NAG Library Routine Document

G01AMF

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

G01AMF finds specified quantiles from a vector of unsorted data.

2 Specification

```
SUBROUTINE G01AMF (N, RV, NQ, Q, QV, IFAIL)

INTEGER            N, NQ, IFAIL
REAL (KIND=nag_wp) RV(N), Q(NQ), QV(NQ)
```

3 Description

A quantile is a value which divides a frequency distribution such that there is a given proportion of data values below the quantile. For example, the median of a dataset is the 0.5 quantile because half the values are less than or equal to it; and the 0.25 quantile is the 25th percentile.

G01AMF uses a modified version of Singleton's 'median-of-three' Quicksort algorithm (Singleton (1969)) to determine specified quantiles of a vector of real values. The input vector is partially sorted, as far as is required to compute desired quantiles; for a single quantile, this is much faster than sorting the entire vector. Where necessary, linear interpolation is also carried out to return the values of quantiles which lie between two data points.

4 References

Singleton R C (1969) An efficient algorithm for sorting with minimal storage: Algorithm 347 *Comm. ACM* **12** 185–187

5 Parameters

1: N – INTEGER *Input*

On entry: the number of elements in the input vector RV.

Constraint: $N > 0$.

2: RV(N) – REAL (KIND=nag_wp) array *Input/Output*

On entry: the vector whose quantiles are to be determined.

On exit: the order of the elements in RV is not, in general, preserved.

3: NQ – INTEGER *Input*

On entry: the number of quantiles requested.

Constraint: $NQ > 0$.

4: Q(NQ) – REAL (KIND=nag_wp) array *Input*

On entry: the quantiles to be calculated, in ascending order. Note that these must be between 0.0 and 1.0, with 0.0 returning the smallest element and 1.0 the largest.

Constraints:

$0.0 \leq \mathrm{Q}(i) \leq 1.0$, for $i = 1, 2, \ldots, \mathrm{NQ}$;
$\mathrm{Q}(i) \leq \mathrm{Q}(i+1)$, for $i = 1, 2, \ldots, \mathrm{NQ} - 1$.

5: QV(NQ) – REAL (KIND=nag_wp) array *Output*

On exit: $\mathrm{QV}(i)$ contains the quantile specified by the value provided in $\mathrm{Q}(i)$, or an interpolated value if the quantile falls between two data values.

6: 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: $\mathrm{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

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

Errors or warnings detected by the routine:

$\mathrm{IFAIL} = 1$

On entry, $\mathrm{N} < 1$.

$\mathrm{IFAIL} = 2$

On entry, $\mathrm{NQ} < 1$.

$\mathrm{IFAIL} = 3$

On entry, some $\mathrm{Q} < 0.0$ or $\mathrm{Q} > 1.0$.

$\mathrm{IFAIL} = 4$

On entry, Q is not in ascending order.

$\mathrm{IFAIL} = 5$

Internal error. Check all array sizes and calls to G01AMF. Please contact NAG.

7 Accuracy

Not applicable.

8 Further Comments

The average time taken by G01AMF is approximately proportional to $\mathrm{N} \times (1 + \log^{arg} \mathrm{NQ})$. The worst case time is proportional to N^2 but this is extremely unlikely to occur.

9 Example

This example computes a list of quantiles from an array of reals and an array of point values.

9.1 Program Text

```
    Program g01amfe

!     G01AMF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: g01amf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, n, nq
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: q(:), qv(:), rv(:)
!     .. Executable Statements ..
      Write (nout,*) 'G01AMF Example Program Results'
      Write (nout,*)

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

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

      Allocate (q(nq),qv(nq),rv(n))

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

!     Read in the required quantiles
      Read (nin,*) q(1:nq)

!     Display data
      Write (nout,*) 'Data Values:'
      Write (nout,99998) rv(1:n)
      Write (nout,*)

!     Calculate the quantiles
      ifail = 0
      Call g01amf(n,rv,nq,q,qv,ifail)

!     Display results
      Write (nout,*) 'Quantile     Result'
      Write (nout,*)
      Write (nout,99999)(q(i),qv(i),i=1,nq)

99999 Format (1X,F7.2,4X,F7.2)
99998 Format (1X,20F7.2)
    End Program g01amfe
```

9.2 Program Data

```
G01AMF Example Program Data
11 3                                        : N and NQ
4.9 7.0 3.9 9.5 1.3 3.1 9.7 0.3 8.5 0.6 6.2 : The N data values
0.25 0.5 1.0                                : The NQ required quantiles
```

9.3 Program Results

```
 G01AMF Example Program Results

 Data Values:
    4.90   7.00   3.90   9.50   1.30   3.10   9.70   0.30   8.50   0.60   6.20
```

```
 Quantile     Result

    0.25       2.20
    0.50       4.90
    1.00       9.70
```