Random Number Generators and Empirical Tests

Pierre L'Ecuyer

ABSTRACT We recall some requirements for "good" random number generators and argue that while the construction of generators and the choice of their parameters must be based on theory, a posteriori empirical testing is also important. We then give examples of tests failed by some popular generators and examples of generators passing these tests.

1 Introduction

Roughly, a random number generator is a simple program producing a periodic sequence of numbers on a computer. This sequence should behave, for practical purposes, as the realization of a sequence of independent and identically distributed (i.i.d.) random variables, say, uniformly over the interval $(0, 1)$, or $U(0, 1)$ for short. In the next section, we provide a more formal definition and describe basic desired properties for a generator. With these properties in view, the construction of a generator must be based on a good theoretical analysis of its structure. In this paper, we do not study the tools for such theoretical analysis (this is done elsewhere), but we focus rather on empirical tests that can be applied a posteriori. Section 3 discusses the idea of empirical statistical testing and what it can do for us. The theoretical distribution functions used in statistical tests are often known only asymptotically or approximately, and one must assess and control the approximation error for finite sample size. Section 3 also discusses how to do this. Sections 4 and 5 describe two specific classes of tests, based on close points in space and on discrete entropy, respectively. They summarize the content of the two papers [LCS96, LCC96]. Section 6 gives the results of a set of tests applied to a set of previously proposed generators. This is mainly to illustrate what the tests can do, and not to recommend any specific generator. Section 7 is a brief conclusion.

2 Random number generators

We define a *random number generator* as a structure comprised of a finite set of *states* S, a probability distribution μ over S, a *transition function* $T : S \to S$, a finite set of *output* symbols U, and an *output function* $G : S \to U$. The *initial state* (or *seed*) $s_0 \in S$ is generated (externally) from the probability distribution μ, and then the state evolves according to $s_n = T(s_{n-1})$, for $n \geq 1$. At step n, the generator outputs $u_n = G(s_n) \in U$. These u_n are called the *random numbers*. Typically, U is the interval $[0, 1)$. Often, the seed s_0 is chosen deterministically, so μ is degenerate. Obviously, the sequence of states s_n is periodic with period length ρ not exceeding the cardinality of the set S. If b bits of memory are used to represent the state, then $\rho \leq 2^b$, and we generally want ρ to be near 2^b.

The unreachable dream is that no statistical test (or computer program) could distinguish between the output sequence of the generator and the realization of an i.i.d. uniform sequence over U; e.g., cannot tell which is which better than by flipping a fair coin. This is obviously impossible, since one can always distinguish them if the sequence is observed long enough, because $\rho < \infty$.

A (perhaps) more reachable dream can be defined via the point of view of computational complexity. Consider a *family* of generators $\{\mathcal{G}_b = (S_b, \mu_b, T_b, U_b, G_b), b \geq 1\}$, where the parameter b indicates the size of the memory used to represent the state. So, the state space S_b of generator $\mathcal{G}_b$ has cardinality 2^b. We assume that the functions T_b and G_b can be computed in polynomial time (in b) and restrict our attention to the statistical tests whose running time is also polynomial in b. Assuming that the period length increases as 2^b, this excludes any statistical test that exhausts the period. The family $\{\mathcal{G}_b\}$ is called *polynomial-time perfect* if there is a constant $\epsilon > 0$ such that, for any polynomial-time statistical test trying to distinguish the output sequence $\{u_n\}$ from the realization of a sequence of i.i.d. $U(0, 1)$ random variables, the probability of the test guessing right is less than $1/2 + e^{-b\epsilon}$. This is equivalent to the requirement that no polynomial-time algorithm, after observing $u_0, \ldots, u_{n-1}$, could correctly predict any given bit of u_n with probability exceeding $1/2 + e^{-b\epsilon}$. This computational complexity setup is well-known in the area of cryptology, where unpredictability is a key issue. For more details, see, e.g., [BBS86, Lag93, LP89]. The idea is that statistical tests not running in polynomial time "quickly" become too long to apply, as b gets large enough, even with the fastest computers. So, polynomial-time perfect generators *provably* pass (essentially) all statistical tests when their size is large enough.

That sounds nice but how large is large enough? These unpredictability properties are only asymptotic and we do not know how safe we are for any given b. Moreover, there are generator families *conjectured* to be polynomial-time perfect, but none has been *proved* to be so. In fact, nobody knows for sure whether or not such a family *exists*. A nice instance is the BBS generator, introduced by Blum, Blum, and Shub [BBS86, LP89], which is proved polynomial-time perfect under the (unproven) assumption that factoring the product of two (unknown) b-bit random primes cannot be done in polynomial time in b.

The generators used in simulation are *not* polynomial-time perfect, but are generally much faster than those conjectured to be so, and some of them seem to have statistical behavior good enough for most simulation applications.

The sequence of states of a generator with period length ρ can be put on a circle, and the generator goes around just like a runner circling a track with ρ steps per lap. In general, the generator may have many disjoint subcycles, and will cycle around one of many "tracks", depending on its initial state s_0. Here, to simplify, we suppose that there is a single cycle. If the seed s_0 is fixed to a default value, as is sometimes the case, then the sequence is always exactly the same, so it is very easy to construct a statistical test that this generator will fail.

Suppose now that the starting point s_0 is picked randomly, uniformly over the ρ values on the circle. For any given n and fixed t, $1 < t \ll \rho$, consider the vector of t successive output values $\boldsymbol{U}_n = (U_n, \ldots, U_{n+t-1})$. This vector $\boldsymbol{U}_n$ is now truly random, uniformly distributed (at best) over a set Ψ of cardinality ρ. Ideally, we would like $\boldsymbol{U}_n$ to be uniformly distributed over the t-dimensional hypercube U^t. Strictly speaking, this is possible only if $|U|$, the cardinality of the set U, is finite and if ρ is a multiple of $|U|^t$. This can occur only for small t. Otherwise, for t such that $|U|^t > \rho$, one would (heuristically) like the set $\Psi = \{\boldsymbol{U}_n, 0 \le n < \rho\}$ to be uniformly spread over U^t. Observe that we do not ask here for Ψ to look like a random set of points, but rather to have a *super-uniform* (or very even) distribution over U^t. Then, a few points picked randomly from Ψ should behave pretty much as i.i.d. uniform points over U^t, assuming that ρ/t is several orders of magnitudes larger than the number of those points. This justifies huge period lengths, much larger than what is actually used in a simulation (e.g., $\rho > 2^{100}$ or more).

As an analogy, suppose one is asked to place 10^{10} balls of 10 different colors in a (huge) box. Every evening, the balls are well-mixed and 16 of them are drawn randomly, without replacement. The 16 successive colors $C_1, \ldots, C_{16}$ are noted (e.g., to determine the winning numbers of a lottery), then the 16 balls are replaced in the box for the next day. The question is: How many balls of each color to place in the box if we want the vector $(C_1, \ldots, C_{16})$ to be approximately uniformly distributed over the set of 10^{16} possible vectors? Obviously, the answer is to place 10^9 balls of each color. Then, if we pick a large fraction of the balls from the box, we get superuniformity, but if we pick only a few (e.g., 16), we get a close approximation to uniformity. Our reasoning is similar in the case of random number generators. Another possibility would be that the colors of the 10^{10} balls placed in the box be independent uniform random variables, so the numbers of balls of each color in the box, say $(X_1, \ldots, X_{10})$, would be multinomial with parameters $n = 10^{10}$ and $p_1 = \cdots = p_{10} = 1/10$. In this case, the *a priori* distribution of $(C_1, \ldots, C_{16})$ is uniform, so this is a perfect strategy if these balls are used only once. However, the distribution of $(C_1, \ldots, C_{16})$ *conditional* on $(X_1, \ldots, X_{10})$ is generally nonuniform and is the same from day to day, so this scheme is not so good if the balls are to be used for several days. According to this argument, if a random number generator is to be constructed once and used for several applications, then, over the entire period of the generator, a superuniform point set might be preferred over a point set whose distribution resembles a typical realization of i.i.d. uniform random variables.

The real interval $[0, 1)$ can be approximated by the finite set $U = \{0, 1/m, 2/m, \ldots, 1-1/m\}$, where m is a large integer. Suppose that $\rho = |\Psi| \approx m^k$. If $t \leq k$, then $\boldsymbol{U}_n$ can be approximately uniformly distributed over U^t and this is what we want. If $t > k$, then $|U^t|/|\Psi| \approx m^{t-k}$, so Ψ can cover only a small fraction of U^t. In that case, following the above argument, we would like the points of Ψ to be evenly distributed over the unit hypercube $[0, 1)^t$. If this holds for all t up to some reasonably large value t_0 and if ρ is large enough, then the vectors $\boldsymbol{U}_n$ should behave pretty much as independent random points in the unit hypercube. This is similar to the "scanning ensemble" framework described by Compagner [Com91, Com95].

The above super-uniformity requirements provide precise theoretical guidelines for constructing specific generators. Their application, in turn, implies a thorough theoretical understanding of the equidistribution properties of the generators. Theoretical analysis is crucial for the design of good generators and has been the subject of several books and papers; see, e.g., [Knu81, L'E94, L'E97, Nie92] and other references given there. For generators based on linear recurrences, for example, this is often achieved by analyzing the underlying lattice structure (sometimes called the "spectral test"). Generators with well-understood structural properties may appear less random, but the less understood ones could hide worse defects.

After a specific generator has been constructed and implemented, it is a good idea to test it empirically. This is what we discuss in the next section. Good theoretical and statistical behavior are not the only desired properties. Speed, low memory usage, ease of implementation, and fast jump-ahead capabilities for splitting, are among the other major requirements; see [L'E94, L'E97] for further details.

3 Empirical statistical tests

Empirical tests view a generator as a black box and compute certain statistics from part of its output sequence in order to catch up significant defects.

From now on, we assume that $U = [0, 1)$ and we view the generator's output (before it is observed) as a sequence of random variables $U_0, U_1, U_2, \ldots$. We want to test the null hypothesis H_0: "The output sequence $\{U_0, U_1, U_2, \ldots\}$ is a sequence of i.i.d. $U(0, 1)$ random variables". We know in advance that H_0 is false, but the question is "how easily can we detect it?". A *statistical test* in our context is defined by any real-valued function T of any finite number of output values, say $\{U_0, \ldots, U_{n-1}\}$. Since

n has no upper bound, there is an infinite number of such tests. The random variable T is called a *statistic* and its distribution under the null hypothesis H_0 (or a good approximation of this distribution) must be known to apply the test. In its simplest version, the test computes the value of T from the output sequence of the generator and rejects H_0 if that value differs significantly from the "typical" values of T that are to be expected under H_0.

Since there is an unlimited number of tests, which tests are the good ones? There is no general answer to this question. Different tests may detect different types of defects. As a general rule, we want to detect the defects that may have a bad impact on our simulation results and do not really care about the other types of defects. So, what we need is a statistic T whose distribution under H_0 is known (or well-approximated) and whose behavior resembles very well that of the random variable Y (say) of interest in our simulation. This is usually too hard to achieve in practice for two reasons:

(a) Typically, if we need simulation to estimate the properties of Y (such as, e.g., its expectation), it is precisely because they are too complicated to be studied directly. In this case, it is unlikely that we can find a T as just described.

(b) The random number generators provided in general purpose softwares cannot be tested this way, because the random variable Y of interest to the user is unknown in advance and varies from user to user.

In some cases, in may be possible to find a T whose behavior is close to that of Y. Then, reason (a) no longer applies. Few users, however, are ready to spend time testing the random number generators. It is therefore important that the generators proposed for general use be submitted to a wide variety of tests that can detect different types of correlations or other harmful structures. New classes of applications bring the need for new families of tests, based on, e.g., spatial statistics, random walks, and so on.

Random number generators are often tested via a *two-level* procedure as follows. Compute N "independent" copies of T, denoted $T_1, \ldots, T_N$, from disjoint segments of the output sequence, and compute their empirical distribution $\hat{F}_N$. Then, compare this empirical distribution to the theoretical distribution of T under H_0, say F. This comparison can be done by a standard goodness-of-fit test, such as those of Kolmogorov-Smirnov (KS), Anderson-Darling, etc. [Dur73, Knu81, Ste86a].

The KS test computes the KS statistics

$$\begin{aligned} D_N^+ &= \sup_{0 \le u \le 1} (\hat{F}(u) - F(u)), \\ D_N^- &= \sup_{0 \le u \le 1} (F(u) - \hat{F}(u)). \end{aligned}$$

TABLE 1.1. Significance levels for D_N^+

N	10	10^2	10^3	10^4	10^5
$\alpha = 0.95$	.410	.134	.043	.013	.004
$\alpha = 0.99$	.495	.161	.051	.016	.005

These two random variables take values, say, d_N^+ and d_N^-, respectively, with significance levels

$$\delta^+ = P[D_N^+ > d_N^+] \quad \text{and} \quad \delta^- = P[D_N^- > d_N^-],$$

where the latter probabilities are under H_0. The distribution functions of D_N^+ and D_N^- under H_0 can be computed as explained in [Dur73], assuming that F is continuous. Under H_0, δ^+ and δ^- are (correlated) $U(0,1)$ random variables. One rejects H_0 if δ^+ or δ^- is far too close to 0 or 1. In case of doubt, one can repeat the test with other segments of the sequence and reject H_0 if δ^+ or δ^- is consistently close to 0 or 1. When the confidence levels are "reasonable", this is somewhat reassuring, but does not prove that the generator will behave correctly for any arbitrary application. No statistical test can prove this. Nevertheless, generally speaking, one would better trust a generator which has passed an extensive and varied set of tests. For better confidence, it is also a good idea to re-run important simulations with totally different random number generators.

For many tests, including those described in the following sections, the exact theoretical distribution of the statistic T under H_0, say $\tilde{F}$, is unknown and only an approximation F (usually an asymptotic), is available. For large N, the test will eventually detect the lack of fit of the approximation and even the best generators will be rejected. We must therefore control the approximation error. Since we have a KS test in mind, the following error measures are convenient:

$$\begin{aligned}
\Delta^+ &\stackrel{\text{def}}{=} \sup_{0 \le u \le 1} (\tilde{F}(u) - F(u)), \\
\Delta^- &\stackrel{\text{def}}{=} \sup_{0 \le u \le 1} (F(u) - \tilde{F}(u)).
\end{aligned}$$

To determine what error size is acceptable, the following table is useful. It gives the values of d_N^+ yielding significance levels α of 0.95 and 0.99, for different values of N. For $N = 100$, for example, rejection at the 99% level occurs only for $d_N^+ > 0.16$, so an error Δ^+ of, say, less than 0.01 on the distribution has little effect on the test. The same holds for Δ^-.

Generally, Δ^+ and Δ^- cannot be computed exactly, but they can at least be estimated by simulation, using simply $\hat{\Delta}^+ = D_N^+$ and $\hat{\Delta}^- = D_N^-$ as estimators, with very large N. This also provides confidence intervals (using the KS distribution). This is what we did here, using different types of "robust" generators, such as G14 in Section 6, and the results agreed well across the different generators.

Knuth [Knu81] describes a "classical" set of tests. Marsaglia [Mar85, Mar96] proposes a more recent battery, called DIEHARD. Some of these tests are more "stringent" than the classical ones, in the sense that more popular generators tend to fail them. References to other statistical tests can be found, for instance, in [L'E92, L'E94, LCS96, LCC96, L'E96b, LW97, Ste86b]. Another extensive testing package called `TestU01` [L'E96b], still under development, implements most of the tests proposed so far, as well as several classes of generators implemented in generic form. The next two sections describe two classes of statistical tests implemented in [L'E96b], but not belonging to earlier packages. Further details on these tests can be found in [LCS96, LCC96].

4 Tests based on close points

Suppose we generate n points $\boldsymbol{U}_i = (U_i, \ldots, U_{i+k-1})$, $i = k, 2k, 3k, \ldots$, in the k-dimensional hypercube $[0,1)^k$. Under H_0, since these vectors are non-overlapping, the points are i.i.d. uniformly distributed in the hypercube. For each pair $i \neq j$, let

$$D_{n,i,j} = \|\boldsymbol{U}_j - \boldsymbol{U}_i\|_p$$

be the distance between $\boldsymbol{U}_i$ and $\boldsymbol{U}_j$ induced by the L_p-norm, defined by

$$\|(x_1, \ldots, x_k)\|_p = \begin{cases} (|x_1|^p + \cdots + |x_k|^p)^{1/p} & \text{for } 1 \leq p < \infty; \\ \max(|x_1|, \ldots, |x_k|) & \text{for } p = \infty. \end{cases}$$

Let V_k be the volume of the unit ball $\{x \in \mathbb{R}^k \mid \|x\|_p \leq 1\}$, $\mu = V_k/(n(n-1)/2)$ and, for all $t \geq 0$, $Y_n(t)$ be the number of pairs (i,j), with $i < j$, such that $D^k_{n,i,j} \leq \mu t$. The next proposition characterizes the stochastic process Y_n. For the proof and more details, see [LCS96, RS78, SB78, Rip87].

Proposition 1 *Under H_0, for each $t_1 > 0$, as $n \to \infty$, the process $\{Y_n(t), 0 \leq t \leq t_1\}$ converges weakly to a Poisson process with unit rate.*

Let

$$\begin{aligned} T_{n,0} &= 0, \\ T_{n,i} &= \inf\{t \geq 0 \mid Y_n(t) \geq i\}, \quad i \geq 1, \\ \Delta_{n,i} &= T_{n,i} - T_{n,i-1}. \end{aligned}$$

The $\Delta_{n,i}$ are the times between the successive jumps of the process Y_n. Therefore, under H_0, they are approximately i.i.d. exponential with unit mean, and the transformed random variables

$$W^*_{n,i} = 1 - \exp[-\Delta_{n,i}]$$

are approximately i.i.d. $U(0,1)$. In particular, the random variable

$$W_n^* = W_{n,1}^* = 1 - \exp[-(D_n^*)^k/\mu],$$

where $D_n^* = \min_{i \neq j} D_{n,i,j}$ is the smallest interpoint distance, is approximately $U(0,1)$.

Based on this, we can define a statistical test using $T = W_n^*$ as a statistic. As usual, N independent copies of T are computed and their empirical distribution is compared with the uniform via a KS test. We call this the *nearest-pair test*.

We can also define a test based on $W_{n,1}^*, \ldots, W_{n,m}^*$ for some fixed $m > 1$. These m values are supposed to be i.i.d. $U(0,1)$, so one possibility is to compare their empirical distribution to the uniform distribution with a KS test, which yields significance levels δ^+ and δ^-. This is replicated N times, and gives rise to two (correlated) tests, the first one using this δ^+ as its statistic T, the second one using δ^-. At the outer level, the empirical distribution of the N values of δ^+ are compared with the uniform distribution, and similarly for δ^-. We call this the *m-nearest-pairs test*.

These two tests are based on asymptotic results in n, which means that the errors Δ^+ and Δ^- should decrease with n, so the question arises of how large n should be for the error to be negligible. For the nearest-pair test, our experiments showed that Δ^+ is negligible (probably 0), but Δ^- is quite significant. For $p = 2$ and $n = 1000$, we obtained estimates $\hat{\Delta}^-$ equal to .02, .06, and .18 in dimensions 4, 6, and 9, respectively. In dimension 9, we still have $\Delta^- \approx .12$ for $n = 10000$. Similar results were obtained for $p = 1$ and $p = \infty$. This can be explained by the boundary effects: To prove Proposition 1, one assumes that for each i and "small" r, the number of points $\boldsymbol{U}_j$, $j \neq i$, in the ball of radius r centered at $\boldsymbol{U}_i$ is roughly Poisson with mean $(n-1)$ times the volume of that ball. But for points at distance less than r from the boundary, this is not true. As a result, W_n^* is stochastically larger than a $U(0,1)$. The fraction of the hypercube volume at distance more than r from the boundary is $(1-2r)^k$, which decreases to zero exponentially with k.

To eliminate the boundary effects, one can transform the hypercube $[0,1)^k$ into a torus, by identifying the opposite faces. In the torus, points facing each other on opposite sides or in opposite corners become close to each other. We found that in the torus, for $n = 1000$, the approximation error becomes practically negligible (less than 0.01) at least up to dimension 12. We obtained similar results for the m-nearest-pairs test.

5 Tests based on entropy

Let us write the binary expansion of each U_i as

$$U_i = .b_{i1}b_{i2}b_{i3}\ldots.$$

Extract the following bit sequence:

$$b_{1,r+1}, \ldots, b_{1,r+s}, b_{2,r+1}, \ldots, b_{2,r+s}, b_{3,r+1}, \ldots, b_{3,r+s}, \ldots, \tag{1.1}$$

where $r \geq 0$ and $s > 0$ are fixed integers. Extract from this sequence n adjacent and disjoint *blocks* of L bits each, thus constructing n integers $X_1, \ldots, X_n$. Under H_0, the X_i are i.i.d. uniform over $\{0, 1, \ldots, C-1\}$, where $C = 2^L$. For each x, let N_x be the number of X_i that are equal to x, and compute the *discrete empirical entropy*

$$\hat{H}_d(C, n) = -\sum_{x=0}^{C-1} (N_x/n) \log_2(N_x/n).$$

This is our statistic T. Repeat this N times, let $T_1, \ldots, T_N$ be the N values of $\hat{H}_d(C, n)$, and define the normalized values

$$S_i = \frac{T_i - E[T_i]}{\sqrt{\mathrm{Var}[T_i]}}.$$

Under H_0, we can obtain expressions for the exact mean and variance of $\hat{H}_d(C, n)$, and we also know that S_i converges to a $N(0, 1)$ as $n \to \infty$ and $C \to \infty$ simultaneously, while n/C converges to some positive constant (see [LCC96]). This justifies the *discrete entropy distribution test*, which compares the empirical distribution of the S_i to the $N(0, 1)$. Of course, the quality of the normal approximation is an issue. As for the nearest-pair test, we have estimated the approximation errors Δ^+ and Δ^-, and obtained (for example) values smaller than 0.01 for $n = 2^L \geq 256$.

One can also test for the independence of the entropy across successive blocks. The *discrete entropy correlation test* computes the sample correlation $\hat{\rho}_N$ between the pairs (S_i, S_{i+1}) and tests for significance. Here, we would rather take n small and N very large, in which case $\sqrt{N}\hat{\rho}_N$ is approximately $N(0, 1)$ under H_0.

To squeeze out more information from the sequence, one may also construct the blocks of bits with overlap. Relabel the first n bits of the sequence (1.1) as $b_1, b_2, \ldots, b_n$, and put them on a circle, thus defining $b_0 = b_n$ and $b_j = b_{j-n}$ for $j > n$. Let X_i be the integer whose binary representation is $b_i \cdots b_{i+L-1}$, and define C, N_x, $\hat{H}_d(C, n)$, $T_1, \ldots, T_N$, and $S_1, \ldots, S_N$ as before. The exact mean and variance of the T_i can be computed by brute force in time $O(2^n)$. The *overlapping average entropy test* computes the statistic

$$\frac{1}{\sqrt{N}} \sum_{i=1}^{N} S_i,$$

which is approximately $N(0, 1)$ for large N, under H_0. The *overlapping entropy correlation test* computes the sample correlation between the pairs (S_i, S_{i+1}) and tests for significance.

TABLE 1.2. List of selected generators.

G1.	LCG with $m = 2^{31} - 1$ and $a = 742938285$ (see [FM86]).
G2.	LCG with $m = 2^{31} - 1$ and $a = 630360016$ (see [LK91]).
G3.	LCG with $m = 2^{31} - 1$ and $a = 16807$ (see [BFS87, LK91]).
G4.	LCG with $m = 2^{32}$, $a = 69069$, and $c = 1$ (see [LK91, Rip90]).
G5.	LCG with $m = 2^{31}$ and $a = 65539$ (RANDU, see [LK91]).
G6.	LCG with $m = 2^{31}$ and $a = 452807053$ (see [DvdMST95]).
G7.	LCG with $m = 2^{31}$, $a = 1103515245$, $c = 12345$ (see [Pla92]).
G8.	Implicit inv. with $m = 2^{31} - 1$, $a_1 = a_2 = 1$ (see [Eic92]).
G9.	Explicit inv. with $m = 2^{31} - 1$, $a = b = 1$ (see [Eic92, Hel95]).
G10.	Implicit inv. with $m = 2^{32}$, $a = b = 1$, $z_0 = 5$ (see [EG92]).
G11.	Explicit inv. of [EI94] with $m = 2^{32}$, $a = 6$, $b = 1$.
G12.	Modified explicit inv. of [Eic96] with $m = 2^{32}$, $a = 6$, $b = 1$.
G13.	Combined LCG in Fig. 3 of [L'E88].
G14.	Combined MRG in Fig. 1 of [L'E96a].

6 Some test results

We selected an arbitrary set of generators previously proposed or widely used, and applied to them some instances of the tests described in the previous sections. The aim here is not to test all generators nor recommend specific ones, but just to illustrate the fact that many fail these tests. The generators are listed in Table 1.2.

G1 to G7 are linear congruential generators (LCG), based on the recurrence $x_i = (ax_{i-1} + c) \bmod m$, with output $u_i = x_i/m$, and well-known parameters. The next five are inversive generators modulo m, with output $u_i = z_i/m$, and where z_i obeys a nonlinear recurrence. G8 is an *implicit* inversive generator, with recurrence $z_i = (a_1 + a_2 z_{i-1}^{-1}) \bmod m$. G9 is an *explicit* inversive generator of the form $x_i = (ai + b) \bmod m$, $z_i = x_i^{-1} \bmod m = x_i^{m-2} \bmod m$. G10 is an implicit inversive generator based on the recurrence $z_i = T(z_{i-1})$ where $T(2^\ell z) = (a_1 + 2^\ell a_2 z^{-1}) \bmod 2^{32}$ for odd z. G11 and G12 are explicit inversive generators, the latter with the recurrence $z_i = i(ai + c)^{-1} \bmod 2^{32}$. G13 and G14 are two combined linear generators, with period lengths of approximately 2^{61} and 2^{185}, respectively.

We applied to these generators the nearest-pair test with the following parameters: $p = 2$ (Euclidean norm), $N = 20$ for $n = 10000, 100000$ and $k = 2, 4, 6, 9, 12$; then $N = 10$ with $n = 500000$ and $k = 2, 4, 6, 9$. We also applied the m-nearest-pairs test with the same parameters and $m = 10$. We just summarize the results here. For more details and more refined versions of these tests, see [LCS96].

Because of their well-known regular lattice structure [Knu81, L'E94, Rip87], we expected the LCGs G1 to G7 to fail the nearest-pair tests and they all did, spectacularly. Most of them passed the tests with $n = 10000$,

TABLE 1.3. Parameters for entropy distribution tests

n	L	r	s
2^{12}	12	0	12
2^{12}	12	0	4
2^{12}	12	20	4
2^{16}	8	0	8
2^{16}	8	0	4
2^{16}	8	20	4
2^{16}	16	0	16
2^{16}	16	0	4
2^{16}	16	20	4

TABLE 1.4. Parameters for the entropy tests with overlapping

n	r	s
10^4	0	30
10^5	0	30
10^6	0	30
10^7	0	30
10^4	20	3
10^5	20	3
10^6	20	3
10^7	20	3

but for larger n, they all failed, in most cases with significance levels less than 10^{-10}. When the points are very evenly distributed over the entire period of a generator, then (as a general rule) the period should be several orders of magnitude larger than the number of values used. This is not the case with these LCGs: Their period length is too small. The inversive generators, on the other hand, do not have their points so regularly spaced; They look more randomly scattered even over the entire period length. They all did well in the tests (all passed). The two combined generators G13 and G14 also passed.

For the entropy tests, we selected the parameters of Table 1.3 with $N = 1000$ for the non-overlapping discrete entropy distribution test, while for the overlapping average and correlation tests, we used the parameters in Table 1.4 with $n = 30$ and $L = 5$.

The results can be summarized as follows. All the generators with power-of-two modulus, namely G4, G5, G6, G7, G10, G11, G12, failed the entropy distribution tests with $r = 20$, and also failed most of the overlapping average entropy with $r = 20$ and many entropy correlation tests, at significance level usually less than 10^{-10}. Those tests with $r = 20$ look at less significant bits of the generators, and it is thus no surprise that those with power-of-two modulus (linear or inversive) fail, because their least significant bits have short period length. G5 and G6 also fail most of the tests based on the most significant bits (with $r = 0$). Other suspicious values with $r = 0$ were observed for the entropy distribution test for G1 and G10 (significance levels near 0.001), and for the overlapping average entropy tests for G4 and G9. For the overlapping entropy correlation tests with the first four parameter sets ($r = 0$), all generators with power-of-two modulus failed some of those tests. G9 also failed the first two overlapping entropy correlation tests with significance levels less than 10^{-10}. The explanation is that the first n values produced by G9 are the inverses (modulo $2^{31}-1$) of the first n positive integers, and with this modulus, the inverses of small integers tend to have low entropy in their high-order bits. In fact, we tried changing the parameter b of G9 to some arbitrary values larger than 10^5, and it passed the tests in all cases. Further discussion and results of entropy tests can be found in [LCC96].

7 Conclusion

Random number generators should be designed based on sound theoretical analysis, and then be tested empirically. The author believes that generators which fail simple tests such as the ones presented in this paper should be discarded. Only generators G8, G13, and G14 passed all the tests reported in this paper. Of course, many other generators not considered here would also pass.

Acknowledgments: This work has been supported by NSERC-Canada grants # OGP0110050 and # SMF0169893, and FCAR-Québec grant # 93-ER-1654. I wish to thank Peter Hellekalek and Harald Niederreiter for the organization of this wonderful MC& QMC'96 conference in Salzburg. I also thank At Compagner, Jean-François Cordeau, and Richard Simard, who contributed to the ideas and implementations of the tests described in this paper.

8 References

[BBS86] L. Blum, M. Blum, and M. Schub. A simple unpredictable pseudo-random number generator. *SIAM Journal on Computing*, 15(2):364–383, 1986.

[BFS87] P. Bratley, B. L. Fox, and L. E. Schrage. *A Guide to Simulation*. Springer-Verlag, New York, second edition, 1987.

[Com91] A. Compagner. The hierarchy of correlations in random binary sequences. *Journal of Statistical Physics*, 63:883–896, 1991.

[Com95] A. Compagner. Operational conditions for random number generation. *Physical Review E*, 52(5-B):5634–5645, 1995.

[Dur73] J. Durbin. *Distribution Theory for Tests Based on the Sample Distribution Function*, volume 9 of *SIAM CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, 1973.

[DvdMST95] E. J. Dudewicz, E. C. van der Meulen, M. G. SriRam, and N. K. W. Teoh. Entropy-based random number evaluation. *American Journal of Mathematical and Management Sciences*, 15:115–153, 1995.

[EG92] J. Eichenauer-Herrmann and H. Grothe. A new inversive congruential pseudorandom number generator with power of two modulus. *ACM Transactions on Modeling and Computer Simulation*, 2(1):1–11, 1992.

[EI94] J. Eichenauer-Herrmann and K. Ickstadt. Explicit inversive congruential pseudorandom numbers with power of two modulus. *Mathematics of Computation*, 62(206):787–797, 1994.

[Eic92] J. Eichenauer-Herrmann. Inversive congruential pseudorandom numbers: A tutorial. *International Statistical Reviews*, 60:167–176, 1992.

[Eic96] J. Eichenauer-Herrmann. Modified explicit inversive congruential pseudorandom numbers with power-of-two modulus. *Statistics and Computing*, 6:31–36, 1996.

[FM86] G. S. Fishman and L. S. Moore III. An exhaustive analysis of multiplicative congruential random number generators with modulus $2^{31} - 1$. *SIAM Journal on Scientific and Statistical Computing*, 7(1):24–45, 1986.

[Hel95] P. Hellekalek. Inversive pseudorandom number generators: Concepts, results, and links. In C. Alexopoulos, K. Kang, W. R. Lilegdon, and D. Goldsman, editors, *Proceedings of the 1995 Winter Simulation Conference*, pages 255–262. IEEE Press, 1995.

[Knu81] D. E. Knuth. *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*. Addison-Wesley, Reading, Mass., second edition, 1981.

[Lag93] J. C. Lagarias. Pseudorandom numbers. *Statistical Science*, 8(1):31–39, 1993.

[LCC96] P. L'Ecuyer, A. Compagner, and J.-F. Cordeau. Entropy-based tests for random number generators. Submitted. Also GERAD technical report number G-96-41, 1996.

[LCS96] P. L'Ecuyer, J.-F. Cordeau, and R. Simard. Close-neighbor tests for random number generators. In preparation, 1996.

[L'E88] P. L'Ecuyer. Efficient and portable combined random number generators. *Communications of the ACM*, 31(6):742–749 and 774, 1988. See also the correspondence in the same journal, 32, 8 (1989) 1019–1024.

[L'E92] P. L'Ecuyer. Testing random number generators. In *Proceedings of the 1992 Winter Simulation Conference*, pages 305–313. IEEE Press, Dec 1992.

[L'E94] P. L'Ecuyer. Uniform random number generation. *Annals of Operations Research*, 53:77–120, 1994.

[L'E96a] P. L'Ecuyer. Combined multiple recursive generators. *Operations Research*, 44(5):816–822, 1996.

[L'E96b] P. L'Ecuyer. TestU01: Un logiciel pour appliquer des tests statistiques à des générateurs de valeurs aléatoires. In preparation, 1996.

[L'E97] P. L'Ecuyer. Random number generation. In Jerry Banks, editor, *Handbook on Simulation*. Wiley, 1997. To appear. Also GERAD technical report number G-96-38.

[LK91] A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, New York, second edition, 1991.

[LP89] P. L'Ecuyer and R. Proulx. About polynomial-time "unpredictable" generators. In *Proceedings of the 1989 Winter Simulation Conference*, pages 467–476. IEEE Press, Dec 1989.

[LW97] H. Leeb and S. Wegenkittl. Inversive and linear congruential pseudorandom number generators in selected empirical tests. *ACM Transactions on Modeling and Computer Simulation*, 1997. To appear.

[Mar85] G. Marsaglia. A current view of random number generators. In *Computer Science and Statistics, Sixteenth Symposium on the Interface*, pages 3–10, North-Holland, Amsterdam, 1985. Elsevier Science Publishers.

[Mar96] G. Marsaglia. Diehard: A battery of tests of randomness. `http://stat.fsu.edu/~geo/diehard.html`, 1996.

[Nie92] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*, volume 63 of *SIAM CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, 1992.

[Pla92] P. J. Plauger. *The Standard C Library*. Prentice Hall, Englewood Cliffs, New Jersey, 1992.

[Rip87] B. D. Ripley. *Stochastic Simulation*. Wiley, New York, 1987.

[Rip90] B. D. Ripley. Thoughts on pseudorandom number generators. *Journal of Computational and Applied Mathematics*, 31:153–163, 1990.

[RS78] B. D. Ripley and B. W. Silverman. Quick tests for spatial interaction. *Biometrika*, 65(3):641–642, 1978.

[SB78] B. Silverman and T. Brown. Short distances, flat triangles and Poisson limits. *Journal of Applied Probability*, 15:815–825, 1978.

[Ste86a] M. S. Stephens. Tests based on EDF statistics. In R. B. D'Agostino and M. S. Stephens, editors, *Goodness-of-Fit Techniques*. Marcel Dekker, New York and Basel, 1986.

[Ste86b] M. S. Stephens. Tests for the uniform distribution. In R. B. D'Agostino and M. S. Stephens, editors, *Goodness-of-Fit Techniques*, pages 331–366. Marcel Dekker, New York and Basel, 1986.

Pierre L'Ecuyer
Département d'informatique et de recherche opérationnelle,
Université de Montréal,
C.P. 6128, Succ. Centre-Ville, Montréal, H3C 3J7, Canada
`http://www.iro.umontreal.ca/~lecuyer`