Random Number Generators

INTRODUCTION

- What are random numbers?
- How do we generate random numbers (for example for simulations)?
- How do we test numbers for being random?
- sequence of numbers is random if, for every block length n, the probability that a particular block of length n is observed is 1/n.
- For example for a sequence of numbers between 1 and 6 to be random it is not enough that each number occurs with probability 1/6.

INTRODUCTION

- **Random number generation** is a process which, often by means of a random number generator (RNG), generates a sequence of numbers or symbols that cannot be reasonably predicted better than by a random chance.
- A simulation of process in which random Component requires a method of generating numbers that are random
- Methods of generating random variates from uniform distribution on the interval [0 1] denoted as U(0,1)
- Random variates generated from U(0,1) distribution will be called as random numbers
- Casting lots, throwing dice, dealing out cards
- Electronic random number indicator equipment(ERNIE) Was used by British GPO to pick winners in lottery

Two Types of Random Number Generators

- Pseudo-Random Number Generators (PRNG)
- True Random Number Generators (TRNG)

Pseudo-Random Number Generators

- PRNG's generate random numbers using a mathematical algorithm or a list of numbers precalculated beforehand.
- PRNG's are said to be efficient because they can create the numbers very quickly and with minimal computing resources.
- They are also said to be deterministic because a given sequence of numbers can be reproduced at a later state if one knows the starting point of the sequence.
- They are also periodic. That is, the sequence of random numbers will eventually repeat itself. Fortunately, the period is so long that it can be ignored for most practical purposes, but it is a predictable behavior (not truly random).

True Random Number Generators

- TRNG's draw from the randomness of some accessible physical phenomena for generation of random numbers.
- Examples include radioactive decay, atmospheric noise, background (white) noise, and electrical or quantum phenomena.
- There is no discernable pattern in true random number generation and they can be counted on to produce something that is truly random.
- They are nondeterministic
  - One would not be able to reproduce a given sequence of numbers.

Random Number Generators

- In practice it is hard to generate true random numbers, in particular on a computer.
- Such numbers will always have patterns (in particular repetitions), and the aim is thus to reduce such patterns.
- Among others the following problems occur with computer generated ’random’ numbers:
- We can define a number of properties that random numbers should have and use them to test PRNGs.

Random Number Generation

Types of Sources of Random Numbers: *Three types of sources of random numbers*

1. ***Physical devices:***
   - Thermal noise in electronic components is random.
   - Last digit in (very precise) computer clock when command is executed is close to being random.

   These types of phenomena can be exploited to generate truly random numbers.

2. ***Tables:***
   - Thick volume published by RAND Corporation, titled *A Million Random Digits with 100,000 Normal Deviates*
   - Historically, this was only practical source of random numbers.
   - They were generated using physical device.
   - Such tables are now available on web.

Random Number Generation

Types of Sources of Random Numbers: *Three types of sources of random numbers*

3. Number-Theoretic Methods: **(*By far most widely used generators*)**

- Technically, not random at all; in fact, every number in sequence is completely determined by first one.
- But sequence is sufficiently complex to appear random and to pass many statistical tests.
- All modern simulation software uses RNG of this type.

In addition to statistical considerations, two practical features are important in an RNG:

- it should be possible to reproduce a sequence of random numbers;
- it should also be possible to obtain an entirely different sequence.

Pseudo-Random Numbers

- Approach: Arithmetically generation (calculation) of random numbers
- "Pseudo", because generating numbers using a known method removes the potential for true randomness.

- Goal: To produce a sequence of numbers in [0,1] that simulates, or imitates, the ideal properties of random numbers (RN).

Pseudo-Random Numbers

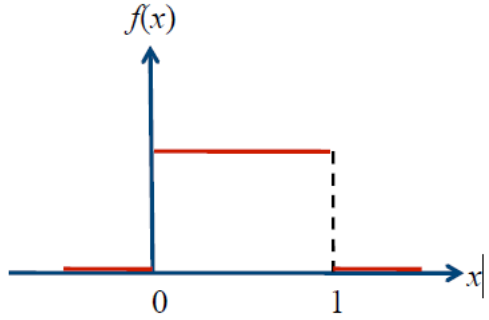
- Important properties of good random number routines:
  - Fast
  - Portable to different computers
  - Have sufficiently long cycle
  - Replicable
    - Verification and debugging
    - Use identical stream of random numbers for different systems
  - Closely approximate the ideal statistical properties of
    - uniformity and
    - independence

Pseudo-Random Numbers: Properties

- Two important statistical properties:
  - Uniformity
  - Independence
- Random number R_i must be independently drawn from a uniform distribution with PDF:

$$f(x) = \begin{cases} 1, & 0 \le x \le 1 \\ 0, & \text{otherwise} \end{cases}$$

$$E(R) = \int_0^1 x dx = \frac{x^2}{2}\Big|_0^1 = \frac{1}{2}$$



PDF for random numbers

Pseudo-Random Numbers

- Problems when generating pseudo-random numbers
  - The generated numbers might not be uniformly distributed
  - The generated numbers might be discrete-valued instead of continuous-valued
  - The mean of the generated numbers might be too high or too low
  - The variance of the generated numbers might be too high or too low

- There might be dependence:
  - Autocorrelation between numbers
  - Numbers successively higher or lower than adjacent numbers
  - Several numbers above the mean followed by several numbers below the mean

Generator Algorithms

Computer systems often come supplied with a RNG built in.

eg `rand()` or `random()`

While these are fast and easy to use, they may not be very random, have short repeat sequences and may have strong correlation patterns.

Generating Random Numbers

- Midsquare method
- Linear Congruential Method (LCM)
- Combined Linear Congruential Generators (CLCG)
- Multiplicative Congruential Generators

Midsquare method

- First arithmetic generator: Midsquare method
  - von Neumann and Metropolis in 1940s
- The Midsquare method:
  - Start with a four-digit positive integer Z_0 (m digits)
  - Compute: $Z_0^2 = Z_0 \times Z_0$ to obtain an integer with up to eight digits(2m digits). If it doesn't become 2N digits we add zeros before the number to make it 2N digits
  - Take the middle four digits for the next four-digit number($\boldsymbol{Z_i}$)
  - To make it in the interval [0,1) divide $\boldsymbol{Z_i}$ by 10

i	Z_i	U_i	$Z_i \times Z_i$
0	7182	-	51581124
1	5811	0.5811	33767721
2	7677	0.7677	58936329
3	9363	0.9363	87665769
…			

Midsquare method

Problem:

- Generated numbers tend to 0
- Not random as predictable
- If we get a random number 50 the square will be 2500 and the midterms are 50 again, we get into this chain of 50

i	Z_i	U_i	$Z_i \times Z_i$
0	7182	-	51581124
1	5811	0,5811	33767721
2	7677	0,7677	58936329
3	9363	0,9363	87665769
4	6657	0,6657	44315649
5	3156	0,3156	09960336
6	9603	0,9603	92217609
7	2176	0,2176	04734976
8	7349	0,7349	54007801
9	78	0,0078	00006084
10	60	0,006	00003600
11	36	0,0036	00001296
12	12	0,0012	00000144
13	1	0,0001	00000001
14	0	0	00000000
15	0	0	00000000

Linear Congruential Method

- Many built-in RNGs use the Linear Congruential Generator or LCG.
- This generator is very fast and has low memory requirements
- For most serious applications where randomness actually matters, it is useless.

Linear Congruential Method

- Let X_0 be the seed value (initial value of the sequence). The method produces a sequence of integers $X_1, X_2, \ldots$ between 0 and m-1 by following a recursive relationship:

$$X_{i+1} = (aX_i + c) \bmod m, \quad i = 0,1,2,...$$

The multiplier

The increment

The modulus

- Assumption: $m > 0$ and $a < m, c < m, X_0 < m$
- The selection of the values for a, c, m, and X_0 drastically affects the statistical properties and the cycle length
- The random integers X_i are being generated in $[0, m\text{-}1]$

Linear Congruential Method

- Convert the integers X_i to random numbers

$$R_i = \frac{X_i}{m}, \quad i = 1,2,\ldots$$

- Note:
  - $X_i \in \{0, 1, \ldots, m\text{-}1\}$
  - $R_i \in [0, (m\text{-}1)/m]$

Linear Congruential Method

- For a 32-bit computer, typical values of the parameters are:

$a = 7^5 = 16{,}807$ and $m = 2^{31} - 1$. c should be such that the only positive integer that (exactly divides) both c and m is 1.

- LCG is bound to “loop.”
- Whenever, *Xi* takes on a value it had previously taken, exactly the same sequence of values is generated and cycle repeats.
- The length of the cycle is called *period* of a generator.
- Length of the period can almost be *m*.
- If in fact the period is *m*, then it is called *full period LCG*.
- If the period is less than *m*, the cycle length depends on X_0.

Linear Congruential Method: Example

- Use $X_0 = 27$, $a = 17$, $c = 43$, and $m = 100$.
- The X_i and R_i values are:

$X_1 = (17 \times 27 + 43) \bmod 100 = 502 \bmod 100 = 2$ -> $R_1 = 0.02$

$X_2 = (17 \times 2 + 43) \bmod 100 = 77$ -> $R_2 = 0.77$

$X_3 = (17 \times 77 + 43) \bmod 100 = 52$ -> $R_3 = 0.52$

$X_4 = (17 \times 52 + 43) \bmod 100 = 27$ -> $R_3 = 0.27$

...

Linear Congruential Method: Example

- Use $a = 13$, $c = 0$, and $m = 64$
- The period of the generator is very low
- Seed X_0 influences the sequence

i	X_i X_0= 1	X_i X_0= 2	X_i X_0= 3	X_i X_0= 4
0	1	2	3	4
1	13	26	39	52
2	41	18	59	36
3	21	42	63	20
4	17	34	51	4
5	29	58	23	
6	57	50	43	
7	37	10	47	
8	33	2	35	
9	45		7	
10	9		27	
11	53		31	
12	49		19	
13	61		55	
14	25		11	
15	5		15	
16	1		3	

Linear Congruential Method

Characteristics of a good Generator

- Maximum Density
  - The values assumed by R_i, $i=1,2,\ldots$ leave no large gaps on [0,1]
  - Problem: Instead of continuous, each R_i is discrete
  - Solution: a very large integer for modulus m
    - Approximation appears to be of little consequence

- Maximum Period
  - To achieve maximum density and avoid cycling
  - Achieved by proper choice of a, c, m, and X_0

- Most digital computers use a binary representation of numbers
  - Speed and efficiency are aided by a modulus, m, to be (or close to) a power of 2.

Linear Congruential Method

- If $c \neq 0$ then the form is called mixed congruential generator.
- If $c=0$, then the form is called multiplicative congruential generator.
- If $a=1$, then the form is called additive congruential generator.

Multiplicative Congruential Method

QUESTION:

Generate 5 random number using multiplicative congruential method with X0=63, a=19, m=100.

SOLUTION

X0=63,a=19 , m=100

So X1= (ax0+c) mod m= (19*63+0) mod 100=97

R1= X1/m=97/100=0.97

X2= (ax1+c) mod m=(19*63+0) mod 100=43

R2= X2/m= 43/100=0.43

X3= (ax2+c) mod m=(19*43+0) mod 100=17

R3= X3/m=17/100=0.17

X4= (ax3+c) mod m=(19*17+0) mod 100=23

R4= X4/m= 23/100=0.23

X5= (ax0+c) mod m==(19*23+0) mod 100=37

R5= X5/m=37/100=0.37

Combined Linear Congruential Generators

- Reason: Longer period generator is needed because of the increasing complexity of simulated systems.
- Approach: Combine two or more multiplicative congruential generators.

- Let $X_{i,1}, X_{i,2}, \ldots, X_{i,k}$ be the i-th output from k different multiplicative congruential generators.
  - The j-th generator $X_{\bullet,j}$:

$$X_{i+1,j} = (a_j X_i + c_j) \bmod m_j$$

  - has prime modulus m_j, multiplier a_j, and period $m_j - 1$
  - produces integers $X_{i,j}$ approx ~ Uniform on $[0, m_j - 1]$
  - $W_{i,j} = X_{i,j} - 1$ is approx ~ Uniform on integers on $[0, m_j - 2]$

Combined Linear Congruential Generators

- Suggested form:

$$X_i = \left(\sum_{j=1}^{k} (-1)^{j-1} X_{i,j} \right) \bmod m_1 - 1 \qquad \text{Hence, } R_i = \begin{cases} \dfrac{X_i}{m_1}, & X_i > 0 \\ \dfrac{m_1 - 1}{m_1}, & X_i = 0 \end{cases}$$

- The maximum possible period is: $P = \dfrac{(m_1 - 1)(m_2 - 1)\ldots(m_k - 1)}{2^{k-1}}$