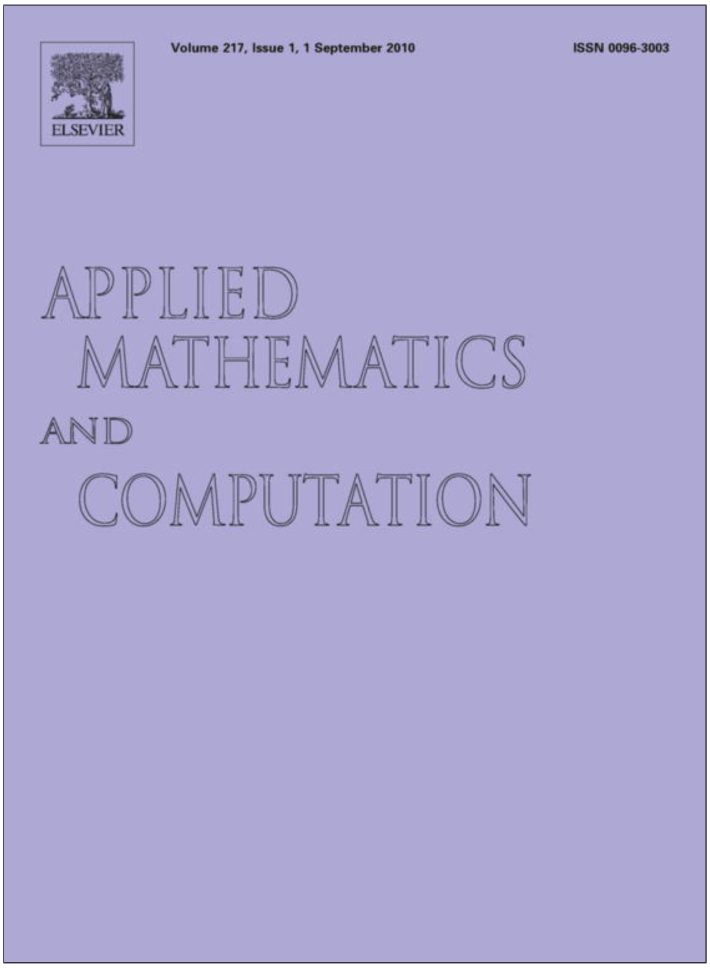

Volume 217, Issue 1, 1 September 2010

ISSN 0096-3003

ELSEVIER

APPLIED MATHEMATICS AND COMPUTATION

Applied Mathematics and Computation 217 (2010) 339–346

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Re-seeding invalidates tests of random number generators

Hans Ekkehard Plesser [a,b,c,*], Anders Grønvik Jahnsen [a]

[a] *Department of Mathematical Science and Technology, Norwegian University of Life Sciences, P.O. Box 5003, 1432 Aas, Norway*
[b] *Center for Biomedical Computing, Simula Research Laboratory, P.O. Box 134, 1325 Lysaker, Norway*
[c] *RIKEN Brain Science Institute, 2-1 Hirosawa, Wako-shi, Saitama 351-0198, Japan*

ARTICLE INFO

Keywords:
Random walk
Gambler's ruin
Pseudorandom number generator

ABSTRACT

Kim et al. [C. Kim, G.H. Choe, D.H. Kim, Test of randomness by the gambler's ruin algorithm, Applied Mathematics and Computation 199 (2008) 195–210] recently presented a test of random number generators based on the gambler's ruin problem and concluded that several generators, including the widely used Mersenne Twister, have hidden defects. We show here that the test by Kim et al. suffers from a subtle, but consequential error: re-seeding the pseudorandom number generator with a fixed seed for each starting point of the gambler's ruin process induces a random walk of the test statistic as a function of the starting point. The data presented by Kim et al. are thus individual realizations of a random walk and not suited to judge the quality of pseudorandom number generators. When generating or analyzing the gambler's ruin data properly, we do not find any evidence for weaknesses of the Mersenne Twister and other widely used random number generators.

1. Introduction

Pseudorandom number generators (PRNGs) are indispensable tools in computational science. They are well covered by an increasing body of scientific literature, e.g. [1,2]. Systematic tests of pseudorandom number generators have received much attention. The TestU01 testsuite by L'Ecuyer and Simard [3] appears to define the current state of the art in the field. The history of pseudorandom number generation is rife with seemingly robust random number generators which failed spectacularly under particular circumstances, e.g., [4]. Donald Knuth has summarized this beautifully: "Every random number generator will fail in at least one application" [1, p. 189]. New reports on deficiencies of random number generators therefore deserve careful attention.

The recent report by Kim et al. [5] demonstrating "hidden defects" in the widely used Mersenne Twister generators [6] and some multiple recursive generators gave thus reason for concern. The analysis by Kim et al. is based on the gambler's ruin problem: at the beginning of a game, the player owns $x_0 = s$ and the bank $N - s$ coins. For each step j in the game, the player either wins a coin from the bank with probability p, so that $x_j = x_{j-1} + 1$, or loses a coin to the bank with probability $1 - p$, i.e., $x_j = x_{j-1} - 1$. The game ends after T steps when the player has no money left (ruin, $x_T = 0$), or the player has won everything (broke the bank, $x_T = N$). This game has fascinated statisticians for centuries and is well covered in textbooks, e.g. [7, Ch. XIV].

After playing K games for a given starting point s, we can compute the average duration

$$\overline{T}(s) = \frac{1}{K}\sum_{j=1}^{K} T_j(s), \tag{1}$$

[*] Corresponding author at: Department of Mathematical Science and Technology, Norwegian University of Life Sciences, P.O. Box 5003, 1432 Aas, Norway.
E-mail addresses: hans.ekkehard.plesser@umb.no, heplesser@gmail.com (H.E. Plesser).

0096-3003/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2010.05.066

where $T_j(s)$ is the duration of the jth trial for starting point s. For $K \gg 1,\ \overline{T}(s)$ should be a random number following a normal distribution by virtue of the central limit theorem [7, Ch. X]. Explicit expressions for the mean $\mu(s)$ and the standard deviation $\sigma(s)$ of the trial duration have been obtained by Kim et al. [5, p. 200]; see also [7, Ch. XIV]. For brevity, we give here only the equations for test case A3 from Kim et al., with a winning probability of $p = \sqrt{2} - 1 \approx 0.414$:

$$\mu_{\mathrm{A3}} = \left(3 + 2\sqrt{2}\right)\left(s - \frac{N\left(1 - 2^{\frac{s}{2}}\right)}{1 - 2^{\frac{N}{2}}}\right), \tag{2}$$

$$\sigma^2_{\mathrm{A3}} = \left(68\sqrt{2} + 96\right)\left(s + \frac{N\left(2 + \left(\sqrt{2}s - 2\right)2^{\frac{s}{2}}\right)}{2\left(2^{\frac{N}{2}} - 1\right)} - \frac{N^2\left(3 \times 2^{\frac{N}{2}} + 2^{\frac{N}{2}}\right)\left(2^{\frac{s}{2}} - 1\right)}{4\sqrt{2}\left(2^{\frac{N}{2}} - 1\right)^2}\right). \tag{3}$$

One may then compute the Z-statistic

$$Z(s) = \frac{\overline{T}(s) - \mu(s)}{\sigma(s)/\sqrt{K}}. \tag{4}$$

If $\overline{T}(s)$ is normally distributed, then the Z-statistic is normally distributed with zero mean and unit variance, whence $|Z| > 2.58$ should be observed for approximately 1% of all values of $\overline{T}(s)$ collected. We will denote the zero-mean, unit-variance distribution as $N(0,1)$.

Interestingly, Kim et al. re-seed the PRNG with a fixed seed $\mathcal{S}$ before beginning the series of K games for each starting point s. They concede that as a consequence "the Z-values corresponding to different starting points are highly correlated" and that one therefore "cannot say whether a PRNG fails or passes by counting the number of starting points for which the PRNG fails" the Z-test. But they then argue that "we can regard a Z-value to each starting point as a separate test result" and conclude that "[i]f a PRNG fails for too many starting points, then we cannot say that the PRNG has good randomness".

We show here that one cannot regard Z-values obtained for different starting points as "separate test result[s]". Quite to the contrary: Due to the re-seeding of the PRNG for each starting point s, the Z-values obtained for subsequent starting points $s = 1, 2, 3\ldots$ form a random walk with properties that are independent of the details of the experiment. If one re-analyzes the results by Kim et al. properly in terms of this random walk, no evidence for "hidden defects" in the Mersenne Twister remains. The same holds true for properly conducted experiments in which the random number generator is seeded only once before the very first game.

We will briefly describe our numerical experiments in the following section, followed by results and analysis in Section 3, before concluding in Section 4.

2. Numerical experiments

We performed numerical experiments for test cases A2 ($p = 1/4$), A3 ($p = \sqrt{2} - 1$), and A4 ($p = 1/e$) as defined by Kim et al. Each test case was simulated under two different regimes:

Re-seeding regime Under the re-seeding regime, we seeded the PRNG with a fixed seed $\mathcal{S}$ for each starting point s and then performed K subsequent games beginning at that starting point. This is the regime used by Kim et al. [5].

Continuous regime Under the continuous regime, we seeded the PRNG once with a given seed $\mathcal{S}$. We then performed K games for starting point $s = 1$, then K games for starting point $s = 2$, etc.

All experiments were performed for a total number of $N = 1000$ coins, starting points $s = 1, 2, \ldots, N - 1 = 1, 2, \ldots, 999$, with $K = 2000$ games per starting point. For each starting point, the average trial duration $\overline{T}(s)$ and its standard deviation $\sqrt{\overline{\Delta T^2}}$ is reported. We will use the term *experiment* to refer specifically to the set of K games played for each of the $N - 1$ starting points for a given seed $\mathcal{S}$. Thus an experiment consists of a total of $(N - 1)K$ games and reports $N - 1$ values $\overline{T}(s)$. We performed ten experiments with seeds $\mathcal{S} = \{1, 2, \ldots, 10\} \times 10^6$ under each of the two regimes.

All experiments were performed with self-written C++-code using the implementation of the Mersenne Twister pseudorandom number generator MT19937ar provided by Makoto Matsumoto.[1] Random numbers were drawn using the `genrand_real2()` function, which generates 32-bit random numbers uniformly distributed in [0,1). Function `init_genrand()` was used with a 32-bit seed to initialize the random number generator. Data was analyzed using MATHEMATICA 7 (Wolfram Research, Inc).

All data reported here was generated on a MacBook Pro computer (Apple, Inc). Preliminary experiments were performed on the Stallo HPC Cluster operated by University of Tromsø as part of the NOTUR framework.

[1] Available at http://www.math.sci.hiroshima-u.ac.jp/m-mat/MT/MT2002/emt19937ar.html, last accessed 2009–07-02.

3. Results

We first consider the continuous regime, where all pseudorandom numbers are drawn from a single sequence without reseeding the generator. A PRNG would typically be used in this manner in a simulation. Fig. 1 shows the *Z*-scores for five experiments (five different initial seeds $\mathcal{S}$) for test case A3 ($p = \sqrt{2} - 1$). The scores clearly scatter in a manner consistent with a normal distribution, with few scores outside the 99% confidence interval. This is corroborated by the normal probability plot in Fig. 2, which shows *Z*-scores for all thirty experiments performed (10 for each of the test cases A2, A3, A4). The expected cumulative distribution function for the $N(0,1)$ normal distribution is superimposed on the experimental data, indicating that the experimental data closely follows the $N(0,1)$-distribution.

We further confirmed this by testing the data against the $N(0,1)$-distribution using the Kolmogorov–Smirnov test [1, Ch. 3.3.1]. The null hypothesis that the *Z*-scores are distributed according to $N(0,1)$ is rejected for two out of thirty experiments at the 5% confidence level. At this level, one would expect 5% of 30 experiments, i.e., 1.5 experiments, to fail the test. We therefore conclude that there is no evidence to reject the null hypothesis that the *Z*-scores for the game durations of the gambler's ruin problem are $N(0,1)$-distributed when using the Mersenne Twister `mt19937ar` PRNG in the continuous regime.

We applied this gambler's ruin test to several other widely used PRNGs:

- `mrg32k3a` [8], using the implementation in the TestU01 library [3], v. 1.2.3;
- `ran2` and `ran3` [9], using the implementation in the GNU Scientific Library [10], v. 1.14;

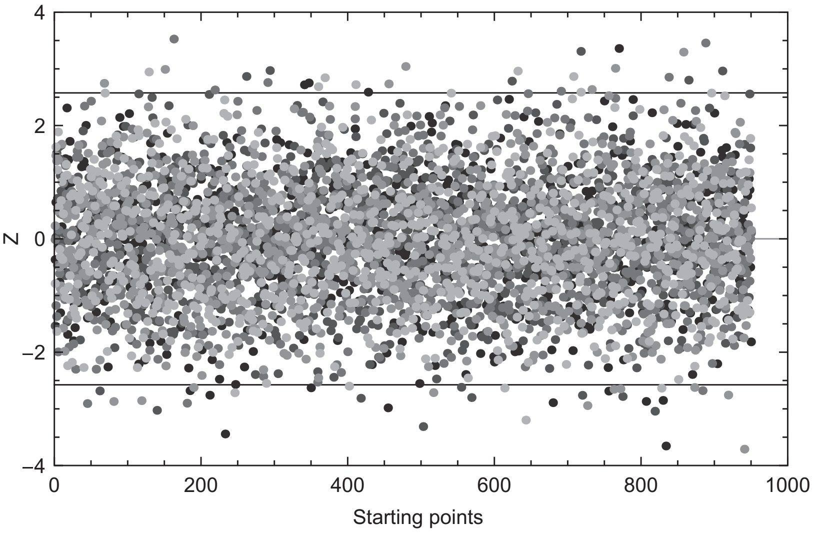

Fig. 1. *Z*-statistic for experiments under the continuous regime for starting points $s = 1, \ldots, 950$: *Z*-values for five experiments for case A3 ($p = \sqrt{2} - 1$); data points for each experiment are shown in a different shade of gray.

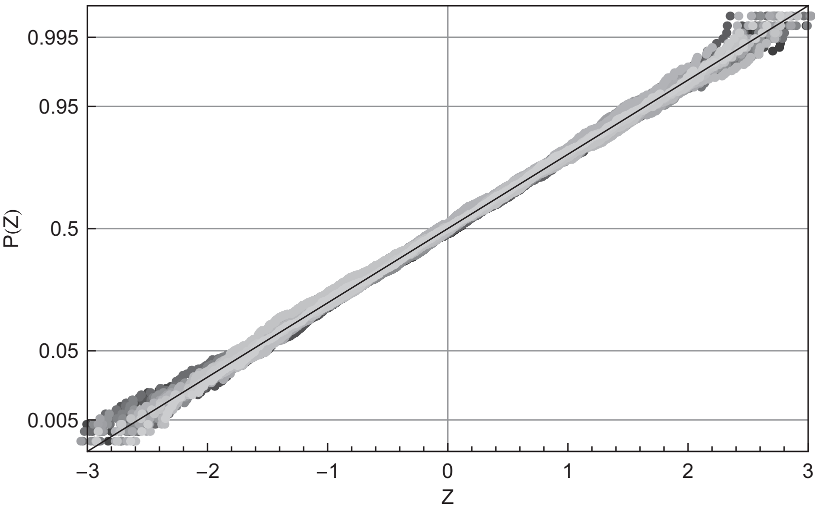

Fig. 2. Normal probability plot of *Z*-scores for all ten experiments for each of the cases A2, A3, A4 under the continuous regime, starting points $s = 1, \ldots, 950$. Each experiment is shown in a different shade of gray, starting with black for the first experiment for case A2, and ending with the lightest gray for the tenth experiment for case A4. The thin black diagonal is the cumulative distribution function for the $N(0,1)$ normal distribution.

- Knuth's lagged Fibonacci generator [1, 9th printing, 2002], using the implementation in the GNU Scientific Library [10], v. 1.14.

All generators passed the test.

3.1. Gambler's ruin with re-seeding

Z-scores obtained in the re-seeding regime are strikingly different as shown in Fig. 3. Here, scores from each of the five experiments form seemingly continuous paths. These paths are very similar in appearance to many of the paths shown by Kim et al., e.g., Fig. 6 in Ref. [5]. It is rather obvious that these *Z*-scores are highly correlated and do not follow a normal distribution. This is confirmed by the normal probability plot in Fig. 4, which again shows data for all thirty experiments. Not a single experiment comes even close to the $N(0,1)$-distribution.

The *Z*-scores for individual experiments in the re-seeding regime, as shown in Fig. 3, strongly resemble random walks in *Z* as a function of *s*, with a seemingly shrinking size of the random steps with increasing *s*. We shall now demonstrate that this interpretation is correct.

3.2. A random walk in Z

Our analysis is based on the fact that for $0 < p < \frac{1}{2}$, almost all games end in ruin; our argument applies equally for $\frac{1}{2} < p < 1$ if one exchanges ruin with breaking the bank. From the probability of ruin for a given starting point given by Feller [7, Eq. XIV.2.4], we can find the starting point S_β for which the probability of ruin is $1 - \beta$:

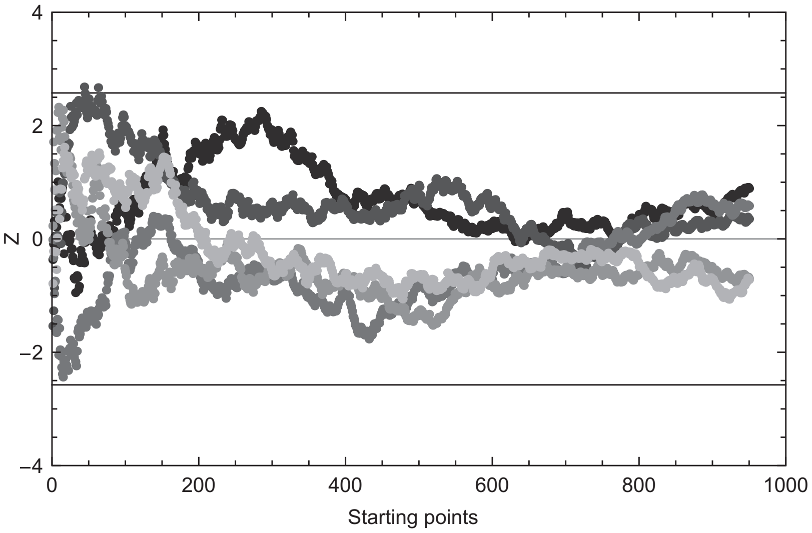

Fig. 3. *Z*-values for five experiments for case A3 under the re-seeding regime. Data are presented in the same way as in Fig. 1.

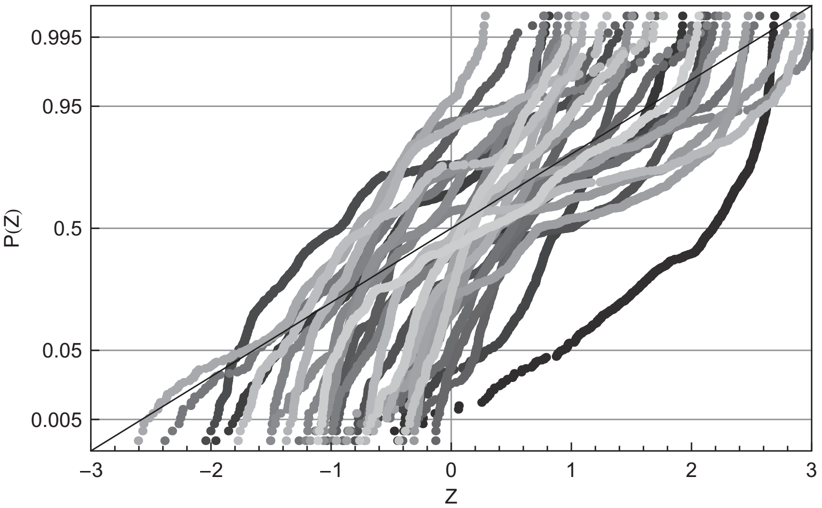

Fig. 4. Normal probability plot of *Z*-values for all experiments and cases in the re-seeding regime. See Fig. 2 for details of presentation.

$$S_\beta = \frac{\ln\left(1+\beta\left[\left(\frac{1}{p}-1\right)^N - 1\right]\right)}{\ln\left(\frac{1}{p}-1\right)}. \tag{5}$$

For any $s \leqslant S_\beta$, the probability of ruin is greater than $1-\beta$. For the three test cases considered here, the probability of ruin is greater than $\beta = 1\text{–}10^{-6}$ for starting points $s \leqslant 950$ (cf. Table A.1 in the appendix). We will therefore restrict ourselves to starting points between 1 and 950 and assume that all games end in ruin.

Based on this assumption we show in Appendix A that the expectation for the Z-score for starting point $s+1$, given the measured Z-score z_s for starting point s, is given by

$$\langle Z(s+1)|s, z_s\rangle = \frac{z_s}{\sqrt{1+\frac{1}{s}}} \pm \frac{1}{\sqrt{1+s}}. \tag{6}$$

The ± term gives the standard deviation.

Thus, the re-seeding regime used by Kim et al. [5] induces a random walk in the Z-statistic as a function of the starting point s. The random walk is characterized by a drift describing the damping of any excursions away from $Z = 0$ (first term in Eq. (6)) and Gaussian fluctuations with an amplitude decreasing with s (second term in Eq. (6)).

According to Eq. (6), the difference

$$d_s = z_{s+1} - z_s \quad \text{for } s = 1, \ldots, 949. \tag{7}$$

of Z-scores for neighboring starting points follows approximately a normal distribution with expectation

$$\langle D_s|z_s\rangle = \frac{z_s}{\sqrt{1+1/s}} - z_s. \tag{8}$$

The Z-statistic for the d_s

$$\zeta_s = \frac{d_s - \langle D_s|z_s\rangle}{1/\sqrt{1+s}} = \sqrt{1+s}\left[z_{s+1} - \frac{z_s}{\sqrt{1+1/s}}\right], \tag{9}$$

should then have approximately a $N(0,1)$ distribution.

Table A.1
Properties of test cases. p: probability of winning a single coin toss; $S_{10^{-6}}$: probability of eventual ruin is greater than $1\text{–}10^{-6}$ for starting points $s < S_{10^{-6}}$, out of a total of 999 starting points; a, b, q: parameters in Eqs. (A.1) and (A.2).

Test case	p	$S_{10^{-6}}$	a	b	q
A2	$1/4$	987	2	6	3
A4	$1/e$	974	$e/(e-2)$	$4e(e-1)/(e-2)^3$	$e-1$
A3	$\sqrt{2}-1$	960	$3+2\sqrt{2}$	$68\sqrt{2}+96$	$\sqrt{2}$

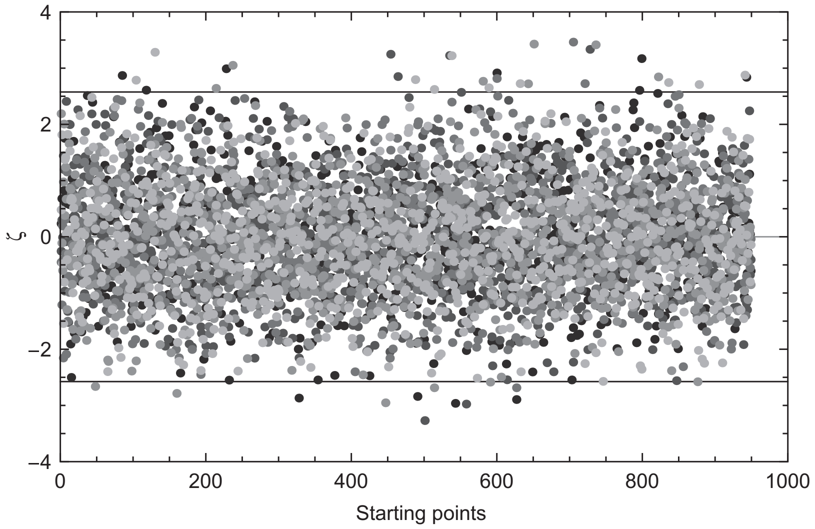

Fig. 5. ζ-values for step size in random walk obtained from Eq. (9) for five experiments for case A3. ζ-values are computed from the same experimental data as the data in Fig. 3. Data are presented in the same way as in Fig. 1.

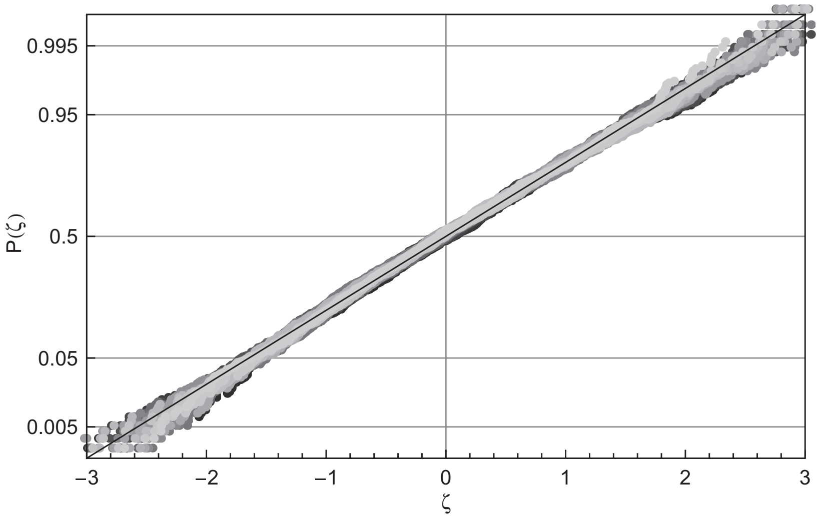

Fig. 6. Normal probability plot of ζ-values for all experiments and cases. ζ-values are computed from the same experimental data as the data in Fig. 4. See Fig. 2 for details of presentation.

This is supported by data in Figs. 5 and 6. Fig. 5 shows individual ζ-scores for the same five experiments as in Fig. 3. The distribution of the scores is in apparent agreement with a normal distribution. There are also no perceptible correlations in the data. Fig. 6 shows the normal probability plot of the ζ-scores for all thirty experiments, i.e., the same data as shown in Fig. 4, but now properly analyzed. The figure indicates excellent agreement with the $N(0,1)$ normal distribution. All thirty trials pass the Kolmogorov–Smirnov test against $N(0,1)$ as null hypothesis at the 5% significance level, which is again compatible with 1.5 expected failures. Thus, also the data from the re-seeding regime provide no evidence against a proper distribution of the underlying pseudorandom numbers.

4. Discussion

Kim et al. [5] tested a range of pseudorandom number generators using three different variants of the gambler's ruin problem. They obtained Z-scores for the average duration of the game across different starting points, and judged the quality of pseudorandom number generators based on the number of starting points yielding Z-scores outside the 99% confidence interval. Based on their analysis, they state in particular that one "should be careful in simulating random walks using ... MT19937ar" and that "[t]he recently improved initialization ... of MT19937 brings bad results".

We have shown here that these results are an artifact induced by re-seeding the pseudorandom number generator for each starting point. Data obtained without re-seeding do not provide any evidence against the reliability of the `mt19937ar` generator, cf. Section 3 and Figs. 1 and 2. Indeed, the re-seeding induces a random walk of the Z-scores as a function of the starting point of the gambler's ruin game, as shown in Section 3.2. When we re-analyzed the data obtained with re-seeding in terms of this random walk, we again found no evidence for weaknesses in the `mt19937ar` generator. We further demonstrated that the data presented by Kim et al. are individual realizations of a random walk of the Z-score as function of the starting point. It is obviously not useful to judge the quality of pseudorandom number generators based on individual realizations of random walk experiments.

The fact that Kim et al. considered only individual realizations of a random walk may explain why they found that the `mt19937ar` generator with improved seeding scheme performed worse than the original `mt19937` [5, Tab. 5]: They generated a single realization for each of the two PRNGs (or, if one takes all of their test cases into account, a very small number) and most likely obtained—entirely by chance—random walks with large excursions for the `mt19937ar` generator.

In this paper, we have only considered test cases A2, A3, and A4 from the paper by Kim et al. Our analysis does not apply to test cases with equal probability of winning and losing (A1, A5, B1, B2), as the probability of ruin and breaking the bank is identical in these cases. But then Kim et al. did not present evidence for weaknesses of random number generators in these cases, with the exception of some generators in test case A5. The latter failures can be ascribed to the fact that A5 tests individual bits, not entire random numbers; Kim et al. report no defects of the Mersenne Twister for case A5. Our random walk analysis applies to test cases C1–C5 even if the probability of winning and losing is equal, because step sizes differ, so that games from a giving starting point either all end in ruin or all in breaking the bank.

As our random walk analysis made no assumption about the underlying pseudorandom number generator except that it is "proper", i.e., generating uncorrelated pseudorandom numbers equidistributed on the unit interval, our results apply also to the other pseudorandom number generators studied by Kim et al., in so far as they invalidate the evidence presented against these generators. Indeed, when we applied the gambler's ruin test without re-seeding to several widely used pseudorandom number generators, we found no evidence for weaknesses in those generators (`mrg32k3a`, `ran2`, `ran3`, Knuth's lagged Fibonacci generator).

Concerning the `mt19937ar` Mersenne Twister, we have demonstrated that the criticism voiced against it by Kim et al. [5] is invalid. This does not make the Mersenne Twister *the* perfect pseudorandom number generator. Panneton et al. [11] recently demonstrated that the Mersenne Twister escapes "zeroland" very slowly: when initialized with a state vector in which only one bit is non-zero, it typically takes $\mathcal{O}(10^6)$ random numbers before approximately half of the bits in the state vector are 1. WELL generators [11] fare much better in this respect. The Mersenne Twister by construction also fails tests looking for linear dependencies in long sequences of bits [3,11].

We would like to conclude this report with a piece of advice: think *very* carefully before re-using random numbers in any simulation study!

Acknowledgements

We are grateful to an anonymous reviewer for suggestions. Tom Tetzlaff made useful comments on the manuscript. HEP is grateful to Honda Research Institute Europe and to the Research Council of Norway (Grant 178892/V30 eNeuro) for financial support.

Appendix A. Re-seeding induced random walk in *Z*-scores

In this appendix, we derive Eq. (6), demonstrating that re-seeding the PRNG for each starting point s induces a random walk in the Z-scores obtained for subsequent starting points.

As discussed in Section 3.2, we assume that all games end in ruin for $s \leqslant 950$. We will also exploit that the expressions for mean and variance of the game duration given by Kim et al. [5, p. 200] for $p \neq \frac{1}{2}$ are all of the form

$$\mu(s) = as + \mathcal{O}\left(\frac{1}{q^{N-s}}\right), \tag{A.1}$$

$$\sigma^2(s) = bs + \mathcal{O}\left(\frac{x}{q^{N-s}}\right). \tag{A.2}$$

Parameters a, b, and q are given in Table A.1. For $N = 1000$, the linear approximations hold very well for $s \leqslant 950$.

We now turn to the random walk analysis. Let us assume that we have performed K games for a given starting point s and have obtained the mean trial duration t_s, where t_s is an individual realization of the random variable $\overline{T}(s)$. As discussed above, we assume that all trials end in ruin. Any game starting at s must have included exactly s more losing than winning coin tosses out of a total of t_s tosses. Each coin toss corresponds to one pseudorandom number drawn from the PRNG. In total, the K games used $r_s = K\,t_s$ pseudorandom numbers, of which Ks more were losing than winning numbers.

Now consider a new set of K games beginning at starting point $u > s$, all of which we assume to end in ruin. At the end of each such game, u more losing than winning random numbers will have been consumed. Since the PRNG is seeded with the same seed before the series of games beginning at s and u, respectively, we know that the first $r_s = K\,t_s$ random numbers consumed by the series of games played from u, will be the same as for the series played from s. We can then estimate that these r_s re-used random numbers have been used to perform the first

$$\hat{k} = \frac{sK}{u} \tag{A.3}$$

games starting from u. The validity of this estimate hinges on the fact that practically all games end in ruin: If a game ends in ruin anyways, it largely does not matter in which precise order wins and losses occur in a game, and we may consider the total surplus of losing numbers across a series of games. We also ignore here the complication that $\hat{k}$ generally will not be an integer number of games.

The remaining $K - \hat{k}$ games for the series with starting point u will then be played using new pseudorandom numbers. Provided that the PRNG generates uncorrelated numbers uniformly distributed on the unit interval, these remaining trials have an expected duration of $\mu(u)$ with standard deviation $\sigma(u)$. The expected number of random numbers consumed by all K games for starting point u is now given by the number of re-used numbers, plus the number of fresh numbers:

$$\langle R(u)|r_s\rangle = r_s + \mu(u)(K - \hat{k}), \tag{A.4}$$

while the standard deviation in the total trial durations arises from the fresh random numbers only, so we obtain

$$\sqrt{\left\langle (\Delta R(u))^2 | r_s \right\rangle} = \sigma(u)\sqrt{K - \hat{k}}. \tag{A.5}$$

From these expressions we immediately obtain the expected mean and standard deviation of the game duration, $\overline{T}(u)$, by dividing by the total trial number K and thus the expected Z score by inserting into Eq. (4)

$$\langle Z(u)|t_s\rangle = \frac{\langle R(u)|r_s\rangle/K - \mu(u)}{\sigma(u)/\sqrt{K}} = \frac{t_s - \mu(u)\frac{s}{u}}{\sigma(u)/\sqrt{K}}, \tag{A.6}$$

and

$$\sqrt{\left\langle (\Delta Z(u))^2 | t_s \right\rangle} = \frac{\sqrt{\left\langle (\Delta R(u))^2 | r_s \right\rangle}/K}{\sigma(u)/\sqrt{K}} = \sqrt{1-\frac{s}{u}}. \tag{A.7}$$

Since the specific values for mean game duration t_s, total number of tosses/numbers r_s and Z-score z_s are directly related through Eqs. (1) and (4), we can used them interchangeably to express that the expectations for the series of games played for starting point u is conditional on the outcome of the series played for starting point s.

To obtain a compact expression predicting the value of $Z(u)$ at $u > s$ for given z_s, we express t_s in Eq. (A.6) in terms of z_s by virtue of Eq. (4) and express the new starting point as $u = s + \delta$ for $\delta \geqslant 1$. Exploiting further the linear approximations for $\mu(u)$ and $\sigma^2(u)$ from Eqs. (A.1) and (A.2), and expressing the expected standard deviation as a ±-term, we obtain

$$\langle Z(s+\delta)|s, z_s\rangle = \frac{z_s}{\sqrt{1+\frac{\delta}{s}}} \pm \frac{1}{\sqrt{1+\frac{s}{\delta}}}. \tag{A.8}$$

The most interesting case is to predict the Z score for the starting point following s, i.e $\delta = 1$, where Eq. (A.8) yields

$$\langle Z(s+1)|s, z_s\rangle = \frac{z_s}{\sqrt{1+\frac{1}{s}}} \pm \frac{1}{\sqrt{1+s}}. \tag{A.9}$$

References

[1] D.E. Knuth, The Art of Computer Programming, third ed., vol. 2, Addison-Wesley, Reading, MA, 1998.

[2] J.E. Gentle, Random Number Generation and Monte Carlo Methods, second ed., Springer Science+Business Media, New York, 2003.

[3] P. L'Ecuyer, R. Simard, TestU01: a C library for empirical testing of random number generators, ACM Transactions on Mathematical Software 33 (2007) (Article 22, 40 pp.).

[4] A.M. Ferrenberg, D.P. Landau, Y.J. Wong, Monte Carlo simulations: hidden errors from "good" random number generators, Physics Review Letters 69 (23) (1992) 3382–3384.

[5] C. Kim, G.H. Choe, D.H. Kim, Test of randomness by the Gambler's ruin algorithm, Applied Mathematics and Computation 199 (2008) 195–210.

[6] M. Matsumoto, T. Nishimura, Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator, ACM Transactions on Modeling and Computer Simulation 8 (1998) 3–30.

[7] W. Feller, An Introduction to Probability Theory and Its Applications, third ed., vol. 1, John Wiley & Sons, New York, 1970.

[8] P. L'Ecuyer, Good parameters and implementations for combined multiple recursive random number generators, Operations Research 47 (1999) 159–164.

[9] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in C, second ed., Cambridge University Press, Cambridge, UK, 1992.

[10] M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, M. Booth, F. Rossi, GNU Scientific Library Reference Manual, Network Theory, Bristol, 2001.

[11] F. Panneton, P. L'Ecuyer, M. Matsumoto, Improved long-period generators based on linear recurrences module 2, AMS Transactions on Mathematical Software 32 (2006) 1–16.