NAG C Library Function Document

nag_rngs_f (g05ldc)

1 Purpose
nag_rngs_f (g05ldc) generates a vector of pseudo-random numbers taken from a $F$ (or Fisher’s variance ratio) distribution with $\mu$ and $\nu$ degrees of freedom.

2 Specification

```c
void nag_rngs_f (Integer df1, Integer df2, Integer n, double x[], Integer igen, Integer iseed[], NagError *fail)
```

3 Description

The distribution has PDF (probability density function)

$$f(x) = \frac{\left(\frac{\mu+\nu-2}{2}\right)! x^{\mu-1}}{\left(\frac{\mu}{2}\right)! \left(\frac{\nu-1}{2}\right)! (1 + \frac{x}{\mu})^{\frac{\mu+\nu}{2}}} \quad \text{if} \quad x > 0,$$

$$f(x) = 0 \quad \text{otherwise.}$$

nag_rngs_f (g05ldc) calculates the values

$$\frac{\nu y_i}{\mu z_i}, \quad i = 1, \ldots, n,$$

where $y_i$ and $z_i$ are generated by nag_rngs_gamma (g05lfc) from gamma distributions with parameters $\left(\frac{\mu}{2}, 2\right)$ and $\left(\frac{\nu}{2}, 2\right)$ respectively (i.e., from $\chi^2$ distributions with $\mu$ and $\nu$ degrees of freedom).

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_f (g05ldc).

4 References


5 Parameters

1: df1 – Integer

On entry: the number of degrees of freedom, $\mu$, of the distribution.
Constraint: $df1 \geq 1$.

2: df2 – Integer

On entry: the number of degrees of freedom, $\nu$, of the distribution.
Constraint: $df2 \geq 1$.

3: n – Integer

On entry: the number, $n$, of pseudo-random numbers to be generated.
Constraint: $n \geq 0$.

4: x[dim] – double

Note: the dimension, dim, of the array x must be at least max(1, n).

[g05ldc.1]
On exit: the \( n \) pseudo-random numbers from the specified \( F \) distribution.

5: \( \text{igen} \) – Integer \hspace{1cm} \text{Input}

\( \text{On entry:} \) 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 \( \text{g05kbc} \) or \( \text{g05kcc} \).

6: \( \text{iseed}[4] \) – Integer \hspace{1cm} \text{Input/Output}

\( \text{On entry:} \) contains values which define the current state of the selected generator.
\( \text{On exit:} \) contains updated values defining the new state of the selected generator.

7: \( \text{fail} \) – \text{NagError} * \hspace{1cm} \text{Input/Output}

The NAG error parameter (see the Essential Introduction).

6 \hspace{1cm} \text{Error Indicators and Warnings}

\textbf{NE_INT}

On entry, \( n = \langle \text{value} \rangle \).
Constraint: \( n \geq 0 \).

On entry, \( \text{df2} = \langle \text{value} \rangle \).
Constraint: \( \text{df2} \geq 1 \).

On entry, \( \text{df1} = \langle \text{value} \rangle \).
Constraint: \( \text{df1} \geq 1 \).

\textbf{NE_BAD_PARAM}

On entry, parameter \( \langle \text{value} \rangle \) had an illegal value.

\textbf{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 \hspace{1cm} \text{Accuracy}

Not applicable.

8 \hspace{1cm} \text{Further Comments}

The time taken by \( \text{g05l} \) increases with \( \mu \) and \( \nu \).

9 \hspace{1cm} \text{Example}

The example program prints five pseudo-random numbers from a \( F \)-distribution with two and three degrees of freedom, generated by a single call to \( \text{g05l} \), after initialisation by \( \text{g05kbc} \).

9.1 \hspace{1cm} \text{Program Text}

/* \text{g05l} Example Program. */
* Copyright 2001 Numerical Algorithms Group.
*/
```c
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg05.h>

int main(void)
{
    /* Scalars */
    Integer i, igen, n;
    Integer exit_status=0;
    NagError fail;
    /* Arrays */
    double *x=0;
    Integer iseed[4];
    INIT_FAIL(fail);
    Vprintf("g05ldc Example Program Results\n\n");
    n=5;
    /* Allocate memory */
    if ( !(x = NAG_ALLOC(n, double)) )
    {
        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);
    g05ldc(2, 3, n, x, igen, iseed, &fail);
    if (fail.code != NE_NOERROR)
    {
        Vprintf("Error from g05ldc.\n", fail.message);
        exit_status = 1;
        goto END;
    }
    for (i = 0; i < n; ++i)
    {
        Vprintf("%10.4f\n", x[i]);
    }
    END:
    if (x) NAG_FREE(x);
    return exit_status;
}

9.2 Program Data
None.

9.3 Program Results

<table>
<thead>
<tr>
<th>g05ldc Example Program Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>14.2359</td>
</tr>
<tr>
<td>0.8889</td>
</tr>
<tr>
<td>0.4055</td>
</tr>
<tr>
<td>2.3299</td>
</tr>
<tr>
<td>0.0689</td>
</tr>
</tbody>
</table>
```

905 – Random Number Generators

---

NP3645/7  g05ldc.3 (last)