NAG Library Routine Document

G01TAF

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

G01TAF returns a number of deviates associated with given probabilities of the Normal distribution.

2 Specification

```
SUBROUTINE G01TAF (LTAIL, TAIL, LP, P, LXMU, XMU, LXSTD, XSTD, X,          &
                   IVALID, IFAIL)

INTEGER             LTAIL, LP, LXMU, LXSTD, IVALID(*), IFAIL
REAL (KIND=nag_wp) P(LP), XMU(LXMU), XSTD(LXSTD), X(*)
CHARACTER(1)        TAIL(LTAIL)
```

3 Description

The deviate, x_{p_i} associated with the lower tail probability, p_i, for the Normal distribution is defined as the solution to

$$P\left(X_i \leq x_{p_i}\right) = p_i = \int_{-\infty}^{x_{p_i}} Z_i(X_i)\, dX_i,$$

where

$$Z_i(X_i) = \frac{1}{\sqrt{2\pi{\sigma_i}^2}} e^{-(X_i-\mu_i)^2/\left(2{\sigma_i}^2\right)},\ -\infty < X_i < \infty.$$

The method used is an extension of that of Wichura (1988). p_i is first replaced by $q_i = p_i - 0.5$.

(a) If $|q_i| \leq 0.3$, z_i is computed by a rational Chebyshev approximation

$$z_i = s_i \frac{A_i\left({s_i}^2\right)}{B_i\left({s_i}^2\right)},$$

where $s_i = \sqrt{2\pi} q_i$ and A_i, B_i are polynomials of degree 7.

(b) If $0.3 < |q_i| \leq 0.42$, z_i is computed by a rational Chebyshev approximation

$$z_i = \text{sign}\, q_i \left(\frac{C_i(t_i)}{D_i(t_i)}\right),$$

where $t_i = |q_i| - 0.3$ and C_i, D_i are polynomials of degree 5.

(c) If $|q_i| > 0.42$, z_i is computed as

$$z_i = \text{sign}\, q_i \left[\left(\frac{E_i(u_i)}{F_i(u_i)}\right) + u_i\right],$$

where $u_i = \sqrt{-2 \times \log\left(\min(p_i, 1-p_i)\right)}$ and E_i, F_i are polynomials of degree 6.

x_{p_i} is then calculated from z_i, using the relationsship $z_{p_i} = \frac{x_i - \mu_i}{\sigma_i}$.

For the upper tail probability $-x_{p_i}$ is returned, while for the two tail probabilities the value $x_{ip_i{}^*}$ is returned, where ${p_i}^*$ is the required tail probability computed from the input value of p_i.

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

Wichura (1988) Algorithm AS 241: the percentage points of the Normal distribution *Appl. Statist.* **37** 477–484

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. Letting Z denote a variate from a standard Normal distribution, and $z_i = \frac{x_{p_i} - \mu_i}{\sigma_i}$, then for $j = ((i-1) \bmod \text{LTAIL}) + 1$, for $i = 1, 2, \ldots, \max(\text{LTAIL}, \text{LP}, \text{LXMU}, \text{LXSTD})$:

TAIL(j) = 'L'
The lower tail probability, i.e., $p_i = P(Z \leq z_i)$.

TAIL(j) = 'U'
The upper tail probability, i.e., $p_i = P(Z \geq z_i)$.

TAIL(j) = 'C'
The two tail (confidence interval) probability, i.e., $p_i = P(Z \leq |z_i|) - P(Z \leq -|z_i|)$.

TAIL(j) = 'S'
The two tail (significance level) probability, i.e., $p_i = P(Z \geq |z_i|) + P(Z \leq -|z_i|)$.

Constraint: TAIL(j) = 'L', 'U', 'C' or 'S', for $j = 1, 2, \ldots,$ 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 probabilities for the Normal 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 $j = 1, 2, \ldots,$ LP.

5: LXMU – INTEGER *Input*

On entry: the length of the array XMU.

Constraint: LXMU > 0.

6: XMU(LXMU) – REAL (KIND=nag_wp) array *Input*

On entry: μ_i, the means with $\mu_i = \text{XMU}(j)$, $j = ((i-1) \bmod \text{LXMU}) + 1$.

7: LXSTD – INTEGER *Input*

On entry: the length of the array XSTD.

Constraint: LXSTD > 0.

8: XSTD(LXSTD) – REAL (KIND=nag_wp) array *Input*

On entry: σ_i, the standard deviations with $\sigma_i = \text{XSTD}(j)$, $j = ((i-1) \bmod \text{LXSTD}) + 1$.

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

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

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

On exit: x_{p_i}, the deviates for the Normal distribution.

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

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

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 x_{p_i}.

IVALID$(i) = 2$

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

IVALID$(i) = 3$

On entry, $\sigma_i \leq 0.0$.

11: 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, XSTD or P 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: LP > 0.

IFAIL $= 4$

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

IFAIL $= 5$

On entry, array size $= \langle value \rangle$.
Constraint: LXSTD > 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 accuracy is mainly limited by the ***machine precision***.

8 Parallelism and Performance

G01TAF is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example reads vectors of values for μ_i, σ_i and p_i and prints the corresponding deviates.

10.1 Program Text

```
    Program g01tafe
!     G01TAF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: g01taf, nag_wp
!     .. Implicit None Statement ..
```

```
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, lout, lp, ltail, lxmu,     &
                                          lxstd
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: p(:), x(:), xmu(:), xstd(:)
      Integer, Allocatable             :: ivalid(:)
      Character (1), Allocatable       :: tail(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, mod, repeat
!     .. Executable Statements ..
      Write (nout,*) 'G01TAF 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,*) lxmu
      Allocate (xmu(lxmu))
      Read (nin,*) xmu(1:lxmu)

      Read (nin,*) lxstd
      Allocate (xstd(lxstd))
      Read (nin,*) xstd(1:lxstd)

!     Allocate memory for output
      lout = max(lp,ltail,lxmu,lxstd)
      Allocate (x(lout),ivalid(lout))

!     Calculate the deviate (inverse CDF)
      ifail = -1
      Call g01taf(ltail,tail,lp,p,lxmu,xmu,lxstd,xstd,x,ivalid,ifail)

      If (ifail==0 .Or. ifail==1) Then
!       Display titles
        Write (nout,*)
        Write (nout,*)                                                        &
          '   TAIL      P        XMU      XSTD       X     IVALID'
        Write (nout,*) repeat('-',56)

!       Display results
        Do i = 1, lout
          Write (nout,99999) tail(mod(i-1,ltail)+1), p(mod(i-1,lp)+1),        &
            xmu(mod(i-1,lxmu)+1), xstd(mod(i-1,lxstd)+1), x(i), ivalid(i)
        End Do
      End If

99999 Format (5X,A1,4X,F6.3,2(4X,F6.3),3X,F7.3,4X,I3)
    End Program g01tafe
```

10.2 Program Data

```
G01TAF Example Program Data
4                         :: LTAIL
'L' 'U' 'C' 'S'           :: TAIL
4                         :: LP
0.975 0.025 0.95 0.05     :: P
1                         :: LXMU
0.0                       :: XMU
1                         :: LXSTD
1.0                       :: XSTD
```

10.3 Program Results

```
 G01TAF Example Program Results


    TAIL      P         XMU       XSTD        X      IVALID
 ------------------------------------------------------------
     L     0.975      0.000      1.000      1.960       0
     U     0.025      0.000      1.000      1.960       0
     C     0.950      0.000      1.000      1.960       0
     S     0.050      0.000      1.000      1.960       0
```