NAG Library Routine Document

G05KHF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

G05KHF allows for the generation of multiple, independent, sequences of pseudorandom numbers using the leap-frog method.

2 Specification

```
SUBROUTINE G05KHF(N, K, STATE, IFAIL)
INTEGER           N, K, STATE(lstate), IFAIL
```

3 Description

G05KHF adjusts a base generator to allow multiple, independent, sequences of pseudorandom numbers to be generated via the leap-frog method (see the G05 Chapter Introduction for details).

If, prior to calling G05KHF the base generator defined by STATE would produce random numbers $x_1, x_2, x_3, \ldots$, then after calling G05KHF the generator will produce random numbers $x_k, x_{k+n}, x_{k+2n}, x_{k+3n}, \ldots$.

One of the initialization routines G05KFF (for a repeatable sequence if computed sequentially) or G05KGF (for a non-repeatable sequence) must be called prior to the first call to G05KHF.

The leap-frog algorithm can be used in conjunction with the NAG basic generator and both the Wichmann–Hill I and Wichmann–Hill II generators.

4 References

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

5 Parameters

1: N – INTEGER *Input*

On entry: n, the total number of sequences required.

Constraint: $\mathrm{N} > 0$.

2: K – INTEGER *Input*

On entry: k, the number of the current sequence.

Constraint: $0 < \mathrm{K} \leq \mathrm{N}$.

3: STATE($lstate$) – INTEGER array *Communication Array*

Note: $lstate =$ LSTATE, the dimension of STATE as supplied to the initialization routine G05KFF or G05KGF.

On entry: contains information on the selected base generator and its current state.

On exit: contains updated information on the state of the generator.

4: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

For environments where it might be inappropriate to halt program execution when an error is detected, the value −1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value −1 or 1 is used it is essential to test the value of IFAIL on exit.**

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry, $N \leq 0$.

IFAIL = 2

On entry, $K > N$.

IFAIL = 3

On entry, STATE vector was not initialized or has been corrupted.

IFAIL = 4

On entry, cannot use the leap-frog method with the base generator defined by STATE.

7 Accuracy

Not applicable.

8 Further Comments

The leap-frog method tends to be less efficient than other methods of producing multiple, independent sequences. See the G05 Chapter Introduction for alternative choices.

9 Example

This example creates three independent sequences using G05KHF, after initialization by G05KFF. Five variates from a uniform distribution are then generated from each sequence using G05SAF.

9.1 Program Text

```
*     G05KHF Example Program Text
*     Mark 22 Release. NAG Copyright 2006.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          MSTATE, MSEED, NV, N
      PARAMETER        (MSTATE=633,MSEED=1,NV=5,N=3)
*     .. Local Scalars ..
      INTEGER          GENID, I, IFAIL, J, LSEED, LSTATE, SUBID
*     .. Local Arrays ..
      DOUBLE PRECISION X(NV)
      INTEGER          SEED(MSEED), STATE(MSTATE,N)
```

```
*     .. External Subroutines ..
      EXTERNAL         G05KFF, G05KHF, G05SAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G05KHF Example Program Results'
      WRITE (NOUT,*)
*     Initialize the seed
      SEED(1) = 1762543
*     GENID and SUBID identify the base generator
      GENID = 1
      SUBID = 1
      LSTATE = MSTATE
      LSEED = MSEED
*     Prepare N streams
      DO 20 I = 1, N
*        Initialize the generator to a repeatable sequence
         IFAIL = 1
         CALL G05KFF(GENID,SUBID,SEED,LSEED,STATE(1,I),LSTATE,IFAIL)
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99998) IFAIL
            GO TO 60
         END IF
*        Prepare the I'th out of N streams
         IFAIL = 1
         CALL G05KHF(N,I,STATE(1,I),IFAIL)
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99997) IFAIL
            GO TO 60
         END IF
   20 CONTINUE
*     Generate a NV variates from a uniform distribution, from each stream
      DO 40 I = 1, N
         WRITE (NOUT,*) 'Stream ', I
         IFAIL = 1
         CALL G05SAF(NV,STATE(1,I),X,IFAIL)
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99996) IFAIL
            GO TO 60
         END IF
*        Display the variates
         WRITE (NOUT,99999) (X(J),J=1,NV)
         WRITE (NOUT,*)
   40 CONTINUE
*
   60 CONTINUE
*
99999 FORMAT (1X,F10.4)
99998 FORMAT (1X,' ** G05KFF returned with IFAIL = ',I5)
99997 FORMAT (1X,' ** G05KHF returned with IFAIL = ',I5)
99996 FORMAT (1X,' ** G05SAF returned with IFAIL = ',I5)
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 G05KHF Example Program Results

 Stream           1
     0.7460
     0.4925
     0.4982
     0.2580
     0.5938

 Stream           2
     0.7983
     0.3843
     0.6717
```

```
     0.6238
     0.2785

 Stream           3
     0.1046
     0.7871
     0.0505
     0.0535
     0.2375
```
