NAG Library Routine Document

G01SCF

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

G01SCF returns a number of lower or upper tail probabilities for the χ^2-distribution with real degrees of freedom.

2 Specification

```
SUBROUTINE G01SCF (LTAIL, TAIL, LX, X, LDF, DF, P, IVALID, IFAIL)

INTEGER             LTAIL, LX, LDF, IVALID(*), IFAIL
REAL (KIND=nag_wp)  X(LX), DF(LDF), P(*)
CHARACTER(1)        TAIL(LTAIL)
```

3 Description

The lower tail probability for the χ^2-distribution with ν_i degrees of freedom, $P = (X_i \leq x_i : \nu_i)$ is defined by:

$$P = (X_i \leq x_i : \nu_i) = \frac{1}{2^{\nu_i/2}\Gamma(\nu_i/2)} \int_{0.0}^{x_i} X_i^{\nu_i/2-1} e^{-X_i/2}\, dX_i, \qquad x_i \geq 0, \nu_i > 0.$$

To calculate $P = (X_i \leq x_i : \nu_i)$ a transformation of a gamma distribution is employed, i.e., a χ^2-distribution with ν_i degrees of freedom is equal to a gamma distribution with scale parameter 2 and shape parameter $\nu_i/2$.

The input arrays to this routine are designed to allow maximum flexibility in the supply of vector parameters by re-using elements of any arrays that are shorter than the total number of evaluations required. See Section 2.6 in the G01 Chapter Introduction for further information.

4 References

Abramowitz M and Stegun I A (1972) *Handbook of Mathematical Functions* (3rd Edition) Dover Publications

Hastings N A J and Peacock J B (1975) *Statistical Distributions* Butterworth

5 Parameters

1: LTAIL – INTEGER *Input*

On entry: the length of the array TAIL.

Constraint: LTAIL > 0.

2: TAIL(LTAIL) – CHARACTER(1) array *Input*

On entry: indicates whether the lower or upper tail probabilities are required. For $j = ((i-1) \bmod \text{LTAIL}) + 1$, for $i = 1, 2, \ldots, \max(\text{LTAIL}, \text{LX}, \text{LDF})$:

TAIL(j) = 'L'

The lower tail probability is returned, i.e., $p_i = P(X_i \leq x_i : \nu_i)$.

TAIL(j) = 'U'

The upper tail probability is returned, i.e., $p_i = P(X_i \geq x_i : \nu_i)$.

Constraint: TAIL(j) = 'L' or 'U', for $j = 1, 2, \ldots, \text{LTAIL}$.

3: LX – INTEGER *Input*

On entry: the length of the array X.

Constraint: LX > 0.

4: X(LX) – REAL (KIND=nag_wp) array *Input*

On entry: x_i, the values of the χ^2 variates with ν_i degrees of freedom with $x_i = \text{X}(j)$, $j = ((i-1) \bmod \text{LX}) + 1$.

Constraint: X$(j) \geq 0.0$, for $j = 1, 2, \ldots, \text{LX}$.

5: LDF – INTEGER *Input*

On entry: the length of the array DF.

Constraint: LDF > 0.

6: DF(LDF) – REAL (KIND=nag_wp) array *Input*

On entry: ν_i, the degrees of freedom of the χ^2-distribution with $\nu_i = \text{DF}(j)$, $j = ((i-1) \bmod \text{LDF}) + 1$.

Constraint: DF$(j) > 0.0$, for $j = 1, 2, \ldots, \text{LDF}$.

7: P$(*)$ – REAL (KIND=nag_wp) array *Output*

Note: the dimension of the array P must be at least $\max(\text{LTAIL}, \text{LDF}, \text{LX})$.

On exit: p_i, the probabilities for the χ^2 distribution.

8: IVALID$(*)$ – INTEGER array *Output*

Note: the dimension of the array IVALID must be at least $\max(\text{LTAIL}, \text{LDF}, \text{LX})$.

On exit: IVALID(i) indicates any errors with the input arguments, with

IVALID$(i) = 0$

No error.

IVALID$(i) = 1$

On entry, invalid value supplied in TAIL when calculating p_i.

IVALID$(i) = 2$

On entry, $x_i < 0.0$.

IVALID$(i) = 3$

On entry, $\nu_i \leq 0.0$.

IVALID$(i) = 4$

The solution has failed to converge while calculating the gamma variate. The result returned should represent an approximation to the solution.

9: 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, because for this routine the values of the output parameters may be useful even if IFAIL $\neq 0$ on exit, the recommended value is -1. **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).

Note: G01SCF may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the routine:

IFAIL $= 1$

On entry, at least one value of X, DF or TAIL was invalid, or the solution failed to converge.
Check IVALID for more information.

IFAIL $= 2$

On entry, array size = ⟨*value*⟩.
Constraint: LTAIL > 0.

IFAIL $= 3$

On entry, array size = ⟨*value*⟩.
Constraint: LX > 0.

IFAIL $= 4$

On entry, array size = ⟨*value*⟩.
Constraint: LDF > 0.

IFAIL $= -999$

Dynamic memory allocation failed.

7 Accuracy

A relative accuracy of five significant figures is obtained in most cases.

8 Further Comments

For higher accuracy the transformation described in Section 3 may be used with a direct call to S14BAF.

9 Example

Values from various χ^2-distributions are read, the lower tail probabilities calculated, and all these values printed out.

9.1 Program Text

```
    Program g01scfe
!     G01SCF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: g01scf, nag_wp
!     .. Implicit None Statement ..
```

```
      Implicit None
!     .. Parameters ..
      Integer, Parameter                 :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                            :: i, ifail, ldf, lout, ltail, lx
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable    :: df(:), p(:), x(:)
      Integer, Allocatable               :: ivalid(:)
      Character (1), Allocatable         :: tail(:)
!     .. Intrinsic Procedures ..
      Intrinsic                          :: max, mod, repeat
!     .. Executable Statements ..
      Write (nout,*) 'G01SCF Example Program Results'
      Write (nout,*)

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

!     Read in the input vectors
      Read (nin,*) ltail
      Allocate (tail(ltail))
      Read (nin,*) tail(1:ltail)

      Read (nin,*) lx
      Allocate (x(lx))
      Read (nin,*) x(1:lx)

      Read (nin,*) ldf
      Allocate (df(ldf))
      Read (nin,*) df(1:ldf)

!     Allocate memory for output
      lout = max(ltail,lx,ldf)
      Allocate (p(lout),ivalid(lout))

!     Calculate probability
      ifail = -1
      Call g01scf(ltail,tail,lx,x,ldf,df,p,ivalid,ifail)

      If (ifail==0 .Or. ifail==1) Then
!       Display titles
        Write (nout,*) '   TAIL     X        DF        P      IVALID'
        Write (nout,*) repeat('-',45)

!       Display results
        Do i = 1, lout
          Write (nout,99999) tail(mod(i-1,ltail)+1), x(mod(i-1,lx)+1), &
            df(mod(i-1,ldf)+1), p(i), ivalid(i)
        End Do
      End If

99999 Format (5X,A1,2(4X,F6.2),4X,F6.3,4X,I3)
    End Program g01scfe
```

9.2 Program Data

```
G01SCF Example Program Data
1                       :: LTAIL
'L'                     :: TAIL
3                       :: LX
8.26 6.2 55.76          :: X
3                       :: LDF
20.0 7.5 45.0           :: DF
```

9.3 Program Results

```
 G01SCF Example Program Results

    TAIL      X         DF         P      IVALID
 ---------------------------------------------
     L      8.26     20.00     0.010       0
     L      6.20      7.50     0.428       0
     L     55.76     45.00     0.869       0
```