NAG C Library Function Document

nag_rngs_gamma (g05lfc)

1 Purpose

nag_rngs_gamma (g05lfc) generates a vector of pseudo-random numbers taken from a gamma distribution with a and b.

2 Specification

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

3 Description

The beta distribution has PDF (probability density function)

$$f(x) = \frac{1}{b^a \Gamma(a)} x^{a-1} e^{-x/b} \quad \text{if } 0 \leq x; \quad a, b > 0.0$$

$$f(x) = 0 \qquad \text{otherwise.}$$

One of three algorithms is used to generate the variates depending upon the value of a:

(i) If $a < 1$, a switching algorithm described by Dagpunar (1988) (called G6) is used. The target distributions are $f_1(x) = cax^{a-1}/t^a$ and $f_2(x) = (1-c)e^{-(x-t)}$, where $c = t(t + ae^{-t})$, and the switching parameter, t, is taken as $1 - a$. This is similar to Ahrens and Dieter's GS algorithm (see Ahrens and Dieter (1974)) in which $t = 1$;

(ii) If $a = 1$, the gamma distribution reduces to the exponential distribution and the method based on the logarithmic transformation of a uniform random variate is used;

(iii) If $a > 1$, the algorithm given by Best (1978) is used. This is based on using a Student's t-distribution with two degrees of freedom as the target distribution in an envelope rejection method.

One of the initialisation functions nag_rngs_init_repeatable (g05kbc) (for a repeatable sequence if computed sequentially) or nag_rngs_init_nonrepeatable (g05kcc) (for a non-repeatable sequence) must be called prior to the first call to nag_rngs_gamma (g05lfc).

4 References

Ahrens J H and Dieter U (1974) Computer methods for sampling from gamma, beta, Poisson and binomial distributions *Computing* **12** 223–46

Best D J (1978) Letter to the Editor *Appl. Statist.* **29** 181

Dagpunar J (1988) *Principles of Random Variate Generation* Oxford University Press

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

5 Parameters

1: **a** – double *Input*

On entry: the parameter, a, of the gamma distribution.

Constraint: **a** > 0.0.

2: **b** – double *Input*

On entry: the parameter, b, of the gamma distribution.

Constraint: **b** > 0.0.

3: **n** – Integer *Input*

On entry: the number, n, of pseudo-random numbers to be generated.

Constraint: **n** ≥ 0.

4: **x**$[dim]$ – double *Output*

Note: the dimension, dim, of the array **x** must be at least $\max(1, \mathbf{n})$.

On exit: the n pseudo-random numbers from the specified gamma distribution.

5: **igen** – Integer *Input*

On entry: must contain the identification number for the generator to be used to return a pseudo-random number and should remain unchanged following initialisation by a prior call to one of the functions nag_rngs_init_repeatable (g05kbc) or nag_rngs_init_nonrepeatable (g05kcc).

6: **iseed**$[4]$ – Integer *Input/Output*

On entry: contains values which define the current state of the selected generator.

On exit: contains updated values defining the new state of the selected generator.

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

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **n** $= \langle value \rangle$.
Constraint: **n** ≥ 0.

NE_REAL

On entry, **b** $= \langle value \rangle$.
Constraint: **b** > 0.0.

On entry, **a** $= \langle value \rangle$.
Constraint: **a** > 0.0.

NE_BAD_PARAM

On entry, parameter $\langle value \rangle$ had an illegal 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.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

The example program prints a set of five pseudo-random numbers from a gamma distribution with parameters $a = 5.0$ and $b = 1.0$, generated by a single call to nag_rngs_gamma (g05lfc), after initialisation by nag_rngs_init_repeatable (g05kbc).

9.1 Program Text

```
/* nag_rngs_gamma(g05lfc) Example Program.
 *
 * Copyright 2001 Numerical Algorithms Group.
 *
 * Mark 7, 2001.
 */

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

int main(void)
{
  /* Scalars */
  Integer  igen, j, n;
  Integer  exit_status=0;
  NagError fail;

  /* Arrays */
  double   *x=0;
  Integer  iseed[4];

  INIT_FAIL(fail);
  Vprintf("g05lfc Example Program Results\n\n");

  n = 5;
  /* Allocate memory */
  if ( !(x = NAG_ALLOC(n, double)) )
    {
      Vprintf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }

  /* Initialise the seed to a repeatable sequence */
  iseed[0] = 1762543;
  iseed[1] = 9324783;
  iseed[2] = 42344;
  iseed[3] = 742355;
  /* igen identifies the stream. */
  igen = 1;
  g05kbc(&igen, iseed);

  Vprintf("Gamma Dist --- A=5.0, B=1.0\n");

  g05lfc(5.0, 1.0, n, x, igen, iseed, &fail);
  if (fail.code != NE_NOERROR)
    {
      Vprintf("Error from g05lfc.\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }
  for (j = 0; j < n; ++j)
    {
      Vprintf("%10.4f\n",   x[j]);
    }
 END:
  if (x) NAG_FREE(x);
  return exit_status;
}
```

9.2 Program Data

None.

9.3 Program Results

```
g05lfc Example Program Results

Gamma Dist --- A=5.0, B=1.0
    3.2806
    4.6512
    4.0683
    4.6252
    7.6745
```