

## NAG C Library Function Document

### nag\_rngs\_discrete\_uniform (g05mac)

#### 1 Purpose

nag\_rngs\_discrete\_uniform (g05mac) generates a vector of pseudo-random integers uniformly distributed over the interval  $[a, b]$ .

#### 2 Specification

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

#### 3 Description

nag\_rngs\_discrete\_uniform (g05mac) generates the next  $n$  values  $y_i$  from a uniform (0,1) generator (see nag\_rngs\_basic (g05kac) for details) and applies the transformation

$$x_i = a + [(b - a + 1)y_i],$$

where  $[z]$  is the integer part of the real value  $z$ . The function ensures that the values  $x_i$  lie in the closed interval  $[a, b]$ .

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\_discrete\_uniform (g05mac).

#### 4 References

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

#### 5 Parameters

- |    |   |                     |
|----|---|---------------------|
| 1: | <b>a</b> – Integer  | <i>Input</i>        |
| 2: | <b>b</b> – Integer  | <i>Input</i>        |
|    | <i>On entry:</i> the end-points $a$ and $b$ of the uniform distribution.  |                     |
|    | <i>Constraint:</i> $\mathbf{a} \leq \mathbf{b}$ .   |                     |
| 3: | <b>n</b> – Integer  | <i>Input</i>        |
|    | <i>On entry:</i> the number, $n$ , of pseudo-random numbers to be generated.  |                     |
|    | <i>Constraint:</i> $\mathbf{n} \geq 0$ .  |                     |
| 4: | <b>x</b> [ <i>dim</i> ] – Integer   | <i>Output</i>       |
|    | <b>Note:</b> the dimension, <i>dim</i> , of the array <b>x</b> must be at least $\max(1, \mathbf{n})$ .   |                     |
|    | <i>On exit:</i> the $n$ pseudo-random numbers from the specified uniform distribution.  |                     |
| 5: | <b>igen</b> – Integer   | <i>Input</i>        |
|    | <i>On entry:</i> 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: | <b>iseed</b> [4] – Integer  | <i>Input/Output</i> |
|    | <i>On entry:</i> 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**  $\geq$  0.

### NE\_INT\_2

On entry, **a** =  $\langle value \rangle$  and **b** =  $\langle value \rangle$ .  
Constraint: **b**  $\geq$  **a**.

### 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 five pseudo-random integers from a discrete uniform distribution between  $-5$  and  $5$ , generated by a single call to `nag_rngs_discrete_uniform` (g05mac), after initialisation by `nag_rngs_init_repeatable` (g05kbc).

### 9.1 Program Text

```
/* nag_rngs_discrete_uniform(g05mac) 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, m;
    Integer exit_status=0;
    NagError fail;

    /* Arrays */
```

```
Integer *x=0;
Integer  iseed[4];

INIT_FAIL(fail);
Vprintf("g05mac Example Program Results\n\n");

m = 5;
/* Allocate memory */
if ( !(x = NAG_ALLOC(m, Integer)) )
{
    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);

g05mac(-5, 5, m, x, igen, iseed, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from g05mac.\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}
for (j = 0; j < m; ++j)
{
    Vprintf("%12ld\n", x[j]);
}
END:
if (x) NAG_FREE(x);
return exit_status;
}
```

## 9.2 Program Data

None.

## 9.3 Program Results

g05mac Example Program Results

```
-5
 5
-1
 3
 5
```

---