NAG Library Function Document

nag_rand_exp_mix (g05sgc)

1 Purpose

nag_rand_exp_mix (g05sgc) generates a vector of pseudorandom numbers from an exponential mix distribution composed of m exponential distributions each having a mean a_i and weight w_i.

2 Specification

```
#include <nag.h>
#include <nagg05.h>
void nag_rand_exp_mix (Integer n, Integer nmix, const double a[],
     const double wgt[], Integer state[], double x[], NagError *fail)
```

3 Description

The distribution has PDF (probability density function)

$$f(x) = \sum_{i=1}^{m} \frac{1}{a_i} w_i e^{-x/a_i} \quad \text{if } x \geq 0,$$
$$f(x) = 0 \quad \text{otherwise,}$$

where $\sum_{i=1}^{m} w_i = 1$ and $a_i > 0$, $w_i \geq 0$.

nag_rand_exp_mix (g05sgc) returns the values x_i by selecting, with probability w_j, random variates from an exponential distribution with argument a_j.

One of the initialization functions nag_rand_init_repeatable (g05kfc) (for a repeatable sequence if computed sequentially) or nag_rand_init_nonrepeatable (g05kgc) (for a non-repeatable sequence) must be called prior to the first call to nag_rand_exp_mix (g05sgc).

4 References

Kendall M G and Stuart A (1969) *The Advanced Theory of Statistics (Volume 1)* (3rd Edition) Griffin

Knuth D E (1981) *The Art of Computer Programming (Volume 2)* (2nd Edition) Addison–Wesley

5 Arguments

1: **n** – Integer *Input*

On entry: n, the number of pseudorandom numbers to be generated.

Constraint: **n** ≥ 0.

2: **nmix** – Integer *Input*

On entry: m, the number of exponential distributions in the mix.

Constraint: **nmix** ≥ 1.

3: **a**[**nmix**] – const double *Input*

On entry: the m parameters a_i for the m exponential distributions in the mix.

Constraint: **a**$[i-1] > 0.0$, for $i = 1, 2, \ldots,$ **nmix**.

4: **wgt**[**nmix**] – const double *Input*

On entry: the m weights w_i for the m exponential distributions in the mix.

Constraints:

$$\sum_{i=1}^{m} \mathbf{wgt}[i-1] = 1.0;$$
$$\mathbf{wgt}[i-1] \geq 0.0, \text{ for } i = 1, 2, \ldots, m.$$

5: **state**[dim] – Integer *Communication Array*

Note: the dimension, dim, of this array is dictated by the requirements of associated functions that must have been previously called. This array MUST be the same array passed as argument **state** in the previous call to nag_rand_init_repeatable (g05kfc) or nag_rand_init_nonrepeatable (g05kgc).

On entry: contains information on the selected base generator and its current state.

On exit: contains updated information on the state of the generator.

6: **x**[**n**] – double *Output*

On exit: the n pseudorandom numbers from the specified exponential mix distribution.

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

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_INT

On entry, **n** = ⟨*value*⟩.
Constraint: **n** ≥ 0.

On entry, **nmix** = ⟨*value*⟩.
Constraint: **nmix** ≥ 1.

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 contact NAG for assistance.

NE_INVALID_STATE

On entry, **state** vector has been corrupted or not initialized.

NE_REAL_ARRAY

On entry, **a**[⟨*value*⟩] = ⟨*value*⟩.
Constraint: **a**$[i-1] > 0.0$.

On entry, sum of **wgt** = ⟨*value*⟩.
Constraint: sum of **wgt** = 1.0.

On entry, **wgt**[⟨*value*⟩] = ⟨*value*⟩.
Constraint: **wgt**$[i-1] \geq 0.0$.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_rand_exp_mix (g05sgc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

Please consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

None.

10 Example

This example prints the first five pseudorandom numbers from an exponential mix distribution comprising three exponential distributions with parameters $a_1 = 1.0$, $a_2 = 5.0$ and $a_3 = 2.0$, and with respective weights 0.5, 0.3 and 0.2. The numbers are generated by a single call to nag_rand_exp_mix (g05sgc), after initialization by nag_rand_init_repeatable (g05kfc).

10.1 Program Text

```
/* nag_rand_exp_mix (g05sgc) Example Program.
 *
 * Copyright 2008, Numerical Algorithms Group.
 *
 * Mark 9, 2009.
 */
/* Pre-processor includes */
#include <stdio.h>
#include <math.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg05.h>

int main(void)
{
  /* Integer scalar and array declarations */
  Integer     exit_status = 0;
  Integer     i, lstate;
  Integer     *state = 0;

  /* NAG structures */
  NagError    fail;

  /* Double scalar and array declarations */
  double      *x = 0;

  /* Set the distribution parameters */
  Integer     nmix = 3;
  double      a[] = { 1.0e0, 5.0e0, 2.0e0 };
  double      wgt[] = { 0.50e0, 0.30e0, 0.20e0 };

  /* Set the sample size */
  Integer     n = 5;

  /* Choose the base generator */
  Nag_BaseRNG genid = Nag_Basic;
  Integer     subid = 0;

  /* Set the seed */
  Integer     seed[] = { 1762543 };
  Integer     lseed = 1;
```

```
  /* Initialise the error structure */
  INIT_FAIL(fail);

  printf("nag_rand_exp_mix (g05sgc) Example Program Results\n\n");

  /* Get the length of the state array */
  lstate = -1;
  nag_rand_init_repeatable(genid, subid, seed, lseed, state, &lstate, &fail);
  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_rand_init_repeatable (g05kfc).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }

  /* Allocate arrays */
  if (!(x = NAG_ALLOC(n, double)) ||
      !(state = NAG_ALLOC(lstate, Integer)))
    {
      printf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }

  /* Initialise the generator to a repeatable sequence */
  nag_rand_init_repeatable(genid, subid, seed, lseed, state, &lstate, &fail);
  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_rand_init_repeatable (g05kfc).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }

  /* Generate the variates*/
  nag_rand_exp_mix(n, nmix, a, wgt, state, x, &fail);
  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_rand_exp_mix (g05sgc).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }

  /* Display the variates*/
  for (i = 0; i < n; i++)
    printf("%10.4f\n", x[i]);

 END:
  NAG_FREE(x);
  NAG_FREE(state);

  return exit_status;
}
```

10.2 Program Data

None.

10.3 Program Results

```
nag_rand_exp_mix (g05sgc) Example Program Results

    0.4520
    2.2398
    1.4649
    0.2253
   11.2884
```