Uniform Random Rational Number Generation

Thomas Morgenstern

FB Automatisierung und Informatik, Hochschule Harz, Wernigerode, Germany
tmorgenstern@hs-harz.de

Summary. Classical floating point random numbers fail simple tests when considered as rational numbers.

A uniform random number generator based on a linear congruential generator with output in the rational numbers is implemented within the two computer algebra systems Maple 10 and MuPAD Pro 3.1.

The empirical tests in Knuth's suite of tests show no evidence against the hypothesis that the output samples independent and identically uniformly distributed random variables.

1 Introduction

Randomized algorithms like Monte Carlo simulation require long sequences of random numbers $(x_i)_{i\in\mathbb{N}}$. These sequences are asked to be samples of independent and identically distributed (i.i.d.) random variables $(X_i)_{i\in\mathbb{N}}$. Of special interest are in the open unit interval $(0,1) \subseteq \mathbb{R}$ uniformly distributed random variables $U_i \sim \mathcal{U}(0,1)$.

Random numbers produced by algorithmic random number generators (RNG) are never random, but should appear to be random to the uninitiated. We call these generators pseudorandom number generators [1]. They should pass statistical tests of the hypothesis [2]:

> H_0: the sequence is a sample of i.i.d. random variables with the given distribution.

In fact, no pseudo RNG can pass all statistical tests. So we may say that bad RNGs are those that fail simple tests, whereas good RNGs fail only complicated tests that are very hard hard to find and to run [4, 5].

1.1 Definitions

We follow the definitions in [2, 4, 3]:

Definition 1. *A* (pseudo-) random number generator *(RNG) is a structure* $(\mathcal{S}, \mu, t, \mathcal{O}, o)$ *where* $\mathcal{S}$ *is a finite set of* states *(the* state space*),* μ *is a probability distribution on* $\mathcal{S}$ *used to select the* initial state *(or* seed*)* s_0*,* $t : \mathcal{S} \to \mathcal{S}$ *is the* transition function*,* $\mathcal{O}$ *is the* output space*, and* $o : \mathcal{S} \to \mathcal{O}$ *is the* output function*.*

The state of the RNG evolves according to the recurrence $s_i = t(s_{i-1})$, for $i \geq 1$, and the output at step i is $u_i = o(s_i) \in \mathcal{O}$. The output values $u_0, u_1, u_2, \ldots$ are called the *random numbers* produced by the RNG.

In order to keep the computation times moderate and to compare our results with the literature we use a generator implemented in Maple 10 [9] and MuPAD Pro 3.1 [10] (and discussed in [4]) and call it LCG10. It is a multiplicative congruential generator ($c = 0$) with multiplier $a = 427\,419\,669\,081$ and modulus $m = 10^{12} - 11 = 999\,999\,999\,989$, i.e. with the recurrence:

$$s_{i+1} := 427\,419\,669\,081 \cdot s_i \bmod 999\,999\,999\,989 \quad . \tag{1}$$

It has period length $\rho = 10^{12} - 12$. To produce values in the unit interval $\mathcal{O} = (0, 1) \subseteq \mathbb{R}$ one usually uses the output function:

$$u_i = o(s_i) := s_i / m \ .$$

1.2 Problems and Indications

Example 1. Consider a micro wave with frequency 10^{10} Hz. We want to determine the signal energy $E = \int_0^1 \cos(2\,\pi\,10^{10} \times t)^2\,dt$ by Monte-Carlo integration. We use the generator Eq. 1 to produce (decimal 10 digits precision floating point) numbers $u_i \in (0, 1)$ and simulate the times $t_i := 2\,\pi\,10^{10} \times u_i$. For n random numbers and $S := \sum_{i=1}^{n} \cos^2(t_i)$ we expect $E \approx S/n$.

Using $n := 10^6$ and 10 iterations starting with seed 1 we get values in the interval $[0.949, 0.950]$ with mean 0.9501, far from the true result 0.50. Using the same numbers to integrate the cosine $\int_0^1 \cos(2\,\pi\,10^{10} \times t)\,dt$ we get values in $[0.899, 0.901]$ with mean 0.9000.

These results are due to the fact that mainly natural multipliers of $2\,\pi$ are generated. To see this, we multiply the number u_i by $m = 10^{12}$ or $m = 999\,999\,999\,989$ and treat the result as rational number r_i (e.g. converte it to a rational number using `convert(u, rational, exact)` or `convert(u, rational,10)` in Maple 10). The smallest common multiplier of the divisors is 1, what is unlikely for true random rational numbers.