
Note: This document is a part of the lectures given to students of IIT Guwahati during the Jan-May 2018 Semester.

Combining Generators:

One can move from the basic linear congruence generators through summation. Wichmann and Hill proposed summing values in the unit interval. L'Ecuyer sums first and then divides.

Example:

To be more explicit, consider J generators with the j-th generator having parameters a_j and m_j. Then,

$$\begin{aligned} x_{j,i+1} &= a_j x_{j,i} \bmod m_j, \\ u_{j,i+1} &= x_{j,i+1}/m_j \ , \ j = 1, 2, \ldots, J. \end{aligned}$$

1. The Wichmann-Hill combination sets u_{i+1} equal to the fractional part of $u_{1,i+1} + u_{2,i+1} + \cdots + u_{J,i+1}$.

2. L'Ecuyer's combination takes the form:

   $$x_{i+1} = \sum_{j=1}^{J} (-1)^{(j-1)} x_{j,i+1} \bmod (m_1 - 1),$$

   and

   $$u_{i+1} = \begin{cases} x_{i+1}/m_1 \ , \ x_{i+1} > 0, \\ (m_1 - 1)/m_1 \ , \ x_{i+1} = 0. \end{cases}$$

   This assumes that m_1 is the largest of the m_j.

A combination of generators can result in a much longer period than any of its components. A long period can also be achieved in a single generator by using a larger modulus, which could create problems with overflow. In combining generators, it is possible to choose the multiplier a_j much smaller than m_j, in order to use the integer implementation. While L'Ecuyer's uses integer arithmetic, Wichmann-Hill uses floating point arithmetic.

Another way of extending the basic linear congruence generator uses a higher order recursion of the form:

$$x_i = (a_1 x_{i-1} + a_2 x_{i-2} + \cdots + a_k x_{i-k}) \bmod m.$$

This is called a multiple recursive generator. A seed for this generator consists of initial values $x_{k-1}, x_{k-2}, \ldots, x_0$. The vector $(x_{i-1}, x_{i-2}, \ldots, x_{i-k})$ can take up to m^k distinct values. The sequence x_i repeats once this vector returns to a previously visited value. If the vector reaches the zero vector then all subsequent x_i are zero. Thus the longest possible period for the multiple recursive generator is $m^k - 1$. See Knuth's book for the conditions on m and $a_1, a_2, \ldots, a_k$ under which this is bound to be achieved.

Other Methods:

This method produces a stream of bits that are concatenated to produce integers and then normalized to produce points in the unit interval. Bits can be produced by linear recursion mod 2.

Example:

$$b_i = (a_1 b_{i-1} + a_2 b_{i-2} + \cdots + a_k b_{i-k}) \bmod m \ , \ a_i = 0 \text{ or } 1.$$

Inverse Congruential Generator:

This method uses recursion of the form

$$x_{i+1} = (a x_i^- + c) \bmod m,$$

where the "mod m inverse x^- of x" (If it is an integer in $\{1, 2, \dots, m-1\}$, then that integer is unique.) satisfies $xx^- = 1 \bmod m$.

Fibonacci Generators:

The original Fibonacci recursion motivates the formula:

$$N_{i+1} = (N_i + N_{i-1}) \bmod M.$$

It turns out that this is not suitable for generating random numbers. The modified approach is:

$$N_{i+1} = (N_{i-\nu} + N_{i-\mu}) \bmod M,$$

for suitable $\nu, \mu \in \mathbb{N}$ is called the lagged Fibonacci generator. For many ν and μ, this approach leads to recommendable generators.

Example:

$U_i = u_{i-17} + U_{i-5}$. In case $U_i < 0$ we set $U_i = U_i + 1.0$. This recursion immediately produces floating point numbers. This generator requires a prologue in which 17 initial U's are generated by means of another method.

General Sampling Methods:

With the introduction of random number generators behind us, we assume the availability of an ideal sequence of random numbers. More precisely, we assume the availability of a sequence $U_1, U_2, \dots$ of independent random variables satisfying,

$$P(U_i \leq u) = \begin{cases} 0\ ,\ u < 0 \\ u\ ,\ 0 \leq u \leq 1 \\ 1\ ,\ u > 1, \end{cases}$$

that is uniformly distributed between 0 and 1. A simulation algorithm transforms these independent uniform variables into sample paths of stochastic process. A typical simulation uses methods for transforming samples from the uniform distribution to samples from other distributions. The most widely used general techniques are:

1. Inverse Transform Method.
2. Acceptance Rejection Method.

Inverse Transform Method:

Suppose we want a sample from a cumulative distribution function $F(x)$, that is we want to generate a random variable X with the property that $P(X \leq x) = F(x)\ \forall\ x$. The inverse transform method sets:

$$X = F^{-1}(U)\ ,\ U \sim \mathcal{U}[0,1],$$

where $\mathcal{U}[0,1]$ is a uniform distribution on $[0,1]$.

Theorem:

Suppose $U \sim \mathcal{U}[0,1]$ and F is a continuous strictly increasing function. Then $F^{-1}(U)$ is a sample from F.

Proof:

Let P denote the underlying probability $U \sim \mathcal{U}[0,1]$ means $P(U \leq \xi) = \xi\ ,\ 0 \leq \xi \leq 1$. Therefore $P(F^{-1}(U) \leq x) = P(U \leq F(x)) = F(x)$.

Examples:

1. Exponential Distribution:

   The exponential distribution with mean θ has distribution

$$F(x) = 1 - e^{-x/\theta}.$$

This is the distribution of the time between jumps of a Poisson process with rate $1/\theta$. Inverting yields

$$X = -\theta \log(1 - U).$$

This can be implemented also as $X = -\theta \log(U)$, since U and $(1 - U)$ have the same distribution.

2. Arc Sin Law:
   The time at which a standard Brownian motion attains it's maximum over the time interval $[0, 1]$ and has distribution:
   $$F(x) = \frac{2}{\pi} \text{ arc sin } \sqrt{x}\ ,\ 0 \le x \le 1.$$
   The inverse transform method for sampling from this distribution is:
   $$X = \sin^2\left(\frac{U\pi}{2}\right) = \frac{1}{2} - \frac{1}{2}\cos(U\pi)\ , 0 \le t \le \pi/2,\ ,\ U \sim \mathcal{U}[0, 1].$$