NAG Library Routine Document

G05TKF

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

G05TKF generates a vector of pseudorandom integers, each from a discrete Poisson distribution with differing parameter.

2 Specification

```
SUBROUTINE G05TKF (M, VLAMDA, STATE, X, IFAIL)
INTEGER            M, STATE(*), X(M), IFAIL
REAL (KIND=nag_wp) VLAMDA(M)
```

3 Description

G05TKF generates m integers x_j, each from a discrete Poisson distribution with mean λ_j, where the probability of $x_j = I$ is

$$P(x_j = I) = \frac{\lambda_j^I \times e^{-\lambda_j}}{I!}, \quad I = 0, 1, \ldots,$$

where

$$\lambda_j \geq 0, \quad j = 1, 2, \ldots, m.$$

The methods used by this routine have low set up times and are designed for efficient use when the value of the parameter λ changes during the simulation. For large samples from a distribution with fixed λ using G05TJF to set up and use a reference vector may be more efficient.

When $\lambda < 7.5$ the product of uniforms method is used, see for example Dagpunar (1988). For larger values of λ an envelope rejection method is used with a target distribution:

$$f(x) = \tfrac{1}{3} \qquad \text{if } |x| \leq 1,$$

$$f(x) = \tfrac{1}{3}|x|^{-3} \quad \text{otherwise.}$$

This distribution is generated using a ratio of uniforms method. A similar approach has also been suggested by Ahrens and Dieter (1989). The basic method is combined with quick acceptance and rejection tests given by Maclaren (1990). For values of $\lambda \geq 87$ Stirling's approximation is used in the computation of the Poisson distribution function, otherwise tables of factorials are used as suggested by Maclaren (1990).

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 G05TKF.

4 References

Ahrens J H and Dieter U (1989) A convenient sampling method with bounded computation times for Poisson distributions *Amer. J. Math. Management Sci.* 1–13

Dagpunar J (1988) *Principles of Random Variate Generation* Oxford University Press

Maclaren N M (1990) A Poisson random number generator *Personal Communication*

5 Arguments

1: M – INTEGER *Input*

On entry: m, the number of Poisson distributions for which pseudorandom variates are required.

Constraint: $\mathrm{M} \geq 1$.

2: VLAMDA(M) – REAL (KIND=nag_wp) array *Input*

On entry: the means, λ_j, for $j = 1, 2, \ldots, \mathrm{M}$, of the Poisson distributions.

Constraint: $0.0 \leq \mathrm{VLAMDA}(j) \leq \mathrm{X02BBF}/2.0$, for $j = 1, 2, \ldots, \mathrm{M}$.

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

Note: the actual argument supplied **must** be the array STATE supplied to the initialization routines 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: X(M) – INTEGER array *Output*

On exit: the m pseudorandom numbers from the specified Poisson distributions.

5: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

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 argument, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: $\mathrm{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry $\mathrm{IFAIL} = 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

$\mathrm{IFAIL} = 1$

On entry, $\mathrm{M} = \langle value \rangle$.
Constraint: $\mathrm{M} \geq 1$.

$\mathrm{IFAIL} = 2$

On entry, at least one element of VLAMDA is less than zero.

On entry, at least one element of VLAMDA is too large.

$\mathrm{IFAIL} = 3$

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

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

Not applicable.

8 Parallelism and Performance

G05TKF is not threaded in any implementation.

9 Further Comments

None.

10 Example

This example prints ten pseudorandom integers from five Poisson distributions with means $\lambda_1 = 0.5$, $\lambda_2 = 5$, $\lambda_3 = 10$, $\lambda_4 = 500$ and $\lambda_5 = 1000$. These are generated by ten calls to G05TKF, after initialization by G05KFF.

10.1 Program Text

```
    Program g05tkfe

!     G05TKF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: g05kff, g05tkf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter              :: lseed = 1, nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                         :: genid, i, ifail, lstate, m, n, subid
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: lambda(:)
      Integer                         :: seed(lseed)
      Integer, Allocatable            :: state(:), x(:)
!     .. Executable Statements ..
      Write (nout,*) 'G05TKF Example Program Results'
      Write (nout,*)

!     Skip heading in data file
      Read (nin,*)

!     Read in the base generator information and seed
      Read (nin,*) genid, subid, seed(1)
```

```
!     Initial call to initializer to get size of STATE array
      lstate = 0
      Allocate (state(lstate))
      ifail = 0
      Call g05kff(genid,subid,seed,lseed,state,lstate,ifail)

!     Reallocate STATE
      Deallocate (state)
      Allocate (state(lstate))

!     Initialize the generator to a repeatable sequence
      ifail = 0
      Call g05kff(genid,subid,seed,lseed,state,lstate,ifail)

!     Read in sample size
      Read (nin,*) n

!     Read in the distribution parameters
      Read (nin,*) m

      Allocate (x(n),lambda(m))

!     Read in rest of distribution parameters
      Read (nin,*) lambda(1:m)

!     Generate N sets of the M variates
      Do i = 1, n
        ifail = 0
        Call g05tkf(m,lambda,state,x,ifail)

!       Display the variates
        Write (nout,99999) i, x(1:m)
      End Do

99999 Format (1X,6(1X,I12))
    End Program g05tkfe
```

10.2 Program Data

```
G05TKF Example Program Data
1  1  1762543             :: GENID,SUBID,SEED(1)
10                        :: N
5                         :: M
0.5 5.0 10.0 500.0 1000.0 :: LAMBDA
```

10.3 Program Results

```
 G05TKF Example Program Results

             1           1           6          12         507        1003
             2           0           9          11         520        1028
             3           1           3           7         483        1041
             4           0           3          11         513        1012
             5           1           5           9         496         940
             6           0           6          17         548         990
             7           1           9           8         512        1035
             8           0           4          10         458        1029
             9           1           6          13         523         971
            10           0           9          16         519         999
```