Pseudo-random Number Generators: Linear and Non-linear Method

Donald Stevens

March 10, 2014

Abstract

Simulation and Modelling project: An attempt to create in the R programming language various kinds of pseudo-random number generators. Commentary regarding the endeavor included.

keywords

1. Linear Congruential Generators (LCG)
2. Multiple Recursive Generator (MRG)
3. Combined LCGs and MRGs (CMRG)
4. Inversive Congruential Generator (ICG)
5. Quadratic Congruential Generators (QCG)
6. Blum, Blum & Shub generator (BBS)

1 Introduction

There are two kinds of pseudo-random number generators in this paper, linear and non-linear. Their characteristics and limitations are discussed below. I will be using code samples made from within the R programming environment.

2 Linear Pseudo-random Number Generation

2.1 Linear Congruential Generators (LCG)

Using a linear equation one can produce a series of values that are reliably non-periodic. That is not easily distinguished repeated order of numbers will occur. The critical components to this generator are:

1. the "modulus" m
2. the "additive constant", or c

3. the "multiplier" or a
4. and finally the "seed" or "start value" r0

$$r_{n+1} = a \times r_n + c \pmod{m}$$

According to A. Gille-Genest, "The maximal period for a LCG is m. It is reached if and only if c is relatively prime to m, a - 1 is a multiple of p for every prime factor p of m and if m is a multiple of 4, then a - 1 is also a multiple of 4."[2]

2.2 Multiple Recursive Generators (MRG)

For this paper I will be using the MRG described at the R-Project.Org[3], specifically the generator, *rstream.mrg32k3a*

The recurrence relation for a MRG is defined by:
$x_n = (a_1 x_{n-1} \ldots + a_k x_{n-k}) \mod m$[2]

2.3 Inversive Congruential Generators (ICG)

defined by: $x_{n+1} = \overline{ax_n + b} \mod m$

2.4 Quadratic Congruential Generators (QCG)

described by Blum, Blum, and Shub: $g = a_1 x_n^2 + a_2 x_n + c$

2.5 Blum, Blum & Shub Generator (BBS)

defined as: $x_n = (x_{n-1})^2 \mod m$, $x_0 = x_2 \mod m$ where $m = pq$ and p and q are both large prime numbers. The seed x_0 is greater than 1 and coprime to m, that is, it has no factors in common with p or q. Additionally, p and q should be congruent to 3 mod 4. So the remainder of p divided by 4 and q divided by 4 is 3, as well as $x_0/4 = 3$ One you have done the square of $x_n - 1 \mod m$ then calculate the least significant bit of the resulting x_n.[1]

2.6 Determining the PRNG performance measure.

There are a few ways one can determine if the number set is random. One simple method is simple to plot the numbers as seen in figures included and to observe the repetitive or cluster patterns seen in a non-random set. Some sets will converge to a single value over time, others will only appear random over a specific interval but beyond this interval the random pattern will recur, this is known as a lattice structure. For this reason picking the right prime numbers and constants is important.[4]

Currently, the best standard measure of randomness for a PRNG is the Dieharder series of tests by Robert G. Brown, Dirk Eddelbuettel, David Bauer from Duke University.[5]

Following the example used by Strata for testing their random number generator, I first created a data set from each random number generator consisting of 3 million values in ascii format. Generally one would then convert it into

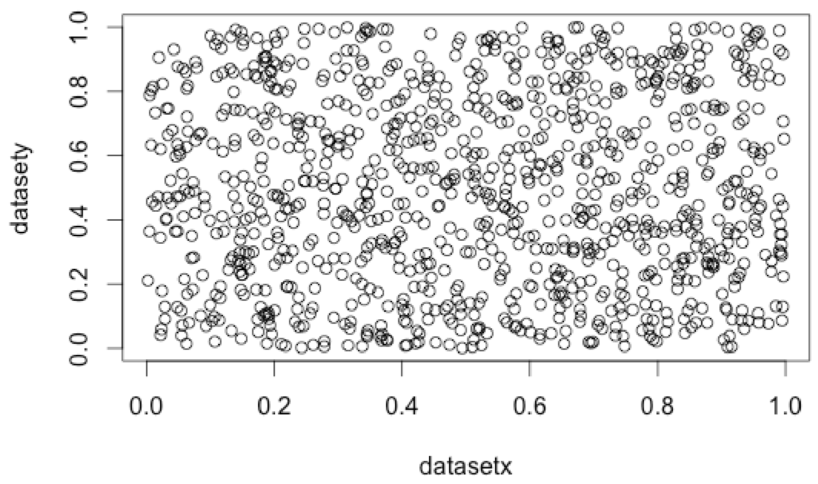

Figure 1: This figure shows two random datasets generated via Linear Congruential generation plotted as x,y coordinates between zero and one.

Number Generator	Dieharder Testing (sample size=3,000,000)
LCG	0.4259
MRG	0.0673
CMRG	1.2618
ICG	0.6254
QCG	13
BBS	13

Table 1: An example table.

hexadecimal and finally into binary as shown on their site, however it appears the current version of Dieharder can run the full suite of tests using a properly formatted ascii text file of numbers.[4]

LaTeX is great at mathematics. Let $X_1, X_2, \ldots, X_n$ be a sequence of independent and identically distributed random variables with $\mathrm{E}[X_i] = \mu$ and $\mathrm{Var}[X_i] = \sigma^2 < \infty$, and let

$$S_n = \frac{X_1 + X_2 + \cdots + X_n}{n} = \frac{1}{n}\sum_{i}^{n} X_i$$

denote their mean. Then as n approaches infinity, the random variables $\sqrt{n}(S_n - \mu)$ converge in distribution to a normal $\mathcal{N}(0, \sigma^2)$.

References

[1] Scott A. Vanstone Alfred J. Menezes, Paul C. van Oorschot. *Handbook of Applied Cryptography.* CRC Press, 2010.

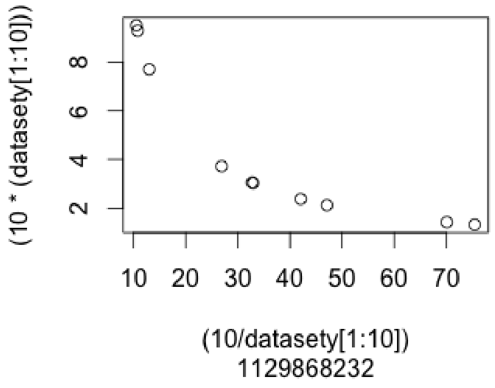

Figure 2: This figure shows a sample of 10 points plotted from datasety in the form (1/x,10x) using 1129868232 as the seed value.

[2] Anne Gille-Genest. Pseudo-random numbers generators, March 2012.

[3] Josef Leydold. Streams of random numbers, August 2012.

[4] StataCorp LP, February 2014.

[5] David Bauer Robert G. Brown, Dirk Eddelbuettel, February 2014.