G05BEFP

NAG Parallel Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check for implementation-dependent details. You are advised to enclose any calls to NAG Parallel Library routines between calls to Z01AAFP and Z01ABFP.

1 Description

G05BEFP generates a vector of pseudo-random numbers of length n from the (negative) exponential distribution with mean a which has the probability density function (PDF)

$$f(x) = \frac{1}{a}e^{-x/a} \quad \text{if} \quad x > 0, \quad a \geq 0,$$

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

A total of 273 statistically independent generators are available; it is possible to select a particular generator and initialize the seeds for the generator by a preceding call to G05BBFP. If G05BBFP is not used, default values for the generator and the seeds are assumed.

The routine G05BEFP always generates exactly the same pseudo-random numbers as would n consecutive calls of G05AEFP.

2 Specification

```
SUBROUTINE G05BEFP(A, N, X)
INTEGER            N
DOUBLE PRECISION   A, X(*)
```

3 Usage

3.1 Definitions

None.

3.2 Global and Local Arguments

All arguments are local.

4 Arguments

1: A — DOUBLE PRECISION *Local Input*

On entry: the mean a of the distribution. If A is negative the absolute value of A is used.

2: N — INTEGER *Local Input/Local Output*

On entry: n, the number of pseudo-random numbers to be generated. If N < 1, no pseudo-random numbers are generated.

On exit: the actual number of pseudo-random numbers which were generated.

3: X(*) — DOUBLE PRECISION array *Local Output*

On exit: the n pseudo-random numbers from the specified (negative) exponential distribution.

5 Errors and Warnings

None.

6 Further Comments

Repeatable sequences of random numbers can be generated by calling G05BBFP to set the seeds and generator number before calling G05BEFP.

G05BEFP may be called without a prior call to Z01AAFP.

6.1 Algorithmic Detail

Each basic generator uses a Wichmann–Hill type generator (Wichmann and Hill [3]), which is a variant of a multiplicative congruential algorithm to produce real pseudo-random numbers u_k in the semi-open interval $[0, 1)$. See G05BCFP for further details. The pseudo-random value v_i from the exponential distribution is then given by $v_i = -a \ln u_i$.

7 References

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

[2] Maclaren N M (1989) The generation of multiple independent sequences of pseudorandom numbers *Appl. Statist.* **38** 351–359

[3] Wichmann B A and Hill I D (1982) AS183 An efficient and portable pseudo-random number generator *Appl. Statist.* **31** 188–190

8 Example

This example generates a series of random numbers on each processor on a 2 by 2 logical grid of processors. The routine G05BBFP is used to initialise the seeds and the generators.

8.1 Example Text

```
*     G05BEFP Example Program Text
*     NAG Parallel Library Release 3. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NOUT, NX
      PARAMETER        (NOUT=6,NX=10)
      INTEGER          MAG
      PARAMETER        (MAG=16909320)
*     .. Local Scalars ..
      DOUBLE PRECISION A
      INTEGER          I, ICNTXT, ICOFF, IFAIL, IGEN, MP, MYCOL,
     +                 MYROW, N, NP, NPCOL, NPROW
      LOGICAL          ROOT
      CHARACTER        CNUMOP, TITOP
      CHARACTER*20     FORMT
*     .. Local Arrays ..
      DOUBLE PRECISION WORK(NX), X(NX)
      INTEGER          IS(5), ISEED(4), IWORK(5)
*     .. External Functions ..
      LOGICAL          Z01ACFP
      EXTERNAL         Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         G05BBFP, G05BEFP, X04BFFP, X04BMFP, Z01AAFP,
     +                 Z01ABFP, Z01ZAFP
```

```
*     .. Intrinsic Functions ..
      INTRINSIC        MOD
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) THEN
         WRITE (NOUT,*) 'G05BEFP Example Program Results'
         WRITE (NOUT,*)
      END IF
*
      MP = 2
      NP = 2
*
*     Declare the processor grid
*
      IFAIL = 0
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
*     Initialise the seeds and the generator
      CALL Z01ZAFP(ICNTXT,NPROW,NPCOL,MYROW,MYCOL)
*
*     Initialize the seeds and choose a generator number that depends
*     on the processor position on the grid.
*
      ISEED(1) = 207*(50*MYROW+19*MYCOL) + 2626279
      ISEED(2) = 451*(70*MYROW+31*MYCOL) + 2727289
      ISEED(3) = 912*(39*MYROW+53*MYCOL) + 9598590
      ISEED(4) = 812*(69*MYROW+14*MYCOL) + 4684748
      IGEN = NP*MYROW*6 + MYCOL*MP*5
*
*     Make sure that the seeds are within the maximum value MAG
*
      DO 40 I = 1, 4
   20    IF (ISEED(I).GT.MAG) THEN
            ISEED(I) = ISEED(I)/2
            GO TO 20
         END IF
   40 CONTINUE
*
*     Make sure that the generator is valid
*
      IGEN = MOD(IGEN,273)
*
*     Print the seeds and the generator on each processor
*

      IS(1) = ISEED(1)
      IS(2) = ISEED(2)
      IS(3) = ISEED(3)
      IS(4) = ISEED(4)
      IS(5) = IGEN
      IF (ROOT) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Seeds and the generator'
         WRITE (NOUT,*)
      END IF
      FORMT = 'I10'
      TITOP = 'Y'
      CNUMOP = 'X'
```

```
      ICOFF = 0
      IFAIL = 0
      CALL X04BMFP(ICNTXT,NOUT,1,5,IS,1,FORMT,TITOP,CNUMOP,ICOFF,IWORK,
     +             1,IFAIL)
*
      CALL G05BBFP(ISEED,IGEN)
*
*     Set the parameter A
*
      A = 1.0D0
*
*     Set the number of random numbers on each processor
*
      N = 5
*
*     Now fill the vectors with random numbers
*
      CALL G05BEFP(A,N,X)
*
*     Print the N random numbers on each processor
*
*
      IF (ROOT) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,*)
     +     'The generated random numbers on each processor'
         WRITE (NOUT,*)
      END IF
      FORMT = 'F12.5'
      TITOP = 'Y'
      CNUMOP = 'X'
      ICOFF = 0
      IFAIL = 0
      CALL X04BFFP(ICNTXT,NOUT,1,N,X,1,FORMT,TITOP,CNUMOP,ICOFF,WORK,1,
     +             IFAIL)

      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
*
      END
```

8.2 Example Data

None.

8.3 Example Results

```
G05BEFP Example Program Results

Seeds and the generator
  Array from logical processor    0,   0
    2626279   2727289   9598590   4684748         0
  Array from logical processor    0,   1
    2630212   2741270   9646926   4696116        10
  Array from logical processor    1,   0
    2636629   2758859   9634158   4740776        12
  Array from logical processor    1,   1
    2640562   2772840   9682494   4752144        22

The generated random numbers on each processor
  Array from logical processor    0,   0
      0.71974     1.52380     0.43078     0.57130     2.06355
  Array from logical processor    0,   1
      1.18314     0.58246     1.37639     2.06612     2.29289
  Array from logical processor    1,   0
      0.56857     0.37826     0.58601     0.76624     0.05285
  Array from logical processor    1,   1
      0.61310     1.31984     1.70097     0.99411     0.35373
```