Considerations on Correlations in Shift–Register Pseudorandom Number Generators and Their Removal

Andreas Heuer and Burkhard Dünweg[1]

Max–Planck–Institut für Polymerforschung, Postfach 3148, D–55021 Mainz, Germany

and

Alan M. Ferrenberg[2]

Institut für Physik, Johannes–Gutenberg–Universität Mainz, Postfach 3980, D–55099 Mainz, Germany

Abstract

We present a simple calculation quantitatively explaining the triplet correlations in the popular shift–register random number generator "R250", which were recently observed numerically by Schmid and Wilding, and are known from general analysis of this type of generator. Starting from these considerations, we discuss various methods to remove these correlations by combining different shift–register generators. We implement and test a particularly simple and fast version, based on an XOR combination of two independent shift–register generators with different time lags. The results indicate that this generator has much better statistical properties than R250, while being only a factor of two slower. This is consistent with previous analytical considerations and successful applications of this type of generator. The known nine–point correlations still present in the generator are quantitatively understood by our simple arguments.

PACS: 02.70.Lq, 05.40.+j, 05.50.+q, 75.40.Mg

Keywords: Random number generators, Shift–register generators
R250, Triplet correlations
Statistical tests, Application tests
Ising model, Wolff algorithm

[1]To whom correspondence should be addressed (phone: +49–6131–379–198, fax: +49–6131–379–340, e–mail: duenweg@mpip-mainz.mpg.de)

[2]Present and permanent address: University Computing and Networking Services, and Center for Simulational Physics, The University of Georgia, Athens GA 30602, USA

1 Introduction

Shift–register pseudorandom number (PN) generators [1–4] have been widely used in many areas of computational and simulational physics. In addition to being simple to implement in a machine–independent manner, these generators are also quite fast, often requiring only a single operation to produce a PN. Moreover, shift–register generators also possess rather long periods which make them particularly well–suited for applications which require many PNs. Unfortunately, several recent studies have pointed out flaws in the statistical properties of these generators, which can result in systematic errors in Monte Carlo simulations. Typical examples included the Wolff algorithm [5–7], cluster properties [8], random and self–avoiding walks [9–11] as well as the 3D Blume–Capel model using local Metropolis updating [12]. It is important to note, however, that despite the known deficiencies in these generators, which can potentially yield erroneous results in Monte Carlo calculations, they remain quite popular and are still in widespread use.

Standard statistical tests of randomness [13], and even more so application tests, can often reveal a substantial amount of information on the hidden correlations in PN generators. Nevertheless, an improved understanding would be obtained if one could directly relate the flaws in the applications to specific statistical correlations. Previous authors have pointed out that the problems with shift–register PN generators are connected with triplet correlations [9–11, 14], which are, even more specifically, directly related to the "time lag" involved in the generator [12, 15]. Actually, they are a natural consequence of the algorithm, and have, on the bit level, been discussed in quite some detail in the comprehensive analysis of Compagner *et al.* [14–16]. In this paper we focus on the correlation observed by Schmid and Wilding [12], and discuss the effect for normalized real PNs rather than for bits only. Starting from the observation that also the reverse correlation can be induced by slightly modified shift–register PN generators, we then discuss ways of removing the triplet correlations. The simplest and fastest (approximately half as fast as the uncorrected generator) of these methods turns out to be one which has been proposed [15, 16] and used [17, 18] previously. Theoretical analysis of this scheme reveals some residual higher–order correlations [16], which we however were unable to resolve in our application tests. The detailed mechanism of *how* the triplet correlations finally introduce the deviations in the applications could not be addressed in the present study, and remains unclear.

One should note that the research effort of recent years has produced PN generators with very good statistical properties. In particular, we would like to mention the RANLUX generator by Lüscher and James [19, 20]. It is based on a map for which there are strong theoretical arguments that its correlations are short–ranged and hence can be removed by discarding large chunks of PNs. Without discarding, the generator is just the well–

known Marsaglia–Zaman generator [21], which has actually performed worse than R250 (the most popular version of shift–register PN generators [4]) in statistical tests [13]. Very good statistical properties are obtained if only roughly 10 % of the PNs are actually used [19, 20]; however, the generator is then rather slow, needing 700 nanoseconds per PN on a Cray–YMP [19]. Conversely, R250 needs only 22 nanoseconds, and the modified version R250/521, which we discuss below, 41 nanoseconds. Thus, our view on the question which generator one should use can be summarized as follows: In case it is affordable, a generator with well–established quality like RANLUX is probably the method of choice. However, for some applications like simple lattice models, where the PN generation is actually the most time–consuming part of the overall program and the interesting regime can only be accessed by high–statistics studies, a faster generator is needed. For these applications, we believe that the generator to be described below is an excellent compromise between speed and statistical quality. In case one needs even better statistical properties, it is possible to systematically improve the generator, as explained below. Hence, for both methods it is possible to trade in speed for quality. However, for these specialized applications this trade–off is much more economical for the improved shift–register generator: If the application requires a production rate of 10 PNs per microsecond YMP–time or above, then clearly the improved shift–register generator (which is at least as good as R250, for all properties) is superior to RANLUX (which then is just the Marsaglia–Zaman generator), as seen from the results of Ref. [13].

Another rather interesting PN generator has recently been proposed and used by Ziff [22]. This generator is also quite fast, needing the same number of operations as the one to be discussed below, while rather good statistical properties have been found [9, 22]. However, theoretical analysis shows that this generator has somewhat larger correlations, as explained in Sec. 3.

The remainder of this paper is organized as follows: Sec. 2 contains the analytical considerations which explain the observations by Schmid and Wilding [12], while Sec. 3 describes the reasoning which leads to the improved generator. Numerical tests are described in Sec. 4, while Sec. 5 summarizes our conclusions.

2 Explanation of Triplet Correlations

Shift–register PN generators can be viewed as a special case of the general lagged–Fibonacci generators [23]

$$Y_n = Y_{n-p} \diamond Y_{n-q}, \tag{1}$$

where the binary operator $\diamond$ acting on the integers Y_{n-p} and Y_{n-q} can be either one of the arithmetic operators $-$, $+$ and $\times$, or the bitwise exclusive–or (XOR, see Table 1) denoted by the symbol $\oplus$. In the case of the arithmetic operators, the operations can lead to intermediate results which fall outside the range of integers for a particular machine and which must, therefore, be manually folded back with a time–consuming modulo operation into the proper range to ensure a machine–independent implementation. The exclusive–or operation, on the other hand, keeps all numbers within the proper range and therefore requires no additional operations. The most common example is the widely employed "R250" generator [4] where $p = 250$, $q = 103$ (special choices of p and q are necessary to ensure maximum period length [4]; instead of (250,103) one could also use, e. g., (521,168) [24]). It should be noted that $q = 147 = 250 - 103$ is equally valid, due to a "time–reversal symmetry" of the generator, and that the routine can be vectorized by splitting up the recursion loop into blocks of length $\min(q, p - q)$.

In very recent work Schmid and Wilding [12] analyzed the three point average $\langle X_n X_{n-k} X_{n-p} \rangle$ (X_n denoting the normalized real random number $X_n = Y_n/(2^N - 1)$ formed from the positive N–bit integer Y_n) for different values of k. Only for $k \neq q$ did they obtain the expected value $(1/2)^3 = 0.125$, whereas for $k = q$ they found a value of approximately 0.107.

In order to understand this, let us consider two N–bit integer numbers

$$A = \sum_{n=0}^{N-1} a_n 2^n, \qquad B = \sum_{n=0}^{N-1} b_n 2^n \tag{2}$$

with $a_n, b_n \in \{0, 1\}$. These numbers are the input for the XOR operation, and for simplicity, we assume that they are bitwise statistically independent. This assumption allows us to straightforwardly calculate the three point average

$$W = \langle X_n X_{n-q} X_{n-p} \rangle = \langle ABC \rangle / (2^N - 1)^3, \tag{3}$$

where

$$C = \sum_{n=0}^{N-1} c_n 2^n = \sum_{n=0}^{N-1} (a_n \oplus b_n) 2^n. \tag{4}$$

This is well–justified since no strong pair correlations have been found in R250, and the generator leaves the mth and nth bit independent for $m \neq n$.

Calculating $\langle ABC \rangle$ by using their explicit binary representation one obtains four different types of terms:

$$d_1 \equiv \langle a_n b_n (a_n \oplus b_n) \rangle, \tag{5}$$

$$d_2 \equiv \langle a_n b_m (a_n \oplus b_n) \rangle \quad (m \neq n), \tag{6}$$

$$d_3 \equiv \langle a_n b_n (a_m \oplus b_m) \rangle \quad (m \neq n), \tag{7}$$

and

$$d_4 \equiv \langle a_n b_m (a_k \oplus b_k) \rangle \quad (k \neq m \neq n), \tag{8}$$

for which we find the values $d_1 = 0$, $d_2 = \langle b_m \rangle \langle a_n c_n \rangle = (1/2)(1/4) = 1/8$, $d_3 = \langle a_n \rangle \langle b_n \rangle \langle c_m \rangle = (1/2)^3 = 1/8$, $d_4 = \langle a_n \rangle \langle b_m \rangle \langle c_k \rangle = (1/2)^3 = 1/8$. Hence $\langle a_n b_m c_k \rangle$ can be always viewed as if a_n, b_m, c_k were independent except for the case $k = m = n$.

Neglecting the anomalies induced by the occurrence of these "diagonal" ($k = m = n$) terms, hence setting $\langle a_n b_n c_n \rangle = 1/8$, one would obtain $W = 1/8$. In reality, however, $\langle a_n b_n c_n \rangle$ vanishes, and for calculating the true value of W one has to subtract the "diagonal" terms:

$$W = (1/8) - (1/8) \frac{\sum_{n=0}^{N-1} (2^n)^3}{(2^N - 1)^3}, \tag{9}$$

since the error in the nth bit occurs with a weight of $(2^n)^3$. Since

$$\sum_{n=0}^{N-1} 8^n = \frac{8^N - 1}{8 - 1} = (1/7)(8^N - 1), \tag{10}$$

we find

$$W = \frac{1}{8} \left[1 - \frac{1}{7} \frac{(8^N - 1)}{(2^N - 1)^3} \right], \tag{11}$$

which, for large N, is $W = 3/28 \approx 0.107$. Conversely, for $N = 1$ we find $W = 0$ since then *only* diagonal terms occur. Hence the numerically observed triplet correlation can be directly traced back to the relation $\langle a_n b_n c_n \rangle = 0$ ($\neq 1/8$).

Furthermore, one can do the same calculation for a variant of R250, where the XOR operation is replaced by the not–exclusive–or (NXOR, $\bar{\oplus}$, see Table 2). It should be noted that such a generator is as valid as the conventional R250, since the NXOR sequence can be viewed as the exact bitwise mirror image of an R250 sequence. This is obvious from the bitwise relation $a \bar{\oplus} b = \mathrm{NOT}(\bar{a} \oplus \bar{b})$ (where $\bar{a} = \mathrm{NOT}(a)$ is a's complement or negative). This means that one can generate the NXOR sequence by either starting from values Y_n and using NXOR, or by starting from $\bar{Y}_n$, then using the standard R250, and afterwards negating the whole sequence. For the NXOR generator one finds $\langle a_n b_n c_n \rangle = 1/4$, while $d_2 = d_3 = d_4 = 1/8$. An analogous calculation to above then yields $W = 1/7 \approx 0.143$, i. e. one obtains the same absolute value of the systematic deviation $|W - 0.125|$, but in the opposite direction. This suggests that a suitable combination of both methods should be free of triplet correlations.

3 Improved Generators

A trivial way to decrease the error $\langle X_n X_{n-k} X_{n-l} \rangle - 0.125$ for $k = q$ and $l = p$ is to randomly choose between two standard R250s, or to randomly mix the output of one R250. While these procedures would only "smear out" the error, we here pursue the idea of combining XOR and NXOR generators, or to combine the XOR operation with bitwise negations.

The most straightforward way to implement this would be to run two generators (one XOR sequence and one NXOR), and then to alternate between the two sequences, such that

$$Y_{2n} = Y_{2n-2p} \oplus Y_{2n-2q}, \qquad Y_{2n+1} = Y_{2n+1-2p} \bar{\oplus} Y_{2n+1-2q}. \tag{12}$$

This algorithm would be quite fast, since only for every second PN is an additional XOR operation for the calculation of the complement required. Moreover, the originally wrong triplet correlation (which here is of course $\langle X_n X_{n-2p} X_{n-2q} \rangle$) is corrected, since the errors from the two sequences exactly cancel out. Nevertheless, we have not pursued this idea further, as the correction comes at the price of another correlation: One immediately sees that the procedure above leads to a wrong high–frequency Fourier component, since $(2M)^{-1} \sum_{n=0}^{2M-1} (-1)^n \langle X_n X_{n-2p} X_{n-2q} \rangle = -1/56$, which differs from the ideal value zero. Moreover, the six–point correlation function $\langle X_n X_{n-2p} X_{n-2q} X_{n+1} X_{n+1-2p} X_{n+1-2q} \rangle$ is of course also wrong, with a relative deviation from the ideal value $(1/2)^6$ of $1/49$.

A generalization would then be to "randomly" alternate between the two generators, i. e. to have a third independent generator decide if the next PN should be taken from the XOR sequence or from the NXOR sequence. One could conveniently use for this third generator an R250 which only runs on the least significant bit, i. e. yields only zeroes or ones. However, although rather good statistical properties are expected, such a generator is rather slow, since (apart from the generation of an additional unused PN, which however seems to be inevitable) the involved random addressing or random branching is rather inefficient on both scalar and vector architectures. Indeed, a timing test on a Cray–YMP with fully vectorized codes, in which we generated 10^9 normalized PNs with 10^4 calls, showed that the simple R250 procedure needed only 22 nsec. per PN, while the procedure with mixing needed 80 nsec..

Given the rather slow speed of this generator, we sought a faster algorithm that would nevertheless be able to get rid of the erroneous triplet correlations. Instead of mixing two independent series one might instead think of a modified procedure, where integers are randomly negated, or bits randomly flipped. Let us discuss this idea in some detail on the bit level.

Let $\{f_n\}$ be a (yet unspecified, but supposedly random) sequence of "flip bits". The original R250 sequence is then given by

$$y_n = y_{n-p} \oplus y_{n-q}, \tag{13}$$

where y_n stands for a single bit. We assume that $\{y_n\}$ and $\{f_n\}$ are statistically independent. From these two, one can generate a new sequence $\{z_n\}$ either by

$$z_n = y_n \oplus f_n, \tag{14}$$

i. e. simple random flipping, or by

$$z_n = z_{n-p} \oplus z_{n-q} \oplus f_n \tag{15}$$

i. e. random flipping with feedback (where the "flipped" number is fed back into the random number table).

Rule 1 (Eqn. 14) simply is an XOR of an R250 sequence with another random sequence, and hence the random properties are not deteriorated in comparison with R250. Conversely, rule 2 (Eqn. 15) establishes a generator which is best viewed as a recursion for *pairs of bits* (z_n, f_n), the details of which depend on the rule for $\{f_n\}$. For the simplest case, where $\{f_n\}$ is also an R250 sequence, $f_n = f_{n-p} \oplus f_{n-q}$, one would obtain

$$(z_n, f_n) = (z_{n-p} \oplus z_{n-q} \oplus f_{n-p} \oplus f_{n-q}, f_{n-p} \oplus f_{n-q}). \tag{16}$$

The same rule is obtained if one replaces z_n by $\zeta_n = z_n \oplus f_n$. While this generator might be an interesting alternative, we do not discuss it further, since to our knowledge the mathematical analysis of period etc. has not been done yet, and the simpler rule 1 provides a possibility to remove the triplet correlations. However, in the case of rule 1 one has to make sure that $\{f_n\}$ is *not* an R250 sequence: Eqn. 14 shows immediately that XOR–ing two R250 sequences with each other will generate just another R250 sequence such that nothing has been gained. On the other hand, there is no compelling reason to use an R250 sequence for $\{f_n\}$. Instead, one could use a sequence based on another pair of "magic numbers", say $(r, s) = (521, 168)$. Therefore, we finally arrive at the following simple recipe: Run two sequences based on two different pairs of "magic numbers" (we will call these, in accordance with our choice, R250 and R521), and get the final output Y by XOR–ing those sequences together (we will call this generator R250/521):

$$U_n = U_{n-p} \oplus U_{n-q} \tag{17}$$

$$V_n = V_{n-r} \oplus V_{n-s} \tag{18}$$

$$Y_n = U_n \oplus V_n. \tag{19}$$

This procedure should make the bits effectively independent, such that $W = 1/8$. Apart from using different values for p, q, r, s, this generator has actually been proposed [15, 16]

and used [17, 18] previously. Very good results were obtained for the Wolff algorithm applied to the two–dimensional Ising model [17], and it was pointed out that now instead of three–point correlations nine–point correlations (and of course higher correlations) do occur [18]. This is seen from the fact that Eqns. 17–19 are equivalent to the eight–point production rule [16]

$$Y_n = Y_{n-p} \oplus Y_{n-q} \oplus Y_{n-r} \oplus Y_{n-s} \oplus Y_{n-p-r} \oplus Y_{n-p-s} \oplus Y_{n-q-r} \oplus Y_{n-q-s}, \qquad (20)$$

which is easily verified using the properties of the XOR operation (of course, this representation is less suitable for the implementation). Generalizing the calculation of Sec. 2, we find that for a generator based on XOR–ing $m-1$ previous values (i. e. $m = 3$ for R250, $m = 9$ for R250/521) the following general formula holds for the correlation of the output value with the input values:

$$B = A_1 \oplus \ldots \oplus A_{m-1} \qquad \Rightarrow \qquad (21)$$

$$\frac{\langle B A_1 \ldots A_{m-1} \rangle}{(2^N - 1)^m} = \frac{1}{2^m} \left(1 + \frac{(-1)^m}{2^m - 1} \right) \qquad (22)$$

(here the small effects of finite word length have been neglected). For R250/521, this yields a very small relative correction 1/511 in the nine–point correlation. For some applications, this might turn out to not be sufficient; in this case one can improve further by introducing a third generator (with a third set of "magic numbers"), and combine its output via XOR with that of R250/521. The lowest–order deviation for such a generator would occur in a 27–point correlation function, etc. [16].

For R250/521, we found that normalized real PNs were produced at a rate of 41 nsec. per PN on a Cray–YMP, i. e. roughly two times slower than the original R250. On workstations, we found similar moderate slowdowns: An IBM RISC/6000 model 390 needed 57 nsec. per R250 PN and 130 nsec. per R250/521 PN, while for an SGI R10000 processor with 194 MHz clock speed these numbers are 22 nsec. and 96 nsec., respectively.

A generator similar to R250/521 was suggested and tested by Grassberger [9], with quite satisfactory results (which is no longer too surprising, in view of the results given above). His second generator was based on a congruential rule, and hence rather slow. The generator by Ziff [22],

$$Y_n = Y_{n-157} \oplus Y_{n-314} \oplus Y_{n-471} \oplus Y_{n-9689}, \qquad (23)$$

which also seems to have quite good statistical properties, needs the same number of input values and operations, and should hence have comparable speed. However, Eqn. 22 shows that its statistical properties are slightly worse, since it exhibits a systematic deviation in the five–point correlation function $\langle X_n X_{n-157} X_{n-314} X_{n-471} X_{n-9689} \rangle$, whose relative size is 1/31.

4 Empirical and Physical Tests of R250/521

As a test of our implementation of the simple R250 generator, which is based upon 31–bit integers (regardless of the machine), we first reproduced the triplet correlation results of Schmid and Wilding [12]. We generated 1000×100250 PNs, and calculated $\langle X_n X_{n-k} X_{n-250} \rangle$ for each of the 1000 sub–blocks separately. Within a block we used all available data, while the block–block fluctuations allowed us to calculate the statistical error (assuming statistical independence of the blocks). As seen from Figs. 1a and b, the correlation function is, for all lags k, consistent with the ideal value 0.125, except for $k = q = 103$, where the value 3/28 is reproduced. Conversely, the corresponding data for R250/521 in Fig. 1c do not show this deviation at $k = q$, as expected. At first glance, it is interesting to note that the data in Fig. 1c seem to be in much better agreement with the ideal value than those in Fig. 1b, which are somewhat below 0.125. However, this behavior does *not* hint to additional flaws in R250. By using different start values, we were able to produce data which were both above as well as below 0.125, within error bars. Note that deviations (within error bars) in the direction of smaller values are expected to be slightly more probable than those in the direction of larger values: The probability density of a random variable x, which is the product of m statistically independent random numbers uniformly distributed between zero and one, is $P(x) = [(m-1)!]^{-1}(-\ln x)^{m-1}$, which is strongly asymmetric. Moreover, one can find analytically the size of the error bar which should be produced if the PNs were strictly statistically independent. A straightforward but somewhat tedious calculation yields an error bar of $\sigma = 2.3 \times 10^{-5}$, which is (for two significant digits) nicely reproduced by the data for R250/521. Conversely, the numerically calculated and plotted error bar for R250 is roughly 2.0×10^{-5}, i. e. slightly too small. We view this as an indication of spurious additional (or secondary) long–range correlations in R250, which cause a systematic underestimation of the error. Finally, the data for all k values are strongly correlated, because they are all based on the same set of raw data. Indeed, an analytical calculation of the root mean square fluctuation between two different k values reveals that its size should be roughly 40% of the size of the error bar.

The long–range correlations in R250, which are already revealed by the underestimation of the error bar mentioned above, are more clearly borne out by the blocking test introduced by Vattulainen *et al.* [10]. We would like to describe this test here in a "magnetic" language (for a formulation closer to the concepts of statistics, see Ref. [10]). First, one subdivides the sequence $\{X_i\}$ of N PNs into blocks of length n, and forms, for each block, the arithmetic mean $n^{-1} \sum_{i=1}^{n} X_i$. If this variable exceeds 1/2, the spin variable S, which is associated with the block, is 1; otherwise it is -1. This procedure maps the sequence of PNs onto a one–dimensional Ising chain of $L = N/n$ spins. One then estimates the magnetization, $m = L^{-1} \sum_{i=1}^{L} S_i$, and checks if the value is consistent with its ideal

value zero within statistical accuracy, assuming independent spins. In particular, the susceptibility–like variable $\chi = Lm^2$ should be of order unity, since for independent spins $\langle\chi\rangle = 1$. In more detail, the probability density of χ is $P(\chi) = (2\pi\chi)^{-1/2}\exp(-\chi/2)$, since m, for large L, should be Gaussian.

In our test, we used $L = 10^6$ and varied n between 100 and 800. The χ values which we obtained are plotted in Fig. 2, as a function of block length n. Consistent with the results obtained in Ref. [10], we observe a dramatic increase in χ for R250, as soon as the block size exceeds $n \approx 250$. This is indicative of "ferromagnetic ordering", i. e. subsequent blocks are statistically more like each other than they should. Conversely, for R250/521, the values are much more moderate. Indeed, the distribution of the 71 χ values is roughly consistent with $P(\chi)$, as demonstrated in Table 3. We also checked the behavior for larger n up to 1600 (data not shown) and observed no qualitative difference.

Furthermore, we performed the Wolff algorithm test originally used by Ferrenberg, Landau and Wong [5]. The two–dimensional Ising model was simulated using the Wolff cluster–flipping algorithm at the critical point, using a square lattice of size 16×16. The simple R250 generator showed rather severe systematic errors in both the average energy and the specific heat [5]. As the comparison of these numbers with the corresponding data for R250/521 (Table 4) shows, the latter generator performs much better than R250, the deviations from the exact values [25] being consistent with the statistical error bars. Quite similar behavior was observed by Talapov *et al.* [17], who ran the same test for R4423/9689 (i. e. $p = 4423$, $q = 1393$, $r = 9689$, $s = 471$) and lattices up to 256×256. This remarkable improvement indicates that the triplet correlations were very probably responsible for the systematic error. However, the reason why the Wolff algorithm is so sensitive to triplet correlations remains a mystery.

5 Conclusions

It has been shown how the numerically observed value for the triplet correlation in shift–register PN generators can be calculated on the basis of the production rule. Interestingly, the reverse correlation is obtained if the logical exclusive–or operation is replaced by a not–exclusive–or operation. These considerations motivate the introduction of a second generator which is used to remove the triplet correlations. The simple and fast method of combining two independent shift–register generators with different time lags via an XOR has shown very satisfactory results in the triplet correlation test, the blocking test, and the Wolff algorithm. These are all tests where R250 failed spectacularly. Since it is also obvious that R250/521 cannot be worse than R250, we did not consider it necessary to run

tests which R250 had passed. We hence believe that this type of generator can prove very useful in many applications, in particular in those where a lot of PNs are needed at a very high production rate, with only small additional cost of the overall program. Of course, one should keep in mind that this speed comes at the price of a weak residual nine–point correlation. It is therefore advisable to not use this generator blindly, but rather compare application results with results produced by either R250 or an even further improved version where three or more shift–register generators are combined, as discussed in Ref. [16]. Such a test on the "convergence" of statistical quality might still be cheaper than using a slow generator.

6 Acknowledgements

We wish to thank F. Schmid and N. B. Wilding for stimulating discussions. AMF would like to acknowledge the hospitality of the Physics Department of the University of Mainz during a recent visit, as well as the support of NATO Grant No. CRG 921202.

References

[1] R. C. Tausworthe, Math. Comput. **19** (1965) 201.

[2] T. G. Lewis and W. H. Payne, J. Assoc. Comput. Mach. **20** (1973) 456.

[3] H. S. Bright, Computing Surveys **11** (1979) 357.

[4] S. Kirkpatrick and E. P. Stoll, Journ. Comp. Phys. **40** (1981) 517.

[5] A. M. Ferrenberg, D. P. Landau and Y. J. Wong, Phys. Rev. Lett. **69** (1992) 3382.

[6] W. Selke, A. L. Talapow and L. N. Shchur, JETP Lett. **58** (1993) 665.

[7] P. D. Coddington, Int. Journ. Mod. Phys. C **5** (1994) 547.

[8] K. Kankaala, T. Ala–Nissila and I. Vattulainen, Phys. Rev. E **48** (1993) R4211.

[9] P. Grassberger, Phys. Lett. **A 181** (1993) 43.

[10] I. Vattulainen, T. Ala–Nissila and K. Kankaala, Phys. Rev. Lett. **73** (1994) 2513.

[11] I. Vattulainen, T. Ala–Nissila and K. Kankaala, Phys. Rev. E **52** (1995) 3205.

[12] F. Schmid and N. B. Wilding, Int. Journ. Mod. Phys. C **6** (1995) 781.

[13] I. Vattulainen, K. Kankaala, J. Saarinen and T. Ala–Nissila, Comp. Phys. Comm. **86** (1995) 209.

[14] A. Compagner and A. Hoogland, J. Comp. Phys. **71** (1987) 391.

[15] A. Compagner, J. Stat. Phys. **63** (1991) 883.

[16] A. Compagner, Phys. Rev. E **52** (1995) 5634.

[17] A. L. Talapov, L. N. Shchur and H. W. J. Blöte, JETP Lett. **62** (1995) 174.

[18] H. W. J. Blöte, E. Luijten and J. R. Heringa, J. Phys. A **28** (1995) 6289.

[19] M. Lüscher, Comp. Phys. Comm. **79** (1994) 100.

[20] F. James, Comp. Phys. Comm. **79** (1994) 111.

[21] G. Marsaglia and A. Zaman, Ann. Appl. Prob. **1** (1991) 462.

[22] R. M. Ziff, Phys. Rev. Lett. **69** (1992) 2670.

[23] F. James, Comp. Phys. Comm. **60** (1990) 329.

[24] N. Zierler, Information and Control **15** (1969) 67.

[25] A. E. Ferdinand and M. E. Fisher, Phys. Rev. **185** (1969) 823.

Figures

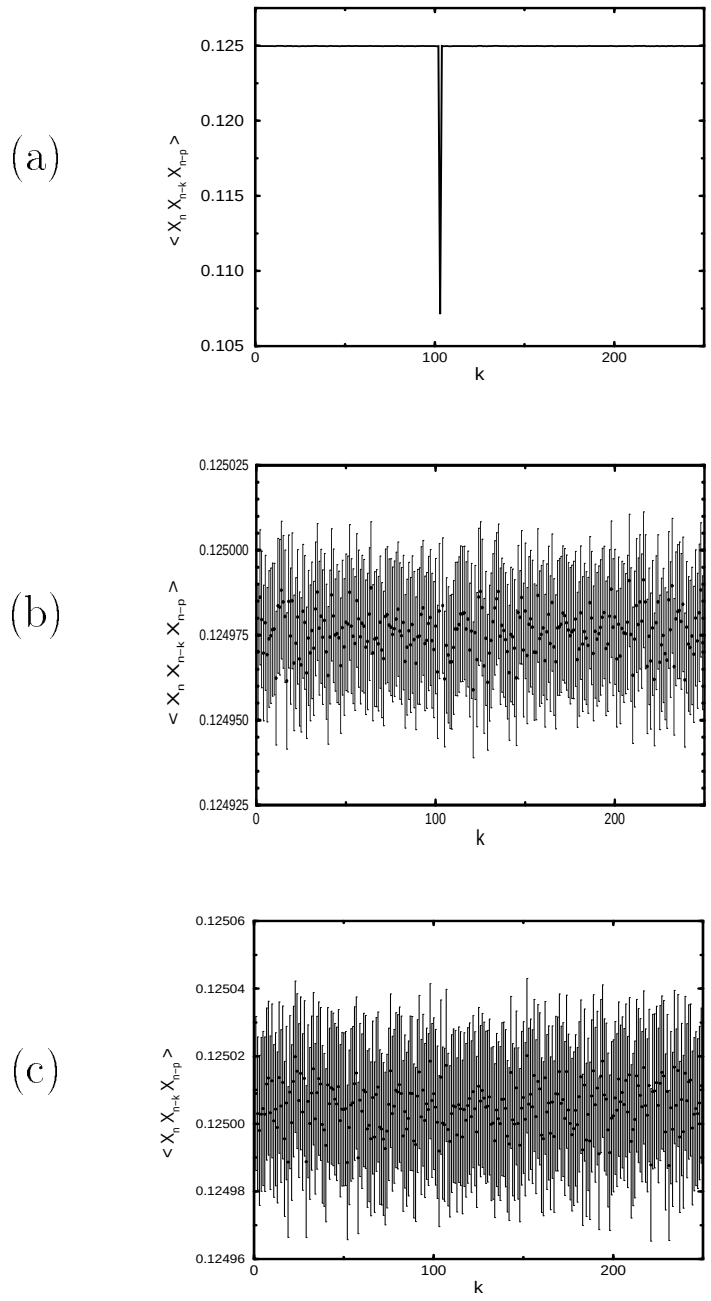

Figure 1: *(a)* The triplet correlation function $\langle X_n X_{n-k} X_{n-250} \rangle$ as a function of the lag parameter k, for data from R250. In accordance with the analytical calculation, we observe for $k = 103$ a value of $0.1071 \approx 3/28$. *(b)* Same as (a), but on an expanded scale, and including statistical error bars. *(c)* Same as (b), but for R250/521.

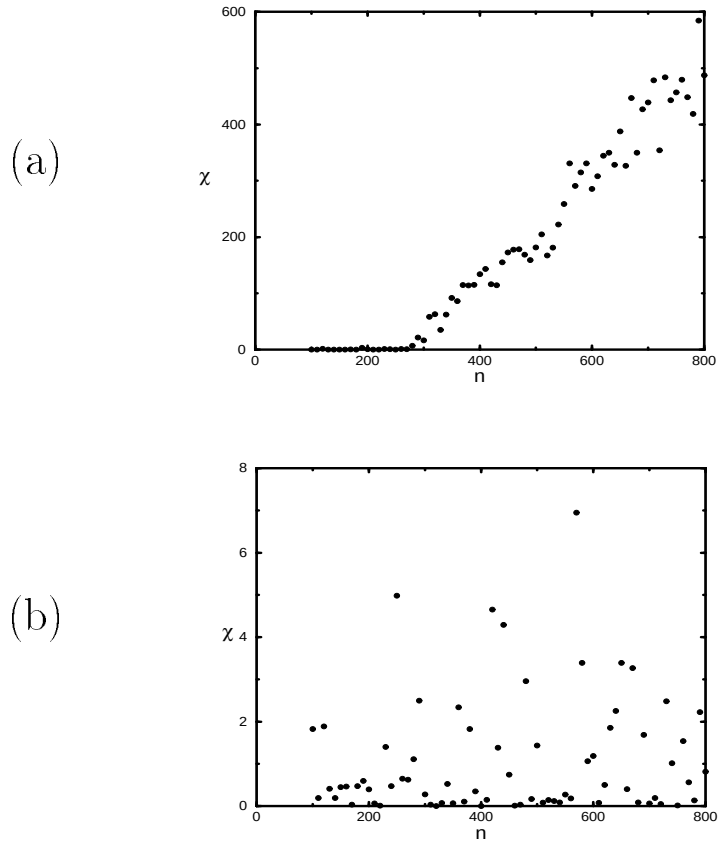

Figure 2: *(a)* The variable χ, as defined in the text, as a function of block length, for R250. *(b)* Same as (a) for R250/521 (note the different scale).

Tables

Tab. 1: The exclusive–or operation: $c = a \oplus b$

a	b	c
0	0	0
0	1	1
1	0	1
1	1	0

Tab. 2: The not–exclusive–or operation: $c = a \bar{\oplus} b = 1 \oplus a \oplus b$

a	b	c
0	0	1
0	1	0
1	0	0
1	1	1

Tab. 3: Blocking test results for R250/521: Comparison of observed χ distribution with the theoretical one.

χ interval	theor. probab.	obs. freq.
$0 < \chi < 0.2$	0.35	0.38
$0.2 < \chi < 1$	0.34	0.25
$1 < \chi < 3$	0.23	0.27
$3 < \chi < \infty$	0.08	0.10

Tab. 4: Wolff algorithm results for the energy per site $\langle E \rangle$ and the specific heat C of the two–dimensional Ising model on the 16×16 square lattice at the critical point.

Generator	R250 (Ref. [5])	R250/521	exact (Ref. [25])
- $\langle E \rangle$	1.455017	1.4530621	1.4530649
error	0.000046	0.0000243	—
deviation	42 σ	0.1 σ	—
C	1.448627	1.498378	1.498711
error	0.000467	0.000217	—
deviation	107 σ	1.5 σ	—