1 SUMMARY

This function generates uniformly distributed pseudo-random numbers. Random numbers are generated in the ranges $0 < \xi < 1$, $-1 < \eta < 1$ and random integers in $1 \leq k \leq N$ where $N$ is specified by the user.

A multiplicative congruent method is used where a 31 bit generator word $g$ is maintained. On each call to the subroutine $g_{n+1}$ is updated to $7^3 g_n \mod (2^{31} - 1)$; the initial value of $g$ is 1. Depending upon the type of random number required the following are computed $\xi = g_{n+1} / (2^{31} - 1)$; $\eta = 2\xi - 1$ or $k = \text{int.part}\{\xi N\} + 1$.

The subroutine also provides the facility for saving the current value of the generator word and for restarting with any specified value.


ATTRIBUTES — Version: 1.0.0. Types: FA04A; FA04AD. Original date: August 1979. Origin: C.R.Kirby and C.L.Winskill, Harwell. Remark: FA04A was formerly called FA04AS. Licence: A third-party licence for this package is available without charge.

2 HOW TO USE THE PACKAGE

2.1 Argument lists and calling sequences

There are four entries.

(i) to obtain random floating point numbers

The single precision version

$R = \text{FA04A}(I)$

The double precision version

$R = \text{FA04AD}(I)$

$I$ is an INTEGER variable which must be set by the user to chose one of two ranges for the random number. Only its sign is significant. If $I$ is non-negative the result is a real value in the range $0 < \xi < 1$ and if $I$ is negative the result is a real value in the range $-1 < \eta < 1$.

FA04A is a REAL valued function subprogram returned set to the random number.

FA04AD is a DOUBLE PRECISION valued function subprogram returned set to the random number.

(ii) to obtain random integers

The single precision version

CALL FA04B(N,K)

The double precision version

CALL FA04BD(N,K)

$N$ is an INTEGER variable which must be set by the user to specify the upper limit $N$ of the range of integers from which the random number is to be taken, i.e. the random number $k$ is chosen from $1 \leq k \leq N$. Restriction: $N$ must be positive.

$K$ is an INTEGER variable which will be set by the subroutine to the random integer $k$. 
(iii) to save the current generator word

   The single precision version
   CALL FA04C(IGEN)

   The double precision version
   CALL FA04CD(IGEN)

IGEN is an INTEGER variable which will be set by the subroutine to the current value of the generator word \( g \).

(iv) to reset the current value of the generator word

   The single precision version
   CALL FA04D(IGEN)

   The single precision version
   CALL FA04DD(IGEN)

IGEN is an INTEGER variable which must be set by the user to the new generator word. It must have a positive integer value less than \( 2^{31} - 1 \) (= 2147483647) and it is recommended that the value of IGEN should have been obtained by a previous call of FA04C/CD.

2.2 The common block

A common block called FA04E is used to hold the generator word. The user may reference this, with caution, through a common statement of the form

   The single precision version
   COMMON/FA04E/ IX

   The double precision version
   COMMON/FA04ED/ IX

IX is an INTEGER variable which contains the current value of the generator. Users are strongly advised to use a combination of FA04C/CD and FA04D/DD to change its value and not change it directly through common. In any event, IX should have a value of at least 1 and at most \( 2^{31} - 1 \) (=2147483646).

3 GENERAL INFORMATION

Use of common: the subroutine uses a common block called FA04E/ED, see §2.2, which has been initialized using a BLOCK DATA subprogram.

Workspace: none.

Other subprograms: none.

Input/Output: none.

Portability: The subroutines will work on any computer which is able to represent integers in the range \((-2^{31}, 2^{31})\). The Fortran code is nearly standard and should be portable to any computer which allows integers in the above range.
4 METHOD

4.1 Method description

The method employed is a multiplicative congruential method. The generator word \( g \) is held as an integer and is updated on each call to \( \text{FA04A or FA04B} \) as follows

\[
g_{n+1} = 7^5 g_n \mod (2^{31} - 1)
\]

The result returned from \( \text{FA04A} \), for a non-negative argument, is \( \xi \), where

\[
\xi = g_{n+1} / (2^{31} - 1)
\]

and for a negative argument is

\[
2\xi - 1
\]

The value of \( k \) returned by \( \text{FA04B} \) is

\[
\text{int.part}\{\xi N\} + 1
\]

\( \text{FA04A/B/C/D} \) are based upon the Fortran coded random number generator published in T.O.M.S., 5, no. 2, June 1979 (pages 132-138), by Linus Schrage.

4.2 Comparison with FA01A

\( \text{FA04A} \) provides the Fortran user with a random number generator which has a cycle length of \( 2^{31} - 1 \) which is twice as long as the cycle length of \( \text{FA01A} \). Moreover \( \text{FA04A} \), which is nearly standard, will return the same values (to machine accuracy) on most computers.