NAG Library Routine Document

G01TBF

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

G01TBF returns a number of deviates associated with given probabilities of Student's t-distribution with real degrees of freedom.

2 Specification

```
SUBROUTINE G01TBF (LTAIL, TAIL, LP, P, LDF, DF, T, IVALID, IFAIL)
INTEGER            LTAIL, LP, LDF, IVALID(*), IFAIL
REAL (KIND=nag_wp) P(LP), DF(LDF), T(*)
CHARACTER(1)       TAIL(LTAIL)
```

3 Description

The deviate, t_{p_i} associated with the lower tail probability, p_i, of the Student's t-distribution with ν_i degrees of freedom is defined as the solution to

$$P\left(T_i < t_{p_i} : \nu_i\right) = p_i = \frac{\Gamma((\nu_i+1)/2)}{\sqrt{\nu_i\pi}\Gamma(\nu_i/2)}\int_{-\infty}^{t_{p_i}}\left(1+\frac{T_i^2}{\nu_i}\right)^{-(\nu_i+1)/2} dT_i, \quad \nu_i \geq 1; -\infty < t_{p_i} < \infty.$$

For $\nu_i = 1$ or 2 the integral equation is easily solved for t_{p_i}.

For other values of $\nu_i < 3$ a transformation to the beta distribution is used and the result obtained from G01FEF.

For $\nu_i \geq 3$ an inverse asymptotic expansion of Cornish–Fisher type is used. The algorithm is described by Hill (1970).

The input arrays to this routine are designed to allow maximum flexibility in the supply of vector arguments 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

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

Hill G W (1970) Student's t-distribution *Comm. ACM* **13(10)** 617–619

5 Arguments

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 which tail the supplied probabilities represent. For $j = ((i-1) \bmod \text{LTAIL}) + 1$, for $i = 1, 2, \ldots, \max(\text{LTAIL}, \text{LP}, \text{LDF})$:

TAIL(j) = 'L'
The lower tail probability, i.e., $p_i = P\left(T_i \leq t_{p_i} : \nu_i\right)$.

TAIL(j) = 'U'
The upper tail probability, i.e., $p_i = P\left(T_i \geq t_{p_i} : \nu_i\right)$.

TAIL(j) = 'C'
The two tail (confidence interval) probability,
i.e., $p_i = P\left(T_i \leq \left|t_{p_i}\right| : \nu_i\right) - P\left(T_i \leq -\left|t_{p_i}\right| : \nu_i\right)$.

TAIL(j) = 'S'
The two tail (significance level) probability,
i.e., $p_i = P\left(T_i \geq \left|t_{p_i}\right| : \nu_i\right) + P\left(T_i \leq -\left|t_{p_i}\right| : \nu_i\right)$.

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

3: LP – INTEGER *Input*

On entry: the length of the array P.

Constraint: LP > 0.

4: P(LP) – REAL (KIND=nag_wp) array *Input*

On entry: p_i, the probability of the required Student's t-distribution as defined by TAIL with $p_i = \text{P}(j)$, $j = ((i-1) \bmod \text{LP}) + 1$.

Constraint: $0.0 < \text{P}(j) < 1.0$, for $j = 1, 2, \ldots, \text{LP}$.

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 Student's t-distribution with $\nu_i = \text{DF}(j)$, $j = ((i-1) \bmod \text{LDF}) + 1$.

Constraint: DF$(j) \geq 1.0$, for $j = 1, 2, \ldots, \text{LDF}$.

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

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

On exit: t_{p_i}, the deviates for the Student's t-distribution.

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

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

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 t_{p_i}.

IVALID$(i) = 2$

On entry, $p_i \leq 0.0$,
or $p_i \geq 1.0$.

IVALID$(i) = 3$

On entry, $\nu_i < 1.0$.

IVALID$(i) = 4$
The solution has failed to converge. 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 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, at least one value of TAIL, P or DF was invalid, or the solution failed to converge. Check IVALID for more information.

IFAIL $= 2$

On entry, array size $= \langle value \rangle$.
Constraint: LTAIL > 0.

IFAIL $= 3$

On entry, array size $= \langle value \rangle$.
Constraint: LP > 0.

IFAIL $= 4$

On entry, array size $= \langle value \rangle$.
Constraint: LDF > 0.

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 results should be accurate to five significant digits, for most argument values. The error behaviour for various argument values is discussed in Hill (1970).

8 Parallelism and Performance

G01TBF is not threaded in any implementation.

9 Further Comments

The value t_{p_i} may be calculated by using a transformation to the beta distribution and calling G01TEF. This routine allows you to set the required accuracy.

10 Example

This example reads the probability, the tail that probability represents and the degrees of freedom for a number of Student's t-distributions and computes the corresponding deviates.

10.1 Program Text

```
    Program g01tbfe
!     G01TBF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: g01tbf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, ldf, lout, lp, ltail
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: df(:), p(:), t(:)
      Integer, Allocatable             :: ivalid(:)
      Character (1), Allocatable       :: tail(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, mod, repeat
!     .. Executable Statements ..
      Write (nout,*) 'G01TBF 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,*) lp
      Allocate (p(lp))
      Read (nin,*) p(1:lp)

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

!     Allocate memory for output
      lout = max(lp,ldf,ltail)
      Allocate (ivalid(lout),t(lout))

!     Calculate deviates (inverse CDF)
      ifail = -1
```

```
      Call g01tbf(ltail,tail,lp,p,ldf,df,t,ivalid,ifail)

      If (ifail==0 .Or. ifail==1) Then
!       Display titles
        Write (nout,*) '   TAIL      P         DF         T      IVALID'
        Write (nout,*) repeat('-',47)

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

99999 Format (5X,A1,4X,F6.3,4X,F6.2,3X,F7.3,4X,I3)
    End Program g01tbfe
```

10.2 Program Data

```
G01TBF Example Program Data
3                       :: LTAIL
'S' 'L' 'C'             :: TAIL
3                       :: LP
0.01 0.01 0.99          :: P
3                       :: LDF
20.0 7.5 45.0           :: DF
```

10.3 Program Results

```
 G01TBF Example Program Results

    TAIL      P         DF         T      IVALID
 ------------------------------------------------
     S     0.010     20.00     2.845      0
     L     0.010      7.50    -2.943      0
     C     0.990     45.00     2.690      0
```