ACORN RANDOM NUMBER GENERATOR
USER DOCUMENTATION

R S Wikramaratna
AEA Petroleum Services
Winfrith
Dorchester
Dorset
DT2 8DH

October 1990

1. Algorithm

The k-th order ACORN (Additive Congruential Random Number) generator is defined from a modulus M, a seed Y^0_0 ($0 < Y^0_0 < M$) and initial values Y^m_0, $m = 1, \ldots, k$ satisfying $0 \leq Y^m_0 < M$ by:

$$Y^0_n = Y^0_{n-1} \qquad n \geq 1 \qquad (1)$$

$$Y^m_n = (Y^{m-1}_n + Y^m_{n-1}) \bmod M \qquad n \geq 1,\ m = 1, \ldots, k \qquad (2)$$

$$X^k_n = Y^k_n / M \qquad n \geq 1 \qquad (3)$$

It can be shown (Wikramaratna, 1990) that the numbers Y^m_n are of the form

$$Y^m_n = \left\{ \sum_{i=0}^{m} Y^i_0 \, Z^{m-i}_n \right\} \bmod M \qquad (4)$$

where

$$Z^m_n = (n+m-1)!/\left[(n-1)!m!\right] \qquad (5)$$

The number X^k_n, $n \geq 1$ are uniformly distributed in $[0,1)$. The main features of the algorithm are (Wikramaratna 1989, 1990)

(i) Long period length (equal to an integer multiple of the modulus, given suitable choice of order and seed).

(ii) Ease of programming in high level languages.

(iii) Can be programmed as function call for simplicity, or as in-line code for maximum efficiency.

(iv) Statistical behaviour of resulting sequences appears to be relatively insensitive to the seed and initial values, provided that seed and modulus are relatively prime, and modulus is sufficiently large.

(v) Amenability to theoretical analysis, as illustrated in the references (a more detailed analysis of the algorithm is the subject of current research).

2. Implementation and Portability

The ACORN generator has been coded in FORTRAN 77 as a FORTRAN - callable double precision function. Two versions of the function are available; either version can be called from a single precision or a double precision calling program. In what follows we assume that the largest integer which can be stored without overflow is at least $(2^{31}-1)$.

The first version of the function (called ACORNI) is for values of the modulus less than or equal to 2^{30}. The second version of the

function (called ACORNJ) is for values of the modulus of the form N^2 with N less than or equal to 2^{30} (ie modulus less than or equal to 2^{60}); in this version of the function, two words are used to store each integer number in order to avoid integer overflow. In each version of the algorithm all the arithmetic operations are performed in exact integer arithmetic apart from the calculation of the random number X^k_n from equation (3), so that the only rounding error associated with the function evaluation is in this calculation. As a result there is no cumulative build-up of rounding errors, and the same sequence of numbers will be generated on any machine (subject only to the accuracy of the machine representation of double precision numbers). It should be noted that for values of N less than or equal to 2^{15}, a call to ACORNJ can be replaced by an equivalent call to ACORNI, thus providing a useful check on the correctness of the coding - see the second test program.

Both the implementations of the ACORN algorithm and the test programs should execute without any modification on any machine with a FORTRAN 77 compiler. They have been tested and run successfully on a wide range of machines, including

CRAY 2
VAX 11/785
VAX 6000-410
SUN SPARC Station 1
HP 9000 Series 340
Amstrad AT
Apple Macintosh IIx

3. Test Programs

Two double precision test programs are included. In each case, the integer values calculated should be identical on any machine; the real values should be the same on any machine to the accuracy of the output and should differ only as a result of variations in machine precision.

The first program TEST1 illustrates the use of the function ACORNI. The program generates 10^5 random numbers using ACORNI, providing output after each of the first ten function calls, and then after 10^2, 10^3, 10^4 and 10^5 calls. Output includes: the current value of IXV(KORDEI+1), the value of the random number generated at this call, the mean and variance of the random numbers generated so far. A suitable test data set is included in the file DATA1. Example output is included in OUT1.

The second program TEST2 illustrates the use of the function ACORNJ. The program generates 10^5 random numbers using ACORNJ, printing output at the same stages as TEST1. Outputs include: the value of IXV1(KORDEJ+1), IXV2(KORDEJ+1), IVAL (see below), the value of the random number generated at this call, the mean and the variance of the random numbers generated so far. If the modulus of the generator is less than or equal to 2^{30} (ie MAXJNT $\leq$ 2^{15}) then IVAL is the equivalent value of IXV(KORDEI+1) for the ACORNI implementation. In the case where MAXJNT $\leq 2^{15}$, the

program then sets up the equivalent set of calls to ACORNI and prints corresponding output for comparison; the values output for ACORNI should be the same as the corresponding values for the ACORNJ; this provides a check on the correctness of the implementation. Suitable test data sets are included in the files DATA2 and DATA3. The example in DATA2 is for a large value of the modulus (equal to 2^{60}). The example in DATA3 is for a smaller value of the modulus (equal to 2^{30}). Example output is included in OUT2 and OUT3.

The objective in printing the mean and variance is to provide a global check on the random numbers being generated and to verify the uniformity of the distribution; more extensive testing of the randomness of the numbers which are generated has been carried out by Wikramaratna (1989). The test programs provided here are intended to demonstrate the correctness of the implementation rather than provide a rigorous test of the randomness.

4. Documentation

Basic documentation and instructions for use of the functions ACORNI and ACORNJ are provided in comments within the body of code. Source listings of the two routines are included in the Appendix, together with source listings of the two test programs, the input data sets and example output.

5. References

[1] R S Wikramaratna, ACORN - A New Method for Generating Sequences of Uniformly Distributed Pseudo-random Numbers, J.Comput.Phys., 83(1) 16-31 (1989)

[2] R S Wikramaratna, Theoretical Analysis of the ACORN Random Number Generator (Paper presented at the SIAM Conference on Applied Probability in Science and Engineering, New Orleans, Louisiana, March 5-7 1990), Winfrith Petroleum Technology Report PRTD(90)R62 (1990).