NAG Library Routine Document

G01SBF

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

G01SBF returns a number of one or two tail probabilities for the Student's t-distribution with real degrees of freedom.

2 Specification

```
SUBROUTINE G01SBF (LTAIL, TAIL, LT, T, LDF, DF, P, IVALID, IFAIL)
INTEGER            LTAIL, LT, LDF, IVALID(*), IFAIL
REAL (KIND=nag_wp) T(LT), DF(LDF), P(*)
CHARACTER(1)       TAIL(LTAIL)
```

3 Description

The lower tail probability for the Student's t-distribution with ν_i degrees of freedom, $P\left(T_i \leq t_i : \nu_i\right)$ is defined by:

$$P\left(T_i \leq t_i : \nu_i\right) = \frac{\Gamma((\nu_i+1)/2)}{\sqrt{\pi\nu_i}\Gamma(\nu_i/2)}\int_{-\infty}^{t_i}\left[1+\frac{{T_i}^2}{\nu_i}\right]^{-(\nu_i+1)/2} dT_i, \quad \nu_i \geq 1.$$

Computationally, there are two situations:

(i) when $\nu_i < 20$, a transformation of the beta distribution, $P_{\beta_i}\left(B_i \leq \beta_i : a_i, b_i\right)$ is used

$$P\left(T_i \leq t_i : \nu_i\right) = \tfrac{1}{2}P_{\beta_i}\left(B_i \leq \frac{\nu_i}{\nu_i + {t_i}^2} : \nu_i/2, \tfrac{1}{2}\right) \quad \text{when } t_i < 0.0$$

or

$$P\left(T_i \leq t_i : \nu_i\right) = \tfrac{1}{2} + \tfrac{1}{2}P_{\beta_i}\left(B_i \geq \frac{\nu_i}{\nu_i + {t_i}^2} : \nu_i/2, \tfrac{1}{2}\right) \quad \text{when } t_i > 0.0;$$

(ii) when $\nu_i \geq 20$, an asymptotic normalizing expansion of the Cornish–Fisher type is used to evaluate the probability, see 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

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

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 returned probabilities should represent. For $j = ((i-1) \bmod \text{LTAIL}) + 1$, for $i = 1, 2, \ldots, \max(\text{LTAIL}, \text{LT}, \text{LDF})$:

TAIL(j) = 'L'
The lower tail probability is returned, i.e., $p_i = P(T_i \leq t_i : \nu_i)$.

TAIL(j) = 'U'
The upper tail probability is returned, i.e., $p_i = P(T_i \geq t_i : \nu_i)$.

TAIL(j) = 'C'
The two tail (confidence interval) probability is returned, i.e., $p_i = P(T_i \leq |t_i| : \nu_i) - P(T_i \leq -|t_i| : \nu_i)$.

TAIL(j) = 'S'
The two tail (significance level) probability is returned, i.e., $p_i = P(T_i \geq |t_i| : \nu_i) + P(T_i \leq -|t_i| : \nu_i)$.

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

3: LT – INTEGER *Input*

On entry: the length of the array T.

Constraint: LT > 0.

4: T(LT) – REAL (KIND=nag_wp) array *Input*

On entry: t_i, the values of the Student's t variates with $t_i = \text{T}(j)$, $j = ((i-1) \bmod \text{LT}) + 1$.

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

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

On exit: p_i, the probabilities for the Student's t distribution.

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

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

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 p_i.

IVALID$(i) = 2$

On entry, $\nu_i < 1.0$.

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 or DF was invalid.
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: LT > 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 computed probability should be accurate to five significant places for reasonable probabilities but there will be some loss of accuracy for very low probabilities (less than 10^{-10}), see Hastings and Peacock (1975).

8 Parallelism and Performance

G01SBF is not threaded in any implementation.

9 Further Comments

The probabilities could also be obtained by using the appropriate transformation to a beta distribution (see Abramowitz and Stegun (1972)) and using G01SEF. This routine allows you to set the required accuracy.

10 Example

This example reads values from, and degrees of freedom for Student's *t*-distributions along with the required tail. The probabilities are calculated and printed.

10.1 Program Text

```
    Program g01sbfe
!     G01SBF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: g01sbf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, ldf, lout, lt, 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,*) 'G01SBF 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,*) lt
      Allocate (t(lt))
      Read (nin,*) t(1:lt)

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

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

```
!     Calculate probability
      ifail = -1
      Call g01sbf(ltail,tail,lt,t,ldf,df,p,ivalid,ifail)

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

!       Display results
        Do i = 1, lout
          Write (nout,99999) tail(mod(i-1,ltail)+1), t(mod(i-1,lt)+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 g01sbfe
```

10.2 Program Data

```
G01SBF Example Program Data
4                       :: LTAIL
'L' 'S' 'C' 'U'         :: TAIL
1                       :: LT
0.85                    :: T
1                       :: LDF
20.0                    :: DF
```

10.3 Program Results

```
 G01SBF Example Program Results

    TAIL      T        DF        P      IVALID
 ------------------------------------------------
     L      0.85     20.00     0.797      0
     S      0.85     20.00     0.405      0
     C      0.85     20.00     0.595      0
     U      0.85     20.00     0.203      0
```