nag_random_beta (g05fec)

1. Purpose

nag_random_beta (g05fec) generates a vector of pseudo-random variates from a beta distribution with parameters a and b.

2. Specification

```
#include <nag.h>
#include <nagg05.h>

void nag_random_beta(double a, double b, Integer n, double x[],
            NagError *fail)
```

3. Description

The beta distribution has PDF (probability density function):

$$\begin{array}{rcll} f(x) & = & \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}x^{a-1}(1-x)^{b-1} & \text{if } 0 \leq x \leq 1;\ a,b > 0.0 \\ f(x) & = & 0 & \text{otherwise} \end{array}.$$

One of four algorithms is used to generate the variates depending on the values of a and b. Let α be the maximum and β be the minimum of a and b. Then the algorithms are as follows:

If $\alpha < 0.5$

Jöhnk's algorithm is used, see for example Dagpunar (1988). This generates the beta variate as $u_1^{1/a}/(u_1^{1/a} + u_2^{1/b})$, where u_1 and u_2 are uniformly distributed random variates.

If $\beta > 1$

The algorithm BB given by Cheng (1978) is used. This involves the generation of an observation from a beta distribution of the second kind by the envelope rejection method using a log-logistic target distribution and then transforming it to a beta variate.

If $\alpha > 1$ and $\beta < 1$

The switching algorithm given by Atkinson (1979) is used. The two target distributions used are $f_1(x) = \beta x^{\beta}$ and $f_2(x) = \alpha(1-x)^{\beta-1}$, along with the approximation to the switching parameter of $t = (1-\beta)/(\alpha+1-\beta)$.

In all other cases

Cheng's BC algorithm, see Cheng (1978), is used with modifications suggested by Dagpunar (1988). This algorithm is similar to BB, used when $\beta > 1$, but is tuned for small values of a and b.

4. Parameters

a

Input: the parameter, a, of the beta distribution.
Constraint: **a** > 0.0.

b

Input: the parameter, b, of the beta distribution.
Constraint: **b** > 0.0.

n

Input: the number, n, of pseudo-random numbers to be generated.
Constraint: **n** ≥ 1.

x[n]

Output: the n pseudo-random variates from the specified beta distribution.

fail

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

5. Error Indications and Warnings

NE_REAL_ARG_LE
On entry, **a** must not be less than or equal to 0.0: **a** = ⟨*value*⟩.
On entry, **b** must not be less than or equal to 0.0: **b** = ⟨*value*⟩.

NE_INT_ARG_LE
On entry, **n** must not be less than or equal to 0: **n** = ⟨*value*⟩.

6. Further Comments

To generate an observation, y, from the beta distribution of the second kind from an observation, x, generated by nag_random_beta the transformation, $y = x/(1-x)$, may be used.

To generate an observation, y, from an F-distribution with degrees of freedom v_1 and v_2 generate an observation from the beta distribution with parameters $v_1/2$ and $v_2/2$ and use the transformation $y = v_2 x/v_1(1-x)$.

6.1. Accuracy

Not applicable.

6.2. References

Atkinson A C (1979) A Family of Switching Algorithms for the Computer Generation of Beta Random Variates *Biometrika* **66** 141–5.
Cheng R C H (1978) Generating Beta Variates with Nonintegral Shape Parameters *Comm. ACM* **21** (4) 317–322.
Dagpunar J (1988) *Principles of Random Variate Generation* Oxford University Press.
Hastings N A J and Peacock J B (1975) *Statistical Distributions* Butterworths.

7. See Also

nag_random_init_repeatable (g05cbc)
nag_random_init_nonrepeatable (g05ccc)

8. Example

The example program prints a set of five pseudo-random variates from a beta distribution with parameters $a = 2.0$ and $b = 2.0$, generated by nag_random_beta after initialisation by nag_random_init_repeatable (g05cbc).

8.1. Program Text

```
/* nag_random_beta(g05fec) Example Program
 *
 * Copyright 1991 Numerical Algorithms Group.
 *
 * Mark 2, 1991.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg05.h>

#define N 5

main()
{
  Integer j;
  double a = 2.0;
  double b = 2.0;
  double x[N];

  Vprintf("g05fec Example Program Results\n");
  g05cbc((Integer)0);
```

```
  Vprintf("Beta Dist --- a=%2.1f, b=%2.1f\n",a,b);
  g05fec(a, b, (Integer)N, x, NAGERR_DEFAULT);
  for (j=0; j<(Integer)N; j++)
    Vprintf("%10.4f\n", x[j]);
  exit(EXIT_SUCCESS);
}
```

8.2. Program Data

None.

8.3. Program Results

```
g05fec Example Program Results
Beta Dist --- a=2.0, b=2.0
    0.7229
    0.4079
    0.8023
    0.2555
    0.0946
```