nag_deviates_beta (g01fec)

1. Purpose

nag_deviates_beta (g01fec) returns the deviate associated with the given lower tail probability of the beta distribution.

2. Specification

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

```
double nag_deviates_beta(double p, double a, double b, double tol,
        NagError *fail)
```

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

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

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

4. Parameters

р

Input: the probability, p, from the required beta distribution. Constraint: $0.0 \leq \mathbf{p} \leq 1.0.$

a

Input: the first parameter, a, of the required beta distribution. Constraint: $0.0 < \mathbf{a} \le 10^6$.

b

Input: the second parameter, b, of the required beta distribution. Constraint: $0.0 < \mathbf{b} \le 10^6$.

\mathbf{tol}

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

fail

The NAG error parameter, see the Essential Introduction to the NAG C Library.

5. Error Indications and Warnings

On any of the error conditions listed below except **NE_RES_NOT_ACC** and **NE_SOL_NOT_CONV** nag_deviates_beta returns 0.0.

NE_REAL_ARG_LT

On entry, **p** must not be less than 0.0: $\mathbf{p} = \langle value \rangle$.

NE_REAL_ARG_GT

On entry, **p** must not be greater than 1.0: $\mathbf{p} = \langle value \rangle$. On entry, **a** must not be greater than 10⁶: $\mathbf{a} = \langle value \rangle$. On entry, **b** must not be greater than 10⁶: $\mathbf{b} = \langle value \rangle$.

NE_REAL_ARG_LE

On entry, **a** must not be less than or equal to 0.0: $\mathbf{a} = \langle value \rangle$. On entry, **b** must not be less than or equal to 0.0: $\mathbf{b} = \langle value \rangle$.

NE_RES_NOT_ACC

The requested accuracy has not been achieved. Use a larger value of **tol**. 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 6.1). The result should be a reasonable approximation of the solution.

NE_SOL_NOT_CONV

The solution has failed to converge.

However, the result should be a reasonable approximation.

Requested accuracy not achieved when calculating beta probability. The user should try setting **tol** larger.

6. Further Comments

The time taken by the function will depend on the shape of the distribution. For highly skewed distributions with one of the values of a, b large and the other small, series (2) will take longer to converge than for distributions which are more symmetric.

6.1. Accuracy

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

6.2. References

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

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

7. See Also

nag_prob_beta_dist (g01eec)

8. Example

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

```
8.1. Program Text
     /* nag_deviates_beta(g01fec) Example Program
       * Copyright 1990 Numerical Algorithms Group.
       *
       * Mark 2 revised, 1992.
       */
     #include <nag.h>
     #include <stdio.h>
     #include <nag_stdlib.h>
     #include <nagg01.h>
     main()
     {
        double a ,b, p, tol, x;
        static NagError fail;
       /* Skip heading in data file */
Vscanf("%*[^\n]");
printf("g01fec Example Program Results\n");
printf(" Probability A B Dev

        printf(" Probability A B Deviat
while (scanf("%lf %lf %lf", &p, &a, &b) != EOF)
                                                         Deviate\n\n");
          {
             tol = 0.0;
             x = g01fec(p, a, b, tol, &fail);
             if (fail.code==NE_NOERROR)
               Vprintf("%9.4f%10.3f%10.3f%10.4f\n", p, a, b, x);
             else
               Vprintf("%9.4f%10.3f%10.3f%10.4f\n Note: %s\n",p,a,b,x,
                         fail.message);
          }
        exit(EXIT_SUCCESS);
     }
```

8.2. Program Data

g01fec Example Program Data 0.5000 1.0 2.0 0.9900 1.5 1.5 0.2500 20.0 10.0

8.3. Program Results

```
g01fec Example Program Results
Probability
                Α
                         В
                                Deviate
  0.5000
             1.000
                      2.000
                                0.2929
             1.500
  0.9900
                       1.500
                                0.9672
  0.2500
            20.000
                      10.000
                                0.6105
```