NAG Toolbox for Matlab

nag_correg_robustm_corr_user (g02hm)

1 Purpose

nag_correg_robustm_corr_user (g02hm) computes a robust estimate of the covariance matrix for user-supplied weight functions. The derivatives of the weight functions are not required.

2 Syntax

```
[user, cov, a, wt, theta, nit, ifail] = nag_correg_robustm_corr_user(ucv, indm, x, a, theta, 'user', user, 'n', n, 'm', m, 'bl', bl, 'bd', bd, 'maxit', maxit, 'nitmon', nitmon, 'tol', tol)
[user, cov, a, wt, theta, nit, ifail] = g02hm(ucv, indm, x, a, theta, 'user', user, 'n', n, 'm', m, 'bl', bl, 'bd', bd, 'maxit', maxit, 'nitmon', nitmon, 'tol', tol)
```

Note: The interface to this routine has changed since earlier releases of the toolbox:

Mark 22: **n** has been made optional

Mark 23: **nitmon**, **tol** now optional.

3 Description

For a set of 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

$$C = \tau^2 \left(A^{\mathrm{T}} A\right)^{-1},$$

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

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

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

and

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

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.

nag_correg_robustm_corr_user (g02hm) 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 (1981) and Marazzi (1987)).

nag_correg_robustm_corr_user (g02hm) finds A using the iterative procedure as given by Huber; see Huber (1981).

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

and

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

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

$$s_{jl} = \begin{cases} -\min\big[\max\big(h_{jl}/D_2, -BL\big), BL\big], & j > l \\ -\min\big[\max\big(\frac{1}{2}\big(h_{jj}/D_2 - 1\big), -BD\big), BD\big], & j = l \end{cases},$$

where

$$D_1 = \sum_{i=1}^{n} w(\|z_i\|)$$

$$D_2 = \sum_{i=1}^{n} u(\|z_i\|)$$

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

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

and BD and BL are suitable bounds.

The value of τ may be chosen so that C is unbiased if the observations are from a given distribution.

nag_correg_robustm_corr_user (g02hm) is based on routines in ROBETH; see Marazzi (1987).

4 References

Huber P J (1981) *Robust Statistics* Wiley

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

5.1 Compulsory Input Parameters

1: **ucv** – function handle or string containing name of m-file

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

```
[user, u, w] = ucv(t, user)
```

Input Parameters

1: **t** – double scalar

The argument for which the functions u and w must be evaluated.

2: **user** – Any MATLAB object

ucv is called from nag_correg_robustm_corr_user (g02hm) with the object supplied to nag_correg_robustm_corr_user (g02hm).

Output Parameters

1: **user** – Any MATLAB object

2: **u** – double scalar

The value of the u function at the point **t**.

3: **w** – double scalar

The value of the w function at the point **t**.

2: **indm** – int64 scalar

Indicates which form of the function v will be used.

indm $= 1$
$v = 1$.

indm $\neq 1$
$v = u$.

3: **x**(ldx,**m**) – double array

ldx, the first dimension of the array, must satisfy the constraint $ldx \geq$ **n**.

x(i,j) must contain the ith observation on the jth variable, for $i = 1,2,\ldots,n$ and $j = 1,2,\ldots,m$.

4: **a**(**m** $\times$ (**m** + **1**)/**2**) – double array

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: **a**$(j \times (j-1)/2 + j) \neq 0.0$, for $j = 1,2,\ldots,m$.

5: **theta**(**m**) – double array

m, the dimension of the array, must satisfy the constraint $1 \leq$ **m** $\leq$ **n**.

An initial estimate of the location parameter, θ_j, for $j = 1,2,\ldots,m$.

In many cases an initial estimate of $\theta_j = 0$, for $j = 1,2,\ldots,m$, will be adequate. Alternatively medians may be used as given by nag_univar_robust_1var_median (g07da).

5.2 Optional Input Parameters

1: **user** – Any MATLAB object

user is not used by nag_correg_robustm_corr_user (g02hm), but is passed to **ucv**. Note that for large objects it may be more efficient to use a global variable which is accessible from the m-files than to use **user**.

2: **n** – int64 scalar

Default: The first dimension of the array **x**.

n, the number of observations.

Constraint: $\mathbf{n} > 1$.

3: **m** – int64 scalar

Default: The dimension of the array **theta** and the second dimension of the array **x**. (An error is raised if these dimensions are not equal.)

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

Constraint: $1 \leq \mathbf{m} \leq \mathbf{n}$.

4: **bl** – double scalar

The magnitude of the bound for the off-diagonal elements of S_k, BL.

Suggested value: $\mathbf{bl} = 0.9$.

Default: 0.9

Constraint: $\mathbf{bl} > 0.0$.

5: **bd** – double scalar

The magnitude of the bound for the diagonal elements of S_k, BD.

Suggested value: $\mathbf{bd} = 0.9$.

Default: 0.9

Constraint: $\mathbf{bd} > 0.0$.

6: **maxit** – int64 scalar

The maximum number of iterations that will be used during the calculation of A.

Suggested value: $\mathbf{maxit} = 150$.

Default: 150

Constraint: $\mathbf{maxit} > 0$.

7: **nitmon** – int64 scalar

Indicates the amount of information on the iteration that is printed.

$\mathbf{nitmon} > 0$
The value of A, θ and δ (see Section [Accuracy]) will be printed at the first and every **nitmon** iterations.

$\mathbf{nitmon} \leq 0$
No iteration monitoring is printed.

When printing occurs the output is directed to the current advisory message channel (See nag_file_set_unit_advisory (x04ab).)

Default: 0

8: **tol** – double scalar

The relative precision for the final estimate of the covariance matrix. Iteration will stop when maximum δ (see Section [Accuracy]) is less than **tol**.

Default: 5e − 5

Constraint: **tol** > 0.0.

5.3 Input Parameters Omitted from the MATLAB Interface

ruser, ldx, wk

5.4 Output Parameters

1: **user** – Any MATLAB object

2: **cov**$(\mathbf{m} \times (\mathbf{m}+\mathbf{1})/\mathbf{2})$ – double array

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), that is C_{ij} is returned in **cov**$(j \times (j-1)/2 + i)$, $i \leq j$.

3: **a**$(\mathbf{m} \times (\mathbf{m}+\mathbf{1})/\mathbf{2})$ – double array

The lower triangular elements of the inverse of the matrix A, stored row-wise.

4: **wt**$(\mathbf{n})$ – double array

wt(i) contains the weights, $wt_i = u(\|z_i\|)$, for $i = 1, 2, \ldots, n$.

5: **theta**$(\mathbf{m})$ – double array

Contains the robust estimate of the location parameter, θ_j, for $j = 1, 2, \ldots, m$.

6: **nit** – int64 scalar

The number of iterations performed.

7: **ifail** – int64 scalar

ifail $= 0$ unless the function detects an error (see [Error Indicators and Warnings]).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **n** ≤ 1,
or **m** < 1,
or **n** $<$ **m**,
or $ldx <$ **n**.

ifail $= 2$

On entry, **tol** ≤ 0.0,
or **maxit** ≤ 0,
or diagonal element of **a** $= 0.0$,
or **bl** ≤ 0.0,
or **bd** ≤ 0.0.

ifail $= 3$

A column of **x** has a constant value.

ifail $= 4$

Value of **u** or **w** returned by **ucv** < 0.

ifail $= 5$

The function has failed to converge in **maxit** iterations.

ifail $= 6$ (*warning*)

Either the sum D_1 or the sum D_2 is zero. This may be caused by the functions u or w being too strict for the current estimate of A (or C). You should either try a larger initial estimate of A or make the u and w functions less strict.

7 Accuracy

On successful exit the accuracy of the results is related to the value of **tol**; see Section [Parameters]. At an iteration let

(i) $d1 =$ the maximum value of $\left|s_{jl}\right|$

(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 (1987)); 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.

If derivatives of the u and w functions are available then the method used in nag_correg_robustm_corr_user_deriv (g02hl) will usually give much faster convergence.

9 Example

```
function nag_correg_robustm_corr_user_example
indm = int64int32nag_int(1);
x = [3.4, 6.9, 12.2;
     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;
     2.7, 8.1, 7.7;
     6.1, 3, 21.9;
     5.3, 2.2, 13.9];
a = [1;
     0;
     1;
     0;
     0;
     1];
theta = [0;
     0;
     0];
user = [4, 2];
```

```
[user, covar, aOut, wt, thetaOut, nit, ifail] = ...
    nag_correg_robustm_corr_user(@ucv, indm, x, a, theta, 'user', user)

function [userp, u, w] = ucv(t, userp)
  cu = userp(1);
  u = 1.0;
  if (t ~= 0)
    t2 = t*t;
    if (t2 > cu)
      u = cu/t2;
    end
  end
  % w function and derivative
  cw = userp(2);
  if (t > cw)
    w = cw/t;
  else
    w = 1.0;
  end
```

```
user =

     4     2


covar =

    3.2779
   -3.6918
    5.2841
    4.7391
   -6.4087
   11.8373


aOut =

    0.5523
    1.0614
    0.9424
   -0.1880
    0.4776
    0.5021


wt =

    1.0000
    1.0000
    1.0000
    1.0000
    0.2339
    1.0000
    1.0000
    0.9385
    0.4012
    0.7579


thetaOut =

    5.6998
    3.8636
   14.7036


nit =
```

```
                    34


ifail =

                     0
```