NAG C Library Function Document

nag_rngs_gen_discrete (g05mzc)

1 Purpose

nag_rngs_gen_discrete (g05mzc) generates a vector of pseudo-random integers from a discrete distribution with a given PDF (probability density function) or CDF (cumulative distribution function) p.

2 Specification

```
void nag_rngs_gen_discrete (Integer mode, const double p[], Integer np,
      Integer ip1, Nag_ComputeType comp_type, Integer n, Integer x[], Integer igen,
      Integer iseed[], double r[], NagError *fail)
```

3 Description

nag_rngs_gen_discrete (g05mzc) generates a sequence of n integers x_i, from a discrete distribution defined by information supplied in **p**. This may either be the PDF or CDF of the distribution. A reference vector is first set up to contain the CDF of the distribution in its higher elements, followed by an index. The full specifications of the reference vector are as follows.

> **r**[0] = the number of elements of index, k.
>
> **r**[1] = a check number to make sure that the values of **ip1** and **comp_type** haven't changed when calling nag_rngs_gen_discrete (g05mzc) with **mode** $= 1$.
>
> **r**[2] = the number of values the variates can take (i.e., the first value of k such that $\mathrm{CDF}(k) = 1$).
>
> **r**[3] = **ip1** $- 1$.
>
> **r**[4] = the space available for indexing $= 8 + 1.4 \times$ **np** $-$ (**r**[2] $+ 5$).
>
> **r**[$i + 5$], for $i = 1, 2, \ldots,$ **r**[2], the CDF.
>
> **r**[i] $= \min\{j | \mathrm{CDF}(j) > (i-1)/k\}$, for $i =$ **r**[2] $+ 6, \ldots, 8 + 1.4 \times$ **np**.

Setting up the reference vector and subsequent generation of variates can each be performed by separate calls to nag_rngs_gen_discrete (g05mzc) or may be combined in a single call.

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_gen_discrete (g05mzc).

4 References

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

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

5 Parameters

1: **mode** – Integer *Input*

On entry: a code for selecting the operation to be performed by the function:

mode $= 0$

Set up reference vector only.

mode $= 1$

Generate variates using reference vector set up in a prior call to nag_rngs_gen_discrete (g05mzc).

mode $= 2$

Set up reference vector and generate variates.

Constraint: $0 \leq$ **mode** ≤ 2.

2: **p**[**np**] – const double *Input*

On entry: the PDF or CDF of the distribution.

3: **np** – Integer *Input*

On entry: The number of values supplied in **p** defining the PDF or CDF of the discrete distribution.

Constraint: **np** > 0.

4: **ip1** – Integer *Input*

On entry: the value of the variate, assumed to be a whole number, to which the probability in **p**[0] corresponds.

5: **comp_type** – Nag_ComputeType *Input*

On entry: **comp_type** indicates the type of information contained in **p**.

If **comp_type** = **Nag_Compute_1**, **p** contains a probability distribution function (PDF).

If **comp_type** = **Nag_Compute_2**, **p** contains a cumulative distribution function (CDF)

Constraint: **comp_type** = **Nag_Compute_1** or **Nag_Compute_2**.

6: **n** – Integer *Input*

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

Constraint: **n** ≥ 1.

7: **x**[**n**] – Integer *Output*

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

8: **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).

9: **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.

10: **r**[dim] – double *Input/Output*

Note: the dimension, dim, of the array **r** must be at least $8 + 1.4 \times$ **np**.

On exit: the reference vector.

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

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, **np** = ⟨*value*⟩.
Constraint: **np** > 0.

On entry, **mode** = ⟨*value*⟩.
Constraint: $0 \leq$ **mode** ≤ 2.

On entry, **np** = ⟨*value*⟩.
Constraint: **np** > 0.

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

NE_NOT_INCREASING

On entry, **comp_type** $= 2$ and the values in **p** are not all in non-descending order.

NE_PREV_CALL

comp_type is not the same as when **r** was set up in a previous call. Previous value of **comp_type** = ⟨*value*⟩, **comp_type** = ⟨*value*⟩.

np or **ip1** is not the same as when **r** was set up in a previous call. Previous value of **np** = ⟨*value*⟩, **np** = ⟨*value*⟩. Previous value of **ip1** = ⟨*value*⟩, **ip1** = ⟨*value*⟩.

NE_REAL_ARRAY_ELEM_CONS

On entry, at least one element of the vector **p** is negative. The first found is at position i = ⟨*value*⟩, **p**$[i-1]$ = ⟨*value*⟩.

NE_REAL_ARRAY_SUM

On entry, the sum of the elements of **p** is not one. The difference from unity in the summation is: ⟨*value*⟩.

NE_BAD_PARAM

On entry, parameter ⟨*value*⟩ 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 20 pseudo-random variates from a discrete distribution whose PDF, $f(n)$, is defined as follows:

n	$f(n)$
-5	0.01
-4	0.02
-3	0.04
-2	0.08
-1	0.20
0	0.30
1	0.20
2	0.08
3	0.04
4	0.02
5	0.01

The reference vector is set up and and the variates are generated by a single call to nag_rngs_gen_discrete (g05mzc), after initialisation by nag_rngs_init_repeatable (g05kbc).

9.1 Program Text

```
/* nag_rngs_gen_discrete(g05mzc) 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)
{
  /* Initialized data */

  static double p[] = { 0.01,0.02,0.04,0.08,0.2,0.3,0.2,0.08,0.04,0.02,0.01 };

  /* Scalars */
  Integer  i, igen, ip1, np, n, nr;
  Integer  exit_status=0;
  NagError fail;
  Nag_ComputeType itype;

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

  INIT_FAIL(fail);
  Vprintf("g05mzc Example Program Results\n\n");
  nr = 60;
  n = 20;
  np = 11;

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

  /* Set the distribution parameters P and M */
  ip1 = -5;
  itype = Nag_Compute_1;
  /* 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);

  /* Choose MODE = 2 */
  g05mzc(2, p, np, ip1, itype, n, x, igen, iseed, r, &fail);
  if (fail.code != NE_NOERROR)
    {
      Vprintf("Error from g05mzc.\n%s\n", fail.message);
      exit_status = 1;
      goto END;
    }
  for (i = 0; i < n; ++i)
    {
      Vprintf("%12ld\n", x[i]);
    }
 END:
  if (r) NAG_FREE(r);
  if (x) NAG_FREE(x);
  return exit_status;
}
```

9.2 Program Data

None.

9.3 Program Results

```
g05mzc Example Program Results

          -2
           3
           0
           1
           3
           0
           0
           0
          -1
           2
           0
           0
           0
           5
          -3
           0
           1
           0
          -3
           1
```