NAG Toolbox

nag_stat_inv_cdf_f_vector (g01td)

1 Purpose

nag_stat_inv_cdf_f_vector (g01td) returns a number of deviates associated with given probabilities of the F or variance-ratio distribution with real degrees of freedom.

2 Syntax

```
[f, ivalid, ifail] = nag_stat_inv_cdf_f_vector(tail, p, df1, df2, 'ltail', ltail, 'lp', lp, 'ldf1', ldf1, 'ldf2', ldf2)
[f, ivalid, ifail] = g01td(tail, p, df1, df2, 'ltail', ltail, 'lp', lp, 'ldf1', ldf1, 'ldf2', ldf2)
```

3 Description

The deviate, f_{p_i}, associated with the lower tail probability, p_i, of the F-distribution with degrees of freedom u_i and v_i is defined as the solution to

$$P\left(F_i \leq f_{p_i} : u_i, v_i\right) = p_i = \frac{u_i^{\frac{1}{2}u_i} v_i^{\frac{1}{2}v_i} \Gamma\left(\frac{u_i+v_i}{2}\right)}{\Gamma\left(\frac{u_i}{2}\right)\Gamma\left(\frac{v_i}{2}\right)} \int_0^{f_{p_i}} F_i^{\frac{1}{2}(u_i-2)} \left(v_i + u_i F_i\right)^{-\frac{1}{2}(u_i+v_i)} \, dF_i,$$

where $u_i, v_i > 0$; $0 \leq f_{p_i} < \infty$.

The value of f_{p_i} is computed by means of a transformation to a beta distribution, $P_{i\beta_i}\left(B_i \leq \beta_i : a_i, b_i\right)$:

$$P\left(F_i \leq f_{p_i} : u_i, v_i\right) = P_{i\beta_i}\left(B_i \leq \frac{u_i f_{p_i}}{u_i f_{p_i} + v_i} : u_i/2, v_i/2\right)$$

and using a call to nag_stat_inv_cdf_beta_vector (g01te).

For very large values of both u_i and v_i, greater than 10^5, a Normal approximation is used. If only one of u_i or v_i is greater than 10^5 then a χ^2 approximation is used; see Abramowitz and Stegun (1972).

The input arrays to this function 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

5 Parameters

5.1 Compulsory Input Parameters

1: **tail**(**ltail**) – CHARACTER(1) array

Indicates which tail the supplied probabilities represent. For $j = ((i-1) \bmod \textbf{ltail}) + 1$, for $i = 1, 2, \ldots, \max(\textbf{ltail}, \textbf{lp}, \textbf{ldf1}, \textbf{ldf2})$:

tail(j) = 'L'

The lower tail probability, i.e., $p_i = P\left(F_i \leq f_{p_i} : u_i, v_i\right)$.

tail(j) = 'U'
The upper tail probability, i.e., $p_i = P\left(F_i \geq f_{p_i} : u_i, v_i\right)$.

Constraint: **tail**(j) = 'L' or 'U', for $j = 1, 2, \ldots,$ **ltail**.

2: **p**(**lp**) – REAL (KIND=nag_wp) array

p_i, the probability of the required F-distribution as defined by **tail** with $p_i =$ **p**(j), $j = ((i-1) \bmod$ **lp**$) + 1$.

Constraints:

if **tail**(k) = 'L', 0.0 $\leq$ **p**$(j) <$ 1.0;
otherwise 0.0 $<$ **p**$(j) \leq$ 1.0.

Where $k = (i-1) \bmod$ **ltail** $+ 1$ and $j = (i-1) \bmod$ **lp** $+ 1$.

3: **df1**(**ldf1**) – REAL (KIND=nag_wp) array

u_i, the degrees of freedom of the numerator variance with $u_i =$ **df1**(j), $j = ((i-1) \bmod$ **ldf1**$) + 1$.

Constraint: **df1**$(j) > 0.0$, for $j = 1, 2, \ldots,$ **ldf1**.

4: **df2**(**ldf2**) – REAL (KIND=nag_wp) array

v_i, the degrees of freedom of the denominator variance with $v_i =$ **df2**(j), $j = ((i-1) \bmod$ **ldf2**$) + 1$.

Constraint: **df2**$(j) > 0.0$, for $j = 1, 2, \ldots,$ **ldf2**.

5.2 Optional Input Parameters

1: **ltail** – INTEGER

Default: the dimension of the array **tail**.

The length of the array **tail**.

Constraint: **ltail** > 0.

2: **lp** – INTEGER

Default: the dimension of the array **p**.

The length of the array **p**.

Constraint: **lp** > 0.

3: **ldf1** – INTEGER

Default: the dimension of the array **df1**.

The length of the array **df1**.

Constraint: **ldf1** > 0.

4: **ldf2** – INTEGER

Default: the dimension of the array **df2**.

The length of the array **df2**.

Constraint: **ldf2** > 0.

5.3 Output Parameters

1: **f**(:) – REAL (KIND=nag_wp) array

The dimension of the array **f** will be max(**ltail**, **lp**, **ldf1**, **ldf2**)

f_{p_i}, the deviates for the F-distribution.

2: **ivalid**(:) – INTEGER array

The dimension of the array **ivalid** will be max(**ltail**, **lp**, **ldf1**, **ldf2**)

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 f_{p_i}.

ivalid$(i) = 2$

On entry, invalid value for p_i.

ivalid$(i) = 3$

On entry, $u_i \leq 0.0$,
or $v_i \leq 0.0$.

ivalid$(i) = 4$
The solution has not converged. The result should still be a reasonable approximation to the solution.

ivalid$(i) = 5$
The value of p_i is too close to 0.0 or 1.0 for the result to be computed. This will only occur when the large sample approximations are used.

3: **ifail** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

6 Error Indicators and Warnings

Note: nag_stat_inv_cdf_f_vector (g01td) may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the function:

ifail $= 1$ (*warning*)

On entry, at least one value of **tail**, **p**, **df1**, **df2** was invalid, or the solution failed to converge. Check **ivalid** for more information.

ifail $= 2$

Constraint: **ltail** > 0.

ifail $= 3$

Constraint: **lp** > 0.

ifail $= 4$

Constraint: **ldf1** > 0.

ifail $= 5$

Constraint: **ldf2** > 0.

ifail $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

ifail $= -399$

Your licence key may have expired or may not have been installed correctly.

ifail $= -999$

Dynamic memory allocation failed.

7 Accuracy

The result should be accurate to five significant digits.

8 Further Comments

For higher accuracy nag_stat_inv_cdf_beta_vector (g01te) can be used along with the transformations given in Section 3.

9 Example

This example reads the lower tail probabilities for several F-distributions, and calculates and prints the corresponding deviates.

9.1 Program Text

```
function g01td_example

fprintf('g01td example results\n\n');

tail = {'L'};
p    = [0.984; 0.9; 0.534];
df1  = [10;    1;  20.25];
df2  = [25.5;  1;   1];

[f, ivalid, ifail] = g01td( ...
                           tail, p, df1, df2);

fprintf('   tail   p      df1     df2      f    ivalid\n');
ltail = numel(tail);
lp    = numel(p);
ldf1  = numel(df1);
ldf2  = numel(df2);
len   = max ([ltail, lp, ldf1, ldf2]);
for i=0:len-1
  fprintf('%5s%7.3f%8.3f%8.3f%8.3f%7d\n', tail{mod(i, ltail)+1}, ...
          p(mod(i,lp)+1), df1(mod(i,ldf1)+1), df2(mod(i,ldf2)+1), ...
          f(i+1), ivalid(i+1));
end
```

9.2 Program Results

```
g01td example results

   tail   p      df1     df2      f    ivalid
    L  0.984  10.000  25.500   2.847      0
    L  0.900   1.000   1.000  39.863      0
    L  0.534  20.250   1.000   2.498      0
```