NAG C Library Function Document

nag_rngs_basic (g05kac)

1 Purpose

nag_rngs_basic (g05kac) returns a pseudo-random number taken from a uniform distribution between 0 and 1.

2 Specification

```
double nag_rngs_basic (Integer igen, Integer iseed[])
```

3 Description

nag_rngs_basic (g05kac) returns the next pseudo-random number from a uniform (0,1) generator.

The particular generator used to generate random numbers is selected by the value set for the input parameter **igen**. Consult the g05 Chapter Introduction for details of the algorithms that can be used.

The current state of the chosen generator is saved in the integer array **iseed** which should not be altered between successive calls. Initial states are set or re-initialised by a call to nag_rngs_init_repeatable (g05kbc) (for a repeatable sequence if computed sequentially) or nag_rngs_init_nonrepeatable (g05kcc) (for a non-repeatable sequence).

nag_rngs_uniform (g05lgc) may be used to generate a vector of n pseudo-random numbers which, if computed sequentially using the same generator, are exactly the same as n successive values of this function. On many machines nag_rngs_uniform (g05lgc) is likely to be much faster.

4 References

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

5 Parameters

1: **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).

2: **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.

6 Error Indicators and Warnings

None.

7 Accuracy

Not applicable.

8 Further Comments

The generator with the smallest period that can be selected is the basic generator. The period of the basic

generator is 2^{57}.

Its performance has been analysed by the Spectral Test, see Section 3.3.4 of Knuth (1981), yielding the following results in the notation of Knuth (1981).

n	ν_n	Upper bound for ν_n
2	3.44×10^8	4.08×10^8
3	4.29×10^5	5.88×10^5
4	1.72×10^4	2.32×10^4
5	1.92×10^3	3.33×10^3
6	593	939
7	198	380
8	108	197
9	67	120

The right-hand column gives an upper bound for the values of ν_n attainable by any multiplicative congruential generator working modulo 2^{59}.

An informal interpretation of the quantities ν_n is that consecutive n-tuples are statistically uncorrelated to an accuracy of $1/\nu_n$. This is a theoretical result; in practice the degree of randomness is usually much greater than the above figures might support. More details are given in Knuth (1981), and in the references cited therein.

Note that the achievable accuracy drops rapidly as the number of dimensions increases. This is a property of all multiplicative congruential generators and is the reason why very long periods are needed even for samples of only a few random numbers.

9 Example

The example program prints the first five pseudo-random numbers from a uniform distribution between 0 and 1, generated by nag_rngs_basic (g05kac) after initialisation by nag_rngs_init_repeatable (g05kbc).

9.1 Program Text

```
/* nag_rngs_basic(g05kac) 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 */
  double x;
  Integer i, igen;
  Integer exit_status=0;

  /* Arrays */
  Integer iseed[4];

  Vprintf("g05kac Example Program Results\n\n");

  /* Initialise the seed */
  iseed[0] = 1762543;
  iseed[1] = 9324783;
  iseed[2] = 42344;
  iseed[3] = 742355;
  /* igen identifies the stream. */
  igen = 1;
```

```
  g05kbc(&igen, iseed);
  for (i = 1; i <= 5; ++i)
    {
      x = g05kac(igen, iseed);
      Vprintf("%10.4f\n",   x);
    }
  return exit_status;
}
```

9.2 Program Data

None.

9.3 Program Results

```
g05kac Example Program Results

    0.0893
    0.9510
    0.4064
    0.7432
    0.9498
```