
DO 70 I = 1, N
F2C = ZERO
DO 20 J = 1, NACT
IND = IACT(J)
F2C = F2C + PAR(J) * X(IND,1)
20 CONTINUE
TEMP = C * F2C + F2C
IF (DABS(TEMP) GE BIG) GO TO 70
IF (IA(I) EQ O) GO TO 30
EITHER = RMILLS(-F2C)
GO TO 40
30 EITHER = RMILLS(F2C)
40 F2C = DEXP(-TEMP) * EITHER * EITHER
DO 60 J = 1, NACT
IND = IACT(J)
TEMP = F2C + X(IND,1)
DO 50 K = J, NACT
IND = IACT(K)
V(K,J) = V(K,J) + TEMP * X(IND,1)
50 CONTINUE
60 CONTINUE
70 CONTINUE

C DIVIDE V(*,*) BY SAMPLE SIZE. FILL OUT
TEMP = DBLE(FLOAT(N))
DO 80 J = 1, NACT
DO 80 I = J, NACT
V(I,J) = V(I,J) / TEMP
80 CONTINUE

C RETURN
END

C*******************************************************************************/
C FIRST AND SECOND PARTIAL DERIVATIVES FOR BINARY EXTREME
C*******************************************************************************/

*DEXV0002
VALUE, REGRESSION PARAMETERIZATION. PARTIALS ARE SCALED
BY SAMPLE SIZE
FAILURE CODE - IFAULT = -1 - AN EXCEPTION WOULD HAVE
OCCURRED WHEN PROCESSING THE I TH OBSERVATION

SUBROUTINE DEXVBN(C, OBJECT, NACT, IACT, PAR, GRAD, HESS, N, IX, IFAULT)
1 X, IA, IFAULT)

INTEGER NACT, IACT(NACT), N, IX, IA(N), IFAULT
REAL X(IX,N)
DOUBLE PRECISION C, OBJECT, PAR(NACT), GRAD(NACT), HESS(NACT,NACT)
LOCAL SCALARS
LOGICAL MLE
INTEGER IND
DOUBLE PRECISION EXTRA, DOT, EXPON, DENFAC, EXECEF, FC, EITHER, HTERM, ZERO, ONE, BIG, BIGI

MACHINE-DEPENDENT CONSTANTS - BIG ROUGHLY CHOSEN SO THAT
DEXP(X) WILL CAUSE EXCEPTION IF /X/ .GT. BIG, BIGI SUCH
THAT 1.000 / DEXP(DEXP(X)) .NE. 1.000 IF X .GT. BIGI
DATA ZERO, ONE, BIG, BIGI /0.000, 1.000, 174.000, -36 200/

IFault = 0
MLE = ONE + C EQ ONE
EXTRA = ONE / PAR(1)
OBJECT = ZERO
DO 10 J = 1, NACT
GRAD(J) = ZERO
DO 10 I = J, NACT
HESS(I,J) = ZERO
10 CONTINUE

MAIN LOOP OVER SAMPLE
DO 90 I = 1, N
DOT = ZERO
DO 20 J = 1, NACT
IND = IACT(J)
DOT = DOT + PAR(IND) * X(IND, I)
20 CONTINUE
IF (DABS(DOT) .GE. BIG) GO TO 110
EXECEF = DEXP(DOT)
DENFAC = ONE - EXECEF
IF (IA(1) EQ. 0) GO TO 40

FAILURE TERM, IA(1) = 1
IF (EXECEF .GE. BIG OR DOT LE. BIGI) GO TO 110
EXECEF = DEXP(EXPON)
DOT = ONE - ONE / EXECEF
IF (MLE) GO TO 30
FC = EXPON / EXECEF
IF (MLE) GO TO 30
FC = FC * C
GO TO 60
110 EXIT

HTERM = EITHER * (DENFAC - EITHER)
IF (MLE) GO TO 30
FC = ONE
GO TO 60

NON FAILURE TERM, IA(1) = 0
40 EITHER = EXECEF
HTERM = EITHER

GO
IF (MLE) GO TO 50
FC = C + (DOT - EXPON)
IF (DABS(FC) GE BIG) GO TO 110
FC = DEXP(FC)
GO TO 60
50 FC = ONE
OBJEKT = OBJECT * EITHER
C
C LOOP TO INCREMENT THE DERIVATIVES
60 EITHER = FC * EITHER
HTERM = FC * HTERM
FC = C * EITHER
DO 80 J = 1, NACT
IND = IACT(J)
DOT = DBLE(X(IND,1))
GRAD(J) = GRAD(J) * DOT * EITHER
EXPON = DOT * DENFAC
IF (J.EQ.1) EXPON = EXPON * EXTRA
EXPON = FC * EXPON + DOT * HTERM
DO 70 K = 1, NACT
IND = IACT(K)
HESS(K,J) = HESS(K,J) + X(IND,1) * EXPON
70 CONTINUE
80 CONTINUE
C
C SCALE PARTIALS BY SAMPLE SIZE. NORMAL EXIT
C EXTRA = DBLE(FLOAT(N))
DO 100 J = 1, NACT
GRAD(J) = GRAD(J) / EXTRA
DO 100 1 = 1, NACT
HESS(I,1) = HESS(I,1) / EXTRA
HESS(J,1) = HESS(J,1)
100 CONTINUE
RETURN
C
C ERROR EXIT - EXCEPTIONS
C IFault = I
C RETURN
END
C
C*****************************************************************************
C COMPUTE V(*.*) FACTOR FOR ASYMPOTIC COVARIANCE MATRIX.
C BINARY EXTREME VALUE CALLED ONLY WHEN C NE 0
C*****************************************************************************

C SUBROUTINE VEXVBNC(N, NACT, IACT, PAR, V, N, IX, X, IA)
C
C ARRGUMENTS
C INTEGER NACT, IACT(NACT), N, IX, IA(N)
C REAL X(IX,N)
C DOUBLE PRECISION C, PAR(NACT), V(NACT,NACT)
C LOCAL SCALARS
C
C INTEGER IND
C DOUBLE PRECISION DOT, EXPON, FC, ZERO, ONE, BIG, BIG1
C C MACHINE-DEPENDENT CONSTANTS BIG ROUGHLY CHOSEN SO THAT
C DEXP(X) WILL CAUSE EXCEPTION IF /X/ GT. BIG, BIG1 SUCH
C THAT 1 0DD / DEXP(DEXP(X)) NE 1 0DD IF X GT BIG1
C DATA ZERO, ONE, BIG, BIG1 /0.0D0, 1.0D0, 174 0DD, -36 200/
C C SET UPPER TRIANGLE OF V(*.*) TO ZERO
C DO 10 J = 1, NACT
C
C*****************************************************************************
DO 10 I = J, NACT
10 V(I,J) = ZERO

C MAIN LOOP OVER SAMPLE
DO 70 I = 1, N
DOT = ZERO
DO 20 J = 1, NACT
IND = IACT(J)
DOT = DOT + PAR(J) * X(IND,I)
20 CONTINUE
IF (DABS(DOT) GE BIG) GO TO 70
EXPON = DEXP(DOT)
IF (IA(I) EQ 0) GO TO 30
IF (DOT LT BIG) GO TO 70
FC = DEXP(EXPON)
DOT = ONE - ONE / FC
FC = EXPON / FC
EXPON = FC / DOT
FC = EXPON * FC ** C
GO TO 40
30 FC = C * (DOT - EXPON)
IF (DABS(FC) GT BIG) GO TO 70
FC = EXPON * DEXP(FC)

C SUMMATOR FOR V(*,*)
40 FC = FC + FC
DO 60 J = 1, NACT
IND = IACT(J)
DOT = FC * X(IND,I)
60 DO 50 K = J, NACT
IND = IACT(K)
V(K,J) = V(K,J) + DOT * X(IND,I)
50 CONTINUE

C DIVIDE V(*,*) BY SAMPLE SIZE. FILL OUT
DO 80 J = 1, NACT
DO 80 I = 1, NACT
V(I,J) = V(I,J) / DOT
80 CONTINUE

C RETURN

C**************************************************************************
C DOUBLE PRECISION MATRIX MULTIPLICATION
C X(N1 BY N3) = Y(N1 BY N2) * Z(N2 BY N3)
C THREE OPTIONS -
C IFLAG = O - X, Y, AND Z ARE DISTINCT
C IFLAG LT. O - X, Y OVERLAP, CORNER OF Y OVERWRITTEN
C IFLAG GT. O - X, Z OVERLAP, CORNER OF Z OVERWRITTEN
C**************************************************************************
C**********************************************************************
C SUBROUTINE DMXMLT(X, IX, N1, Y, IY, N2, Z, IZ, N3, WORK, LWORK,!
IFLAG, IFAULT)
C**********************************************************************
C ARGUMENTS
INTEGER IX, N1, IY, N2, IZ, N3, LWORK, IFLAG, IFAULT
DOUBLE PRECISION X(IX,N3), Y(IY,N2), Z(IZ,N3), WORK(LWORK)
C LOCAL SCALARS
DOUBLE PRECISION TEMP, ZERO
DATA ZERO /0 0 0 0 0 /
C ERROR EXITS
IFAIL = 1
IF (MINO(N1,N2,N3) LT 1 OR MINO(Ix,IV) LT N1 OR IZ LT 1)
N2) RETURN
IFAIL = 2
IF (IFLAG LT 0 AND (N3 GT N2 OR LWORK LT N3))
RETURN
IFAIL = 3
IF (IFLAG GT 0 AND (N1 GT IZ OR LWORK LT N1))
RETURN
IFAIL = 0
IF (IFLAG NE 0) GO TO 30
C STRAIGHTFORWARD, NO OVERWRITING
DO 20 I = 1, N1
DO 20 J = 1, N3
DO 10 K = 1, N2
TEMP = TEMP + Y(I,K) * Z(K,J)
10 X(I,J) = TEMP
20 CONTINUE
C RETURN
C CORNER OF MATRIX Z IS OVERWRITTEN
DO 30 I = 1, N1
DO 30 J = 1, N3
DO 20 K = 1, N2
TEMP = TEMP + Y(I,K) * Z(K,J)
20 WORK(I) = TEMP
30 CONTINUE
C END
C***********************************************************************
C DOUBLE PRECISION FUNCTION ALNORM(X,UPPER)
C***********************************************************************
C ARGUMENTS
C DOUBLE PRECISION X
C LOCAL SCALARS
C DOUBLE PRECISION LTONE, ZTONE, HALF, ONE, CON, Z, Y
C MACHINE-DEPENDENT CONSTANTS - LTONE = (N + 9) / 3, WHERE
C N IS NO. OF DECIMAL DIGITS IN DOUBLE PRECISION NUMBER,
C UTZERO SUCH THAT DEXP(-*X/2.000) WILL CAUSE AN EXCEPTION
C IF X GT. UTZERO
C CONSTANTS IN EXPRESSIONS ARE AS IN AS 66.
C DATA LTONE, UTZERO, ZERO, HALF, ONE, CON /8.000, 10.65DO, 0.000,
C 1. 05DO, 1.000, 1.28D0/.
C C UP = UPPER
Z = X
IF (Z GE ZERO) GO TO 10
UP = NOT UP
Z = -2
10 IF (Z LE LTONE OR UP AND Z .LE. UTZERO) GO TO 20
ALNORM = ZERO
GO TO 40
20 Y = HALF + Z * Z
IF (Z GT CON) GO TO 10 30
ALNORM = HALF + Z * 0.39894228044440D0-0.39990343850404D0*Y/(Y + 5.
175885480458D0-28 8213857808D0/Y + 2.6243312167DO+48.
26959390920D0/Y + 5.92885724343D0)))
GO TO 40
C 30 ALNORM = 0.398942280385D0 + DEXP(-Y) / (Z - 3.8052D-8 + 1.
100000615302D0/Z + 3.8906479D-4 + 1.986153813640D0/Z - 0.
2151679116635D0+5 2930324926D0/Z + 4.836591288D0-15
31508972451D0/Z + 0.74230924027D0+30.768033034D0/(Z + 3.
4990194170D0)))
GO TO 40
C 40 IF (. NOT. UP) ALNORM = ONE - ALNORM
C RETURN
END
C***********************************************************************
C RECIPIROCAL OF MILLS RATIO, Z(X) / G(X), X A STANDARD NORMAL
C VARIATE, BASED ON FUNCTION ALNORM(), ALGORITHM AS 66.
C***********************************************************************
C DOUBLE PRECISION FUNCTION MILLS(X)
C***********************************************************************
C ARGUMENTS
C DOUBLE PRECISION X
C LOCAL SCALARS
C DOUBLE PRECISION Z, Y, LTONE, UTZERO, ZERO, HALF, ONE, CON, FP11, FP12
C MACHINE-DEPENDENT CONSTANTS - LTONE = (N + 9) / 3, WHERE
C N IS NO. OF DECIMAL DIGITS IN DOUBLE PRECISION NUMBER,
C UTZERO SUCH THAT DEXP(-*X/2.000) WILL CAUSE AN EXCEPTION
C IF X LT UTZERO
C CONSTANTS IN EXPRESSIONS ARE AS IN AS 66.
C DATA LTONE, UTZERO, ZERO, HALF, ONE, CON, FP11, FP12 /8.000,
C RML10001
1 18 6500, 0 000, 0 500, 1 000, 1 2800, 7 9788456080286536D-1, RML0019
2 3 9994228040143268001, RML0020
C TRIVIAL CASES
1 IF (X NE ZERO) GO TO 10
RMILLS + FP11
RETURN
10 IF (X GT ZERO) GO TO 20
RMILLS + ZERO
RETURN
C USUAL SITUATION
20 UP + TRUE.
Z + X
IF (Z GE ZERO) GO TO 30
UP + FALSE.
Z - Z
30 Y + HALF + Z + Z
IF (Z GT CON) GO TO 50
C CENTRAL PORTION - X LT ABS(X) LT 128
RMILLS + Z + (0 39894228044000 3990343850400Y/Y + 5.
1/885480580D-29 821355780D0/Y + 2 6243312679D048
2/959930692D0/Y + 5 9288572438D0)))
Y + FP12 + DEXP(-Y)
UP GO TO 40
RMILLS + Y/(HALF + RMILLS)
RETURN
40 RMILLS + Y/(HALF - RMILLS)
RETURN
C OUTER PORTION - ABS(X) GE 128
50 IF (UP OR Z LE LTONE) GO TO 60
C SPECIAL CASE - LOWER TAIL AND Q ESSENTIALLY 1
RMILLS + FP12 + DEXP(-Y)
RETURN
C USUAL SITUATION
60 RMILLS = (Z - 3 8052D 8 + 1 0000615302D0/(Z + 3 98064794D-4 + 1. RMIL0058
198615381364D0/(Z - 0 151679116350005 293303248326D0/(Z + 4
283859128080D0 15 15089724510D0/(Z + 0 742380924027D0 + 30 789930340D/000001)
3(Z + 3 99019417011D0)))
C IF (UP)
V + FP12 + DEXP(Y)
RMILLS + Y/(ONE - Y/RMILLS)
RETURN
END
C***********************************************************************C
C SAMPLE MAIN PROGRAM AND INPUT SUBROUTINE FOR SELF-CRITICAL       C
C BINARY ESTIMATION. IT IS RECOMMENDED THAT BINARY() AND         C
C ITS AUXILIARY PROCEDURES BE COMPILED AND PLACED IN AN            C
C OBJECT CODE LIBRARY, SO THE USER CAN CALL BINARY() FROM         C
C ADDITIONAL PROGRAMS.                                           C
C***********************************************************************C

INTEGER N, IX, IA(750), NPAR, ISUB(10), ISTART(10), IDEP, IDIST. MAIN0007
1       MAXIT, IPRINT, IFLAG, ICOV, LMEM, MEMORY(490), IFault, NC, MAIN0008
2      NMAX, NCMax, IREAD, MAIN0009
REAL X(10, 750) MAIN10
DOUBLE PRECISION C(5), RELTOL, ABSTOL, BETA(10), XLOGL, COV(10, 10) MAIN011
DIMENSION SPECIFICATIONS FOR ARRAYS. MAIN013
C THIS MAIN PROGRAM CAN HANDLE UP TO 750 OBSERVATIONS.        MAIN014
C WITH UP TO 10 PARAMETERS AND 5 DIFFERENT VALUES OF       MAIN015
C FOR ESTIMATION                                            MAIN016
DATA IX, ICov, LMEM, NMAX, NCMax /10, 10, 490, 750, 5/         MAIN017
C BASIC INFORMATION TO PASS TO BINARY() - IPRINT STANDS        MAIN018
C FOR LOGICAL OUTPUT UNIT 6                                  MAIN019
DATA MAXIT, RELTOL, ABSTOL, IPRINT, IFLAG /15, 1.00-7, 1.00-7, 6, MAIN020
1       0/                                                   MAIN021
C IREAD IS LOGICAL INPUT UNIT 5                              MAIN022
DATA IREAD /5/                                            MAIN023
C SPECIFY EVERYTHING BY HAND IN SUBROUTINE INPUT - ARRAY      MAIN024
C MEMORY(*) PASSED AS WORKSPACE                               MAIN025
CALL INPUT(Ix, N, NMAX, X, IA, NPAR, BETA, ISUB, ISTART, NC,   MAIN026
1      NCMax, C, IDEP, IDIST, MEMORY, IREAD, IPRINT, IFault)  MAIN027
   IF (IFault EQ 0) GO TO 10                                   MAIN028
   IF (IFault EQ 1) WRITE (IPrint, 50) N, NMax               MAIN029
   IF (IFault EQ 2) WRITE (IPrint, 60) NPAR, IX              MAIN030
   IF (IFault EQ 3) WRITE (IPrint, 70) NC, NCMax             MAIN031
   WRITE (IPrint, 80)                                        MAIN032
   STOP                                                      MAIN033
C LOOP OVER THE VALUES OF C REQUESTED FOR ESTIMATION          MAIN034
10  DO 40 I = 1, NC                                          MAIN035
   CALL BINARY(N, X, IA, NPAR, ISUB, ISTART, IDEP, IDIST, C(I)) MAIN036
   RELTOL, ABSTOL, MAXIT, IPRINT, IFLAG, BETA, XLOGL, COV,    MAIN037
2      COV, LMEM, MEMORY, IFault)                            MAIN038
   IF (IFault EQ 0) GO TO 20                                  MAIN039
   WRITE (IPrint, 90) IFault                                   MAIN040
   STOP                                                       MAIN041
40  IF (I .GT. 1) GO TO 40                                    MAIN042
C AFTER THE FIRST ESTIMATION, SET ISTART(*) TO 1, SO          MAIN043
C LATEST ESTIMATES CAN BE USED AS STARTING VALUES             MAIN044
30  DO 30 J = 1, NPAR                                        MAIN045
       ISTART(J) = 1                                        MAIN046
30  CONTINUE                                                MAIN047
C 50 FORMAT ('ERROR - SAMPLE SIZE OF', 110, 'EXCEEDS DIMENSION OF', MAIN048
      10)                                                    MAIN049
C 60 FORMAT ('ERROR - ', 13, 'PARAMETERS EXCEEDS DIMENSION OF', 13)  MAIN050
C 70 FORMAT ('ERROR - ', 13, 'C VALUES EXCEEDS DIMENSION OF', 13)   MAIN051
C 80 FORMAT ('RECOMPILE MAIN PROGRAM WITH NEW DIMENSIONS')     MAIN052
C***********************************************************************
**CRUDE INPUT ROUTINE SETS ESTIMATION CONTROL PARAMETERS**

**SUBROUTINE INPUT(I) N, NMAX, X, IA, NPAR, BETA, ISUB, ISTART, NC, NCMAX, C, IDEP, IDIST, WORK, IREAD, IPRINT, IFault)**

**ARGUMENTS**

INTEGER IX, N, NMAX, IA(1), NPAR, ISUB(1), ISTART(1), NC, NCMAX, IDEP, IDIST, IPRINT, IFault

REAL XI(IX), XI(N), XI(1), XI(2), XI(3), XI(4), XI(5), XI(6), XI(7), XI(8), XI(9), XI(10)

DOUBLE PRECISION BETA(1), C(1)

**LOCAL TYPE DECLARATION**

**LOGICAL FLAG**

**SET DETAILS OF PROBLEM SIZE**

N = 480
NPAR = 5
NC = 4

**CHECK FOR SIZE ERRORS**

IFault = 1
IF (N GT NMAX) RETURN
IFault = 2
IF (NPAR GT IX) RETURN
IFault = 3
IF (NC GT NCMAX) RETURN
IFault = 0

**MODELING DETAILS**

DEPENDENT VARIABLE IN FIRST ROW

IDEP = 1

**CONSTANT IN SECOND ROW**

**GAUSSIAN TOLERANCE DISTRIBUTION**

IDIST = 2

**VALUES OF C FOR SELF-CRITICAL**

C(1) = 0 060
C(2) = 0 100
C(3) = 0 200
C(4) = 0 300

**INITIALIZE AVERAGES TO 0**

YBAR = 0 0
YSE = 0 0
DO 10 I = 1, NPAR
10 WORK(I) = 0 0

**LOOP OVER SAMPLE TO READ IN DATA**

FLAG = NPAR LE 2
DO 40 I = 1, N
       READ (IREAD,20) VC, IA(1), ALPHA, S
20       FORMAT (F10.0, 12, F8.0)

**EXPRESS S IN METERS, SCALE VC**

VC = ALOG(VC/1000 0)
S = 0 3048 * EXP(S)
C FILL THE DATA MATRIX - CONSTANT IN SECOND ROW
  X(IDEp,1) = VC
  X(ICONST,1) = 1.0
  X(3,1) = ALPHA
  X(4,1) = S
  X(5,1) = ALPHA * S

C UPDATE MEAN AND STD ERROR OF DEPENDENT VARIABLE IN A
  WAY WHICH DOESN'T LOSE SIGNIFICANT FIGURES
  TEMP = VC - YBAR
  YBAR = YBAR + TEMP / FLOAT(I)
  YSE = YSE + TEMP * (VC - YBAR)

C INCREMENT COVARIATE AVERAGES (3D ROW AND UP)
  IF (FLAG) GO TO 40
  DO 30 J = 3, NPAR
     WORK(J) = WORK(J) + X(J,1)
  30

C 40 CONTINUE

C SET ISUB(*) AND ISTART(*), PROVIDING STARTING VALUES
C FOR INTERCEPT AND SCALE
  DO 50 I = 1, NPAR
     ISUB(I) = 1
     ISTART(I) = 0
  50 CONTINUE
  TEMP = FLOAT(N)
  YSE = SORT(YSE/TEMP)
  ISTART(IDEp) = 1
  BETA(IDEp) = DBLE(YSE)
  ISTART(ICONST) = 1
  BETA(ICONST) = DBLE(YBAR)

C WRITE OUT INFORMATION ON STD. ERROR AND AVERAGES
  WRITE (IPRINT,60) YBAR, YSE
  60 FORMAT ('DEPENDENT (STRESS) VARIABLE'/'OMEAN =', E14.6,
      ' STANDARD DEVIATION =', E14.6)
  IF (FLAG) RETURN
  WRITE (IPRINT,70)
  70 FORMAT ('OCOVARIATES HAVE BEEN CENTERED BY THEIR MEANS'/'
      'OVARIALE', 10X, 'MEAN')
  DO 90 I = 3, NPAR
     WORK(I) = WORK(I) / TEMP
     WRITE (IPRINT,80) ISUB(I), WORK(I)
  80 FORMAT ('0', 13, E14.6)
  90 CONTINUE

C SUBTRACT MEANS FROM COVARIATES
  DO 110 I = 1, N
     DO 100 J = 3, NPAR
        X(J,1) = X(J,1) - WORK(J)
      100
  110 CONTINUE

C RETURN
END
An Algorithm for the Computation of Generalized Likelihood or Self-Critical Estimators for Binary Data

T.A. Delehanty
A.S. Paulson

Rensselaer Polytechnic Institute
Troy, New York 12180

U. S. Army Research Office
Post Office Box 12211
Research Triangle Park, NC 27709

Approved for public release; distribution unlimited.

This paper describes the computational algorithms for computing generalized likelihood estimators for parametric proportional hazards models.
END
5-87
DTIC