A collection of selected pseudorandom number generators with linear structures

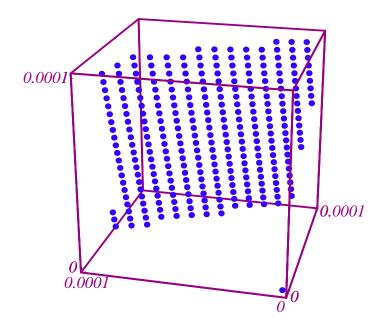
Karl Entacher*

August 21, 1997

Abstract

This is a collection of selected linear pseudorandom number that were implemented in commercial software, used in applications, and some of which have extensively been tested. The quality of these generators is examined using scatter plots and the spectral test. In addition, the spectral test is applied to study the applicability of linear congruential generators on parallel architectures.

Additional Key Words and Phrases: Pseudorandom number generator, linear congruential generator, multiple recursive generator, combined pseudorandom number generators, parallel pseudorandom number generator, lattice structure, spectral test.



^{*}Research supported by the Austrian Science Foundation (FWF), project no. P11143-MAT.

Contents

1	Line	ear congruential generator: LCG	ţ
	1.1	$LCG(2^{31}, 1103515245, 12345, 12345)$ ANSIC	ļ
	1.2	$LCG(2^{31}-1, a = 7^5 = 16807, 0, 1)$ MINSTD	ļ
	1.3	$LCG(2^{31}, 2^{16} + 3 = 65539, 0, 1)$ RANDU	(
	1.4	$LCG(2^{31}-1,630360016,0,1)$ SIMSCRIPT	,
	1.5	$LCG(2^{35}, 5^{15}, 7261067085, 0)$ BCSLIB	,
	1.6	$LCG(2^{32}, 2147001325, 715136305, 0)$ BCPL	7
	1.7	$LCG(2^{31}, 452807053, 0, 1)$ URN12	7
	1.8	$LCG(2^{35}, 5^{13} = 1220703125, 0, 1)$ APPLE	8
	1.9	$LCG(2^{32}, 69069, 0, 1)$ Super-Duper	8
	1.10	$LCG(2^{31}-1,a,0,1)$ HoaglinLCGs	į.
		LCG(m, a, 0, 1) FishmanLCGs	10
	1.12	Knuth - and Borosh and Niederreiter LCGs	10
	1.13	$LCG(2^{59}, 13^{13}, 0, 123456789(2^{32} + 1))$ NAG	1
	1.14	$LCG(2^{48}, 25214903917, 11, 0)$ DRAND48	13
	1.15	$LCG(2^{48}, 44485709377909, 0, 1)$ CRAY	1
	1.16	$LCG(10^{12}-11,427419669081,0,1)$ MAPLE	12
		$LCG(2^{32}, 3141592653, 1, 0)$ DERIVE	12
	1.18	LCG(m, a, 0, 1) CRAND	13
	1.19	LCG(m, a, 0, 1) L'EcuyerLCGs	13
2	Par	allel LCGs	14
3	Mu	tiple recursive generator: MRG	15
	3.1	GrubeMRGs	16
	3.2	L'EcuyerMRGs	16
4	Con	nbined LCGs: cLCG	17
	4.1	Wichman and Hill cLCG	1'
	4.2	L'Ecuyer cLCGs	18
	4.3	NAG PVM Library (G05AAFP) - cLCGs	18
5	Fur	ther "linear" methods and an unified framework	19
6	Cor	clusion	19

Foreword by Peter Hellekalek

Good random number generators are not so easy to find. This task becomes considerably more difficult once parallelization is involved.

There is a basic guideline for stochastic simulation in parallel environments. Generators that produce strongly correlated random numbers with single processors will be unreliable for parallel processors. In many cases, the random number generators are not appropriate for the splitting techniques used in parallel Monte Carlo methods. The streams of random numbers will be strongly correlated.

Linear methods are the most widely used algorithms to produce random numbers, also on parallel machines. It is well-known that parallelization techniques like the "leap-frog" method may lead to disaster, even for generators that are trustworthy in the single processor case.

There are safeguards against unpleasant surprises in your simulation. Practical experience of more than a decade has shown that the *spectral test* is a very reliable figure of merit to assess random number generators. Generators that perform well in the spectral test in a range of dimensions merit a strong recommendation. They will cope with most simulation problems.

In this paper, Karl Entacher carries out a systematic study of random number generators in actual use. He computes the values of the spectral test for the original sequence of random numbers in dimensions up to 8 and investigates the performance of subsequences. The results show how dangerous it is to use parallelization techniques without analyzing the substreams of random numbers they produce.

The reader should note that with random number generators, there are *no guarantees*. Nevertheless, there are *safeguards*. The results of this paper provide this kind of information which is highly relevant to practitioners and designers of generators alike. A systematic study of this scope has never before been published.

Introduction

The present paper gives a collection of linear pseudorandom number generators (PRNGs) that were implemented in commercial software, used in applications, and some of which have extensively been tested. In addition, we study shortcomings of linear congruential generators in parallel applications. A summary of generators used from the 1960s to 1980s is given in Park and Miller [91, Sect. 4] and Dudewicz and Ralley [29, Chapter. 1]. Surveys on pseudorandom numbers are contained in [61, 88, 90, 67, 65, 86, 83, 96, 63, 49, 57, 42]; surveys for parallel PRNGs are [6, 31].

Linear congruential generators (LCGs) are the best analyzed and most widely used PRNGs. LCGs allow an easy (number-) theoretical analysis based on the lattice structure formed by s-dimensional vectors $\mathbf{x}_n = (x_n, \dots, x_{n+s-1}), n \geq 0$ of generated numbers x_n . The quality of LCGs heavily depends on the coarseness of the lattice [61]. In order to find "optimal" parameters for LCGs, several ratings of the lattice structure have been proposed. The most popular measure is the spectral test which gives the maximal distance d_s between adjacent parallel hyperplanes, the maximum being taken over all families of parallel hyperplanes that cover all vectors \mathbf{x}_n (see [18, 61, 64]).

Section 1 contains a collection of many classical- and recent LCGs. References concerning theory (**T**), empirical properties (**E**), implementations (**I**), and literature (**L**) are given. The references are supplemented by quotations (**Quote** [reference]) of several authors which point out the basic properties of generators in an essential way or give critical remarks. We further present two "fingerprints" of LCGs: a zoom into the two- or three-dimensional unit cube which exhibits the lattice structure and results of a normalized spectral test $S_s := d_s^*/d_s$, $2 \le s \le 8$, for which $0 \le S_s \le 1$. The constants d_s^* are absolute lower bounds on d_s based on Hermite constants, see [61, p. 105].

In Section 2 we apply the spectral test to study the applicability of LCGs for parallel simulation environments. The standard technique to get parallel streams of PRNs is to split a sequence of PRNs into suitable subsequences. The spectral test offers an easy way to exhibit subsequences with poor quality. We examined bad subsequences with small-step sizes for almost all LCGs from Section 1. Subsequences may occur also in serial simulations. Hence, if a LCG is proposed due to its good lattice structure, the lattice of the underlying subsequences with step-sizes up to a certain bound should be analyzed as well.

In Section 3 and 4 we consider multiple recursive generators and combined linear congruential generators. These types of generators can be implemented very efficiently and still offer long periods. Similar to LCGs, the set of all s-dimensional vectors \mathbf{x}_n of multiple recursive generators form a lattice structure. Moreover, combined LCGs are equivalent to LCGs with high periods, and they provide some interesting splitting properties to get "independent" parallel streams of PRNs. Finally, in Section 5 we provide some references for further linear methods.

Our spectral tests have been calculated using a *Mathematica* implementation of the Fincke-Pohst algorithm for finding the shortest vector in a lattice by Wilberd van der Kallen¹.

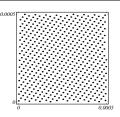
¹The package ShortestVector.m and related Packages are available on the World-Wide-Web at http://www.math.ruu.nl/people/vdkallen/kallen.html and http://random.mat.sbg.ac.at/.

1 Linear congruential generator: LCG

The linear congruential generator (LCG) was proposed by Lehmer in 1948 [77]. We denote this pseudorandom number generator with underlying recursion $y_{n+1} \equiv ay_n + b \pmod{m}$ and seed y_0 by $LCG(m, a, b, y_0)$.

A Classical LCGs

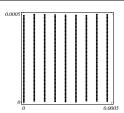
1.1 $LCG(2^{31}, 1103515245, 12345, 12345)$ **ANSIC**



ANSIC is the generator employed by the ANSI C rand() function, BSD version. Note that some versions of the unix online-manual incorrectly claim this generator's period to be 2^{32} .

- **T:** Park and Miller [91] writes: its deficiencies are well known- according to Berkeley 4.2 documentation, the low bits of the numbers generated are not very random. Spectral test for dim. $2 \le s \le 8$: 0.84 0.52 0.63 0.49 0.68 0.43 0.54
- **E:** See [36, 76, 73, 110, 40]
- I: Source code in [93, p. 276 (not recommended!)].
- L: Times for various generators in C (incl. ANSIC) are given in Ripley [96, p. 160]. Quote [96]: The Unix generator rand has been replaced by drand48 (see 1.14) which is far too slow and by a feedback generator random of the type F(r, s, +).

1.2 $LCG(2^{31}-1, a = 7^5 = 16807, 0, 1)$ MINSTD



Quote[93]: First suggested by Lewis, Goodman, and Miller in 1969 [78] (based largely on the fact that this generator is a full period generator), this generator has in subsequent years passed all new theoretical tests, and (perhaps more importantly) has accumulated a large amount of successful use. Park and Miller[91] do not claim that the generator is "perfect" (we will see below that it is not), but only that it is a good minimal standard against which other generators should be judged.

Quote[91]: Our guess is that at some future point we will switch to either a = 48271 or a = 69621. This choice will lead to better spectral test LCGs (see below).

T: "Lattice" tests, execution time results, packing measures, discrepancy bounds and others can be found in [43, 44, 42, 29, 55, 7]. An analysis of the lattice structure is given in [82, 64], and critical distances in [19]. Spectral test for dimension $2 \le s \le 8$:

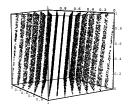
1	16807	0.3375	0.4412	0.5752	0.7361	0.6454	0.5711	0.6096
4	48271	0.8960	0.8269	0.8506	0.7332	0.8078	0.5865	0.4364
(69621	0.7836	0.9205	0.8516	0.7318	0.7667	0.6628	0.7845

E: See [43, 42, 76, 29, 66, 73, 59, 107, 108, 109, 5, 13, 62, 60, 49, 110, 50, 40].

I: For implementations in commercial software and source code see [91, 43, 42, 29, 96, 9]. Source code of MINSTD and MINSTD with a shuffling algorithm added are given in [93]. For fast implementations using multiplier 48271 and 69621 see [11, 93]. Tests for these multipliers are given in [42].

L: [57, 54, 67, 94, 42]

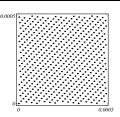
1.3 $LCG(2^{31}, 2^{16} + 3 = 65539, 0, 1)$ **RANDU**



Quote[91]: Many multiplicative linear congruential generators are descendents of the infamous RANDU (System/360 Scientific Subroutine Package, Version III, Programmer's Manual. IBM, White Plains, New York, 1968, p. 77.). This generator was first introduced in the early 1960s; its use soon became widespread (examples are given). In retrospect RANDU was a mistake. The non-prime modulus was selected to facilitate the mod operation and the multiplier was selected primarily because of the simplicity of its binary representation. Research and experience has now made it clear that RANDU represents a flawed generator with no significant redeeming features. It does not have a full period and it has some distinctly non-random characteristics. Knuth calls it "really horrible" [61, p. 173].

- **T:** Lattice ratings in: [61, 82, 43, 42, 29]; Critical distances in [23]. Spectral test for dimension $2 \le s \le 8$: 0.931 | **0.0119** | **0.0594** | 0.157 | 0.293 | 0.453 | 0.617
- **E:** [43, 42, 76, 29, 49, 92, 5, 110]
- I: Three implementations: KERAND (IRCC assambler version of RANDU), original RANDU (IBM Corporation (1970) in FORTRAN) and RANDU as used in SPSS are given in [29, pp. 73]. Further implementations can be found in [91, Sect. 4]. For combined generators using RANDU see [29, pp. 28].
- L: [57, 9]; A history of the beginning of RANDU is given in [29, p. 2].

1.4 $LCG(2^{31}-1,630360016,0,1)$ SIMSCRIPT

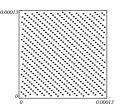


Implemented in the SIMSCRIPT II simulation programming language. For references, empirical tests, execution times and lattice tests see [43, 44, 42, 64, 66, 73, 96, 62, 9, 55].

Spectral test for dim. $2 \le s \le 8$:

	0.82	0.43	0.78	0.80	0.57	0.68	0.72
--	------	------	------	------	------	------	------

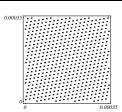
1.5 $LCG(2^{35}, 5^{15}, 7261067085, 0)$ **BCSLIB**



Quote [6]: This generator comes from Knuth [61, p. 102] and is contained in the totally portable random number generator HSRPUN from BCSLIB (Boeing Computer Services). The age of this generator is apparent from its modulus, which dates back to the days of 36-bit computers. The multiplicative version $LCG(2^{35}, 5^{15}, 0, 1)$ is implemented in the programming language SIMULA. Empirical and theoretical tests of this generator are given in [23, 49]. Spectral tests for dimension $2 \le s \le 8$:

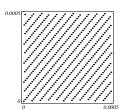
BCSLIB	0.7460	0.8784	0.7995	0.4851	0.6919	0.5233	0.6934
SIMULA	0.5809	0.4145	0.8004	0.6401	0.6951	0.6379	0.7473

1.6 $LCG(2^{32}, 2147001325, 715136305, 0)$ **BCPL**



This generator was used in the BCPL language. References and empirical results are given in [92]. Excellent spectral test: 0.91 | 0.85 | 0.88 | 0.78 | 0.55 | 0.60 | 0.65

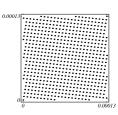
1.7 $LCG(2^{31}, 452807053, 0, 1)$ **URN12**



This generator corresponds to the URN12 generator in [29]. The latter book contains empirical tests and implementations. Further empirical results are given in [73]. **Quote** [29]: This generator was for example used in the CUPL language (ref. given) and was studied by Coveyou and MacPherson [18]. Spectral test for dimension $2 \le s \le 8$:

0.7656 0.7	724 0.4896	0.3982	0.7046	0.6351	0.4365
-------------	------------	--------	--------	--------	--------

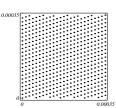
1.8 $LCG(2^{35}, 5^{13} = 1220703125, 0, 1)$ **APPLE**



Implemented on Apple Computers [58, 98]. Spectral test for dimension $2 \le s \le 8$ (see also Knuth [61, p. 102]): $0.4746 \ 0.3715 \ 0.6376 \ 0.6124 \ 0.7416 \ 0.6781 \ 0.7473$

Quote:[61, p. 104] The generators (with multiplier 5^{13} and 5^{15} (BCSLIB)) are reminders of the good old days – they were once used extensively since O. Taussky first suggested them in the early 1950s. Better spectral test results exhibits a LCG with the same multiplier but modulus $m = 2^{35} - 31$ which is mentioned in several papers (see [82]).

1.9 $LCG(2^{32}, 69069, 0, 1)$ Super-Duper



This generator, proposed by Marsaglia [82], is part of a combined Generator called SUPER-DUPER² (combined with a shift-register generator). **Quote** [82]: As a canditate for the best of all multipliers, I nominate $69069 = 3 \cdot 7 \cdot 11 \cdot 13 \cdot 23$. This palindromically convoluted multiplier is easy to remember and has a nearly cubic lattice for moduli 2^{32} , 2^{35} , 2^{36} . This generator is sometimes implemented in the form $LCG(2^{32}, 69069, 1, 0)$, see [49, p. 89] and [109].

T: Lattice considerations can be found in [82, 44] and spectral tests in [6, 29, 61, 42, 7]. Niederreiter [85, p. 1027] gave an upper bound for the discrepancy in dimension 2 of this generator and found multipliers (see also [44, p. 35], [1, p. 81] and [8]) with smaller discrepancy: Quote: Specific multipliers have also been proposed on the basis of the "cubic-lattice criterion"... These data (discrepancy estimates) demonstrate more eloquently than anything else the inutility of the "cubic-lattice" criterion. In fact, one would be better off choosing multipliers blindly rather than using this criterion; Spectral tests for dimension $2 \le s \le 8$ and different moduli:

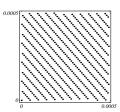
²Sometimes this LCG itself is called Super-Duper [43, 44].

						0.4967	0.6852
2^{35}	0.6935	0.8595	0.6347	0.7	0.2664	0.3690	0.5284
2^{36}	0.4904	0.6822	0.7760	0.6094	0.4746	0.3342	0.4845

The version $LCG(2^{32}, 69069, 1, 0)$ yields "better" spectral test results (The spectral test results of this version are given in [6], instead of the original). The exact discrepancy calculation in dimension 2 for this generator has been made in [3]. For the versions with modulus 2^{35} and 2^{36} critical distances are given in [23]. Super-Duper exhibits bad splitting properties, see Section 2.

- **E:** Empirical results of $LCG(2^{32}, 69069, 1, 0)$ are published in [49, 59, 107, 108, 109, 5, 73, 54, 53].
- I: Quote[83]: The (combined) generator Super-Duper is part of the McGill Random Number Package widely used at several hundred locations. The LCG was implemented on IBM computers [44]. The version $LCG(2^{32}, 69069, 1, 0)$ is part of the VAX VMS-Library (see [49, p. 89] and [95, 96, 57]) and was implemented by the Convex Corp. (see [109]).

1.10 $LCG(2^{31}-1, a, 0, 1)$ HoaglinLCGs



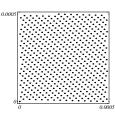
 $a \in \{1078318381, 1203248318, \mathbf{397204094}, 2027812808, 1323257245, 764261123\}$

These are the best spectral and lattice test multipliers from a study of Hoaglin [52]. Theoretical and empirical tests are given in [52, 43, 29, 55] and implementations in [29]. For implementations of multiplier 397204094 (SAS and IMSL Lib.) and its theoretical and empirical results see [44, 105]. Fishman[44, p. 40] writes: Our top five multipliers do not dominate (respectively discrepancy bounds) a = 397204094 and 630360016 (SIMSCRIPT) unambiguously, as in the earlier tables. This lack of discrimination on the part of the lower bounds on discrepancy may be due to the fact that discrepancy is not a rotation invariant measure.

Spectral tests for dim. $2 \le s \le 8$:

$a \backslash s$	2	3	4	5	6	7	8
1078318381	0.9442	0.6790	0.6684	0.7137	0.7252	0.6482	0.6265
1203248318	0.9086	0.8673	0.8092	0.6997	0.6389	0.6324	0.6695
397204094	0.5564	0.5748	0.6674	0.7678	0.5947	0.5815	0.7181
2027812808	0.6883	0.6012	0.7059	0.6976	0.6356	0.7432	0.6917
1323257245	0.7298	0.6106	0.6715	0.7393	0.6389	0.6726	0.6430
764261123	0.7309	0.7081	0.7153	0.7710	0.6887	0.6143	0.6484

1.11 LCG(m, a, 0, 1) FishmanLCGs



The parameters a in the table below determine the top five generators respectively m in an exhaustive analysis of multiplicative congruential random number generators made by Fishman and Moore [44, 41, 42].

Quote [44]: Here a multiplier is said to be optimal if the distance between adjacent parallel hyperplanes on which k-tuples lie does not exceed the minimal achievable distance by more than 25 percent for k = 2, ..., 6. This means that the spectral test values are greater than 0.8 (see below and for the first two generators see also [64, 55]). Empirical result of the second generator can be found in [66, 73, 49, 110, 76, 40]. Generator 1 and 6 are used in [54] and [53] to study transformation methods for non-uniform variates. An implementation is given in [56].

nr .	m	$a \backslash s$	2	3	4	5	6	7	8
1		742938285	0.8672	0.8607	0.8627	0.8319	0.834	0.6239	0.7067
2		950706376	0.857	0.8985	0.8691	0.8337	0.8274	0.5916	0.5981
3	$2^{31}-1$	1226874159	0.8411	0.8787	0.8255	0.8378	0.8444	0.6277	0.5981
4		62089911	0.8930	0.8903	0.8575	0.8630	0.8249	0.7622	0.6020
5		1343714438	0.8236	0.8324	0.8245	0.8262	0.8254	0.7440	0.4915
nr .	m	$a \backslash s$	2	3	4	5	6	7	8
6		1099087573	0.8920	0.8563	0.8603	0.8420	0.8325	0.5547	0.7506
7		2396548189	0.8571	0.9238	0.8316	0.8248	0.8247	0.6664	0.709
8	2^{32}	2824527309	0.9220	0.8235	0.850	0.8451	0.8332	0.6983	0.5512
9		3934873077	0.8675	0.8287	0.8278	0.8361	0.8212	0.7188	0.7616
10		392314069	0.9095	0.8292	0.8536	0.8489	0.8198	0.6047	0.6084
nr.	m	$a \backslash s$	2	3	4	5	6	7	8
11		68909602460261	0.8253	0.8579	0.8222	0.8492	0.8230	0.6971	0.5275
12		33952834046453	0.9282	0.8476	0.8575	0.8353	0.8215	0.5289	0.6383
13	2^{48}	43272750451645	0.8368	0.8262	0.8230	0.8400	0.8213	0.5993	0.6171
14		127107890972165	0.8531	0.8193	0.8216	0.8495	0.8224	0.4504	0.5934
15		55151000561141	0.9246	0.8170	0.9240	0.8278	0.8394	0.6274	0.4471

1.12 Knuth - and Borosh and Niederreiter LCGs

A list of 30 LCGs including RANDU, BCSLIB, APPLE, Super-Duper, DERIVE, and their spectral tests is given in Knuth[61]. These LCGs have been selected according to various criteria ("random" multiplier, multiplier that guarantee fast implementations, multiplier close to power of two which produce bad lattice structures [61, 85]). Some of them stem from a search for optimal multipliers with respect to two-dimensional discrepancy made by Borosh and Niederreiter[8]. Empirical and theoretical results of these generators can be found in [105, 106, 23, 53, 54, 42].

B Some recent LCGs

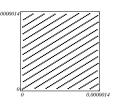
1.13
$$LCG(2^{59}, 13^{13}, 0, 123456789(2^{32} + 1))$$
 NAG

This is the basic generator for pseudorandom numbers in many different distributions implemented in the NAG Fortran Library [104, Sect. G05] (also contained in the NAG C Library). Empirical results for this generator are given in [109].

T: Usable limit considerations in [81]. A first lattice analysis is given in [94]. Spectral test and Beyer quotient calculations up to dimension 20 are published in [2]. Spectral test for dim. $2 \le s \le 8$ (see also [104]):

0.8423	0.7288	0.7426	0.5771	0.6351	0.5217	0.5455

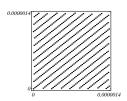
1.14 $LCG(2^{48}, 25214903917, 11, 0)$ **DRAND48**



DRAND48 is the generator employed by the ANSI C drand48() function, BSD version.

T: Spectral test for dim. $2 \le s \le 8$: 0.51 0.80 0.45 0.58 0.66 0.80 0.60 Execution times are given in [96].

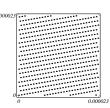
1.15 $LCG(2^{48}, 44485709377909, 0, 1)$ **CRAY**



This generator was implemented on CRAY systems (see [6, 109, 98, 25, 23, 20, 42, 30]) and used in PASCLIB, a collection of utility subprograms callable from PASCAL on CDC CYBER computers (see [41, 94]). Empirical and theoretical tests of this generator are given in the papers above.

Spectral test for dim. $2 \le s \le 8$: 0.827 | 0.742 | 0.398 | 0.731 | 0.618 | 0.667 | 0.564

1.16 $LCG(10^{12}-11, 427419669081, 0, 1)$ **MAPLE**



This is the PRNG implemented in the mathematical software Maple.

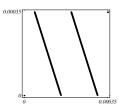
Email(From ddubois@maplesoft.com Mar 13 1996): $10^{12}-11$ is a previously determined prime with 427419669081 a proven primitive element in \mathbb{Z}_p . NOTE: the results of the spectral test for this generator are very good.

There was an article written by Zaven A. Karian and Rohit Goyal that appeared in Volume 1, Number 1, Spring 1994 of the Maple Technical Newsletter entitled "Random Number Generation and Testing". It offers a detailed analysis of Maple's rand() command (For orders and information please contact: Birkhauser Verlag AG, Klosterberg 23, P.O. Box 133 CH-4010, Basel, Switzerland, Phone: 41 61 271 7400).

Spectral test for dimensions $2 \le s \le 8$ (see also [98]):

0.7513 0.7366 0.6491	0.7307	0.6312	0.5598	0.5559
----------------------	--------	--------	--------	--------

1.17 $LCG(2^{32}, 3141592653, 1, 0)$ **DERIVE**



Email(From swh@aloha.com Mar 14 1996): The DERIVE random number generator maintains a random integer in the interval $[0, 2^{32} - 1]$ to calculate user requested random integers in a given range. Given a random integer n, the next random integer is generated by the formula MOD (3141592653*n + 1, 2^{32}). Information: Soft Warehouse, Inc. (the authors of DERIVE, A Mathematical Assistant), 3660 Waialae Avenue, Suite 304 Honolulu, HI 96816-3236 U.S.A. Web page: http://www.derive.com

The multiplier probably stems from Knuth [61, p. 32,44,102], who studied

$$LCG(2^{35}, 3141592653, 2718281829, 0).$$

Quote:[61, p. 103] The latter LCG shows a "random" multiplier; this generator has satisfactorily passed numerous empirical tests for randomness, but it does not have especially good spectral test values for dimensions $2 \le s \le 4$. Empirical results of this generator are given in [105].

Spectral test for dimensions $2 \le s \le 8$ (observe the poor lattice of DERIVE in dim. 2):

2^{32}	0.0972	0.5551	0.5479	0.3216	0.6426	0.5210	0.6731
2^{35}	0.2749	0.2776	0.3258	0.2122	0.4544	0.6721	0.6984

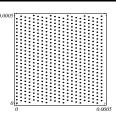
1.18 LCG(m, a, 0, 1) **CRAND**



nr.	m	a	s = 2	3	4	5	6	7	8
1	2^{32}	663608941	0.88	0.60	0.80	0.64	0.68	0.61	0.74
2	2^{54}	2783377641436325	0.41	0.71	0.42	0.64	0.51	0.61	0.62
3	2^{54}	2783377640906189	0.87	0.76	0.79	0.78	0.75	0.73	0.69
4	2^{54}	2783377640450829	0.89	0.84	0.69	0.68	0.72	0.79	0.70
5	2^{54}	2783377640871525	0.93	0.81	0.81	0.67	0.76	0.73	0.72

These generators have been implemented in several versions of C-RAND [98, 99, 100, 51], a package for generating nonuniform random variates. They were proposed by Ahrens et al. [4, 28, 98]. Empirical results of the first generator are given in [49, 53].

1.19 LCG(m, a, 0, 1) L'EcuyerLCGs



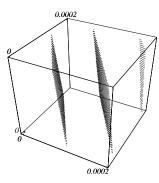
Pierre L'Ecuyer used spectral tests and Beyer ratios to search for good multipliers a for LCG(m,a,0,1) with selected prime moduli m. In order to propose combined LCGs [64] he obtained the multipliers $a \leq \sqrt{m}$ (for faster implementations) given in Section 4.2. The latter paper contains portable implementations and results of a battery of empirical tests for Generator 4.2.3. This LCG is also used as an example to show that LCGs are not recommended to generate random variates by the ratio of uniforms method [54]. For a similar purpose Generator 4.2.1 is applied in [55] as an example to show that LCGs with small multiplier are useless to generate pseudorandom numbers for other distributions via the rejection method. Discrepancy bounds, spectral tests and Beyer quotients for the generators 4.2.1 to 4.2.6 are given in [42].

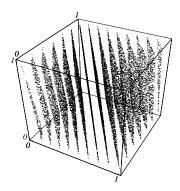
In [72] L'Ecuyer et. al. suggested the following "good-lattice" LCGs (see also [65]).

nr.	m	$a \backslash s$	2	3	4	5	6	7	8
1	$2^{31} - 1$	1385320287	0.767	0.784	0.722	0.809	0.774	0.664	0.676
2	$2^{31}-1$	41358	0.830	0.704	0.701	0.733	0.665	0.687	0.602
3	$2^{47} - 115$	71971110957370	0.756	0.708	0.688	0.706	0.683	0.762	0.687
4	$2^{47} - 115$	-10018789	0.786	0.749	0.773	0.806	0.773	0.744	0.714
5	$2^{63} - 25$	2307085864	0.707	0.713	0.752	0.660	0.803	0.774	0.656

Recent tables of LCGs with different moduli $2^8 \le m \le 2^{128}$ and good lattice structures up to high dimensions are given in [70].

2 Parallel LCGs





An obvious way to get parallel streams of PRNs is to partition a sequence of a given generator into suitable subsequences. This can be achieved in different ways. By varying the initial seed one can partition a sequence of a LCG into consecutive blocks of a given length. An extensive discussion of this method for the LCG with respect to parallelization is given in [20] (see also [21, 22, 19, 23, 24, 25, 30]). It turns out that only small fractions of sequences produced by a LCG can safely be used because of the well known long-range correlations which may cause high correlations between the parallel streams. Critical lengths of the partitions for several LCGs are given e.g. CRAY [21, 19, 23], L'Ecuyer LCG 3 [19], MINSTD [19], Super-Duper and RANDU [23].

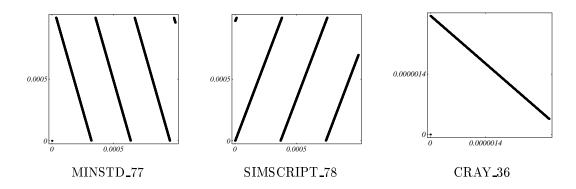
Another way to partition a sequence $(x_n)_{n=0}^{T-1}$ of a LCG with period T is to generate subsequences of the form $(x_{kn+i})_{n>0}$, $k \geq 2$, $0 \leq i \leq k-1$.

Consider a full-period $LCG(m, a, b, y_0)$. For this generator, the subsequence above is equivalent to the output of $LCG(m, a^k \pmod{m}, b^{(k)}, y_0^{(i)})$ with suitable parameters $b^{(k)}$, $y_0^{(i)}$, see [39]. Note that the period of the subsequence equals T/gcd(k, T). Even if one chooses subsequences with maximal period T, these sequences can be of inferior quality. This can be shown by the spectral test. The table below gives some examples. G_k denotes the LCG which generates a full-period subsequence $(x_{kn})_{n\geq 0}$ of generator G. Further results are given in [39, 38].

LCG	Sect.	s=2	3	4	5	6	7	8
ANSIC_25	1.1	0.0822	0.7978	0.6059	0.7767	0.6327	0.5936	0.6096
MINSTD_25	1.2	0.5967	0.0782	0.4427	0.5401	0.478	0.5036	0.56
BCSLIB_45	1.5	0.7494	0.7596	0.0766	0.2122	0.4544	0.7216	0.659
Super-Duper_59	1.9	0.0910	0.6132	0.5992	0.7485	0.6342	0.5902	0.6648
Super-Duper_81	1.9	0.0594	0.6492	0.8133	0.6886	0.6249	0.6729	0.6771
Super-Duper_99	1.9	0.0202	0.2071	0.6	0.2933	0.418	0.4603	0.635
Super-Duper_135	1.9	0.6758	0.6143	0.5708	0.0999	0.1281	0.2016	0.2781
Super-Duper_153	1.9	0.1729	0.0567	0.2795	0.7636	0.5872	0.536	0.4071
FISH2_29	1.11	0.0903	0.5806	0.286	0.615	0.7007	0.5815	0.5641
FISH7_65	1.11	0.0968	0.4785	0.5557	0.7437	0.4426	0.6962	0.7433
FISH7_105	1.11	0.926	0.00945	0.05	0.1367	0.2608	0.4103	0.566
FISH8_3	1.11	0.7092	0.0806	0.4301	0.5704	0.5862	0.6166	0.6307
FISH15_23	1.11	0.2562	0.0600	0.0114	0.0462	0.1275	0.2031	0.2077
NAG_13	1.14	0.0875	0.7036	0.2369	0.7165	0.6532	0.6219	0.5455
DERIVE_33	1.18	0.4475	0.0591	0.1459	0.3601	0.3438	0.559	0.6495
CRAND1_17	1.19	0.0811	0.7742	0.4712	0.5983	0.5382	0.552	0.5512
CRAND1_83	1.19	0.0919	0.5215	0.6726	0.4304	0.6951	0.619	0.7282
LCG1.20.1 _ 97	1.20	0.6802	0.0410	0.2317	0.57	0.6822	0.5278	0.5883
LCG4.2.3_33	4.2	0.0088	0.3013	0.3845	0.6992	0.6632	0.6371	0.5961

Note, that FISH15_23, FISH7_105 exhibit spectral test results worse than RANDU. The zoom into the 3-dimensional unit cube above stems from FISH15_23. The second graphics shows the 15 hyperplanes that cover all two-dimensional overlapping tuples generated by FISH7_105. A parallel application using CRAND1 is given in [79].

If the subsequence-LCGs do not have full period, the underlying lattice structure can be of reduced quality as well. We demonstrate this behavior using zooms into the unit square. Consider for example the generators MINSTD_77, SIMSCRIPT_78 and CRAY_36 below. Compare the zooms in the Sections 1.2, 1.4 and 1.14.



Remark: Using the spectral test it is also possible to analyze non-full period subsequences, see [74]. Recently, we used this test to study such subsequences of CRAY and got bad values for step size 128 and dimensions $s \ge 6$. Simple Monte Carlo integrations supplement our results [37].

3 Multiple recursive generator: MRG

Quote[72, p. 2]: As computational power gets cheaper, increasingly long sequences of random numbers are used in applications. Generators with longer periods and which are more reliable are then necessary. One alternative is the class of multiple recursive generators (MRGs) (ref. are given in the paper) based on the higher order recursion:

$$x_n := (a_1 x_{n-1} + \dots + a_k x_{n-k}) \mod m.$$

For a given prime modulus m, such generators can reach a period length of $m^k - 1$ and the good ones have much better structural properties than the simple MLCGs with the same modulus, while being almost as fast and easy to implement [65].

Proposed by Tausworthe [102] for modulus 2 and by Knuth [61] for arbitrary prime modulus, MRGs have been studied extensively by Grube in his PhD-thesis [47, 46]. Recent computer searches for good parameters can be found in [72]. A statistical study of different PRNGs where the MRGs exhibited the best results is [101].

Similar to LCGs, the set of all overlapping s-tuples of values $u_n = x_n/m$ forms a lattice structure in $[0, 1[^s$ (for details see [47, 45, 27, 74]). **Quote:** [27] MRGs of order k behave in \mathbf{R}^{sk} like LCGs in \mathbf{R}^s . This lattice structure may be analyzed by the spectral test as well.

3.1 GrubeMRGs

In [46] Grube proposed the following parameters for $m = 2^{31} - 1$:

k	a_1	a_2	a_3
1	241639237		
2	337190270	268152554	
3	518175991	510332243	71324449

Further MRGs for k = 2, 3 and the corresponding lattice analysis are given in Dieter[27, 26]. Empirical and theoretical results of MRGs contained in the latter papers (including MRG 3 above) are given in [49]. For spectral test results of MRG 2,3 see [71].

3.2 L'EcuyerMRGs

Using a spectral test- and a lattice test (Beyer ratio) search, L'Ecuyer et. al. [72] obtained the following MRGs (implementations in Pascal and C are given):

m	$2^{31} - 1$	$2^{31} - 1$	$2^{31}-1$	$2^{31}-1$	$2^{31}-1$		
k	2	2	3	3	3		
a_1	1498809829	46325	65338	1476728729	2021422057		
a_2	1160990996	1084587	0	0	1826992351		
a_3			64636	1155643113	1977753457		
m	$2^{31} - 1$	$2^{31}-1$	$2^{31}-1$	$2^{31}-1$	$2^{31} - 19$	$2^{31} - 19$	$2^{31} - 19$
k	4	4	5	6	7	7	7
a_1	2001982722	64886	107374182	177786	1975938786	2109532706	1071 064
a_2	1412284257	0	0	0	875540239	0	0
a_3	1155380217	0	0	0	433188390	0	0
a_4	1668339922	65322	0	0	451413575	0	0
a_5			104480	0	1658907683	0	0
a_6				64654	1513645334	0	0
a_7					1428037821	1651737654	2113 664
m		$2^{47} - 115$	$2^{47} - 115$	$2^{47} - 115$	$2^{63} - 25$	$2^{63} - 711$	
k		2	2	3	2	5	
a_1	650697	701955467	11138366	11209406	2975962250	2949964090	
a_2	1235979	951337197	11808124	0	2909704450	0	
a_3				11721934		0	
a_4						0	
a_5						2946716567	

Parameters for $1 \le k \le 7$ and $m \in \{2^{15} - 19, 2^{31} - 1\}$ obtained by an earlier search of L'Ecuyer et. al. are published in [71]. For empirical results of one of these generators see [49]. Recently, L'Ecuyer studied combined MRGs [68].

4 Combined LCGs: cLCG

Combining different streams $(x_n^{(j)})_{n\geq 0}$, $1\leq j\leq r$, of PRNGS into a new stream $(x_n)_{n\geq 0}$, $x_n\equiv x_n^{(1)}+\ldots+x_n^{(r)}\pmod 1$ yields an easy way to achieve long periods while keeping the computational costs of generating the numbers low by choosing suitable parameters for each underlying generator. For references and an overview of this technique and its difficulties see [67, Sect. 9] and [90, Sect. 4.2]. Other combination methods (composite congruential generators, compound generators, ...) are given in [97, 29, 12, 64, 75, 57, 42]. Quote[67]: Theoretical results in [83, 10] (see also [49, p. 70]) appear to support the view (at first glance) that combined generators should have better statistical behavior in general than their individual components. However, as explained in [65], applying those theoretical results to "deterministic" generators is a somewhat shaky reasoning. Combination can conceivably worsen things. Nevertheless, empirical results strongly support combinations [83, 66].

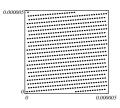
However, the more theoretical properties of a combined generator are known, the less undesired side-effects are possible.

In contrast to hybrid combinations, the combination of (multiplicative) LCGs only yields the advantage that the underlying structure of s-dimensional overlapping tuples is known. This is due to the fact that cLCGs are equivalent to single LCGs with large moduli [111, 49, 75].

Combined LCGs provide an easy way to partition their output sequences into disjoint parallel streams by varying the seeds [49, 48]. From the equivalence to LCGs, it follows that splitting a cLCG needs certain precautions as well [24]. Using a modified spectral test, MacLaren [80] has suggested a method to test the independence of parallel streams for cLCGs (and LCGs). The latter paper is the basis for the combined LCGs implemented in the Nag PVM Library.

A combination of multiplicative LCGs was first suggested by Wichman and Hill [111].

4.1 Wichman and Hill cLCG



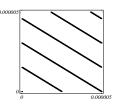
Wichman and Hill combined three multiplicative LCGs:

LCG	s=2	3	4	5	6	7	7
LCG(30269, 171, 0, 1)	0.9147	0.183	0.4082	0.6605	0.8212	0.4813	0.5507
LCG(30307, 172, 0, 1)	0.9195	0.6228	0.7039	0.7854	0.785	0.7014	0.584
LCG(30323, 170, 0, 1)	0.9085	0.6821	0.4639	0.7507	0.6049	0.7415	0.584

Zeisl (1986) [111] completed that this generator is equivalent to

LCG(27817185604309, 16555425264690, 0, 1).

4.2 L'Ecuyer cLCGs



In order to implement combined LCGs, the following "good-lattice" LCGs are suggested in [64]. The zoom above stems from LCG(32504802982957, 30890646900944, 0, 1) which is equivalent to the combined generators using LCG 9, 12 and 13 (observe the weak lattice in dimension 2).

nr.	m	$a \backslash s$	2	3	4	5	6	7	8
1	$2^{31}-1$	39373	0.791	0.755	0.787	0.758	0.754	0.779	0.560
2	2147483563	40014	0.803	0.836	0.788	0.828	0.808	0.474	0.663
3	2147483399	40692	0.817	0.818	0.805	0.891	0.819	0.474	0.663
4	2147482811	41546	0.834	0.787	0.811	0.808	0.821	0.709	0.692
5	2147482801	42024	0.844	0.811	0.857	0.783	0.810	0.558	0.740
6	2147482739	45742	0.919	0.851	0.783	0.820	0.799	0.322	0.449
7	32749	162	0.833	0.796	0.710	0.658	0.763	0.505	0.578
8	32749	219	0.930	0.793	0.726	0.718	0.763	0.733	0.721
9	32363	157	0.812	0.851	0.827	0.782	0.788	0.584	0.669
10	32143	160	0.830	0.754	0.807	0.728	0.777	0.675	0.611
11	32119	172	0.893	0.719	0.735	0.776	0.740	0.653	0.511
12	31727	146	0.763	0.722	0.727	0.758	0.729	0.676	0.547
13	31657	142	0.743	0.762	0.824	0.785	0.779	0.655	0.698

Results of a battery of empirical tests are given for generator 3 and for two combined generators using generator 2 and 3 (32bit) and generator 9, 12 and 13 (16bit). Portable implementations are given as well. An implementation (RANECU) of the combined generator using LCG 2 and 3 is published in [57].

4.3 NAG PVM Library (G05AAFP) - cLCGs

The NAG PVM (parallel virtual machine) Library Chapter G05 contains parameters for 273 combined multiplicative LCGs, four LCGs combined respectively (for example the first combination consists of LCG(16770647, 125, 0, 1), LCG(16770643, 117, 0, 1), LCG(16770623, 127, 0, 1) and LCG(16770617, 126, 0, 1)). The different generators are used to produce "independent" parallel PRNGs. This approach is based on the paper of MacLaren [80]. The "independence" is tested with a modified spectral test.

5 Further "linear" methods and an unified framework

Recently L'Ecuyer studied different combinations of MRGs [68]. The paper analyzes the period and lattice of these PRNGs and contains specific generators and portable implementations. It turns out that combined MRGs are equivalent (or approximately equivalent), to an MRG with large modulus.

As a similar result, the add-with-carry (AWC) and subtract with borrow (SWB) pseudorandom number generator proposed by Marsaglia and Zaman [84] is equivalent to a LCG with large prime modulus [103]. The latter paper also illustrates the fact that AWC and SWB generators have extremely bad lattice structure in high dimensions (see also [14]). Bad lattice structures for vectors of non-successive values produced by several linear methods (LCG, MRG, lagged-Fibonacci, AWC/SWB) have been studied by L'Ecuyer [69, 74]. Example 5 in [69] considers the widely available combined generator RANMAR (see [57]). The bad lattice structure is examined by a MRG which closely approximates RANMAR [16]. A generalization of the family of AWC generators is given by the multiply-with-carry (MWC) family proposed by Marsaglia (see [15]). The s-dimensional uniformity of MWC generators is studied in [17]. The paper also contains a method for finding good parameters in terms of the spectral test.

An efficient algorithm of the spectral test which facilitates the analysis of lattices generated by vectors of successive or non-successive values produced by linear congruential generators with moduli of essentially unlimited sizes was derived by [74].

The analog to the multiplicative LCG for pseudorandom *vector* generation is the *matrix* method. For references and an overview of this method see Niederreiter [90, Sect. 5.1].

Niederreiter introduced an unified framework for "linear" methods, the multiple-recursive matrix method (see [90, Sect. 4.1 and 5.2]). Quote[90]: The method includes several of the methods discussed earlier as special cases, such as the multiplicative linear congruential method (with prime modulus), the multiple-recursive congruential method (with prime modulus), the GFSR method, and the twisted GFSR generator.

6 Conclusion

From the previous sections we conclude that almost all pseudorandom number generators used in simulations are linear methods. These PRNGs allow an easy theoretical analysis based on the linear structure of s-dimensional overlapping tuples.

A disadvantage of linear methods is their weakness with respect to splitting, which restricts the use of these methods in parallel simulations. Previous tests are necessary, if subsequences of a linear generator are used for a particular simulation problem.

Finally we want to note that inversive pseudorandom number generators guarantee the absence of lattice structures [32, 33, 35, 88, 90, 50]. Especially explicit inversive congruential PRNGs are very robust with respect to splitting their output sequences into subsequences and the splitting procedure is easy to handle [34, 87, 89, 40].

Inversive generators are significantly slower than LCGs. Nevertheless, these generators provide substantially different PRNs to control simulation results obtained by linear methods.

References

- [1] L. Afflerbach. Die Gütebewertung von Pseudo-Zufallszahlen-Generatoren aufgrund theoretischer Analysen und algorithmischer Berechnungen. *Grazer Mathematische Berichte*, **309**, 1990.
- [2] L. Afflerbach and G. Gruber. Assessment of random number generators in high accuracy. In S. Morito, H. Sakasegawa, M. Fushimi, and K. Nakano, editors, New Directions in Simulation for Manufacturing and Communications, pages 128–133. OR Society of Japan, 1994.
- [3] L. Afflerbach and R. Weilbächer. The Exact Determination of Rectangle Discrepancy for Linear Congruential Pseudorandom Numbers. *Math. Comp.*, **53**(187):343–354, 1989.
- [4] J. Ahrens, U. Dieter, and A. Grube. Pseudo-random numbers: a new proposal for the choice of multiplicators. *Computing*, **6**:121–138, 1970.
- [5] N.S. Altman. Bit-wise behavior of random number generators. SIAM J. Sci. Stat. Comput., 9:941-949, 1988.
- [6] S.L. Anderson. Random number generators on vector supercomputers and other advanced architectures. SIAM Rev., 32:221-251, 1990.
- [7] A.C. Atkinson. Tests of Pseudo-random Numbers. Appl. Statist., 29(2):164–171, 1980.
- [8] I. Borosh and H. Niederreiter. Optimal multipliers for pseudo-random number generation by the linear congruential method. *BIT*, **23**:65–74, 1983.
- [9] P. Bratley, B.L. Fox, and L.E. Schrage. A Guide to Simulation. Springer, New York, 2nd edition, 1987.
- [10] M. Brown and H. Solomon. On combining pseudorandom number generators. *Annals of Statistics*, 1:691–695, 1979.
- [11] D.G. Carta. Two fast implementations of the "minimal standard" random number generator. *Comm. ACM*, **33**:87–88, 1990.
- [12] B.J. Collings. Compound random number generators. J. Amer. Statist. Assoc., 82:525–527, 1987.
- [13] A. Compagner. Operational conditions for random-number generation. *Phys. Rev. E.*, **52**:5634–5645, 1995.
- [14] R. Couture and P. L'Ecuyer. On the lattice structure of certain linear congruential sequences related to AWC/SWB generators. *Math. Comp.*, **62**:799–808, 1994.
- [15] R. Couture and P. L'Ecuyer. Linear Recurrences with Carry as Uniform Random Number Generators. In C. Alexopoulos and D. Goldsman K. Kang, W.R. Lilegdon, editors, Proceedings of the 1995 Winter Simulation Conference, pages 263–267, 1995.
- [16] R. Couture and P. L'Ecuyer. Orbits and Lattices for Linear Random Number Generators with Composite Moduli. *Math. Comp.*, **65**:189–201, 1996.
- [17] R. Couture and P. L'Ecuyer. Distribution Properties of Multiply-with-Carry Random Number Generators. *Math. Comp.*, **66**:591–607, 1997.
- [18] R.R. Coveyou and R.D. MacPherson. Fourier analysis of uniform random number generators. J. Assoc. Comput. Mach., 14:100–119, 1967.
- [19] A. De Matteis, J. Eichenauer-Herrmann, and H. Grothe. Computation of critical distances within multiplicative congruential pseudorandom number sequences. J. Comp. Appl. Math., 39:49–55, 1992.

- [20] A. De Matteis and S. Pagnutti. Parallelization of random number generators and long-range correlations. *Numer. Math.*, **53**:595–608, 1988.
- [21] A. De Matteis and S. Pagnutti. A class of parallel random number generators. Parallel Comput., 13:193–198, 1990.
- [22] A. De Matteis and S. Pagnutti. Long-range correlations in linear and non-linear random number generators. *Parallel Comput.*, **14**:207–210, 1990.
- [23] A. De Matteis and S. Pagnutti. Critical distances in pseudorandom sequences generated with composite moduli. *Intern. J. Computer Math.*, **43**:189–196, 1992.
- [24] A. De Matteis and S. Pagnutti. Long-range correlation analysis of the Wichmann-Hill random number generator. *Statistics and Computing*, **3**:67–70, 1993.
- [25] A. De Matteis and S. Pagnutti. Controlling correlations in parallel Monte Carlo. Parallel Comput., 21:73–84, 1995.
- [26] U. Dieter. Probleme bei der Erzeugung gleichverteilter Zufallszahlen. In L. Afflerbach and J. Lehn, editors, Kolloquium über Zufallszahlen und Simulationen, pages 7–20. Teubner-Verlag, Stuttgart, 1986.
- [27] U. Dieter. Erzeugung von gleichverteilten Zufallszahlen. In Jahrbuch Überblicke Mathematik 1993, pages 25–44, Braunschweig, 1993. Vieweg.
- [28] U. Dieter and J.H. Ahrens. Uniform random numbers. Inst. f. Math. Stat., Technische Hochschule Graz, Graz, 1974.
- [29] E.J. Dudewicz and T.G. Ralley. The Handbook of Random Number Generation and Testing With TESTRAND Computer Code, volume 4 of American Series in Mathematical and Management Sciences. American Sciences Press, Inc., Columbus, Ohio, 1981.
- [30] M.J. Durst. Using linear congruential generators for parallel random number generation. In E.A. MacNair, K.J. Musselman, and P. Heidelberger, editors, *Proceedings of the 1989 Winter Simulation Conference*, pages 462–466, 1989.
- [31] W.F. Eddy. Random number generators for parallel processors. J. Comp. Appl. Math., 31:63–71, 1990.
- [32] J. Eichenauer and J. Lehn. A non-linear congruential pseudo random number generator. Statist. Papers, 27:315–326, 1986.
- [33] J. Eichenauer-Herrmann. Inversive congruential pseudorandom numbers avoid the planes. *Math. Comp.*, **56**:297–301, 1991.
- [34] J. Eichenauer-Herrmann. Statistical independence of a new class of inversive congruential pseudorandom numbers. *Math. Comp.*, **60**:375–384, 1993.
- [35] J. Eichenauer-Herrmann. Compound nonlinear congruential pseudorandom numbers. Monatsh. Math., 117:213-222, 1994.
- [36] K. Entacher. Selected random number generators in run tests. Preprint, Department of Mathematics, University of Salzburg, Austria.
- [37] K. Entacher. A Remark on a Cray System Pseudorandom Number Generator. Preprint, Department of Mathematics, University of Salzburg, Austria, 1997.
- [38] K. Entacher. The PLAB Picturebook: Part III, Bad Subsequences of LCGs The Results. Report no. 06, PLAB reports, University of Salzburg, 1997. Available on the internet at http://random.mat.sbg.ac.at/team/.
- [39] K. Entacher. Bad subsequences of well-known linear congruential pseudorandom number generators. ACM Transactions on Modeling and Computer Simulation, 8(1), 1998, to appear.

- [40] K. Entacher and S. Wegenkittl. On the Relevance of Splitting Properties and the Compound Method in Parallel Applications of Pseudorandom Number Generators. In M. Trobec, M. Vajtersic, P. Zinterhof, and B. Robic, editors, *Proceedings of the interna*tional Workshop Parallel Numerics' 96, pages 64-74, Gozd Martuljek-Slovenia, 1996.
- [41] 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.
- [42] G.S. Fishman. Monte Carlo: Concepts, Algorithms, and Applications, volume 1 of Springer Series in Operation Research. Springer, New York, 1996.
- [43] 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.
- [44] G.S. Fishman and L.R. Moore. An exhaustive analysis of multiplicative congruential random number generators with modulus 2³¹ 1. SIAM J. Sci. Statist. Comput., 7:24–45, 1986. See erratum, ibid., 7:1058, 1986.
- [45] H. Grothe. Matrixgeneratoren zur Erzeugung gleichverteilter Pseudozufallsvektoren. PhD thesis, Technische Hochschule Darmstadt, 1988.
- [46] A. Grube. Mehrfach rekursiv-erzeugte Pseudo-Zufallszahlen. Z. für angewandte Math. und Mechanik, 53:T223-T225, 1973.
- [47] A. Grube. Mehrfach rekursiv erzeugte Zufallszahlen. PhD thesis, University of Karlruhe, 1973.
- [48] F. Härtel. On combined random number generators. In *International Workshop on Mathematical Methods and Tools in Computer Simulation*, pages 7–8. V. I. Smirnov Scientific Research Institute of Mathematics and Mechancis, 1994.
- [49] F. Härtel. Zufallszahlen für Simulationsmodelle. PhD thesis, Hochschule St. Gallen für Wirtschafts-, Rechts- und Sozialwissenschaften, St. Gallen, 1994.
- [50] P. Hellekalek. Inversive pseudorandom number generators. In C. Alexopoulos and D. Goldsman K. Kang, W.R. Lilegdon, editors, Proceedings of the 1995 Winter Simulation Conference, 1995.
- [51] P. Hellekalek, T. Auer, K. Entacher, H. Leeb, O. Lendl, and S. Wegenkittl. The PLAB www-server. http://random.mat.sbg.ac.at. Also accessible via ftp.
- [52] D. Hoaglin. Theoretical Properties of Congruential Random-Number Generators: An Empirical View. Memorandum NS-340. Harvard University, Department of Statistics, 1976.
- [53] W. Hörmann. The quality of non-uniform random numbers. In H. Dyckhoff, U. Derigs, M. Salomon, and H.C. Tijms, editors, Operations Research Proceedings 1993, DGOR/NSOR, pages 329–335. Springer-Verlag, 1993.
- [54] W. Hörmann. A Note on the Quality of Random Variates Generated by the Ratio of Uniforms Method. ACM Transactions on Modeling and Computer Simulation, 4(1):96– 106, 1994.
- [55] W. Hörmann and G. Derflinger. A portable random number generator well suited for the rejection method. ACM Transactions on Mathematical Software, 19(4):489–495, December 1993.
- [56] P.F. Hulquist. A good random number generator for microcomputers. *Simulation*, **57:4** :258–259, 1991.

- [57] F. James. A review of pseudorandom number generators. *Comp. Phys. Comm.*, **60**:329–344, 1990.
- [58] L.P. Jennergren. Another method for random number generation on microcomputers. Simulation, 41:79, 1983.
- [59] K. Kankaala, T. Ala-Nissila, and I. Vattulainen. Bit-level correlations in some pseudorandom number generators. Phys. Rev. E, 48:4211–4214, 1993.
- [60] S. Kirkpatrick and E.P. Stoll. A very fast shift-register sequence random number generator. J. Comp. Physics, 40:517–526, 1981.
- [61] D.E. Knuth. *The Art of Computer Programming*, volume 2: Seminumerical Algorithms. Addison-Wesley, Reading, MA, 2nd edition, 1981.
- [62] P. Labbe and M.J. Bureau. Results from Statistical Analysis of Popular Pseudorandom Number Generators for Simulation. In 1993 Canadian Conference on Electrical and Computer Engineering, volume 1, pages 131–135, Vancouver, 1993. IEEE Canada.
- [63] J.C. Lagarias. Pseudorandom numbers. Statistical Science, 8:31–39, 1993.
- [64] P. L'Ecuyer. Efficient and portable combined random number generators. Comm. ACM, 31:742-749 and 774, 1988.
- [65] P. L'Ecuyer. Random numbers for simulation. Comm. ACM, 33:85-97, 1990.
- [66] P. L'Ecuyer. Testing random number generators. In *Proceedings of the 1992 Winter Simulation Conference*, pages 305–313. IEEE Press, 1992.
- [67] P. L'Ecuyer. Uniform random number generation. Ann. Oper. Res., 53:77-120, 1994.
- [68] P. L'Ecuyer. Combined Multiple Recursive Random Number Generators. Operations Research, 44:816–822, 1996.
- [69] P. L'Ecuyer. Bad Lattice Structures for Vectors of Non-Successive Values Produced by Some Linear Recurrences. INFORMS Journal on Computing, 9:57-60, 1997.
- [70] P. L'Ecuyer. Tables of Linear Congruential Generators of Different Sizes and Good Lattice Structure. *Math. Comp.*, 1998, to appear.
- [71] P. L'Ecuyer and F. Blouin. Linear congruential generators of order k > 1. In M. Abrams, P. Haigh, and J. Comfort, editors, Proceedings of the 1988 Winter Simulation Conference, IEEE Press, pages 432–439, 1988.
- [72] P. L'Ecuyer, F. Blouin, and R. Couture. A Search for Good Multiple Recursive Generators. *ACM Transactions on Modeling and Computer Simulation*, **3**:87–98, 1993.
- [73] P. L'Ecuyer, A. Compagner, and J.F. Cordeau. Entropy-Based tests for Random Number Generators, 1997. Submitted for publication.
- [74] P. L'Ecuyer and R. Couture. An Implementation of the Lattice and Spectral Tests for Multiple Recursive Linear Random Number Generators. INFORMS Journal on Computing., 9, 1997. To appear.
- [75] P. L'Ecuyer and Tezuka S. Structural properties for two classes of combined random number generators. *Math. Comp.*, **57**(196):735–746, 1991.
- [76] H. Leeb and S. Wegenkittl. Inversive and linear congruential pseudorandom number generators in selected empirical tests. *ACM Transactions on Modeling and Computer Simulation*, **7**(2):272–286, 1997. To appear.
- [77] 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. Harvard University Press.

- [78] P.A. Lewis, A.S. Goodman, and J.M. Miller. A pseudo-random number generator for the System/360. *IBM Syst. J.*, 8:136–146, 1969.
- [79] Ch. Ma. Implementation of a Monte Carlo code on a parallel computer system. *Parallel Computing*, **20**:991–1005, 1994.
- [80] N.M. MacLaren. The Generation of Multiple Independent Sequences of Pseudorandom Numbers. Appl. Statist., **38**:351–359, 1989.
- [81] N.M. MacLaren. A limit on the usable length of a pseudorandom sequence. J. Statist. Comput. Simul., 42:47–54, 1992.
- [82] G. Marsaglia. The structure of linear congruential sequences. In S. K. Zaremba, editor, Applications of Number Theory to Numerical Analysis, pages 248–285. Academic Press, New York, 1972.
- [83] 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.
- [84] G. Marsaglia and A. Zaman. A new class of random number generators. *The Annals of Applied Probability*, 1:462–480, 1991.
- [85] H. Niederreiter. Quasi-Monte Carlo methods and pseudo-random numbers. Bull. Amer. Math. Soc., 84:957–1041, 1978.
- [86] H. Niederreiter. Recent trends in random number generation and random vector generation. *Ann. Oper. Res.*, **31**:323–345, 1991.
- [87] H. Niederreiter. New methods for pseudorandom number and pseudorandom vector generation. *Proc.* 1992 Winter Simulation Conference (Arlington, Va., 1992), IEEE Press, Piscataway, N.J., pages 264–269, 1992.
- [88] H. Niederreiter. Random Number Generation and Quasi-Monte Carlo Methods. SIAM, Philadelphia, 1992.
- [89] H. Niederreiter. On a new class of pseudorandom numbers for simulation methods. J. Comput. Appl. Math., **56**:159–167, 1994.
- [90] H. Niederreiter. New developments in uniform pseudorandom number and vector generation. In H. Niederreiter and P. Jau-Shyong Shiue, editors, Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing, volume 106 of Lecture Notes in Statistics. Springer, 1995.
- [91] S.K. Park and K.W. Miller. Random number generators: good ones are hard to find. Comm. ACM, 31:1192-1201, 1988.
- [92] O. E. Percus and P. A. Whitlock. Theory and Application of Marsaglia's Monkey Test for Pseudorandom Number Generators. ACM Transactions on Modeling and Computer Simulation, 5 (2):87–100, 1995.
- [93] W. H. et al Press. *Numerical Recipes in C.* The Art of Scientific Computing. Cambridge University Press, 1992.
- [94] B.D. Ripley. The lattice structure of pseudo-random number generators. Proc. Roy. Soc. London Ser. A, 389:197–204, 1983.
- [95] B.D. Ripley. Uses and abuses of statistical simulation. Mathematical Programming, 42:53–68, 1988.
- [96] B.D. Ripley. Thoughts on pseudorandom number generators. *J. Comput. Appl. Math.*, **31**:153–163, 1990.

- [97] Y.S. Sherif and R.G. Dear. Development of a new composite pseudo random number generator. *Microelectronics and Reliability*, **30**:545–553, 1990.
- [98] E. Stadlober. Die Gitterstruktur linearer und kombinierter Kongruenzgeneratoren. Preprint, Department of Statistics, TU Graz, Austria, 1992.
- [99] E. Stadlober and R. Kremer. Sampling from discrete and continuous distributions with C-Rand. In G. Pflug and U. Dieter, editors, Simulation and Optimization, volume 374 of Lecture Notes in Economics and Math. Systems, pages 154–162. Springer-Verlag, Berlin, 1992.
- [100] E. Stadlober and F. Niederl. C-Rand: A package for generating nonuniform random variates. In *Compstat '94*, Software Descriptions, pages 63–64, 1994.
- [101] R.S. Szankovich. Ein statistischer Vergleich vierer Typen von Pseudozufallszahlen-Generatoren. Master's thesis, Sozial- und Wirtschaftswissenschaftliche Fakultät, Universität Wien, Austria, 1996.
- [102] R.C. Tausworthe. Random numbers generated by linear recurrence modulo two. *Math. Comp.*, **19**:201–209, 1965.
- [103] S. Tezuka, P. L'Ecuyer, and R. Couture. On Add-with-Carry and Subtract-with-Borrow Random Number Generators. ACM Transactions on Modeling and Computer Simulation, 3:315-331, 1993.
- [104] The Numerical Algorithms Group Limited. The NAG Fortran Library Manual, Mark 15, 1 edition, 1991.
- [105] G. Ugrin-Šparac. Stochastic Investigations of Pseudo-Random Number Generators. *Computing*, **46**:53–65, 1991.
- [106] G. Ugrin-Šparac and D. Ugrin-Šparac. On a Possible Error of Type II in Statistical Evaluation of Pseudo-Random Number Generators. *Computing*, **56**:105–116, 1996.
- [107] I. Vattulainen, T. Ala-Nissila, and K. Kankaala. Physical tests for random numbers in simulations. Phys. Rev. Lett., 73:2513–2516, 1994.
- [108] I. Vattulainen, T. Ala-Nissila, and K. Kankaala. Physical models as tests of randomness. Phys. Rev. E, 52:3205–3214, 1995.
- [109] I. Vattulainen, K. Kankaala, J. Saarinen, and T. Ala-Nissila. A comparative study of some pseudorandom number generators. Comp. Phys. Comm., 86:209-226, 1995.
- [110] S. Wegenkittl. Empirical testing of pseudorandom number generators. Master's thesis, University of Salzburg, 1995.
- [111] B.A. Wichmann and I.D. Hill. An efficient and portable pseudo-random number generator. *Appl. Statist.*, **31**:188–190, 1982. Corrections and remarks in the same journal by Wichmann and Hill **33** (1984) 123; McLeod **34** (1985) 198-200; Zeisl **35** (1986) 89.

Acknowledgments:

Research supported by the Austrian Science Foundation (FWF), project no. P11143-MAT. The author wishes to thank the PLAB-group, Salzburg, for supporting his work, especially the head of the group Peter Hellekalek for many helpful suggestions.