NAG C Library Function Document

nag_rngs_binomial (g05mjc)

1 Purpose

nag_rngs_binomial (g05mjc) generates a vector of pseudo-random integers from the discrete binomial distribution with parameters \( m \) and \( p \).

2 Specification

```c
void nag_rngs_binomial (Integer mode, Integer m, double p, Integer n, Integer x[],
                        Integer igen, Integer iseed[], double r[], NagError *fail)
```

3 Description

nag_rngs_binomial (g05mjc) generates \( n \) integers \( x_i \) from a discrete binomial distribution, where the probability of \( x_i = I \) is

\[
P(x_i = I) = \frac{m!}{I!(m-I)!} p^I (1-p)^{m-I}, \quad I = 0, 1, \ldots, m,
\]

where \( 0 \leq m \) and \( 0 \leq p \leq 1 \). This represents the probability of achieving \( I \) successes in \( m \) trials when the probability of success at a single trial is \( p \).

The variates can be generated with or without using a search table and index. If a search table is used then it is stored with the index in a reference vector and subsequent calls to nag_rngs_binomial (g05mjc) with the same parameter values can then use this reference vector to generate further variates.

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_binomial (g05mjc).

4 References


5 Parameters

1: \( \text{mode} \) – Integer

\textit{Input}

\textit{On entry:} a code for selecting the operation to be performed by the function:

- \( \text{mode} = 0 \) 
  Set up reference vector only.
- \( \text{mode} = 1 \) 
  Generate variates using reference vector set up in a prior call to nag_rngs_binomial (g05mjc).
- \( \text{mode} = 2 \) 
  Set up reference vector and generate variates.
- \( \text{mode} = 3 \) 
  Generate variates without using the reference vector.

\textit{Constraint:} \( 0 \leq \text{mode} \leq 3 \).
2: \( m \) – Integer
   \textit{Input}
   \textit{On entry}: the number of trials, \( m \), of the distribution.
   \textit{Constraint}: \( m \geq 0 \).

3: \( p \) – double
   \textit{Input}
   \textit{On entry}: the probability of success \( p \) of the binomial distribution.
   \textit{Constraint}: \( 0.0 \leq p \leq 1.0 \).

4: \( n \) – Integer
   \textit{Input}
   \textit{On entry}: the number, \( n \), of pseudo-random numbers to be generated.
   \textit{Constraint}: \( n \geq 1 \).

5: \( x[n] \) – Integer
   \textit{Output}
   \textit{On exit}: the \( n \) pseudo-random numbers from the specified binomial distribution.

6: \( igen \) – Integer
   \textit{Input}
   \textit{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 \texttt{nag_rngs_init_repeatable (g05kbc)} or \texttt{nag_rngs_init_nonrepeatable (g05kcc)}.

7: \( iseed[4] \) – Integer
   \textit{Input/Output}
   \textit{On entry}: contains values which define the current state of the selected generator.
   \textit{On exit}: contains updated values defining the new state of the selected generator.

8: \( r[dim] \) – double
   \textit{Input/Output}
   \text{Note}: the dimension, \( dim \), of the array \( r \) must be at least \( 22 + 20 \sqrt{m \times p(1 - p)} \) when \( \text{mode} < 3 \)
   and at least 1 otherwise.
   \textit{On exit}: the reference vector.

9: \( fail \) – \texttt{NagError *}
   \textit{Input/Output}
   The NAG error parameter (see the Essential Introduction).

6 \hspace{1em} \textbf{Error Indicators and Warnings}

\textbf{NE_INT}
   \textit{On entry}, \( \text{mode} = \langle \text{value} \rangle \).
   \textit{Constraint}: \( 0 \leq \text{mode} \leq 3 \).
   \textit{On entry}, \( m = \langle \text{value} \rangle \).
   \textit{Constraint}: \( m \geq 0 \).
   \textit{On entry}, \( n = \langle \text{value} \rangle \).
   \textit{Constraint}: \( n \geq 1 \).

\textbf{NE_PREV_CALL}
   \( p \) or \( m \) is not the same as when \( r \) was set up in a previous call. Previous value of \( p = \langle \text{value} \rangle \),
   \( p = \langle \text{value} \rangle \). Previous value of \( m = \langle \text{value} \rangle \), \( p = \langle \text{value} \rangle \).

\textbf{NE_REAL}
   \textit{On entry}, \( p < 0.0 \) or \( p > 1.0 \): \( p = \langle \text{value} \rangle \).
NE_BAD_PARAM
On entry, parameter \(i\) 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 integers from a binomial distribution with parameters \(m = 6000\) and \(p = 0.8\), generated by a single call to nag_rngs_binomial (g05mjc), after initialisation by nag_rngs_init_repeatable (g05kbc).

9.1 Program Text
/* nag_rngs_binomial(g05mjc) Example Program. *
 * Copyright 2001 Numerical Algorithms Group.
 */
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg05.h>

int main(void)
{
    /* Scalars */
    double p;
    Integer i, igen, m, n, nr;
    Integer exit_status=0;
    NagError fail;
    /* Arrays */
    double *r=0;
    Integer *x=0;
    Integer iseed[4];

    INIT_FAIL(fail);
    Vprintf("g05mjc Example Program Results\n\n");
    n = 20;
    nr = 6007;

    /* 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 \(M\) */
    p = 0.8;

...
m = 6000;
/* 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 */
g05mjc(2, m, p, n, x, igen, iseed, r, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from g05mjc.\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

g05mjc Example Program Results

4758
4851
4793
4820
4851
4795
4807
4792
4787
4842
4801
4794
4806
4878
4745
4790
4832
4789
4743
4812