NAG Toolbox

nag_stat_inv_cdf_beta (g01fe)

1 Purpose

nag_stat_inv_cdf_beta (g01fe) returns the deviate associated with the given lower tail probability of the beta distribution.

2 Syntax

```
[result, ifail] = nag_stat_inv_cdf_beta(p, a, b, 'tol', tol)
[result, ifail] = g01fe(p, a, b, 'tol', tol)
```

Note: the interface to this routine has changed since earlier releases of the toolbox:

At Mark 23: **tol** was made optional (default 0).

3 Description

The deviate, β_p, associated with the lower tail probability, p, of the beta distribution with parameters a and b is defined as the solution to

$$P\left(B \leq \beta_p : a, b\right) = p = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \int_0^{\beta_p} B^{a-1}(1-B)^{b-1}\, dB, \quad 0 \leq \beta_p \leq 1; a, b > 0.$$

The algorithm is a modified version of the Newton–Raphson method, following closely that of Cran *et al.* (1977).

An initial approximation, β_0, to β_p is found (see Cran *et al.* (1977)), and the Newton–Raphson iteration

$$\beta_i = \beta_{i-1} - \frac{f(\beta_{i-1})}{f'(\beta_{i-1})},$$

where $f(\beta) = P(B \leq \beta : a, b) - p$ is used, with modifications to ensure that β remains in the range $(0, 1)$.

4 References

Cran G W, Martin K J and Thomas G E (1977) Algorithm AS 109. Inverse of the incomplete beta function ratio *Appl. Statist.* **26** 111–114

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

5 Parameters

5.1 Compulsory Input Parameters

1: **p** – REAL (KIND=nag_wp)

p, the lower tail probability from the required beta distribution.

Constraint: $0.0 \leq \mathbf{p} \leq 1.0$.

2: **a** – REAL (KIND=nag_wp)

a, the first parameter of the required beta distribution.

Constraint: $0.0 < \mathbf{a} \leq 10^6$.

3: **b** – REAL (KIND=nag_wp)

b, the second parameter of the required beta distribution.

Constraint: $0.0 < \mathbf{b} \leq 10^6$.

5.2 Optional Input Parameters

1: **tol** – REAL (KIND=nag_wp)

Default: 0.0

The relative accuracy required by you in the result. If nag_stat_inv_cdf_beta (g01fe) is entered with **tol** greater than or equal to 1.0 or less than $10 \times$ ***machine precision*** (see nag_machine_precision (x02aj)), then the value of $10 \times$ ***machine precision*** is used instead.

5.3 Output Parameters

1: **result**

The result of the function.

2: **ifail** – INTEGER

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

6 Error Indicators and Warnings

Note: nag_stat_inv_cdf_beta (g01fe) may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the function:

If on exit **ifail** $= 1$ or 2, then nag_stat_inv_cdf_beta (g01fe) returns 0.0.

ifail $= 1$

On entry, $\mathbf{p} < 0.0$,
or $\mathbf{p} > 1.0$.

ifail $= 2$

On entry, $\mathbf{a} \leq 0.0$,
or $\mathbf{a} > 10^6$,
or $\mathbf{b} \leq 0.0$,
or $\mathbf{b} > 10^6$.

ifail $= 3$ (*warning*)

There is doubt concerning the accuracy of the computed result. 100 iterations of the Newton–Raphson method have been performed without satisfying the accuracy criterion (see Section 7). The result should be a reasonable approximation of the solution.

ifail $= 4$ (*warning*)

Requested accuracy not achieved when calculating beta probability. The result should be a reasonable approximation to the correct solution. You should try setting **tol** larger.

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 required precision, given by **tol**, should be achieved in most circumstances.

8 Further Comments

The typical timing will be several times that of nag_stat_prob_beta (g01ee) and will be very dependent on the input argument values. See nag_stat_prob_beta (g01ee) for further comments on timings.

9 Example

This example reads lower tail probabilities for several beta distributions and calculates and prints the corresponding deviates until the end of data is reached.

9.1 Program Text

```
     function g01fe_example

fprintf('g01fe example results\n\n');

p = [    0.50    0.99    0.25];
a = [    1.0     1.5    20.0 ];
b = [    2.0     1.5    10.0 ];
dev  = p;

fprintf('     p       a       b    deviate\n');
for j = 1:numel(p)
   [dev(j), ifail] = g01fe( ...
     p(j), a(j), b(j));
end

fprintf('%8.3f%8.3f%8.3f%8.3f\n', [p; a; b; dev]);
```

9.2 Program Results

```
     g01fe example results

     p       a       b    deviate
   0.500   1.000   2.000   0.293
   0.990   1.500   1.500   0.967
   0.250  20.000  10.000   0.611
```