NAG Library Routine Document

G05NEF

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

G05NEF selects a pseudorandom sample, without replacement and allowing for unequal probabilities.

2 Specification

```
SUBROUTINE G05NEF (ORDER, WT, POP, IPOP, N, ISAMPL, M, STATE, IFAIL)

INTEGER             IPOP(*), N, ISAMPL(M), M, STATE(*), IFAIL
REAL (KIND=nag_wp) WT(N)
CHARACTER(1)        ORDER, POP
```

3 Description

G05NEF selects m elements from either the set of values $(1, 2, \ldots, n)$ or a supplied population vector of length n. The probability of selecting the ith element is proportional to a user-supplied weight, w_i. Each element will appear at most once in the sample, i.e., the sampling is done without replacement.

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 G05NEF.

4 References

None.

5 Parameters

1: ORDER – CHARACTER(1) *Input*

On entry: a flag indicating the sorted status of the WT vector.

ORDER = 'A'
WT is sorted in ascending order,

ORDER = 'D'
WT is sorted in descending order,

ORDER = 'U'
WT is unsorted and G05NEF will sort the weights prior to using them.

Irrespective of the value of ORDER, no checks are made on the sorted status of WT, e.g., it is possible to supply ORDER = 'A', even when WT is not sorted. In such cases the WT array will not be sorted internally, but G05NEF will still work correctly except, possibly, in cases of extreme weight values.

It is usually more efficient to specify a value of ORDER that is consistent with the status of WT.

Constraint: ORDER = 'A', 'D' or 'U'.

2: WT(N) – REAL (KIND=nag_wp) array *Input*

On entry: w_i, the relative probability weights. These weights need not sum to 1.0.

Constraints:

WT(i) ≥ 0.0, for $i = 1, 2, \ldots, \mathrm{N}$;
at least M values must be nonzero.

3: POP – CHARACTER(1) *Input*

On entry: a flag indicating whether a population to be sampled has been supplied.

POP = 'D'
the population is assumed to be the integers $(1, 2, \ldots, \mathrm{N})$ and IPOP is not referenced,

POP = 'S'
the population must be supplied in IPOP.

Constraint: POP = 'D' or 'S'.

4: IPOP($*$) – INTEGER array *Input*

Note: the dimension of the array IPOP must be at least N if POP = 'S'.

On entry: the population to be sampled. If POP = 'D' then the population is assumed to be the set of values $(1, 2, \ldots, \mathrm{N})$ and the array IPOP is not referenced. Elements of IPOP with the same value are not combined, therefore if WT(i) $\neq 0$, WT(j) $\neq 0$ and $i \neq j$ then there is a nonzero probability that the sample will contain both IPOP(i) and IPOP(j). If IPOP(i) = IPOP(j) then that value can appear in ISAMPL more than once.

5: N – INTEGER *Input*

On entry: n, the size of the population.

Constraint: N ≥ 1.

6: ISAMPL(M) – INTEGER array *Output*

On exit: the selected sample.

7: M – INTEGER *Input*

On entry: m, the size of the sample required.

Constraint: $0 \leq \mathrm{M} \leq \mathrm{N}$.

8: 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.

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.

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.**

On exit: 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 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, ORDER $\neq$ 'A', 'D' or 'U'

IFAIL $= 2$

On entry, WT$(i) < 0.0$.

IFAIL $= 3$

On entry, POP $\neq$ 'D' or 'S'

IFAIL $= 5$

On entry, N < 1.

IFAIL $= 7$

On entry, M < 0 or M $>$ N.

IFAIL $= 8$

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

IFAIL $= 21$

On entry, there are less than M nonzero weights.

7 Accuracy

Not applicable.

8 Further Comments

G05NEF internally allocates $(\mathrm{N} + 1)$ reals and N integers.

Although it is possible to use G05NEF to sample using equal probabilities, by setting all elements of the input array WT to the same positive value, it is more efficient to use G05NDF. To sample with replacement, G05TDF can be used when the probabilities are unequal and G05TLF when the probabilities are equal.

9 Example

This example samples from a population of 25.

9.1 Program Text

```
    Program g05nefe

!     G05NEF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: g05kff, g05nef, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
```

```
      Integer, Parameter               :: lseed = 1, nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: genid, i, ifail, lipop, lstate, m,   &
                                          n, subid
      Character (1)                    :: order, pop
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: wt(:)
      Integer, Allocatable             :: ipop(:), isampl(:), state(:)
      Integer                          :: seed(lseed)
!     .. Executable Statements ..
      Write (nout,*) 'G05NEF 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 initialiser 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 population size, sample size and order
      Read (nin,*) n, m, pop
      Read (nin,*) order

      Select Case (pop)
      Case ('S','s')
        lipop = n
      Case Default
        lipop = 0
      End Select

      Allocate (ipop(lipop),wt(n),isampl(m))

      If (lipop==n) Then
!       Read in the population and weights
        Do i = 1, n
          Read (nin,*) ipop(i), wt(i)
        End Do
      Else
!       Read in just the weights
        Do i = 1, n
          Read (nin,*) wt(i)
        End Do
      End If

!     Generate the sample without replacement, unequal weights
      Call g05nef(order,wt,pop,ipop,n,isampl,m,state,ifail)

!     Display the results
      Write (nout,99999)(isampl(i),i=1,m)

99999 Format (10(1X,I4))
    End Program g05nefe
```

9.2 Program Data

```
G05NEF Example Program Data
3  0  1762543   :: GENID,SUBID,SEED(1)
25 10 'S'       :: N,M,POP
'U'             :: ORDER
171  85.54
 52  71.78
172 118.13
139  13.68
196 153.60
125 165.35
 36 122.35
 70  35.87
 25 151.78
 86 128.33
 76 178.27
 37 183.37
185 165.81
 40 101.41
 90 145.16
 27  42.01
 79  59.08
118  17.53
142  87.14
127  69.20
101  31.13
 22  60.26
 41  21.00
199  85.06
 59 119.73      :: End of IPOP,WT
```

9.3 Program Results

```
 G05NEF Example Program Results

  125   41  185   40   37  196   22   25   76  172
```