Inversive and linear congruential pseudorandom number generators in empirical tests

Hannes Leeb and Stefan Wegenkittl

Austrian Science Foundation (FWF), project no. P9285, P11143-MAT

We present results from a series of empirical tests of pseudorandom number generators. The tests cover a broad range of designs due to bit-oriented, efficient test statistics and a testing procedure in which we vary the sample size, dimension, and the statistics' resolution within vast bounds. Inversive generation methods pass the tests for a broader range of test parameters than linear generators with equal period length. The results exemplify how the lattice structure of linear generators can affect a stochastic simulation and suggest the use of inversive generators for cross–checking the results.

Categories and Subject Descriptors: G.3 [**PROBABILITY AND STATISTICS**]: Random number generation; G.3 [**PROBABILITY AND STATISTICS**]: Statistical software; I.6.0 [**SIMULATION AND MODELING**]: General

General Terms: Pseudorandom Numbers, Statistical Tests, Stochastic Simulation

Additional Key Words and Phrases: Random number generation, inversive and linear congruential generators, empirical tests, (overlapping) serial test

1. INTRODUCTION

Inversive (ICG) and explicit (EICG) inversive congruential pseudorandom number generators (PRNGs) were introduced by Eichenauer and Lehn [4] and by Eichenauer-Herrmann [6], respectively. Both exhibit remarkably good theoretical properties [5; 7]. For these generators, efficient implementations and extensive tables of parameters are available [2; 14]. In our study, we complement the theoretical analysis of inversive generators by empirical testing. We compare these generation methods with the well understood and widely used linear congruential generator (LCG [19]) by applying the serial test [10; 15] and the overlapping serial test (also called m-tuple test) [1; 12; 21; 25] using a bit-oriented partitioning scheme.

Affiliation: Institut für Mathematik, Universität Salzburg
Address: Hellbrunnerstr. 34, A-5020 Salzburg, Austria. WWW: http://random.mat.sbg.ac.at/
e-mail: leeb@random.mat.sbg.ac.at, ste@random.mat.sbg.ac.at

Our tests are designed to distinguish between PRNGs of the same period length by means of their load capability in standard equidistribution tests, i.e. by the range of test parameters for which they pass. The results show different behavior of inversive and linear generators if test parameters like the sample size, the dimension, or the number of significant bits are varied.

We introduce the LCG, ICG and EICG in Section 2. In Section 3, we introduce the test statistics and the notion of "load test". For various linear and inversive generators, test results are presented and discussed in Section 4. Testing LCGs, ICGs and EICGs with comparable period lengths, we find that the inversive generators generally pass our tests for a broader range of test parameters than the linear ones. Conclusions are given in Section 5.

2. PSEUDORANDOM NUMBER GENERATORS

A pseudorandom number generator is a deterministic algorithm producing a sequence $(x_i)_{i\geq 0}$ of numbers in $[0,1)$ which, for virtually all generators used for computer simulation, is purely periodic. Confined to the period, these numbers should behave like independent realizations of a random variable uniformly distributed on $[0,1)$. See [17] for a survey on PRNGs.

The linear congruential generator is today's most widely used and most thoroughly understood pseudorandom number generator. Defined by integer parameters M, a, b, and u_0, a linear congruential generator (LCG(M,a,b,u_0), for short) produces a sequence $(u_i)_{i\geq 0}$ of integers by $u_{i+1} = au_i + b (\mathrm{mod}\, M)$, i.e. u_{i+1} is the integer remainder of dividing $au_i + b$ by M. A sequence $(x_i)_{i\geq 0}$ of pseudorandom numbers in $[0,1)$ is defined by $x_i = u_i/M$. The LCG's period is thus at most M and depends on the choice of parameters [23, p.169].
For the full or maximal period LCGs considered below and fixed dimension $s \geq 1$, the s-tuples $(x_i, \ldots, x_{i+s-1})$, $i \geq 0$, form a grid in s dimensions [23, Theorem 7.6]. Moreover, the same applies to the points (x_i, x_{i+s}), $i \geq 0$: they form a two-dimensional grid, which causes strong long-range correlations for specific critical values [3] of s.

The inversive congruential generator is a method based on modular inversion. For a (large) prime p and an integer u with $0 \leq u < p$, let $\overline{u} = u^{p-2} (\mathrm{mod}\, p)$ if $u \neq 0 (\mathrm{mod}\, p)$ and $\overline{u} = 0$ otherwise. With suitable parameters p, a, b, and u_0, the inversive congruential generator (ICG(p,a,b,u_0), for short) produces integers $(u_i)_{i\geq 0}$ by $u_{i+1} = a\overline{u_i} + b (\mathrm{mod}\, p)$, and pseudorandom numbers $(x_i)_{i\geq 0}$ by $x_i = u_i/p$. The maximal period length p is obtained if and only if $x^2 - bx - a$ is an IMP-polynomial over the finite field of p elements $\mathbf{Z}_p$ [11]. Tables of parameters were compiled in [14]. In the following, we only consider maximal period ICGs.
In contrast to the LCG, the overlapping s-tuples produced by the ICG literally avoid the hyperplanes in s dimensions and form no grid structure [5]. As a consequence of this result, the points (x_i, x_{i+s}), $i \geq 0$, form no grid in the plane [18, Proposition 5.4].

The explicit inversive generator is very similar to the ICG when theoretical properties are concerned but choosing proper parameters is significantly easier. For a (large) prime p and integers a and b, the explicit inversive generator (EICG(p,a,b), for short) produces a sequence $(u_i)_{i\geq 0}$ of integers by $u_i = \overline{ai + b}$, and pseudorandom numbers $(x_i)_{i\geq 0}$ by $x_i = u_i/p$. Any choice of a and b with $a \neq 0 (\mathrm{mod}\, p)$ gives

Name	Generator	Period
RANDU	$\mathrm{LCG}(2^{31}, 65539, 0, 1)$	2^{29}
ANSIC	$\mathrm{LCG}(2^{31}, 1103515245, 12345, 12345)$	2^{31}
MINSTD	$\mathrm{LCG}(2^{31}-1, 16807, 0, 1)$	$2^{31}-2$
FISH	$\mathrm{LCG}(2^{31}-1, 950706376, 0, 1)$	$2^{31}-2$
ICG	$\mathrm{ICG}(2^{31}-1, 1, 1, 0)$	$2^{31}-1$
EICG1	$\mathrm{EICG}(2^{31}-1, 1, 0)$	$2^{31}-1$
EICG7	$\mathrm{EICG}(2^{31}-1, 7, 0)$	$2^{31}-1$

Table 1. Selected generators

a sequence of maximal period p. Note that this generator is not a recursive one and allows the explicit computation of x_i given i. As a result, it is particularly easy to parallelize the EICG by splitting its output sequence into disjoint parts. Concerning hyperplane structures and long-range correlations, the EICG has similar favorable properties as the ICG [7].

For our empirical tests, we have selected the generators in Table 1. The LCGs cover the wide range of quality delivered by linear generators, ranging from RANDU (worst) to FISH (best). RANDU, formerly part of IBM's Scientific Subroutine Package, exhibits a devastating defect in three dimensions: its points (x_i, x_{i+1}, x_{i+2}) all lie in just fifteen parallel planes. ANSIC is the generator employed by the ANSI C `rand()` function, BSD version. MINSTD, introduced by Lewis et. al. as random number generator for IBM's System/360 [20], was later proposed as a 'minimal standard' generator by Park and Miller [24]. Finally, FISH is one of the best found by Fishman and Moore [10] in an exhaustive search among all maximum period LCGs with $M = 2^{31} - 1$ and $b = 0$; among the more than 534 million candidate generators, they selected the 414 best with respect to their lattice structure in dimensions 2 to 6. ICG, EICG1, and EICG7 have been chosen arbitrarily among the maximal period inversive generators with modulus $p = 2^{31} - 1$. Note that except for RANDU which is known to be defective anyway, the generators' periods provide for a fair comparison.

3. TEST STATISTICS

We subject the generators presented in the previous section to two equidistribution tests in various dimensions, the serial test [15] and the overlapping serial test [12; 21] (see also `http://stat.fsu.edu/~geo/diehard.html`), in both cases using a generalization of the standard partitioning technique. With each test statistic, we use a two-level testing procedure following the recommendations in [15]. The procedure's basic structure is this: we repeatedly compute the (overlapping) serial test from consecutive samples in the sequence in the first step and compare the empirical distribution of the results to their desired, theoretical one in the second. In the following, we describe this procedure for both test statistics in detail.

To compute the serial test in fixed dimension $s \geq 1$, one first transforms a sample $x_1, \ldots, x_{sN}$ of sN pseudorandom numbers ($N \gg 1$) to a sample $y_1, \ldots, \ldots, y_{sN}$ of integers in the set $\mathbf{Z}_m = \{0, \ldots, m-1\}$ for some m. A set of N vectors of length s is obtained by forming non-overlapping s-tuples from consecutive integers $y_1, \ldots, y_{sN}$. Finally, the chi-square test statistic is computed from these points

using, as partition, the set $\{\{\mathbf{p}\} : \mathbf{p} \in \mathbf{Z}_m^s\}$. As transformation, Knuth [15] suggests $y_i = \lfloor m x_i \rfloor$ for each i ($\lfloor x \rfloor$ denotes the largest integer not exceeding x). We use $y_i = d_{k,l}(x_i)$ with $d_{k,l}(x) = \lfloor 2^{k+l-1} x \rfloor \pmod{2^l}$, where $x \in [0,1)$ and k and l are fixed, positive integers; this maps $[0,1)$ to $\mathbf{Z}_{2^l}$, i.e. $m = 2^l$. Note that $d_{k,l}(x)$ depends only on the k-th to the $(k+l-1)$-th most significant bits of x (excluding the sign bit). The sample size is set to $N = 6 \cdot 2^{sl}$ in order to get a good approximation of the chi-square distribution. Let $T(s,k,l)$ denote the value of the serial test statistic thus defined.

In the procedure's second step, we compute K values $T_1(s,k,l), \ldots, T_K(s,k,l)$, whose empirical distribution is compared to the χ^2-distribution with $2^{sl} - 1$ degrees of freedom. Denote the distribution function of the latter by F, and let $\tilde{F}(t)$ be the empirical distribution of the K values; i.e. $\tilde{F}(t) = \#\{i : T_i(s,k,l) \leq t, 1 \leq i \leq K\}/K$, where $\#\mathcal{S}$ denotes the cardinality of a set $\mathcal{S}$. We measure the distance between $\tilde{F}$ and F with respect to the sup-norm using the two-sided Kolmogorov-Smirnov (KS) test: $KS(T_i(s,k,l) : 1 \leq i \leq K) = \sqrt{K} \sup_{-\infty < t < \infty} |\tilde{F}(t) - F(t)|$. For the sample size in this second step, we choose $K = 64$; the remaining parameters s, k and l are chosen from a range of values as described in Section 4. The critical region of $KS(T_i(s,k,l) : 1 \leq i \leq 64)$ at the 1% level of significance is $[1.63, \infty)$ [13, p.183f].

The same basic steps are performed for the overlapping serial test: let M and s be fixed, positive integers. We transform a sample $x_1, \ldots, x_M$ of pseudorandom numbers to an integer sample by $y_i = d_{1,4}(x_i)$ $(1 \leq i \leq M)$. With these integers, the overlapping serial test is computed as the difference of two chi-squares from $(y_1, \ldots, y_s), (y_2, \ldots, y_{s+1}), \ldots, (y_M, y_1, \ldots, y_{s-1})$ as described in [12]. This gives a test statistic $T^{(o)}(s,M)$ which is asymptotically χ^2-distributed with $2^{4s} - 2^{4(s-1)}$ degrees of freedom.

In the second step, we compute L independent values ${T^{(o)}}_1(s,M), \ldots, {T^{(o)}}_L(s,M)$, which we subject to the two-sided KS test as above. For the overlapping serial test, we choose $L = 32$. The critical region for $KS({T^{(o)}}_i(s,M) : 1 \leq i \leq 32)$ at the 1% level of significance is approximately $[1.59, \infty)$ [13, p.183f].

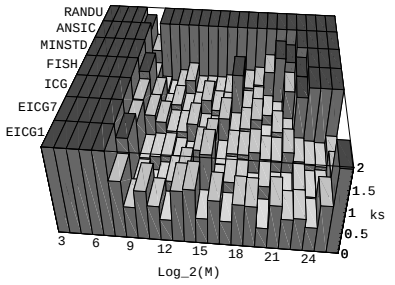


Fig. 1.a: $KS({T^{(o)}}_i(4,M) : 1 \leq i \leq 32)$

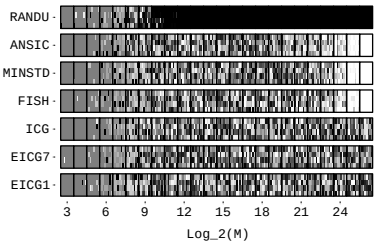


Fig. 1.b: $U_i(4,M) : 1 \leq i \leq 32$

Fig. 1. Overlapping serial test results for dimension $s = 4$ and various sample sizes M

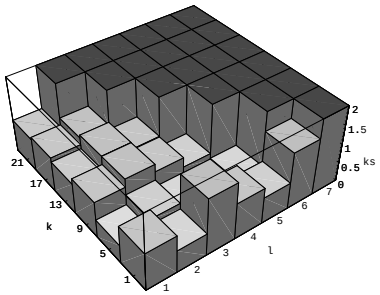
Figure 1 illustrates the graphics (a discussion of the results is given in Section 4). In all the results, we have truncated the two-sided KS statistic to $\min\{KS, 2\}$; values in the critical region at the 1%-level are colored dark grey. Figure 1.a shows the behavior of PRNGs in the overlapping serial test in dimension $s = 4$ for sample sizes $M = 2^3, 2^4, \ldots, 2^{26}$. For $M < 2^8$, the actual distribution of $T^{(o)}(4, M)$ has not converged sufficiently to the asymptotic χ^2, which causes the test to be failed due to the lack of fit. Observe that almost all generators enter the range of acceptable KS values at $M = 2^8$. For $M \geq 2^8$, a generator gives reasonable results up to some upper bound which *differs* from generator to generator. We call this setup a "load test" since we can measure the load-capability, i.e. the range of sample sizes for which a generator passes the test.
In Figure 1.b, for the same values of s and M, and for the same generators, the individual values ${T^{(o)}}_i(s, M)$ have been transformed to the upper tail probabilities $U_i(s, M) = 1 - F({T^{(o)}}_i(s, M))$, $1 \leq i \leq 32$, where F is the distribution function of the χ^2 with $2^{4s} - 2^{4(s-1)}$ degrees of freedom. In the graphic, this gives 32 values for each M and each generator, which we plot as shades of grey, ranging from black ($U_i(s, M) = 0$) to white ($U_i(s, M) = 1$); note that, according to the null-hypothesis, these 32 values should look like uncorrelated noise generated by a uniformly distributed source.

4. RESULTS

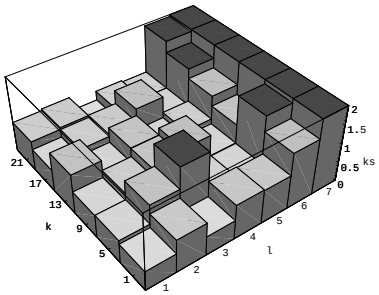
We apply the serial test in two setups: in the first, we keep the dimension s fixed and vary k and l; in the second, we fix l and vary s and k. For $s = 3$, $k = 1, 5, 9, 13, 17, 21$, and $l = 1, 2, \ldots, 7$, Figure 2 shows the results for the linear generators ANSIC and FISH; for the same parameters, results of the inversive generators ICG and EICG1 are shown in Figure 3. For $l = 2$, $s = 1, 2, \ldots, 10$ and $k = 1, 5, 9, \ldots, 29$, results for ANSIC and FISH are shown in Figure 4 and results for ICG and EICG1 in Figure 5. Note that in this second setup, only two bits of each pseudorandom number are taken into account. We do not show the serial test results of RANDU, MINSTD and EICG7 here; RANDU fails the serial test for almost all parameters, MINSTD gives results between those of ANSIC and FISH, and EICG7 behaves essentially like EICG1.
For the overlapping serial test, we show the results of all generators for sample size $M = 2^{18}, \ldots, 2^{26}$ in dimension $s = 2$, $s = 3$ and $s = 5$ in Figure 6, 7 and 8, respectively. The results for $s = 4$ were already given in Figure 1. Note that in the given configuration, the overlapping serial test considers only the first four bits of each pseudorandom number.

The inversive generators pass both the serial test and the overlapping serial test for a wider range of test-parameters than the linear generators. If either the number of bits taken into account (Figures 2,3), or the dimension (Figures 4,5), or the sample size (Figure 1 and Figures 6 to 8) gets too large, all the linear generators from Table 1 exhibit defects. We conclude that with respect to our tests, the inversive generators have a higher load-capability than the linear ones.
We attribute this effect to the presence of grid structures in the s-dimensional points produced by the LCGs and the absence of such structures in points obtained from ICGs and EICGs. The reason for this is twofold: first, the LCGs' behavior in our tests coincides with the structure of their corresponding lattice as measured by the

spectral test [15]; second, the piece-wise linear transformation $d_{k,l}$ used to partition the unit-cube is prone to give interference effects with linear structures in the point set. This is exemplified in the plots of $U_i(s, M)$ in Figure 1 and Figures 6 to 8: for the linear generators, the rejected values in the KS test tend to correspond to either regular patterns or almost constant values of the $U_i(s, M)$. White squares indicate that the $T^{(o)}$ statistic has encountered a too regular distribution of the integer s-tuples in $\mathbf{Z}_{16}^s$. RANDU even yields black squares indicating that certain values in $\mathbf{Z}_{16}^s$ occur far too often. In both cases, the respective generator has lost its ability to randomize the test statistic.

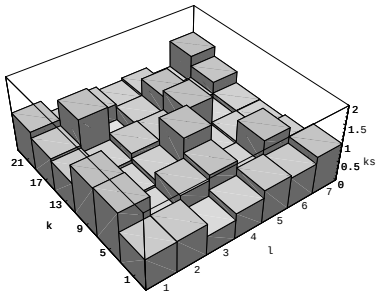


ANSIC: $KS(T_i(3, k, l) : 1 \leq i \leq 64)$

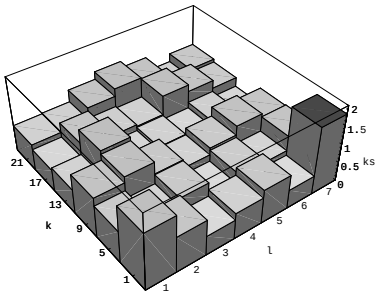


FISH: $KS(T_i(3, k, l) : 1 \leq i \leq 64)$

Fig. 2. Serial test results for dimension $s = 3$ and various k and l



ICG: $KS(T_i(3, k, l) : 1 \leq i \leq 64)$

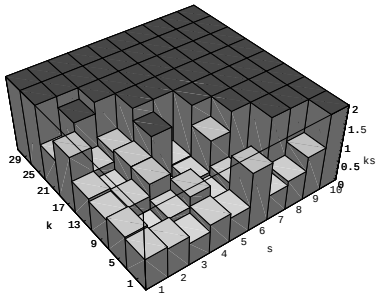


EICG1: $KS(T_i(3, k, l) : 1 \leq i \leq 64)$

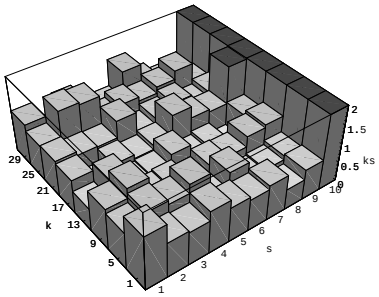
Fig. 3. Serial test results for dimension $s = 3$ and various k and l

5. CONCLUSIONS

In the last 20 years, empirical evidence has shown that theoretical analysis of equidistribution properties is an important indicator for the performance of PRNGs in empirical tests and stochastic simulations. Discrepancy estimates available for inversive generators thus have led us to expect that EICGs and ICGs should come off well in empirical tests.

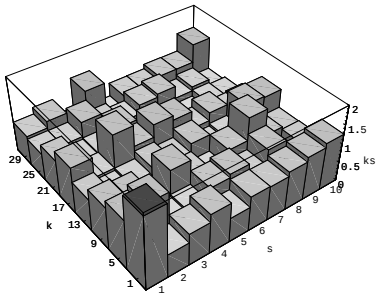


ANSIC: $KS(T_i(s,k,2) : 1 \le i \le 64)$

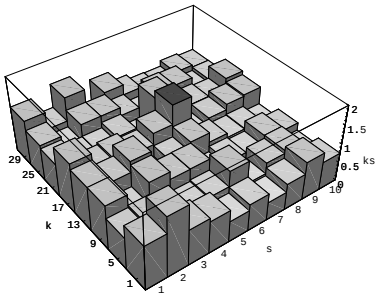


FISH: $KS(T_i(s,k,2) : 1 \le i \le 64)$

Fig. 4. Serial test results for $l = 2$ and various k and s



ICG: $KS(T_i(s,k,2) : 1 \le i \le 64)$



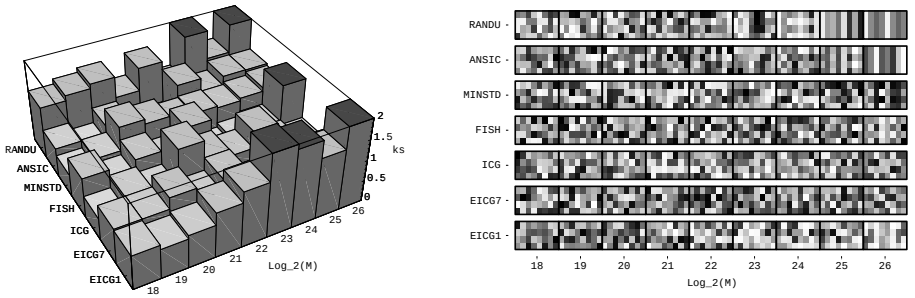
EICG1: $KS(T_i(s,k,2) : 1 \le i \le 64)$

Fig. 5. Serial test results for $l = 2$ and various k and s

Our empirical study has confirmed the theoretical results: we have encountered two test statistics that are able to distinguish between linear and inversive generation methods by considering the range of bits (Figures 2,3), the range of dimensions (Figures 4,5), or the range of sample sizes (Figures 1,6 to 8) for which the PRNGs pass the test. The results show that inversive generators perform well with respect to these three criteria.

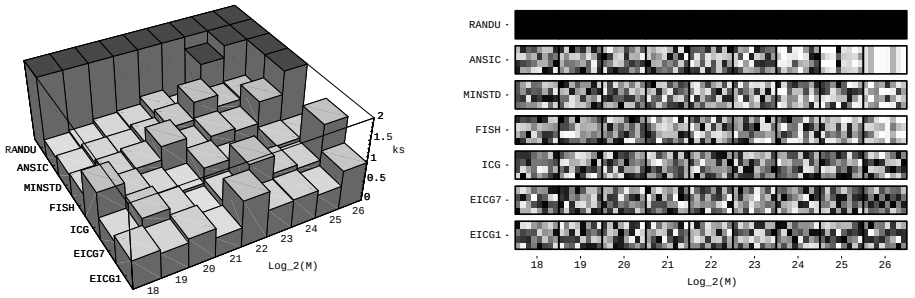
Both the serial test and the overlapping serial test have been applied to random number generators before. For example, Fishman and Moore applied the serial test for $s = 1, 2, 3$, $N = \lfloor 200000/s \rfloor$, $k = 1$, $l = 12/s$, and $M = 100$ to FISH and comparable linear generators and found no significant defects [10]. Altman applied the overlapping serial test as a one-level test (no additional KS-test) for $k = s = 3$ and $M = 2000$ to RANDU and MINSTD, and found that RANDU fails while MINSTD passes. Vattulainen et. al. extended Altman's work to $k = s = 3$, $M = 5000$ and $L = 1000$; they found no bit-level correlations in MINSTD. A detailed comparison shows that our result are consistent with the preceeding work, and that our approach of varying the tests' parameters within as large as possible ranges enchances the stringency of the tests.

One drawback of inversive generators is efficiency, which restricts their use in two



$KS(T^{(o)}{}_i(2, M) : 1 \le i \le 32)$ $U_i(2, M) : 1 \le i \le 32$

Fig. 6. Overlapping serial test results for dimension $s = 2$ and various sample sizes M



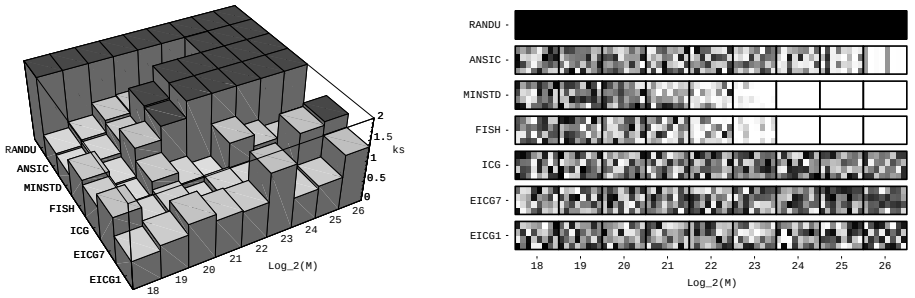
$KS(T^{(o)}{}_i(3, M) : 1 \le i \le 32)$ $U_i(3, M) : 1 \le i \le 32$

Fig. 7. Overlapping serial test results for dimension $s = 3$ and various sample sizes M

respects. First, even efficient implementations [2] are about three times slower than LCGs of the same period length. This is an issue if the simulation spends a large fraction of time for generating random numbers; if this fraction is small, however, inversive generators do not slow down the simulation significantly. Second, while more sophisticated linear methods like the combined linear generator, the multiple recursive linear generator [15], or digital methods like the twisted GFSR [22] give huge periods far beyond 2^{31} and hence outperform the LCGs in Table 1 as well [16], no comparable technique is yet available for the inversive methods (combined inversive generators are possible [8], but their computational efficiency decreases with the number of components). However, the ICG and EICG are relatively new methods, and possibly more efficient inversive generators are currently being investigated [9].

Regularities are present in every sequence of PRNs [17]. The question is always whether the actual application is sensitive to these regularities and yields biased results, see [18; 26]. However, we did not yet encounter a test statistic that is sensitive to regularities within the class of inversive generation methods.

Based on the theoretically predicted and empirically confirmed stability of inversive generators, we suggest to use ICGs and EICGs *in parallel* to classical generators whenever possible to improve confidence in simulation results.



$KS(T^{(o)}{}_i(5, M) : 1 \leq i \leq 32)$ $U_i(5, M) : 1 \leq i \leq 32$

Fig. 8. Overlapping serial test results for dimension $s = 5$ and various sample sizes M

ACKNOWLEDGMENTS

The authors wish to thank their theses' supervisor, Peter Hellekalek, head of the pLab research group, for his support.

REFERENCES

[1] N. S. Altman. Bit–wise behavior of random number generators. *SIAM J. Sci. Stat. Comput.*, **9**(5):941–949, 1988.

[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] 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.

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

[5] J. Eichenauer-Herrmann. Inversive congruential pseudorandom numbers: a tutorial. *Int. Statist. Rev.*, **60**:167–176, 1992.

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

[7] J. Eichenauer-Herrmann. Pseudorandom number generation by nonlinear methods. *Int. Statist. Rev.*, **63**:247–255, 1995.

[8] J. Eichenauer-Herrmann and F. Emmerich. A review of compound methods for pseudorandom number generation. In P. Hellekalek, G. Larcher, and P. Zinterhof, editors, *Proceedings of the 1st Salzburg Minisymposium on Pseudorandom Number Generation and Quasi-Monte Carlo Methods, Salzburg, Nov 18, 1994*, volume ACPC/TR 95-4 of *Technical Report Series*, pages 5–14. ACPC – Austrian Center for Parallel Computation, University of Vienna, Austria, 1995.

[9] J. Eichenauer-Herrmann and H. Niederreiter. Digital inversive pseudorandom numbers. *ACM Trans. Model. Comput. Simul.*, **4**:339–349, 1994.

[10] 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. See erratum, ibid., **7**:1058, 1986.

[11] 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.

[12] I. J. Good. The serial test for sampling numbers and other tests for randomness. *Proc. Cambridge Philosophical Society*, **49**:276–284, 1953.

[13] J. Hartung, B. Elpelt, and K.H. Klösener. *Statistik*. R. Oldenburg, Munich, 9th edition, 1993.

[14] P. Hellekalek, K. Entacher, S. Wegenkittl, and A. Weingartner. Extension of the tables of IMP-polynomials. Report D5H-5, CEI-PACT Project, WP5.1.2.1.2, Research Institute for Software Technology, University of Salzburg, Austria, 1995.

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

[16] P. L'Ecuyer. Testing random number generators. *Proc. 1992 Winter Simulation Conference (Arlington, Va., 1992), IEEE Press, Piscataway, N.J.*, pages 305–313, 1992.

[17] P. L'Ecuyer. Uniform random number generation. *Ann. Oper. Res.*, **53**:77–120, 1994.

[18] 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/team/`.

[19] 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.

[20] P.A.W. Lewis, A.S. Goodman, and J.M. Miller. A pseudo-random number generator for the System/360. *IBM System's Journal*, **2**:136–146, 1969.

[21] 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.

[22] M. Matsumoto and Y. Kurita. Twisted GFSR generators II. *ACM Trans. Model. Comput. Simul.*, **4**:254–266, 1994.

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

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

[25] 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.

[26] S. Wegenkittl. Empirical testing of pseudorandom number generators. Master's thesis, University of Salzburg, 1995. Available on the internet at `http://random.mat.sbg.ac.at/team/`.