Module 20.5: nag_beta_dist

Probabilities and Deviate for a Beta Distribution

`nag_beta_dist` provides procedures for computing the probabilities and the deviate for a beta distribution.

Contents

Procedure: nag_beta_prob

1 Description

nag_beta_prob returns the lower or upper tail probability for a beta distribution with parameters a and b.

2 Usage

```
USE nag_beta_dist
[value =] nag_beta_prob(tail, beta, a, b  [, optional arguments])
```

The function result is a scalar of type real(kind=*wp*).

3 Arguments

3.1 Mandatory Arguments

tail — character(len=1), intent(in)

Input: the type of tail probability to be returned:

if tail = 'L' or 'l', the lower tail probability is returned;

if tail = 'U' or 'u', the upper tail probability is returned.

Constraints: tail = 'L', 'l', 'U' or 'u'.

beta — real(kind=*wp*), intent(in)

Input: the value of the beta variate.

Constraints: $0.0 \leq$ beta ≤ 1.0.

a — real(kind=*wp*), intent(in)

Input: the first parameter of the beta distribution.

Constraints: $0.0 <$ a $\leq 10^6$.

b — real(kind=*wp*), intent(in)

Input: the second parameter of the beta distribution.

Constraints: $0.0 <$ b $\leq 10^6$.

3.2 Optional Arguments

Note. Optional arguments must be supplied by keyword, not by position. The order in which they are described below may differ from the order in which they occur in the argument list.

tol — real(kind=*wp*), intent(in), optional

Input: the relative accuracy for the probability.

Default: tol = 10×EPSILON(1.0_*wp*).

Note: if tol < 10×EPSILON(1.0_*wp*) or tol ≥ 1.0, the default value is used.

error — type(nag_error), intent(inout), optional

The NAG *fl*90 error-handling argument. See the Essential Introduction, or the module document nag_error_handling (1.2). You are recommended to omit this argument if you are unsure how to use it. If this argument is supplied, it *must* be initialized by a call to nag_set_error before this procedure is called.

4 Error Codes

Fatal errors (error%level = 3):

error%code	Description
301	An input argument has an invalid value.

Warnings (error%level = 1):

error%code	Description
101	The user required accuracy has not been achieved. Try using a larger value for `tol`, but the result returned should still be a good approximation.
102	`beta` is too far out into the tails for the probability to be evaluated exactly. The result returned is 0.0 or 1.0; this should be a good approximation to the required solution.

5 Examples of Usage

A complete example of the use of this procedure appears in Example 1 of this module document.

6 Further Comments

6.1 Mathematical Background

The probability density function of the beta distribution with parameters a and b is

$$f(B:a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} B^{a-1}(1-B)^{b-1}, \quad 0 \le B \le 1, \quad a,b > 0.$$

The lower tail probability, $P(B \le \beta : a,b)$, is defined by

$$P(B \le \beta : a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \int_0^\beta B^{a-1}(1-B)^{b-1} dB = I_\beta(a,b), \quad 0 \le \beta \le 1, \quad a,b > 0.$$

The function $I_x(a,b)$ is also known as the incomplete beta function.

6.2 Algorithmic Detail

The method used is similar to that described by Majumder and Bhattacharjee [3], and uses the following three relations for the incomplete beta function (see page 944 of Abramowitz and Stegun [1]).

$$I_x(a,b) = \frac{\Gamma(a+b)}{\Gamma(a+1)\Gamma(b)} x^a (1-x)^{b-1} + I_x(a+1,b-1). \quad (1)$$

$$I_x(a,b) = \frac{\Gamma(a+b)}{\Gamma(a+1)\Gamma(b)} x^a (1-x)^{b} + I_x(a+1,b). \quad (2)$$

$$I_x(a,b) = 1 - I_{1-x}(b,a). \quad (3)$$

If a is less than $(a+b)x$, then a and b are interchanged and $(1-x)$ replaces x, with equation (3) being used to obtain the final result.
Equation (1) is applied repeatedly until the second parameter is reduced to b', where $0 < b' \le 1$. This produces a power series of finite length, in $x/(1-x)$, whose sum is found. If $b' = 1$, this sum equals $I_x(a,b)$, since $I_x(c,1) = x^c/c$ for all $c > 0$. Otherwise $(0 < b' < 1)$, the integral $I_x(c,d)$, where $c = a + b - b'$ and $d = b'$, is evaluated using equation (2) repeatedly; this gives a convergent power series in x of infinite length.

6.3 Accuracy

The series generated by (2) is assumed to have converged when an upper bound on the sum of the remaining terms is less than `tol`. Summation also ceases if the relative change in the sum of the series is less than `EPSILON(1.0_`*wp*`)`, in which case full accuracy cannot be guaranteed.
The accuracy is limited by the error in evaluating the gamma function.

6.4 Timing

The time taken by the procedure depends on the shape of the distribution. If the distribution is highly skewed with one of the values of a, b large and the other small, the series generated by (2) will take longer to converge than for a distribution that is nearly symmetric.

Procedure: nag_beta_deviate

1 Description

nag_beta_deviate returns the deviate associated with a given lower tail probability of a beta distribution with parameters a and b.

2 Usage

```
USE nag_beta_dist
[value =] nag_beta_deviate(p, a, b  [, optional arguments])
```

The function result is a scalar of type real(kind=wp).

3 Arguments

3.1 Mandatory Arguments

p — real(kind=wp), intent(in)

Input: the lower tail probability of the beta distribution.

Constraints: $0.0 \leq$ p < 1.0.

a — real(kind=wp), intent(in)

Input: the first parameter of the beta distribution.

Constraints: $0.0 <$ a $\leq 10^6$.

b — real(kind=wp), intent(in)

Input: the second parameter of the beta distribution.

Constraints: $0.0 <$ b $\leq 10^6$.

3.2 Optional Arguments

Note. Optional arguments must be supplied by keyword, not by position. The order in which they are described below may differ from the order in which they occur in the argument list.

tol — real(kind=wp), intent(in), optional

Input: the relative accuracy which you want for the result.

Default: tol = 10×EPSILON(1.0_wp).

Note: if tol < 10×EPSILON(1.0_wp) or tol ≥ 1.0, the default value is used.

error — type(nag_error), intent(inout), optional

The NAG *fl*90 error-handling argument. See the Essential Introduction, or the module document nag_error_handling (1.2). You are recommended to omit this argument if you are unsure how to use it. If this argument is supplied, it *must* be initialized by a call to nag_set_error before this procedure is called.

4 Error Codes

Fatal errors (error%level = 3):

error%code	Description
301	An input argument has an invalid value.

Warnings (error%level = 1):

error%code	Description
101	The accuracy of the result is doubtful. This is because 100 iterations of the underlying Newton–Raphson method have been performed without satisfying the accuracy criterion. Nevertheless, the result should be a reasonable approximation to the solution.
102	The requested accuracy was not achieved when calculating the deviate. Although the result should be a reasonable approximation to the correct solution, a larger value for `tol` should probably be used.

5 Examples of Usage

A complete example of the use of this procedure appears in Example 1 of this module document.

6 Further Comments

6.1 Mathematical Background

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

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

6.2 Algorithmic Detail

The algorithm is a modified version of the Newton–Raphson method, following closely that of Cran *et al.* [2].
An initial approximation, β_0, to β_p is found (see [2]), 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).

6.3 Accuracy

The required precision given by `tol` should be achieved in most circumstances.

Example 1: Calculation of probabilities and the deviate for a beta distribution

This example program shows how `nag_beta_prob` returns the lower tail probability or upper tail probability for a beta distribution with parameters a and b. It also shows how `nag_beta_deviate` calculates the deviate (`beta_deviate`) associated with a given lower tail probability.

1 Program Text

Note. The listing of the example program presented below is double precision. Single precision users are referred to Section 5.2 of the Essential Introduction for further information.

```
    PROGRAM nag_beta_dist_ex01

      ! Example Program Text for nag_beta_dist
      ! NAG fl90, Release 3. NAG Copyright 1997.

      ! .. Use Statements ..
      USE nag_examples_io, ONLY : nag_std_out, nag_std_in
      USE nag_beta_dist, ONLY : nag_beta_prob, nag_beta_deviate
      ! .. Implicit None Statement ..
      IMPLICIT NONE
      ! .. Intrinsic Functions ..
      INTRINSIC KIND
      ! .. Parameters ..
      INTEGER, PARAMETER :: wp = KIND(1.0D0)
      ! .. Local Scalars ..
      REAL (wp) :: a, b, beta, beta_deviate, prob, probl
      CHARACTER (1) :: tail
      ! .. Executable Statements ..

      WRITE (nag_std_out,*) 'Example Program Results for nag_beta_dist_ex01'

      READ (nag_std_in,*)          ! Skip heading in data file

      WRITE (nag_std_out,*)
      WRITE (nag_std_out,*) &
       ' tail    beta      a       b        prob    deviate'
      WRITE (nag_std_out,*)

      DO
        READ (nag_std_in,*,end=20) tail, beta, a, b

        prob = nag_beta_prob(tail,beta,a,b)

        probl = prob
        IF (tail=='u' .OR. tail=='U') probl = 1.0_wp - prob

        beta_deviate = nag_beta_deviate(probl,a,b)

        WRITE (nag_std_out,'(3X,A,4x,5(F7.4,2X))') tail, beta, a, b, prob, &
         beta_deviate
      END DO
20    CONTINUE

    END PROGRAM nag_beta_dist_ex01
```

2 Program Data

```
Example Program Data for nag_beta_dist_ex01
'l'    0.25  1.0  2.0      :tail, beta, a, b
'U'    0.75  1.5  1.5
'L'    0.50  2.0  1.0
```

3 Program Results

```
 Example Program Results for nag_beta_dist_ex01

  tail    beta      a       b        prob    deviate

   l     0.2500   1.0000   2.0000   0.4375   0.2500
   U     0.7500   1.5000   1.5000   0.1955   0.7500
   L     0.5000   2.0000   1.0000   0.2500   0.5000
```

Additional Examples

Not all example programs supplied with NAG *fl*90 appear in full in this module document. The following additional examples, associated with this module, are available.

`nag_beta_dist_ex02`

Calculation of the deviate associated with a given lower tail probability for a beta distribution with known parameters.

References

[1] Abramowitz M and Stegun I A (1972) *Handbook of Mathematical Functions* Dover Publications (3rd Edition)

[2] Cran G W, Martin K J and Thomas G E (1977) Algorithm AS109. Inverse of the incomplete beta function ratio *Appl. Statist.* **26** 111–114

[3] Majumder K L and Bhattacharjee G P (1973) Algorithm AS63. The incomplete beta integral *Appl. Statist.* **22** 409–411