NAG Library Routine Document

G01ALF

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

G01ALF calculates a five-point summary for a single sample.

2 Specification

```
SUBROUTINE G01ALF (N, X, IWRK, RES, IFAIL)
INTEGER            N, IWRK(N), IFAIL
REAL (KIND=nag_wp) X(N), RES(5)
```

3 Description

G01ALF calculates the minimum, lower hinge, median, upper hinge and the maximum of a sample of n observations.

The data consist of a single sample of n observations denoted by x_i and let z_i, for $i = 1, 2, \ldots, n$, represent the sample observations sorted into ascending order.

Let $m = \dfrac{n}{2}$ if n is even and $\dfrac{(n+1)}{2}$ if n is odd,

and $k = \dfrac{m}{2}$ if m is even and $\dfrac{(m+1)}{2}$ if m is odd.

Then we have

$$
\begin{array}{lll}
\text{Minimum} & = z_1, & \\
\text{Maximum} & = z_n, & \\
\text{Median} & = z_m & \text{if } n \text{ is odd,} \\
 & = \dfrac{z_m + z_{m+1}}{2} & \text{if } n \text{ is even,} \\
\text{Lower hinge} & = z_k & \text{if } m \text{ is odd,} \\
 & = \dfrac{z_k + z_{k+1}}{2} & \text{if } m \text{ is even,} \\
\text{Upper hinge} & = z_{n-k+1} & \text{if } m \text{ is odd,} \\
 & = \dfrac{z_{n-k} + z_{n-k+1}}{2} & \text{if } m \text{ is even.}
\end{array}
$$

4 References

Erickson B H and Nosanchuk T A (1985) *Understanding Data* Open University Press, Milton Keynes

Tukey J W (1977) *Exploratory Data Analysis* Addison–Wesley

5 Arguments

1: N – INTEGER *Input*

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

Constraint: N $\geq$ 5.

2: X(N) – REAL (KIND=nag_wp) array *Input*

On entry: the sample observations, $x_1, x_2, \ldots, x_n$.

3: IWRK(N) – INTEGER array *Workspace*

4: RES(5) – REAL (KIND=nag_wp) array *Output*

On exit: RES contains the five-point summary.

RES(1)
The minimum.

RES(2)
The lower hinge.

RES(3)
The median.

RES(4)
The upper hinge.

RES(5)
The maximum.

5: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation 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 argument, 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, N < 5.

IFAIL $= -99$

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

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

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

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The computations are stable.

8 Parallelism and Performance

G01ALF is not threaded in any implementation.

9 Further Comments

The time taken by G01ALF is proportional to n.

10 Example

This example calculates a five-point summary for a sample of 12 observations.

10.1 Program Text

```
    Program g01alfe

!     G01ALF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: g01alf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: ifail, n
!     .. Local Arrays ..
      Real (Kind=nag_wp)               :: res(5)
      Real (Kind=nag_wp), Allocatable  :: x(:)
      Integer, Allocatable             :: iwrk(:)
!     .. Executable Statements ..
      Write (nout,*) 'G01ALF Example Program Results'
      Write (nout,*)

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

!     Read in the problem size
      Read (nin,*) n

      Allocate (x(n),iwrk(n))

!     Read in data
      Read (nin,*) x(1:n)

!     Calculate summary statistics
      ifail = 0
      Call g01alf(n,x,iwrk,res,ifail)

!     Display results
      Write (nout,99999) 'Maximum                  ', res(5)
      Write (nout,99999) 'Upper Hinge (75% quantile)', res(4)
      Write (nout,99999) 'Median      (50% quantile)', res(3)
      Write (nout,99999) 'Lower Hinge (25% quantile)', res(2)
      Write (nout,99999) 'Minimum                  ', res(1)

99999 Format (1X,A,F16.4)
    End Program g01alfe
```

10.2 Program Data

```
G01ALF Example Program Data
 12
 12.0  9.0  2.0  5.0  6.0  8.0  2.0  7.0  3.0  1.0  11.0  10.0
```

10.3 Program Results

```
 G01ALF Example Program Results

 Maximum                           12.0000
 Upper Hinge (75% quantile)         9.5000
 Median      (50% quantile)         6.5000
 Lower Hinge (25% quantile)         2.5000
 Minimum                            1.0000
```