G02HLF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G02HLF calculates a robust estimate of the covariance matrix for user-supplied weight functions and their derivatives.

2 Specification

```
      SUBROUTINE G02HLF(UCV, USERP, INDM, N, M, X, LDX, COV, A, WT,
     1                  THETA, BL, BD, MAXIT, NITMON, TOL, NIT, WK, IFAIL)
      INTEGER           INDM, N, M, LDX, MAXIT, NITMON, NIT, IFAIL
      real              USERP(*), X(LDX,M), COV(M*(M+1)/2),
     1                  A(M*(M+1)/2), WT(N), THETA(M), BL, BD, TOL,
     2                  WK(2*M)
      EXTERNAL          UCV
```

3 Description

For a set n observations on m variables in a matrix X, a robust estimate of the covariance matrix, C, and a robust estimate of location, θ, are given by:

$$C = \tau^2 (A^T A)^{-1}$$

where τ^2 is a correction factor and A is a lower triangular matrix found as the solution to the following equations.

$$z_i = A(x_i - \theta)$$

$$\frac{1}{n} \sum_{i=1}^{n} w(\|z_i\|_2) z_i = 0$$

and

$$\frac{1}{n} \sum_{i=1}^{n} u(\|z_i\|_2) z_i z_i^T - v(\|z_i\|_2) I = 0,$$

where x_i is a vector of length m containing the elements of the ith row of X,

z_i is a vector of length m,

I is the identity matrix and 0 is the zero matrix,

and w and u are suitable functions.

G02HLF covers two situations:

(i) $v(t) = 1$ for all t,

(ii) $v(t) = u(t)$.

The robust covariance matrix may be calculated from a weighted sum of squares and cross-products matrix about θ using weights $wt_i = u(\|z_i\|)$. In case (i) a divisor of n is used and in case (ii) a divisor of $\sum_{i=1}^{n} wt_i$ is used. If $w(.) = \sqrt{u(.)}$, then the robust covariance matrix can be calculated by scaling each row of X by $\sqrt{wt_i}$ and calculating an unweighted covariance matrix about θ.

In order to make the estimate asymptotically unbiased under a Normal model a correction factor, τ^2, is needed. The value of the correction factor will depend on the functions employed, (see Huber [1] and Marazzi [2]).

G02HLF finds A using the iterative procedure as given by Huber.

$$A_k = (S_k + I)A_{k-1}$$

and

$$\theta_{j_k} = \frac{b_j}{D_1} + \theta_{j_{k-1}}$$

where $S_k = (s_{jl})$, for $j, l = 1, 2, \ldots, m$ is a lower triangular matrix such that:

$$s_{jl} = \begin{cases} -\min[\max(h_{jl}/D_3, -BL), BL], & j > l \\ -\min[\max((h_{jj}/(2D_3 - D_4/D_2)), -BD), BD], & j = l \end{cases}$$

where

$$D_1 = \sum_{i=1}^{n} \left\{ w(\|z_i\|_2) + \tfrac{1}{m} w'(\|z_i\|_2)\|z_i\|_2 \right\}$$

$$D_2 = \sum_{i=1}^{n} \left\{ \tfrac{1}{p}(u'(\|z_i\|_2)\|z_i\|_2 + 2u(\|z_i\|_2))\|z_i\|_2 - v'(\|z_i\|_2) \right\} \|z_i\|_2$$

$$D_3 = \tfrac{1}{m+2} \sum_{i=1}^{n} \left\{ \tfrac{1}{m}(u'(\|z_i\|_2)\|z_i\|_2 + 2u(\|z_i\|_2)) + u(\|z_i\|_2) \right\} \|z_i\|_2^2$$

$$D_4 = \sum_{i=1}^{n} \left\{ \tfrac{1}{m} u(\|z_i\|_2)\|z_i\|_2^2 - v(\|z_i\|_2^2) \right\}$$

$$h_{jl} = \sum_{i=1}^{n} u(\|z_i\|_2) z_{ij} z_{il}, \text{ for } j > l$$

$$h_{jj} = \sum_{i=1}^{n} u(\|z_i\|_2)(z_{ij}^2 - \|z_{ij}\|_2^2/m)$$

$$b_j = \sum_{i=1}^{n} w(\|z_i\|_2)(x_{ij} - b_j)$$

and BD and BL are suitable bounds.

G02HLF is based on routines in ROBETH, see Marazzi [2].

4 References

[1] Huber P J (1981) *Robust Statistics* Wiley

[2] Marazzi A (1987) Weights for bounded influence regression in ROBETH *Cah. Rech. Doc. IUMSP, No. 3 ROB 3* Institut Universitaire de Médecine Sociale et Préventive, Lausanne

5 Parameters

1: UCV — SUBROUTINE, supplied by the user. *External Procedure*

UCV must return the values of the functions u and w and their derivatives for a given value of its argument.

Its specification is:

```
SUBROUTINE UCV(T, USERP, U, UD, W, WD)
real           T, USERP(*), U, UD, W, WD
```

1: T — ***real*** *Input*

On entry: the argument for which the functions u and w must be evaluated.

2: USERP(∗) — ***real*** array *User Workspace*

The array USERP is included so that the user may pass parameter values to the routine UCV. The values of USERP are not altered by G02HLF.

3: U — ***real*** *Output*

On exit: the value of the u function at the point T.

Constraint: U $\geq$ 0.0.

4: UD — ***real*** *Output*

On exit: the value of the derivative of the u function at the point T.

5: W — ***real*** *Output*

On exit: the value of the w function at the point T.

Constraint: W $\geq$ 0.0.

6: WD — ***real*** *Output*

On exit: the value of the derivative of the w function at the point T.

UCV must be declared as EXTERNAL in the (sub)program from which G02HLF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

2: USERP(*) — ***real*** array *User Workspace*

Note: the dimension of the array USERP must be at least 1.

The array USERP is included so that the user may pass parameter values to the routine UCV. The values of USERP are not altered by G02HLF.

3: INDM — INTEGER *Input*

On entry: indicates which form of the function v will be used.

If INDM = 1, then $v = 1$.
If INDM $\neq$ 1, then $v = u$.

4: N — INTEGER *Input*

On entry: the number of observations, n.

Constraint: N > 1.

5: M — INTEGER *Input*

On entry: number of columns of the matrix X, i.e., number of independent variables, m.

Constraint: 1 $\leq$ M $\leq$ N.

6: X(LDX,M) — ***real*** array *Input*

On entry: X(i,j) must contain the ith observation on the jth variable, for $i = 1,2,\ldots,n$; $j = 1,2,\ldots,m$.

7: LDX — INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which G02HLF is called.

Constraint: LDX $\geq$ N.

8: COV(M*(M+1)/2) — ***real*** array *Output*

On exit: COV contains a robust estimate of the covariance matrix, C. The upper triangular part of the matrix C is stored packed by columns (lower triangular stored by rows), C_{ij} is returned in COV$(j \times (j-1)/2 + i)$, $i \leq j$.

9: A(M*(M+1)/2) — ***real*** array *Input/Output*

On entry: an initial estimate of the lower triangular real matrix A. Only the lower triangular elements must be given and these should be stored row-wise in the array.

The diagonal elements must be $\neq 0$, and in practice will usually be > 0. If the magnitudes of the columns of X are of the same order, the identity matrix will often provide a suitable initial value for A. If the columns of X are of different magnitudes, the diagonal elements of the initial value of A should be approximately inversely proportional to the magnitude of the columns of X.

Constraint: $\mathrm{A}(j \times (j-1)/2 + j) \neq 0.0$, for $j = 1, 2, \ldots, m$.

On exit: the lower triangular elements of the inverse of the matrix A, stored row-wise.

10: WT(N) — ***real*** array *Output*

On exit: WT(i) contains the weights, $wt_i = u(\|z_i\|_2)$, for $i = 1, 2, \ldots, n$.

11: THETA(M) — ***real*** array *Input/Output*

On entry: an initial estimate of the location parameter, θ_j, for $j = 1, 2, \ldots, m$.

In many cases an inital estimate of $\theta_j = 0$, for $j = 1, 2, \ldots, m$ will be adequate. Alternatively medians may be used as given by G07DAF.

On exit: THETA contains the robust estimate of the location parameter, θ_j, for $j = 1, 2, \ldots, m$.

12: BL — ***real*** *Input*

On entry: the magnitude of the bound for the off-diagonal elements of S_k, BL.

Suggested value: 0.9.

Constraint: BL > 0.0.

13: BD — ***real*** *Input*

On entry: the magnitude of the bound for the diagonal elements of S_k, BD.

Suggested value: 0.9.

Constraint: BD > 0.0.

14: MAXIT — INTEGER *Input*

On entry: the maximum number of iterations that will be used during the calculation of A.

Suggested value: 150.

Constraint: MAXIT > 0.

15: NITMON — INTEGER *Input*

On entry: indicates the amount of information on the iteration that is printed.

If NITMON > 0, then the value of A, θ and δ (see Section 7) will be printed at the first and every NITMON iterations.
If NITMON ≤ 0, then no iteration monitoring is printed.

When printing occurs the output is directed to the current advisory message unit (see X04ABF).

16: TOL — ***real*** *Input*

On entry: the relative precision for the final estimates of the covariance matrix. Iteration will stop when maximum δ (see Section 7) is less than TOL.

Constraint: TOL > 0.0.

17: NIT — INTEGER *Output*

On exit: the number of iterations performed.

18: WK(2*M) — ***real*** array *Workspace*

19: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, N $\leq$ 1,
or M < 1,
or N < M,
or LDX < N.

IFAIL = 2

On entry, TOL $\leq$ 0.0,
or MAXIT $\leq$ 0.0,
or Diagonal element of A = 0.0,
or BL $\leq$ 0.0,
or BD $\leq$ 0.0.

IFAIL = 3

A column of X has a constant value.

IFAIL = 4

Value of U or W returned by the user-supplied subroutine UCV < 0.

IFAIL = 5

The routine has failed to converge in MAXIT iterations.

IFAIL = 6

One of the following is zero: D_1, D_2 or D_3.

This may be caused by the functions u or w being too strict for the current estimate of A (or C). The user should try either a larger initial estimate of A or make u and w less strict.

7 Accuracy

On successful exit the accuracy of the results is related to the value of TOL, see Section 5. At an iteration let

(i) $d1 =$ the maximum value of $|s_{jl}|$
(ii) $d2 =$ the maximum absolute change in $wt(i)$
(iii) $d3 =$ the maximum absolute relative change in θ_j

and let $\delta = \max(d1, d2, d3)$. Then the iterative procedure is assumed to have converged when $\delta <$TOL.

8 Further Comments

The existence of A will depend upon the function u, (see Marazzi [2]), also if X is not of full rank a value of A will not be found. If the columns of X are almost linearly related, then convergence will be slow.

9 Example

A sample of 10 observations on three variables is read in along with initial values for A and THETA and parameter values for the u and w functions, c_u and c_w. The covariance matrix computed by G02HLF is printed along with the robust estimate of θ. The subroutine UCV computes the Huber's weight functions:

$$u(t) = 1, \quad \text{if } t \le c_u^2$$

$$u(t) = \frac{c_u}{t^2}, \quad \text{if } t > c_u^2$$

and

$$w(t) = 1, \quad \text{if } t \le c_w$$

$$w(t) = \frac{c_w}{t}, \quad \text{if } t > c_w$$

and their derivatives.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G02HLF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX, LDX
      PARAMETER        (NMAX=10,MMAX=3,LDX=NMAX)
*     .. Local Scalars ..
      real             BD, BL, TOL
      INTEGER          I, IFAIL, INDM, J, K, L1, L2, M, MAXIT, MM, N,
     +                 NIT, NITMON
*     .. Local Arrays ..
      real             A(MMAX*(MMAX+1)/2), COV(MMAX*(MMAX+1)/2),
     +                 THETA(MMAX), USERP(2), WK(MMAX*(MMAX+1)/2),
     +                 WT(NMAX), X(LDX,MMAX)
*     .. External Subroutines ..
      EXTERNAL         G02HLF, UCV, X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G02HLF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      CALL X04ABF(1,NOUT)
*     Read in the dimensions of X
      READ (NIN,*) N, M
      IF (N.GT.0 .AND. N.LE.NMAX .AND. M.GT.0 .AND. M.LE.MMAX) THEN
*        Read in the X matrix
         DO 20 I = 1, N
            READ (NIN,*) (X(I,J),J=1,M)
   20    CONTINUE
*        Read in the initial value of A
         MM = (M+1)*M/2
```

```
         READ (NIN,*) (A(J),J=1,MM)
*        Read in the initial value of THETA
         READ (NIN,*) (THETA(J),J=1,M)
*        Read in the values of the parameters of the ucv functions
         READ (NIN,*) USERP(1), USERP(2)
*        Set the values of remaining parameters
         INDM = 1
         BL = 0.9e0
         BD = 0.9e0
         MAXIT = 50
         TOL = 0.5e-4
*        * Change NITMON to a positive value if monitoring information
*          is required *
         NITMON = 0
         IFAIL = 0
*
         CALL G02HLF(UCV,USERP,INDM,N,M,X,LDX,COV,A,WT,THETA,BL,BD,
     +               MAXIT,NITMON,TOL,NIT,WK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'G02HLF required ', NIT,
     +     ' iterations to converge'
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Robust covariance matrix'
         L2 = 0
         DO 40 J = 1, M
            L1 = L2 + 1
            L2 = L2 + J
            WRITE (NOUT,99998) (COV(K),K=L1,L2)
   40    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Robust estimates of THETA'
         DO 60 J = 1, M
            WRITE (NOUT,99997) THETA(J)
   60    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,A,I4,A)
99998 FORMAT (1X,6F10.3)
99997 FORMAT (1X,F10.3)
      END
*
      SUBROUTINE UCV(T,USERP,U,UD,W,WD)
*     .. Scalar Arguments ..
      real              T, U, UD, W, WD
*     .. Array Arguments ..
      real              USERP(2)
*     .. Local Scalars ..
      real              CU, CW, T2
*     .. Executable Statements ..
*     u function and derivative
      CU = USERP(1)
      U = 1.0e0
      UD = 0.0e0
      IF (T.NE.0) THEN
         T2 = T*T
         IF (T2.GT.CU) THEN
```

```
            U = CU/T2
            UD = -2.0e0*U/T
         END IF
      END IF
*     w function and derivative
      CW = USERP(2)
      IF (T.GT.CW) THEN
         W = CW/T
         WD = -W/T
      ELSE
         W = 1.0e0
         WD = 0.0e0
      END IF
      END
```

9.2 Program Data

```
G02HLF Example Program Data
    10    3                     : N  M
  3.4  6.9  12.2                : X1  X2  X3
  6.4  2.5  15.1
  4.9  5.5  14.2
  7.3  1.9  18.2
  8.8  3.6  11.7
  8.4  1.3  17.9
  5.3  3.1  15.0
  2.7  8.1   7.7
  6.1  3.0  21.9
  5.3  2.2  13.9                : End of X1 X2 and X3 values
  1.0 0.0 1.0 0.0 0.0 1.0       : A
  0.0 0.0 0.0                   : THETA
  4.0 2.0                       : CU CW
```

9.3 Program Results

```
 G02HLF Example Program Results

 G02HLF required   25 iterations to converge

 Robust covariance matrix
      3.278
     -3.692     5.284
      4.739    -6.409    11.837

 Robust estimates of THETA
      5.700
      3.864
     14.704
```