G05ABFP

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

G05ABFP selects a Wichmann–Hill random number generator from the 273 available generators and initializes the seeds for the generator on a logical grid of processors.

2 Specification

```
SUBROUTINE G05ABFP(ICNTXT, ISEED, IGEN, IFAIL)
INTEGER            ICNTXT, ISEED(4), IGEN, IFAIL
```

3 Data Distribution

3.1 Definitions

Not applicable.

3.2 Global and Local Arguments

The input argument IFAIL is global, so must have the same value on each processor. The remaining arguments are local to each processor. The output argument IFAIL is global and so will have the same value on exit from the routine on each processor.

4 Arguments

1: ICNTXT — INTEGER *Local Input*

On entry: the BLACS context used by the communication mechanism, usually returned by a call to Z01AAFP.

2: ISEED(4) — INTEGER array *Local Input*

On entry: the seeds for the random number generator.

Constraint: ISEED$(i) > 0, \quad i = 1, 2, 3, 4$.

Suggested values: the value of each element of ISEED should be a large integer (i.e., ISEED$(i) \gtrsim 10^6, \quad i = 1, 2, 3, 4$).

3: IGEN — INTEGER *Local Input*

On entry: the generator number.

Constraint: $273 \geq$ IGEN ≥ 1.

4: IFAIL — INTEGER *Global Input/Global Output*

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in the Essential Introduction) the recommended values are:

IFAIL $= 0$, if multigridding is **not** employed;
IFAIL $= -1$, if multigridding is employed.

On exit: IFAIL $= 0$ unless the routine detects an error (see Section 5).

5 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output from the root processor (or processor {0,0} when the root processor is not available) on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = −2000

The routine has been called with an invalid value of ICNTXT on one or more processors.

IFAIL = −1000

The logical processor grid and library mechanism (Library Grid) have not been correctly defined, see Z01AAFP.

IFAIL = −1

One or more elements of the seed vector, ISEED, was less than 1.

IFAIL = −2

The generator number IGEN was less than 1 or greater than 273.

6 Further Comments

The values of ISEED are saved internally by the routine. To generate different sequences of random numbers on different processors, ensure that ISEED and/or IGEN are initialized to different values on each processor before starting the sequence.

If small seeds are supplied to two different generators, the first few (typically 3 or 4) random numbers produced by each generator will be similar. If it is important to avoid this then large (i.e., 6 or 7 digit) seeds should be supplied.

7 References

[1] Dongarra J J and Whaley R C (1995) A users' guide to the BLACS v1.0. *LAPACK Working Note 94 (Technical Report CS-95-281)* Department of Computer Science, University of Tennessee, 107 Ayres Hall, Knoxville, TN 37996-1301, USA.
URL: http://www.netlib.org/lapack/lawns/lawn94.ps

8 Example

This example generates a series of random numbers on each processor in a 2 by 2 logical grid of processors. The seeds and generator number are different on each processor so that statistically independent sequences are produced on each processor. The seeds are then restored to their original values and a second sequence of random numbers is produced on each processor. The second sequence is identical to the first on each processor, demonstrating how G05ABFP may be used to produce repeatable sequences.

8.1 Example Text

```
*     G05ABFP Example Program Text
*     NAG Parallel Library Release 2. NAG Copyright 1996.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          M
      PARAMETER        (M=5)
*     .. Local Scalars ..
      INTEGER          GEN, I, ICNTXT, IFAIL, IFLAG, J, K, MP, MYCOL,
     +                 MYROW, NP, NPCOL, NPROW, NRC, NRR
      LOGICAL          ROOT
```

```
*     .. Local Arrays ..
      DOUBLE PRECISION RVEC1(M), RVEC2(M)
      INTEGER          ISAV(4), ISEED(4)
*     .. External Functions ..
      DOUBLE PRECISION G05AAFP
      LOGICAL          Z01ACFP
      EXTERNAL         G05AAFP, Z01ACFP
*     .. External Subroutines ..
      EXTERNAL         BLACS_GRIDINFO, DGERV2D, DGESD2D, G05ABFP,
     +                 Z01AAFP, Z01ABFP, Z01BAFP
*     .. Executable Statements ..
      ROOT = Z01ACFP()
      IF (ROOT) WRITE (NOUT,*) 'G05ABFP Example Program Results'
*
      MP = 2
      NP = 2
*
*     Declare the processor grid
*
      IFAIL = 0
      CALL Z01AAFP(ICNTXT,MP,NP,IFAIL)
*
      CALL BLACS_GRIDINFO(ICNTXT,NPROW,NPCOL,MYROW,MYCOL)
*
*     Initialize the seeds and choose a generator number that depends
*     on the processor position in the grid.
*
      ISEED(1) = 1207*(5000*MYROW+1579)
      ISEED(2) = 8044*(7065*MYROW+3070)
      ISEED(3) = 4812*(6982*MYROW+1204)
      ISEED(4) = 5544012*MYROW + 4298753
      ISAV(1) = ISEED(1)
      ISAV(2) = ISEED(2)
      ISAV(3) = ISEED(3)
      ISAV(4) = ISEED(4)
      GEN = NP*MYROW+MYCOL+1
*
      IFAIL = 0
      CALL G05ABFP(ICNTXT,ISEED,GEN,IFAIL)
*
*     Now fill the first vector with random numbers
*
      DO 20 I = 1, M
         RVEC1(I) = G05AAFP()
   20 CONTINUE
*
*     Restore the original seeds and generate a second sequence
*
      ISEED(1) = ISAV(1)
      ISEED(2) = ISAV(2)
      ISEED(3) = ISAV(3)
      ISEED(4) = ISAV(4)

      IFAIL = 0
      CALL G05ABFP(ICNTXT,ISEED,GEN,IFAIL)
*
*     Now fill the second vector with random numbers
*
```

```
      DO 40 I = 1, M
         RVEC2(I) = G05AAFP()
   40 CONTINUE
*
*     Print the local vectors
*
      IF (ROOT) THEN
         WRITE (NOUT,99999) 0, 0
         WRITE (NOUT,99997) (RVEC1(K),K=1,M)
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 0, 0
         WRITE (NOUT,99997) (RVEC2(K),K=1,M)
         WRITE (NOUT,*)
      END IF
*
*     Print the vectors from the other processors
*     First find out ROOT coordinate position
*
      IFAIL = 1
      CALL Z01BAFP(ICNTXT,NRR,NRC,IFLAG)

      IF (ROOT) THEN
*
*             Receive the vector from processor I,J and print
*
         DO 80 J = 0, NP - 1
            DO 60 I = 0, MP - 1
               IF (I.NE.NRR .OR. J.NE.NRC) THEN
                  WRITE (NOUT,99999) I, J
                  CALL DGERV2D(ICNTXT,M,1,RVEC1,M,I,J)
                  WRITE (NOUT,99997) (RVEC1(K),K=1,M)
                  WRITE (NOUT,*)
                  WRITE (NOUT,99998) I, J
                  CALL DGERV2D(ICNTXT,M,1,RVEC2,M,I,J)
                  WRITE (NOUT,99997) (RVEC2(K),K=1,M)
                  WRITE (NOUT,*)
               END IF
   60       CONTINUE
   80    CONTINUE
      ELSE
*
*                Send the local vectors to the root processor
*
         CALL DGESD2D(ICNTXT,M,1,RVEC1,M,NRR,NRC)
         CALL DGESD2D(ICNTXT,M,1,RVEC2,M,NRR,NRC)
      END IF
*
      IFAIL = 0
      CALL Z01ABFP(ICNTXT,'N',IFAIL)
*
      STOP
*
99999 FORMAT ('First sequence from processor (',I2,',',I2,' )')
99998 FORMAT ('Second sequence from processor (',I2,',',I2,' )')
99997 FORMAT (1X,F10.4)
      END
```

8.2 Example Data

None.

8.3 Example Results

```
 G05ABFP Example Program Results
First sequence from processor ( 0, 0 )
     0.5607
     0.6533
     0.5597
     0.1352
     0.6257

Second sequence from processor ( 0, 0 )
     0.5607
     0.6533
     0.5597
     0.1352
     0.6257

First sequence from processor ( 1, 0 )
     0.2383
     0.4006
     0.9303
     0.5238
     0.0484

Second sequence from processor ( 1, 0 )
     0.2383
     0.4006
     0.9303
     0.5238
     0.0484

First sequence from processor ( 0, 1 )
     0.5812
     0.8899
    -0.0636
     0.1431
     0.0794

Second sequence from processor ( 0, 1 )
     0.5812
     0.8899
    -0.0636
     0.1431
     0.0794

First sequence from processor ( 1, 1 )
     0.1150
     0.4773
     0.0161
     0.3399
     0.6121

Second sequence from processor ( 1, 1 )
     0.1150
     0.4773
```

```
0.0161
0.3399
0.6121
```
