

NAG Library Function Document

nag_rand_basic (g05sac)

1 Purpose

nag_rand_basic (g05sac) generates a vector of pseudorandom numbers taken from a uniform distribution between 0 and 1.

2 Specification

```
#include <nag.h>
#include <nagg05.h>
void nag_rand_basic (Integer n, Integer state[], double x[], NagError *fail)
```

3 Description

nag_rand_basic (g05sac) generates n values from a uniform distribution over the half closed interval $(0, 1]$.

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_basic (g05sac).

4 References

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: $n \geq 0$.
- 2: **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.
- 3: **x**[**n**] – double *Output*
On exit: the n pseudorandom numbers from a uniform distribution over the half closed interval $(0, 1]$.
- 4: **fail** – NagError * *Input/Output*
The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_BAD_PARAM

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

NE_INT

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

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.

7 Accuracy

Not applicable.

8 Parallelism and Performance

Not applicable.

9 Further Comments

None.

10 Example

This example prints the first five pseudorandom numbers from a uniform distribution between 0 and 1, generated by `nag_rand_basic` (g05sac) after initialization by `nag_rand_init_repeatable` (g05kfc).

10.1 Program Text

```

/* nag_rand_basic (g05sac) 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 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_basic (g05sac) Example Program Results\n\n");

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

/* Generate the variates */
nag_rand_basic(n, state, x, &fail);
if (fail.code != NE_NOERROR)
{
    printf("Error from nag_rand_basic (g05sac).\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_basic (g05sac) Example Program Results

0.6364
0.1065
0.7460
0.7983
0.1046
