NAG Library Routine Document

G02BUF

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

G02BUF calculates the sample means and sums of squares and cross-products, or sums of squares and cross-products of deviations from the mean, in a single pass for a set of data. The data may be weighted.

2 Specification

```
SUBROUTINE G02BUF (MEAN, WEIGHT, N, M, X, LDX, WT, SW, WMEAN, C, IFAIL)
INTEGER            N, M, LDX, IFAIL
REAL (KIND=nag_wp) X(LDX,M), WT(*), SW, WMEAN(M), C((M*M+M)/2)
CHARACTER(1)       MEAN, WEIGHT
```

3 Description

G02BUF is an adaptation of West's WV2 algorithm; see West (1979). This routine calculates the (optionally weighted) sample means and (optionally weighted) sums of squares and cross-products or sums of squares and cross-products of deviations from the (weighted) mean for a sample of n observations on m variables X_j, for $j = 1, 2, \ldots, m$. The algorithm makes a single pass through the data.

For the first $i-1$ observations let the mean of the jth variable be $\bar{x}_j(i-1)$, the cross-product about the mean for the jth and kth variables be $c_{jk}(i-1)$ and the sum of weights be W_{i-1}. These are updated by the ith observation, x_{ij}, for $j = 1, 2, \ldots, m$, with weight w_i as follows:

$$\begin{aligned} W_i &= W_{i-1} + w_i \\ \bar{x}_j(i) &= \bar{x}_j(i-1) + \tfrac{w_i}{W_i}\big(x_j - \bar{x}_j(i-1)\big), \quad j = 1, 2, \ldots, m \end{aligned}$$

and

$$c_{jk}(i) = c_{jk}(i-1) + \frac{w_i}{W_i}\big(x_j - \bar{x}_j(i-1)\big)\big(x_k - \bar{x}_k(i-1)\big)W_{i-1}, \quad j = 1, 2, \ldots, m \text{ and } k = j, j+1, \ldots, m.$$

The algorithm is initialized by taking $\bar{x}_j(1) = x_{1j}$, the first observation, and $c_{ij}(1) = 0.0$.

For the unweighted case $w_i = 1$ and $W_i = i$ for all i.

Note that only the upper triangle of the matrix is calculated and returned packed by column.

4 References

Chan T F, Golub G H and Leveque R J (1982) *Updating Formulae and a Pairwise Algorithm for Computing Sample Variances* Compstat, Physica-Verlag

West D H D (1979) Updating mean and variance estimates: An improved method *Comm. ACM* **22** 532–555

5 Parameters

1: MEAN – CHARACTER(1) *Input*

On entry: indicates whether G02BUF is to calculate sums of squares and cross-products, or sums of squares and cross-products of deviations about the mean.

MEAN = 'M'

The sums of squares and cross-products of deviations about the mean are calculated.

MEAN = 'Z'
The sums of squares and cross-products are calculated.

Constraint: MEAN = 'M' or 'Z'.

2: WEIGHT – CHARACTER(1) *Input*

On entry: indicates whether the data is weighted or not.

WEIGHT = 'U'
The calculations are performed on unweighted data.

WEIGHT = 'W'
The calculations are performed on weighted data.

Constraint: WEIGHT = 'W' or 'U'.

3: N – INTEGER *Input*

On entry: n, the number of observations in the dataset.

Constraint: N ≥ 1.

4: M – INTEGER *Input*

On entry: m, the number of variables.

Constraint: M ≥ 1.

5: X(LDX, M) – REAL (KIND=nag_wp) array *Input*

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

6: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which G02BUF is called.

Constraint: LDX $\geq$ N.

7: WT$(*)$ – REAL (KIND=nag_wp) array *Input*

Note: the dimension of the array WT must be at least N if WEIGHT = 'W', and at least 1 otherwise.

On entry: the optional weights of each observation.

If WEIGHT = 'U', WT is not referenced.

If WEIGHT = 'W', WT(i) must contain the weight for the ith observation.

Constraint: if WEIGHT = 'W', WT$(i) \geq 0.0$, for $i = 1, 2, \ldots, n$.

8: SW – REAL (KIND=nag_wp) *Output*

On exit: the sum of weights.

If WEIGHT = 'U', SW contains the number of observations, n.

9: WMEAN(M) – REAL (KIND=nag_wp) array *Output*

On exit: the sample means. WMEAN(j) contains the mean for the jth variable.

10: C$((\text{M} \times \text{M} + \text{M})/2)$ – REAL (KIND=nag_wp) array *Output*

On exit: the cross-products.

If MEAN = 'M', C contains the upper triangular part of the matrix of (weighted) sums of squares and cross-products of deviations about the mean.

If MEAN = 'Z', C contains the upper triangular part of the matrix of (weighted) sums of squares and cross-products.

These are stored packed by columns, i.e., the cross-product between the jth and kth variable, $k \geq j$, is stored in C$(k \times (k-1)/2 + j)$.

11: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL $= 0$ unless the routine detects an error or a warning has been flagged (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 or warnings detected by the routine:

IFAIL $= 1$

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

IFAIL $= 2$

On entry, MEAN $\neq$ 'M' or 'Z'.

IFAIL $= 3$

On entry, WEIGHT $\neq$ 'W' or 'U'.

IFAIL $= 4$

On entry, WEIGHT = 'W', and a value of WT < 0.0.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

7 Accuracy

For a detailed discussion of the accuracy of this algorithm see Chan *et al.* (1982) or West (1979).

8 Parallelism and Performance

Not applicable.

9 Further Comments

G02BWF may be used to calculate the correlation coefficients from the cross-products of deviations about the mean. The cross-products of deviations about the mean may be scaled using F06EDF (DSCAL) or F06FDF to give a variance-covariance matrix.

The means and cross-products produced by G02BUF may be updated by adding or removing observations using G02BTF.

Two sets of means and cross-products, as produced by G02BUF, can be combined using G02BZF.

10 Example

A program to calculate the means, the required sums of squares and cross-products matrix, and the variance matrix for a set of 3 observations of 3 variables.

10.1 Program Text

```
    Program g02bufe

!     G02BUF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: dscal, g02buf, nag_wp, x04ccf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter    :: one = 1.0_nag_wp
      Integer, Parameter               :: inc1 = 1, nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: alpha, sw
      Integer                          :: i, ifail, lc, ldx, lwt, m, n
      Character (1)                    :: mean, weight
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: c(:), wmean(:), wt(:), x(:,:)
!     .. Executable Statements ..
      Write (nout,*) 'G02BUF Example Program Results'
      Write (nout,*)
      Flush (nout)

!     Skip heading in data file
      Read (nin,*)

!     Read in problem size
      Read (nin,*) mean, weight, m, n

      If (weight=='W' .Or. weight=='w') Then
        lwt = n
      Else
        lwt = 0
      End If
      ldx = n
      lc = (m*m+m)/2
      Allocate (wt(lwt),x(ldx,m),wmean(m),c(lc))

!     Read in data
```

```
      If (lwt>0) Then
        Read (nin,*) wt(1:n)
      End If
      Read (nin,*)(x(i,1:m),i=1,n)

!     Calculate sums of squares and cross-products matrix
      ifail = 0
      Call g02buf(mean,weight,n,m,x,ldx,wt,sw,wmean,c,ifail)

!     Display results
      Write (nout,*) 'Means'
      Write (nout,99999) wmean(1:m)
      Write (nout,*)
      Write (nout,*) 'Weights'
      Write (nout,99999) wt(1:n)
      Write (nout,*)
      Flush (nout)
      ifail = 0
      Call x04ccf('Upper','Non-unit',m,c,'Sums of squares and cross-products', &
        ifail)

!     Convert the sums of squares and cross-products to a variance matrix
      If (sw>one) Then
        alpha = one/(sw-one)
!       The NAG name equivalent of dscal is f06edf
        Call dscal(lc,alpha,c,inc1)
        Write (nout,*)
        Flush (nout)
        ifail = 0
        Call x04ccf('Upper','Non-unit',m,c,'Variance matrix',ifail)
      End If

99999 Format (1X,6F14.4)
    End Program g02bufe
```

10.2 Program Data

```
G02BUF Example Program Data
 'M'      'W'     3     3
  0.1300   1.3070   0.3700
  9.1231   3.7011   4.5230
  0.9310   0.0900   0.8870
  0.0009   0.0099   0.0999
```

10.3 Program Results

```
 G02BUF Example Program Results

 Means
         1.3299        0.3334        0.9874

 Weights
         0.1300        1.3070        0.3700

 Sums of squares and cross-products
             1          2          3
 1      8.7569     3.6978     4.0707
 2                 1.5905     1.6861
 3                            1.9297

 Variance matrix
             1          2          3
 1     10.8512     4.5822     5.0443
 2                 1.9709     2.0893
 3                            2.3912
```