NAG Library Routine Document

G05YNF

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

G05YNF initializes a scrambled quasi-random generator prior to calling G05YJF, G05YKF or G05YMF. It must be preceded by a call to one of the pseudorandom initialization routines G05KFF or G05KGF.

2 Specification

```
 SUBROUTINE G05YNF(GENID, STYPE, IDIM, IREF, LIREF, ISKIP, NSDIGI, STATE,
1                  IFAIL)
 INTEGER           GENID, STYPE, IDIM, IREF(LIREF), LIREF, ISKIP, NSDIGI,
1                  STATE(lstate), IFAIL
```

3 Description

G05YNF selects a quasi-random number generator through the input value of GENID, a method of scrambling through the input value of STYPE and initializes the IREF communication array for use in the routines G05YJF, G05YKF or G05YMF.

Scrambled quasi-random sequences are an extension of standard quasi-random sequences that attempt to eliminate the bias inherent in a quasi-random sequence whilst retaining the low-discrepancy properties. The use of a scrambled sequence allows error estimation of Monte Carlo results by performing a number of iterates and computing the variance of the results.

This implementation of scrambled quasi-random sequences is based on TOMS Algorithm 823 and details can be found in the accompanying paper, Hong and Hickernell (2003). Three methods of scrambling are supplied; the first a restricted form of Owen's scrambling (Owen (1995)), the second based on the method of Faure and Tezuka (2000) and the last method combines the first two.

Scrambled versions of the Niederreiter sequence and two sets of Sobol sequences are provided. The first Sobol sequence is obtained using GENID $= 1$. The first 8300 direction numbers for this sequence are based on the work of Joe and Kuo (2008). For dimensions greater than 8300 the direction numbers are randomly generated using the pseudorandom generator specified in STATE (see Jäckel (2002) for details). The second Sobol sequence is obtained using GENID $= 2$ and referred to in the documentation as 'Sobol (A659)'. The first 1111 direction numbers for this sequence are based on Algorithm 659 of Brately and Fox (1988) with the extension proposed by Joe and Kuo (2003). For dimensions greater than 1111 the direction numbers are once again randomly generated. The Niederreiter sequence is obtained by setting GENID $= 3$.

4 References

Brately P and Fox B L (1988) Algorithm 659: Implementing Sobol's Quasirandom Sequence Generator *ACM Trans. Math. Software* **14 (1)** 88–100

Faure H and Tezuka S (2000) Another random scrambling of digital (t,s)-sequences *Monte Carlo and Quasi-Monte Carlo Methods* Springer-Verlag, Berlin, Germany (eds K T Fang, F J Hickernell and H Niederreiter)

Hong H S and Hickernell F J (2003) Algorithm 823: Implementing Scrambled Digital Sequences *ACM Trans. Math. Software* **29:2** 95–109

Jäckel P (2002) *Monte Carlo Methods in Finance* Wiley Finance Series, John Wiley and Sons, England

Joe S and Kuo F Y (2003) Remark on Algorithm 659: Implementing Sobol's quasirandom sequence generator *ACM Trans. Math. Software* **29** 49–57

Joe S and Kuo F Y (2008) Constructing Sobol sequences with better two-dimensional projects *SIAM J. Sci. Comput.* **30** 2635–2654

Niederreiter H (1988) Low-discrepancy and low dispersion sequences *Journal of Number Theory* **30** 51–70

Owen A B (1995) Randomly permuted (t,m,s)-nets and (t,s)-sequences *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing, Lecture Notes in Statistics* **106** Springer-Verlag, New York, NY 299–317 (eds H Niederreiter and P J-S Shiue)

5 Parameters

1: GENID – INTEGER *Input*

On entry: must identify the quasi-random generator to use.

GENID = 1
Sobol generator.

GENID = 2
Sobol (A659) generator.

GENID = 3
Niederreiter generator.

Constraint: GENID = 1, 2 or 3.

2: STYPE – INTEGER *Input*

On entry: must identify the scrambling method to use.

STYPE = 0
No scrambling. This is equivalent to calling G05YLF.

STYPE = 1
Owen like scrambling.

STYPE = 2
Faure–Tezuka scrambling.

STYPE = 3
Owen and Faure–Tezuka scrambling.

Constraint: STYPE = 0, 1, 2 or 3.

3: IDIM – INTEGER *Input*

On entry: the number of dimensions required.

Constraints:

if GENID = 1, $1 \leq \text{IDIM} \leq 50,000$;
if GENID = 2, $1 \leq \text{IDIM} \leq 50,000$;
if GENID = 3, $1 \leq \text{IDIM} \leq 318$.

4: IREF(LIREF) – INTEGER array *Communication Array*

On exit: contains initialization information for use by the generator routines G05YJF, G05YKF and G05YMF. IREF must not be altered in any way between initialization and calls of the generator routines.

5: LIREF – INTEGER *Input*

On entry: the dimension of the array IREF as declared in the (sub)program from which G05YNF is called.

Constraint: LIREF $\geq 32 \times$ IDIM $+ 7$.

6: ISKIP – INTEGER *Input*

On entry: the number of terms of the sequence to skip on initialization for the Sobol and Niederreiter generators.

Constraint: $0 \leq$ ISKIP $\leq 2^{30}$.

7: NSDIGI – INTEGER *Input*

On entry: controls the number of digits (bits) to scramble when GENID $= 1$ or 2, otherwise NSDIGI is ignored. If NSDIGI < 1 or NSDIGI > 30 then all the digits are scrambled.

8: 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 pseudorandom base generator and its current state.

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

9: 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

IFAIL $= 1$

On entry, GENID $\neq 1$, 2 or 3.

IFAIL $= 2$

STYPE $\neq 0$, 1, 2 or 3.

IFAIL $= 3$

On entry, IDIM < 1,
or IDIM is too large.

IFAIL $= 5$

On entry, LIREF is too small.

IFAIL $= 6$

The value of ISKIP < 0 or ISKIP is too large.

IFAIL = 8

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

7 Accuracy

Not applicable.

8 Further Comments

The additional computational cost in using a scrambled quasi-random sequence over a non-scrambled one comes entirely during the initialization. Once G05YNF has been called the computational cost of generating a scrambled sequence and a non-scrambled one is identical.

9 Example

This example calls G05KFF, G05YMF and G05YNF to estimate the value of the integral

$$\int_0^1 \cdots \int_0^1 \prod_{i=1}^{s} |4x_i - 2| dx_1, dx_2, \ldots, dx_s = 1,$$

where s, the number of dimensions, is set to 8.

9.1 Program Text

```
*     G05YNF Example Program Text
*     Mark 22 Release. NAG Copyright 2006.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          MN, MIDIM, ORDRC
      PARAMETER        (MN=200,MIDIM=8,ORDRC=1)
      INTEGER          LDQUAS, TDQUAS, LIREF, MSTATE, MSEED
      PARAMETER        (LDQUAS=MIDIM,TDQUAS=MN,LIREF=32*MIDIM+7,
     +                 MSTATE=633,MSEED=1)
*     .. Local Scalars ..
      DOUBLE PRECISION SUM, TMP, VSBL
      INTEGER          D, GENID, I, IDIM, IFAIL, ISKIP, J, LSEED,
     +                 LSTATE, N, NSDIGI, PGENID, PSUBID, STYPE
*     .. Local Arrays ..
      DOUBLE PRECISION QUAS(LDQUAS,TDQUAS)
      INTEGER          IREF(LIREF), SEED(MSEED), STATE(MSTATE)
*     .. External Subroutines ..
      EXTERNAL         G05KFF, G05YMF, G05YNF
*     .. Intrinsic Functions ..
      INTRINSIC        ABS, DBLE
*     .. Executable Statements ..
      WRITE (NOUT,99999) 'G05YNF Example Program Results'
*     Initialize the psuedo-random generator used in the
*     scrambling to a repeatable sequence
      SEED(1) = 1762543
      PGENID = 1
      PSUBID = 1
      LSTATE = MSTATE
      LSEED = MSEED
      IFAIL = 1
      CALL G05KFF(PGENID,PSUBID,SEED,LSEED,STATE,LSTATE,IFAIL)
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,99997) IFAIL
         GO TO 80
      END IF
*     Problem size
      IDIM = 8
      N = MN
*     Skip the first few variates in the sequence
      ISKIP = 1000
```

```
*     Initialize the Sobol generator
      GENID = 1
*     Use Owen type scrambling
      STYPE = 1
*     Use the default value for NSDIGI
      NSDIGI = 0
*     Call the initializer for the quasi-random sequence
      IFAIL = 1
      CALL G05YNF(GENID,STYPE,IDIM,IREF,LIREF,ISKIP,NSDIGI,STATE,IFAIL)
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,99996) IFAIL
         GO TO 80
      END IF
*     Generate N quasi-random variates
      IFAIL = 1
      CALL G05YMF(N,ORDRC,QUAS,LDQUAS,IREF,IFAIL)
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,99995) IFAIL
         GO TO 80
      END IF
*     Evaluate the function, and sum
      SUM = 0.0D0
      DO 40 I = 1, N
         TMP = 1.0D0
         DO 20 D = 1, IDIM
            TMP = TMP*ABS(4.0D0*QUAS(D,I)-2.0D0)
   20    CONTINUE
         SUM = SUM + TMP
   40 CONTINUE
*     Convert sum to mean value
      VSBL = SUM/DBLE(N)
      WRITE (NOUT,99999)
      WRITE (NOUT,99999) 'Value of integral = ', VSBL
*     Dump the first 10 variates
      WRITE (NOUT,99999)
      WRITE (NOUT,99999) 'First 10 variates'
      DO 60 I = 1, 10
         WRITE (NOUT,99998) I, (QUAS(J,I),J=1,IDIM)
   60 CONTINUE
*
   80 CONTINUE
*
99999 FORMAT (1X,A,F8.4)
99998 FORMAT (1X,I3,20(1X,F8.4))
99997 FORMAT (1X,' ** G05KFF returned with IFAIL = ',I5)
99996 FORMAT (1X,' ** G05YNF returned with IFAIL = ',I5)
99995 FORMAT (1X,' ** G05YMF returned with IFAIL = ',I5)
      END
```

9.2 Program Data

None.

9.3 Program Results

```
 G05YNF Example Program Results

 Value of integral =   1.0169

 First 10 variates
   1   0.8688   0.9719   0.5375   0.0876   0.4721   0.3800   0.2977   0.1010
   2   0.6287   0.3611   0.4963   0.8648   0.0753   0.0174   0.7011   0.2532
   3   0.1244   0.5349   0.8645   0.2621   0.7523   0.7212   0.0538   0.6231
   4   0.1353   0.4013   0.6656   0.4691   0.9096   0.9272   0.5481   0.4164
   5   0.6154   0.6962   0.0321   0.9000   0.2307   0.3186   0.1989   0.7102
```

6	0.8870	0.0880	0.9947	0.1775	0.3148	0.2059	0.8033	0.9249
7	0.3603	0.7579	0.3633	0.6995	0.5127	0.5328	0.4496	0.2013
8	0.3304	0.1096	0.5034	0.3596	0.0137	0.3643	0.1719	0.8774
9	0.9207	0.7834	0.1357	0.7596	0.8138	0.8825	0.5831	0.2493
10	0.5828	0.4226	0.8287	0.0370	0.7336	0.5189	0.4143	0.4015