On the digit test

Hannes Leeb *

This article was published in:
Tagungsband zum 1. Salzburger Minisymposium über
Pseudozufallszahlen und Quasi-Monte Carlo Methoden
am 18. Nov. 1994,
P. Hellekalek, G. Larcher, and P. Zinterhof editors,
Mathematics Institute, University of Salzburg, 1995

Abstract

In a set of stochastic simulations, which we collectively call the *digit test*, we compare the widely used linear congruential with the new inversive random number generators. The inversive generators are found to perform always at least as good as any of the linear congruential generators; in some simulation runs, they perform significantly better.

Contents

[*] Research project supported by the Austrian Science Foundation, no. FWF P9285

1 Introduction

A random number generator is a deterministic algorithm which produces a sequence $(x_n)_{n=0}^{N-1}$ of numbers[1] in $[0,1[$. These numbers should behave like a realization of the sequence $(X_n)_{n=0}^{N-1}$ of independent, on $[0,1[$ equidistributed random variables.

The linear congruential generator (due to Lehmer [23]; see also [31] and [32]) is the most widely used and the best known generator today. A linear congruential generator with integer parameters M, a, b, and u_0 (LCG(M, a, b, u_0), for short) produces a sequence $(u_n)_{n=0}^{N-1}$ of integers by $u_{n+1} := au_n + b \bmod M$, i.e. u_{n+1} is the integer remainder of dividing $au_n + b$ by M. A sequence $(x_n)_{n=0}^{N-1}$ of numbers in $[0,1[$ is obtained by setting $x_n := u_n/M$. The LCG's period is at most M and depends on the choice of parameters; see [20, Section 3.2] or [28, p.169].
There are two types of inversive generators, which were recently proposed by Eichenauer-Hermann and Lehn in [9] and Eichenauer-Hermann in [12], respectively. With integer parameters p (prime), a, b, and u_0, an inversive congruential generator ICG(p, a, b, u_0) produces a sequence $(u_n)_{n=0}^{N-1}$ of integers by $u_{n+1} := a\overline{u_n} + b \bmod p$, and a sequence $(x_n)_{n=0}^{N-1}$ by $x_n := u_n/p$; the number $\overline{u_n}$ is defined to be the multiplicative inverse[2] of u_n in $\mathbf{Z}_p$ if $u_n \neq 0 \bmod p$, and 0 otherwise. The ICG's period N is at most p and depends on the choice of parameters[3].
Similar to the ICG, an explicit inversive generator EICG(p, a, b, n_0) is defined by integer parameters p (prime), $a \neq 0 \bmod p$, b, and n_0. It produces an integer sequence $(u_n)_{n=0}^{N-1}$ by $u_n := \overline{a(n_0 + n) + b} \bmod p$, and a sequence $(x_n)_{n=0}^{N-1}$ in $[0,1[$ by $x_n := u_n/p$. The period N equals p, for any choice of parameters as above.
There is a variety of number-theoretical results which indicate that inversive generators have very attractive statistical properties. For example, their ouput lacks just those regular structures which are inherent to the linear congruential generators (see Section 5 and the references given there).

While linear congruential generators were extensively tested in practice (see [1], [4], [13], [14], [15], [17], [21], [24], [26], and [33]), no rigorous empirical analysis of inversive generators has yet been done. This paper is a first step to change this situation. We set out to compare linear and inversive generators with a statistical test, the digit test, and to find empirical evidence of the inversive generators' theoretical advantages.

2 On the relevance of the digit test

Ranking random number generators by means of statistical tests is a somewhat unrewarding business: these tests are termed 'statistical', because they always incorporate a small but positive probability of making the wrong decision. Therefore, the quality of

[1] Actually, virtually all random number generators used for computer simulation produce infinite, *purely periodic* sequences of some period N. If the algorithm is used to produce more than N numbers, it gives $x_N = x_0, x_{N+1} = x_1, \ldots$.

[2] Since p is prime, the set $\mathbf{Z}_p$ of residues modulo p is a finite field with addition and multiplication defined modulo p. Hence, for each $u \in \mathbf{Z}_p \setminus \{0\}$, there is a uniquely defined u^{-1} such that $uu^{-1} = 1$ in $\mathbf{Z}_p$.

[3] See [16] for a formal condition on the parameters to achieve the maximum period p, [18] for an algorithm to find such parameters, and [19] for tables of parameters.

a random number generator can never be proven by such tests. In fact, since random numbers are expected to behave random, they should fail statistical tests at a certain rate[4]. Finally, even a miserable failure of a generator in one statistical test does not render it useless. Since virtually every real-world Monte Carlo model[5] is simulated more than once, using different parameters, varying boundary conditions, etc., tail-enders will be detected and sorted out.

However, statistical defects can be disastrous if they emerge not just once in a while, but with a certain regular pattern. Suppose we are ignorant of elementary probability theory. Further, suppose we simulate a variety of strategies for playing roulette, searching for the one to make a fortune. Finally, suppose we perform our simulations using a defective random number generator which exhibits a consistent gain if only we play at least 10 rounds, betting at least $100,- in each. If we think the simulated results are correct, and if we apply our strategy, we are likely to loose quite a lot of money.
In the following, we show by example that such a situation can in fact occur if the simulation is solely based on the widely used linear congruential random number generators. Instead of simulating roulette strategies, compute the digit test. Since the new inversive random number generators produce satisfactory results, we recommend that they should be used for verification.

The digit test comes in two flavours: the basic and the extended digit test. The basic digit test simulates a random variable T_1 which is equidistributed on $[0,1[$. The extended digit test simulates a random variable T_2 whose distribution is described by the so-called Kolmogorov-Smirnov distribution function[6]; in particular, T_2 is rather likely to be small: $P(T_2 \leq 1.63) = 0.99$.

For a fixed random number generator producing the sequence $(x_n)_{n=0}^{N-1}$, we use yet to be specified integer parameters s, k, and l, called dimension, block-length, and block-start, respectively, to compute a simulation $t_1(s,k,l)$ of T_1 and a simulation $t_2(s,k,l)$ of T_2. Simulating T_1 and T_2 with the standard ANSI C random number generator[7] for $s=2$, $k \in \{1,5,9,\ldots,21\}$ and $l \in \{1,2,\ldots,10\}$ yields the following[8]:

[4] This rate is usually called the level of significance of the statistical test.

[5] By 'real-world' Monte Carlo models, we mean quantities whose stochastic behavior is – in contrary to that of statistical tests – not completely known in advance; examples include problems such as the simulation of particle depositions in bifurcations of the human lung [3] or predator/prey populations [34].

[6] A definition and critical values for the Kolmogorov-Smirnov distribution function are given in [5, Section 6.11 – 6.14].

[7] The `rand()` function in ANSI C implements a linear congruential generator; see Section 3 below.

[8] In this and the following pictures, the values of $t_2(s,k,l)$ are truncated to $\min\{t_2(s,k,l),2\}$ to keep the graphics in scale.

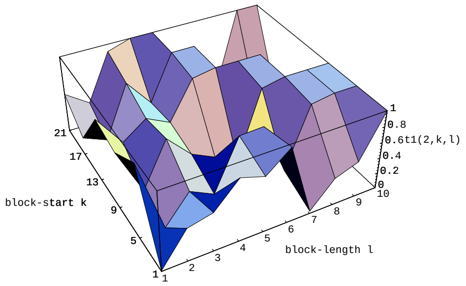


Figure 1a

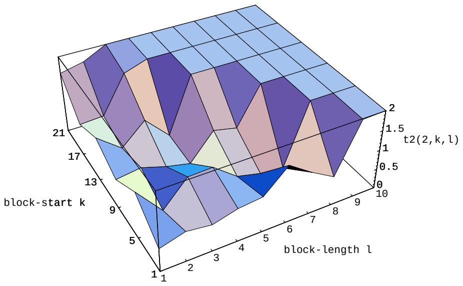


Figure 1b

With the ANSI C generator, the simulations of T_1 apparently exhibit a strong tendency to extremely large or extremely small values; likewise, the simulations of T_2 tend to give very improbable, large values. As we shall see below, the simulations for varying s, k, and l are not independent; anyway, this generator seems to be inadequate for simulating the digit test: if we did not know the expected behavior of T_1 and T_2, the simulation results could lure us to erroneous conclusions.

3 Computing the digit test

In this section, let $(x_n)_{n=0}^{N-1}$ be the sequence produced by a random number generator, and let s, k, and l be fixed, positive integers.

A simulation of the basic digit test T_1 is obtained, basically, by applying a χ^2-test to the non-overlapping s-tuples $\mathbf{x}_n := (x_{ns}, \ldots, x_{ns+s-1})$: the unit cube $[0,1[^s$ is partitioned in $b := 2^{sl}$ bins $B_0, \ldots, B_{b-1}$, each of equal size $1/b$. From a sample of $6b$ points $\mathbf{x}_0, \ldots, \mathbf{x}_{6b-1}$, the corresponding χ^2-statistic χ is computed. With this, we set $t_1(s,k,l) := 1 - G(\chi)$, where G is the distribution function of the χ^2-distribution with $b-1$ degrees of freedom. It is easy to see that $t_1(s,k,l)$ has the desired properties: if the χ^2-statistic χ would depend not on $s \times 6b$ numbers but instead on $s \times 6b$ independent random variables, each equidistributed on $[0,1[$, then $1 - G(\chi)$ would be asymptotically equidistributed on $[0,1[$; this approximation is traditionally considered adequate if, as in our case, the expected number of hits per bin is at least 5.

As implied by its name, the extended digit test T_2 is simulated by repeadedly computing K simulations[9] $t_1^{(0)}(s,k,l), \ldots, t_1^{(K-1)}(s,k,l)$ of T_1. The empirical distribution function F_K of these values is defined as

$$F_K(t) := \frac{1}{K} \# \left\{ i \in \{0, \ldots, K-1\} : \ t_1^{(i)}(s,k,l) \le t \right\}.$$

The empirical distribution F_K of the sample is compared to the desired, uniform distribution with respect to the sup-norm by

$$D_K := \sqrt{K} \sup_{0 \le t < 1} |F_K(t) - t| \,.$$

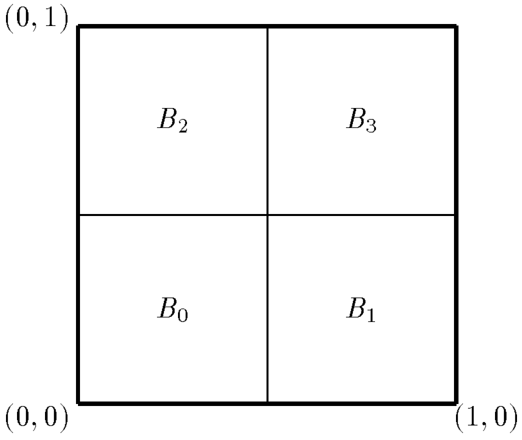
[9] Each $t_1^{(i)}(s,k,l)$ is computed from the sample $\{\mathbf{x}_{6bi}, \mathbf{x}_{6bi+1} \ldots, \mathbf{x}_{6bi+6b-1}\}$.

We set $t_2(s,k,l) := D_{64}$. To see that this yields a useful simulation of T_2, observe the following: if D_K is computed not from the K values $t_1^{(i)}(s,k,l)$ but from K independent realizations of T_1, then, for $K \to \infty$, the distribution function of D_K converges to the Kolmogorov-Smirnov distribution function. For $K > 40$, the approximation is fairly accurate.

So far, our simulations coincide with those already studied by Fishman and Moore [14, Test H_1, H_2, and H_3], Knuth [20, §3.3.2, Test A and B], or L'Ecuyer [21, Test (1) – (5)]. The crucial feature and the reason for naming them 'digit test' is the choice of the bins B_m $(0 \leq m < b)$. Whether a given s-tuple is contained in B_m depends on the binary representation[10] of its coordinates or, more accurately, on their k-th to their $(k+l-1)$-th digits:

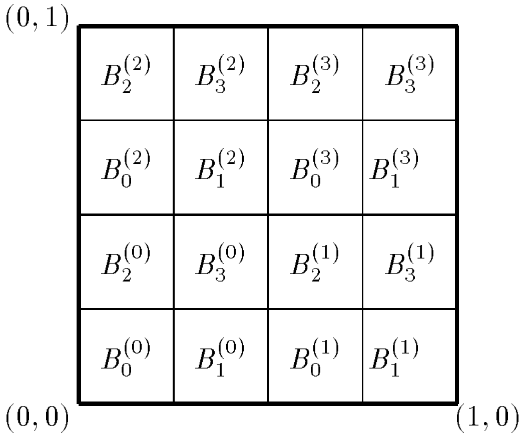
$$B_m := \left\{ \left(\sum_{j \geq 1} b_j^{(i)} 2^{-j} \right)_{i=0}^{s-1} \in [0,1[^s\colon\ m = \sum_{i=0}^{s-1} 2^{li} \left(\sum_{j=0}^{l-1} b_{k+j}^{(i)} 2^j \right) \right\}$$

for $0 \leq m < b = 2^{sl}$. The block-start k determines the first relevant digit, and the block-length l determines the number of relevant digits. To get an idea of the resulting B_m, consider the following example of two partitions in dimension $s = 2$ with block-length $l = 1$ and block-start $k = 1$ and $k = 2$, respectively:



$s = 2, k = 1, l = 1$

Figure 2a



$s = 2, k = 2, l = 1$

Figure 2b

In Figure 2a, B_1 consists of all points whose coordinates have the binary representation $(0.1\Box\Box\ldots, 0.0\Box\Box\ldots)$, where each '□' can be either 0 or 1.

In Figure 2b, B_1 is the union of the $B_1^{(i)}$, $(i = 0, \ldots, 3)$, and it consists of all points of

[10] The binary representation of $x \in [0,1[$ is not unique if $x = \sum_{i=1}^{n-1} b_i 2^{-i} + 2^{-n} = \sum_{i=1}^{n-1} b_i 2^{-i} + \sum_{i>n} 2^{-i}$ $(b_i \in \{0,1\})$. To avoid ambiguity in this case, we adopt the usual convention of representing such x by the finite sum only, i.e. with $b_i \neq 1$ for infinitely many i.

the form $(0.\square 1\square\ldots, 0.\square 0\square\ldots)$. Note that each quarter of the unit square in Figure 2b looks like a copy of Figure 2a.

In general, it is easy to see that the B_m partition $[0,1[^s$. Moreover, each B_m is the disjoint union of $2^{s(k-1)}$ half-open cubes of volume $1/2^{s(k+l-1)}$. Hence, the volume of each B_m is $1/2^{sl} = 1/b$.

With this, $t_1(s,k,l)$ and $t_2(s,k,l)$ are well defined. We have to stress that two simulation $t_1(s,k,l)$ and $t_1(s',k',l')$ of T_1 are, in general, *not independent*: the corresponding digit-blocks may overlap. However, independence holds for $k+l \leq k'$ or $k'+l' \leq k$. The same applies to simulations of T_2.

In our calculations, either s was kept fixed while k and l were varied, or l was fixed while s and k were varied. The basic and the extended digit test were computed for the values below (for parameters in parentheses, only T_1 was simulated):

digit test parameters		
dimension s	block-start k	block-length l
1	$1, 5, 9, \ldots, 21$	$1, 2, \ldots, 10$
2	$1, 5, 9, \ldots, 21$	$1, 2, \ldots, 10$
3	$1, 5, 9, \ldots, 21$	$1, 2, \ldots, 7$
4	$1, 5, 9, \ldots, 21$	$1, 2, \ldots, 5$
5	$1, 5, 9, \ldots, 21$	$1, 2, \ldots, 4$
6	$1, 5, 9, \ldots, 21$	$1, 2, 3$
(7)	$(1), (5), (9), \ldots, (21)$	$(1), (2), (3)$
(8)	$(1), (5), (9), \ldots, (21)$	$(1), (2)$
$1, 2, \ldots, 10, (11)$	$1, 5, 9, \ldots, 29$	2

Table 1

The simulations were performed with following generators:

generators			
nickname	generator	period	$1/\nu_3$
`randu`	$\mathrm{LCG}(2^{31}, 65539, 0, 1)$	2^{29}	0.0920575
`ansi`	$\mathrm{LCG}(2^{31}, 1103515245, 12345, 12345)$	2^{31}	0.00132673
`std`	$\mathrm{LCG}(2^{31}-1, 16807, 0, 1)$	$2^{31}-2$	0.00156518
`fish`	$\mathrm{LCG}(2^{31}-1, 950706376, 0, 1)$	$2^{31}-2$	0.000768506
`eicg1`	$\mathrm{EICG}(2^{31}-1, 1, 0, 0)$	$2^{31}-1$	
`icg`	$\mathrm{ICG}(2^{31}-1, 1, 1, 0)$	$2^{31}-1$	

Table 2

For each line in Table 1 and each generator in Table 2, we produced graphics similar to Figure 1.

The above LCGs were chosen to cover the wide variation of quality obtainable by linear congruential generators, ranging from `randu` (worst) to `fish` (best). `randu`, former part of IBM's Scientific Subroutine Package, exhibits abominable defects in three

dimensions; for $s = 3$, its points $\mathbf{x}_n$ all lie on just fifteen planes in the unit cube. `ansi` was chosen because it is the default random number generator `rand()` in the widely-used ANSI C programming language (BSD version[11]). `std` was proposed as a 'minimal standard' generator by Park and Miller [29]. Finally, `fish` is one of the best found by Fishman and Moore [15] in an exhaustive search among all maximum period LCGs with $M = 2^{31} - 1$ and $b = 0$.
The inversive generators were chosen arbitrarily because no significant differences, neither theoretically nor empirically, among full period inversive generators are known. There is no known way to choose 'good' parameters for the EICG and ICG, except for the obvious choice of parameters which give the largest possible period. Note that the sequence of EICG$(2^{31} - 1, k, 0, 0)$ is obtained by selecting every k-th number produced by `eicg1`. Therefore, EICG$(2^{31} - 1, k, 0, 0)$ correspond to *subsequences* of `eicg1` of stride k.

4 Selected results

The simulations produced by linear congruential generators all are more or less similar to those in Figure 1: for sufficiently large values of s, k, or l, these generators show a suspicious tendency to extreme values in the simulation. The inversive generators, on the other hand, produce results which coincide with the expected behavior of T_1 and T_2.

Here, we only show the results for $s = 3$ and varying k and l. The complete collection of results, as well as the source code of the generators is available on the internet [2].

`randu` for $s = 3$, $k = 1, 5, 9, \ldots, 21$ and $l = 1, 2, \ldots, 7$:

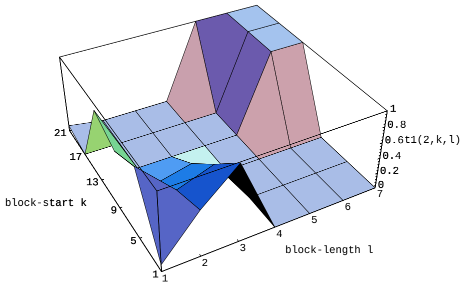


Figure 3a

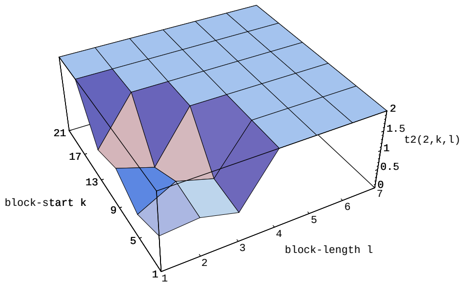


Figure 3b

[11] Note that some versions of the unix online-manual incorrectly state the period of this generator to be 2^{32}.

`ansi` for $s = 3$, $k = 1, 5, 9, \ldots, 21$ and $l = 1, 2, \ldots, 7$:

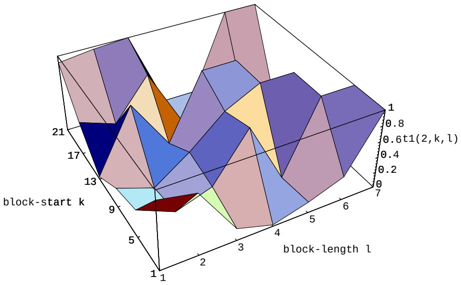


Figure 4a

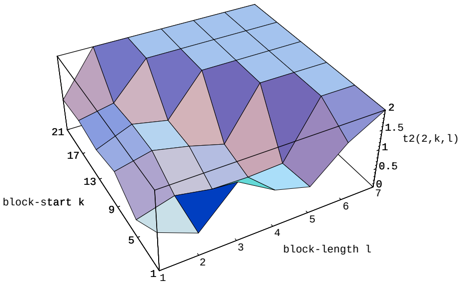


Figure 4b

`std` for $s = 3$, $k = 1, 5, 9, \ldots, 21$ and $l = 1, 2, \ldots, 7$:

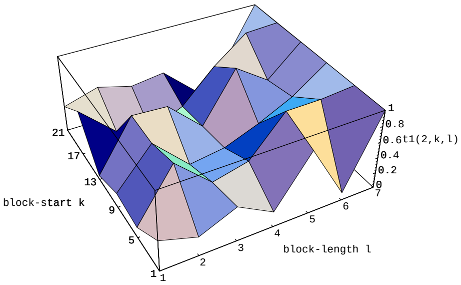


Figure 5a

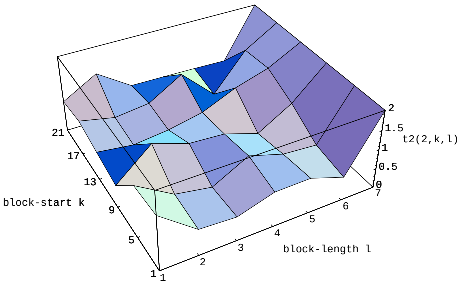


Figure 5b

`fish` for $s = 3$, $k = 1, 5, 9, \ldots, 21$ and $l = 1, 2, \ldots, 7$:

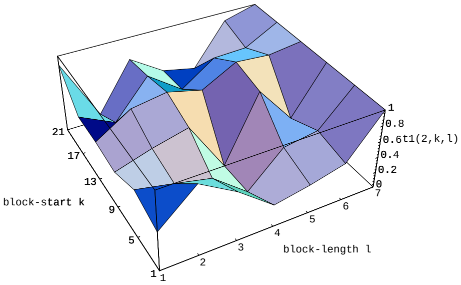


Figure 6a

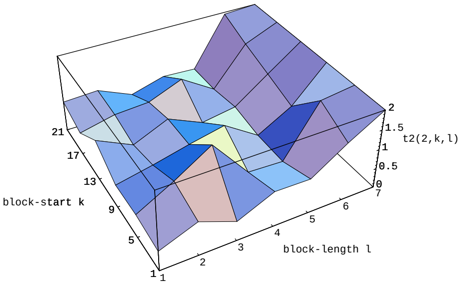


Figure 6b

`eicg1` for $s = 3$, $k = 1, 5, 9, \ldots, 21$ and $l = 1, 2, \ldots, 7$:

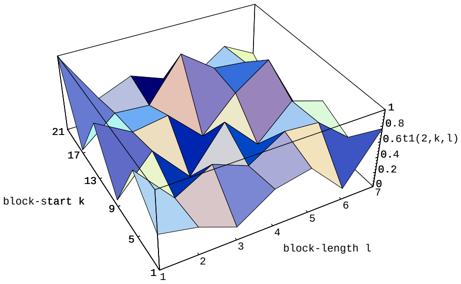


Figure 7a

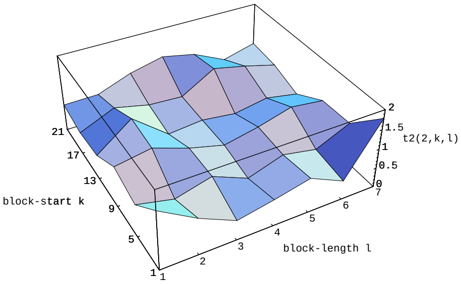


Figure 7b

`icg` for $s = 3$, $k = 1, 5, 9, \ldots, 21$ and $l = 1, 2, \ldots, 7$:

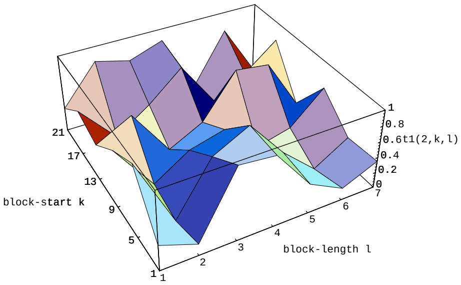


Figure 8a

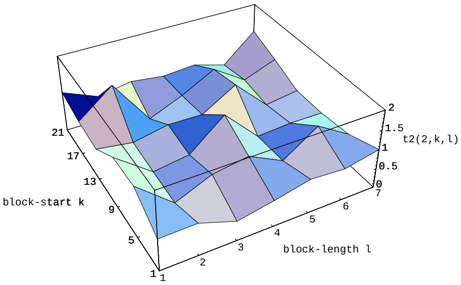


Figure 8b

5 Interpretation

Apparently, the digit test is sensible to some defect which even the best LCG (`fish`) exhibits for some simulation parameters, and which is absent in the inversive generators. We conjecture the digit test is sensible on *grid structures* or *long-range correlations*.

The first reason for this conjecture is that grid structures and certain kinds of long-range correlations are inherent to all linear and absent in all inversive generators[12]. As Marsaglia discovered in [25], the s-tuples $\mathbf{x}_n$ $(0 \leq n < N)$ produced by an LCG "will be found to lie in a relatively small number of parallel hyperplanes." Moreover, these points were shown to have a lattice-like structure [30]. As a consequence, the points (x_n, x_{n+c}) $(0 \leq n < N)$ produced by an LCG lie on a relatively small number of parallel lines, too (see [8], [11], and [22, Chapter 5 and Appendix B]). For some 'shifts' c, the number of lines can be very small and the (x_n, x_{n+c}) can be highly correlated ([4], [7]). On the other hand, inversive generators lack just these defects. The points $\mathbf{x}_n$ $(0 \leq$

[12] This and the following statements only apply to full or maximum period LCGs as described in [28, p.169] or [30], and to full period ICGs and EICGs.

$n < N$) produced by an ICG or EICG literally avoid s-dimensional hyperplanes (due to [10], [27]; see also [22, Chapter 5]). And, for reasonable shifts c, the points (x_n, x_{n+c}) $(0 \leq n < N)$ obtained from these generators also avoid the lines in two dimensions, so they cannot exhibit long-range correlations as the LCG does (see [22, Chapter 5]).

The second reason for this conjecture is that the LCGs' behavior in the digit test appears to correspond to the coarseness of their grid. Let $1/\nu_s$ be the maximal distance of parallel hyperplanes covering an LCG's s-dimensional points $\mathbf{x}_n$ $(0 \leq n < N)$. The so-called spectral test $1/\nu_s$, introduced by Coveyou and MacPherson [6], is a widely-used figure of merit to find 'good' LCG (where small values of $1/\nu_s$ are preferred). In the exhaustive search of Fishman and Moore [15] among over 534 million LCGs, the 414 best with respect to the spectral test in dimension $s = 2, \ldots, 6$ were selected, and `fish` is one of these. For the LCGs we studied here, the value $1/\nu_3$ of the spectral test in three dimensions can be obtained from Table 2. Comparing these values to the behavior of the LCGs in Section 4, we find that larger values of $1/\nu_s$, i.e. coarser grids, correspond to 'worse' simulation results.

6 Conclusion

The digit test shows the existence of stochastic models which, when simulated by LCGs, give erroneous results in some computations. Since these errors appear not just once in a while but as regular *tendencies*, they can lead to false conclusions about the stochastic model.
On the other hand, the results show that concerning the digit test, inversive generators are always at least as good as the linear ones.

Of course: the fact that the LCGs perform 'bad' in the digit test does not necessarily imply they will be 'bad' in another (say, your) simulation problem. Likewise, the inversive generators are not necessarily 'good'. Anyhow, our results indicate that caution is necessary, especially if the simulation problem is suspected to be sensible on grid structures or long-range correlations. In this case, inversive generators should be used to verify LCG-based results.

Author's address:

Hannes Leeb
Institut für Mathematik
Universität Salzburg
Hellbrunnerstr. 34
A-5020 Salzburg
Austria
e-mail address: `leeb@random.mat.sbg.ac.at`

References

[1] S.L. Anderson. Random number generators on vector supercomputers and other advanced architectures. *SIAM Rev.*, **32**:221–251, 1990.

[2] T. Auer, K. Entacher, P. Hellekalek, H. Leeb, O. Lendl, and S. Wegenkittl. The PLAB www-server. `http://random.mat.sbg.ac.at`. Also accessible via ftp.

[3] I. Balásházy and W. Hofmann. *Particle deposition in airway bifurcations for inspiratory flow. Part II: calculations of particle trajectories and deposition patterns.* Hungarian Academy of Sciences, Central Research Institute for Physics, Budapest, 1992.

[4] K.O. Bowman and M.T. Robinson. Studies of random number generators for parallel processing. In M.T. Heath, editor, *Proc. Second Conference on Hypercube Multiprocessors*, pages 445–453, Philadelphia, 1987. SIAM.

[5] K.V. Bury. *Statistical Models in Applied Science.* Robert E. Krieger Publishing Company, Inc., Malabar, Florida, reprint edition, 1986.

[6] R.R. Coveyou and R.D. MacPherson. Fourier analysis of uniform random number generators. *J. Assoc. Comp. Mach.*, **14**:100–119, 1967.

[7] A. De Matteis, J. Eichenauer-Hermann, and H. Grothe. Computation of critical distances within multiplicative congruential pseudorandom number sequences. *J. Comp. Appl. Math.*, **39**:49–55, 1992.

[8] A. De Matteis and S. Pagnutti. Parallelization of random number generators and long-range correlations. *Numer. Math.*, **53**:595–608, 1988.

[9] J. Eichenauer and J. Lehn. A non-linear congruential pseudo random number generator. *Statist. Papers*, **27**:315–326, 1986.

[10] J. Eichenauer-Hermann. Inversive congruential pseudorandom numbers avoid the planes. *Math. Comp.*, **56**:297–301, 1991.

[11] J. Eichenauer-Hermann and H. Grothe. A remark on long-range correlations in multiplicative congruential pseudo random number generators. *Numer. Math.*, **56**:609–611, 1989.

[12] J. Eichenauer-Herrmann. Statistical independence of a new class of inversive congruential pseudorandom numbers. *Math. Comp.*, **60**:375–384, 1993.

[13] G.S. Fishman. Multiplicative congruential random number generators with modulus 2^β: an exhaustive analysis for $\beta = 32$ and a partial analysis for $\beta = 48$. *Math. Comp.*, **54**:331–344, 1990.

[14] G.S. Fishman and L.R. Moore. A statistical evaluation of multiplicative congruential random number generators with modulus $2^{31} - 1$. *J. Amer. Statist. Assoc.*, **77**:129–136, 1982.

[15] G.S. Fishman and L.R. Moore. An exhaustive analysis of multiplicative congruential random number generators with modulus $2^{31} - 1$. *SIAM J. Sci. Statist. Comput.*, **7**:24–45, 1986.

[16] M. Flahive and H. Niederreiter. On inversive congruential generators for pseudorandom numbers. In G.L. Mullen and Shiue P.J.-S., editors, *Finite Fields, Coding Theory, and Advances in Communications and Computing*, pages 75–80, New York, 1992. Dekker.

[17] M. Fushimi. Random number generation with the recursion $X_t = X_{t-3p} \oplus X_{t-3q}$. *J. Comp. Appl. Math.*, **31**:105–118, 1990.

[18] P. Hellekalek. Study of algorithms for primitive polynomials. Report D5H-1, CEI-PACT Project, WP5.1.2.1.2, Research Institute for Software Technology, University of Salzburg, Austria, 1994. Available on the internet at `http://random.mat.sbg.ac.at`.

[19] P. Hellekalek, M. Mayer, and A. Weingartner. Implementation of algorithms for IMP-polynomials. Report D5H-2, CEI-PACT Project, WP5.1.2.1.2, Research Institute for Software Technology, University of Salzburg, Austria, 1994. Available on the internet at `http://random.mat.sbg.ac.at`.

[20] D.E. Knuth. *The Art of Computer Programming*, volume 2: Seminumerical Algorithms. Addison-Wesley, Reading, MA, 2nd edition, 1981.

[21] P. L'Ecuyer. Efficient and portable combined random number generators. *Comm. ACM*, **31**:742–774, 1988.

[22] H. Leeb. Random numbers for computer simulation. Master's thesis, University of Salzburg, 1995. Available on the internet at `http://random.mat.sbg.ac.at`.

[23] D.H. Lehmer. Mathematical methods in large-scale computing units. In *Proc. 2nd Sympos. on Large-Scale Digital Calculating Machinery, Cambridge, MA, 1949*, pages 141–146, Cambridge, MA, 1951. Havard University Press.

[24] N.M. MacLaren. A limit on the usable length of a pseudorandom sequence. *J. Statist. Comput. Simul.*, **42**:47–54, 1992.

[25] G. Marsaglia. Random numbers fall mainly in the planes. *Proc. Nat. Acad. Sci. USA*, **61**:25–28, 1968.

[26] G. Marsaglia. A current view of random number generators. In L. Billard, editor, *Computer Science and Statistics: The Interface*, pages 3–10, Amsterdam, 1985. Elsevier Science Publishers B.V.

[27] H. Niederreiter. On a new class of pseudorandom numbers for simulation methods. *J. Comput. Appl. Math.* To appear.

[28] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods.* SIAM, Philadelphia, 1992.

[29] S.K. Park and K.W. Miller. Random number generators: good ones are hard to find. *Comm. ACM*, **31**:1192–1201, 1988.

[30] B.D. Ripley. The lattice structure of pseudo-random number generators. *Proc. Roy. Soc. London Ser. A*, **389**:197–204, 1983.

[31] A. Rotenberg. A new pseudo-random number generator. *J. Assoc. Comput. Mach.*, **7**:75–77, 1960.

[32] W.E. Thomson. A modified congruence method for generating pseudorandom numbers. *Comp. J.*, **1**:83–86, 1958.

[33] J.P.R. Tootill, W.D. Robinson, and A.G. Adams. The runs up-and-down performance of Tausworthe pseudo-random number generators. *J. Assoc. Comput. Mach.*, **18**:381–399, 1971.

[34] R. v.Hanxleden and L.R. Scott. Correctness and determinism of parallel Monte Carlo processes. *Parallel Comput.*, **18**:121–132, 1992.