NAG C Library Function Document

nag_prob_non_central_beta_dist (g01gec)

1 Purpose

nag_prob_non_central_beta_dist (g01gec) returns the probability associated with the lower tail of the non-central beta distribution.

2 Specification

```
#include <nag.h>
#include <nagg01.h>
```

```
double nag_prob_non_central_beta_dist (double x, double a, double b,
          double lambda, double tol, Integer max_iter, NagError *fail)
```

3 Description

The lower tail probability for the non-central beta distribution with parameters a and b and non-centrality parameter λ, $P(B \leq \beta : a, b; \lambda)$, is defined by

$$P(B \leq \beta : a, b; \lambda) = \sum_{j=0}^{\infty} e^{-\lambda/2} \frac{(\lambda/2)}{j!} P(B \leq \beta : a, b; 0) \quad (1)$$

where

$$P(B \leq \beta : a, b; 0) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \int_0^{\beta} B^{a-1}(1-B)^{b-1} dB,$$

which is the central beta probability function or incomplete beta function.

Recurrence relationships given in Abramowitz and Stegun (1972) are used to compute the values of $P(B \leq \beta : a, b; 0)$ for each step of the summation (1).

The algorithm is discussed in Lenth (1987).

4 Parameters

1: **x** – double *Input*

On entry: the deviate, β, from the beta distribution, for which the probability $P(B \leq \beta : a, b; \lambda)$, is to be found.

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

2: **a** – double *Input*

On entry: the first parameter, a, of the required beta distribution.

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

3: **b** – double *Input*

On entry: the second parameter, b, of the required beta distribution.

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

4: **lambda** – double *Input*

On entry: the non-centrality parameter, λ, of the required beta distribution.

Constraint: $0.0 \leq$ **lambda** $\leq -2.0 \times \log(U)$, where U is the safe range parameter as defined by nag_real_safe_small_number (X02AMC).

5: **tol** – double *Input*

On entry: the relative accuracy required by the user in the results. If nag_prob_non_central_beta_dist is entered with **tol** greater than or equal to 1.0 or less than $10\times$***machine precision*** (see nag_machine_precision (X02AJC)), then the value of $10\times$***machine precision*** is used instead.

See Section 6.1 for the relationship between **tol** and **max_iter**.

6: **max_iter** – Integer *Input*

On entry: the maximum number of iterations that the algorithm should use.

See Section 6.1 for suggestions as to suitable values for **max_iter** for different values of the parameters.

Suggested value: 500.

Constraint: **max_iter** ≥ 1.

7: **fail** – NagError * *Input/Output*

The NAG error parameter (see the Essential Introduction).

5 Error Indicators and Warnings

NE_REAL_ARG_CONS

On entry, **x** = <*value*>.
This parameter must satisfy $0.0 <$ **x** ≤ 1.0.

On entry, **a** = <*value*>.
This parameter must satisfy $0.0 <$ **a** $\leq$ 1.0e6.

On entry, **b** = <*value*>.
This parameter must satisfy $0.0 <$ **b** $\leq$ 1.0e6.

On entry, **lambda** = <*value*>.
This parameter must satisfy $0.0 \leq$ **lambda** $\leq -2.0 * \log$(X02AMC).

NE_INT_ARG_LT

On entry, **max_iter** must not be less than 1: **max_iter** = <*value*>.

NE_CONV

The solution has failed to converge in <*value*> iterations, consider increasing **max_iter** or **tol**.

NE_PROB_LIMIT

The probability is too close to 0.0 or 1.0 for the algorithm to be able to calculate the required probability. nag_prob_non_central_beta_dist will return 0.0 or 1.0 as appropriate. This should be a reasonable approximation.

NE_PROB_B_INIT

The required accuracy was not achieved when calculating the initial value of the beta distribution. The user should try a larger value of **tol**. The returned value will be an approximation to the correct value.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

6 Further Comments

The central beta probabilities can be obtained by setting **lambda** $= 0.0$.

6.1 Accuracy

Convergence is theoretically guaranteed whenever $P(Y >$**max_iter**$) \leq$ **tol** where Y has a Poisson distribution with mean $\lambda/2$. Excessive round-off errors are possible when the number of iterations used is high and **tol** is close to ***machine precision***. See Lenth (1987) for further comments on the error bound.

6.2 References

Lenth R V (1987) Algorithm AS226: Computing noncentral beta probabilities *Appl. Statist.* **36** 241–244

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

7 See Also

None.

8 Example

Values for several beta distributions are read, and the lower tail probabilities calculated and printed, until the end of data is reached.

8.1 Program Text

```
/* nag_prob_non_central_beta_dist (g01gec) Example Program.
 *
 * Copyright 2000 Numerical Algorithms Group.
 *
 * Mark 6, 2000.
 */


#include <stdio.h>
#include <nag.h>
#include <nagg01.h>


int main(void)
{
  double a, b, prob, lambda, tol, x;
  Integer max_iter;
  Integer exit_status=0;
  NagError fail;

  INIT_FAIL(fail);
  Vprintf("g01gec Example Program Results\n");

/*  Skip heading in data file */
  Vscanf("%*[^\n]");
```

```
  Vprintf("\n     x         a        b        lambda   prob\n\n");
  tol = 5e-6;
  max_iter = 50;
  while ((scanf("%lf %lf %lf %lf %*[^\n]", &x, &a, &b, &lambda)) != EOF)
    {
      prob = g01gec(x, a, b, lambda, tol, max_iter, &fail);
      if (fail.code != NE_NOERROR)
 {
   Vprintf("Error from g01gec.\n%s\n", fail.message);
   exit_status=1;
          goto END;
 }
      Vprintf("%8.3f %8.3f %8.3f %8.3f %8.4f\n", x, a, b, lambda, prob);
    }
 END:
  return exit_status;
}
```

8.2 Program Data

```
g01gec Example Program Data
 0.25  1.0  2.0  1.0         :x a lambda
 0.75  1.5  1.5  0.5         :x a lambda
 0.5   2.0  1.0  0.0         :x a lambda
```

8.3 Program Results

```
g01gec Example Program Results

     x         a        b        lambda   prob

   0.250    1.000    2.000    1.000   0.3168
   0.750    1.500    1.500    0.500   0.7705
   0.500    2.000    1.000    0.000   0.2500
```