Nonlinear Dyn (2021) 104:727–737
https://doi.org/10.1007/s11071-021-06246-0

ORIGINAL PAPER

Improving chaos-based pseudo-random generators in finite-precision arithmetic

Aleksandra V. Tutueva · Timur I. Karimov · Lazaros Moysis · Erivelton G. Nepomuceno · Christos Volos · Denis N. Butusov

Received: 11 September 2020 / Accepted: 21 January 2021 / Published online: 24 February 2021

Abstract One of the widely-used ways in chaos-based cryptography to generate pseudo-random sequences is to use the least significant bits or digits of finite-precision numbers defined by the chaotic orbits. In this study, we show that the results obtained using such an approach are very prone to rounding errors and discretization effects. Thus, it appears that the generated sequences are close to random even when parameters correspond to non-chaotic oscillations. In this study, we confirm that the actual source of pseudo-random properties of bits in a binary representation of numbers can not be chaos, but computer simulation. We propose a technique for determining the maximum number of bits that can be used as the output of a pseudo-random sequence generator including chaos-based algorithms. The considered approach involves evaluating the difference of the binary representation of two points obtained by different numerical methods of the same order of accuracy. Experimental results show that such estimation can significantly increase the performance of the existing chaos-based generators. The obtained results can be used to reconsider and improve chaos-based cryptographic algorithms.

Keywords Chaos · Pseudo-random number generator · Floating-point data type · IEEE754-2008 · NIST tests

A. V. Tutueva
Department of Computer-Aided Design, Saint-Petersburg Electrotechnical University "LETI", 5, Professora Popova st., Saint Petersburg, Russia 197376
e-mail: avtutueva@etu.ru

T. I. Karimov · D. N. Butusov (✉)
Youth Research Institute, Saint-Petersburg Electrotechnical University "LETI", 5, Professora Popova st., Saint Petersburg, Russia 197376
e-mail: dnbutusov@etu.ru

T. I. Karimov
e-mail: tikarimov@etu.ru

L. Moysis · C. Volos
Laboratory of Nonlinear Systems - Circuits and Complexity (LaNSCom), Physics Department, Aristotle University of Thessaloniki, Thessaloniki, Greece
L. Moysis
e-mail: lmousis@physics.auth.gr

C. Volos
e-mail: volos@physics.auth.gr

E. G. Nepomuceno
Control and Modelling Group (GCOM), Department of Electrical Engineering, Federal University of São João del-Rei, São João del-Rei, MG 36307-352, Brazil
e-mail: nepomuceno@ufsj.edu.br

1 Introduction

Recently, chaos theory has found numerous applications in technical and engineering problems including secure data transfer and processing [1–11]. Topological mixing and sensitivity of chaotic systems allow developing cryptographic primitives for high-performance and secure communication systems [12,13]. Usually, chaos-based stream encryption schemes are based on pseudo-random number generators (PRNG) [14,15].

Some generation algorithms, such as [16–19], produce bits from chaotic orbits simulated with finite-precision arithmetic. From all the variety of available formats for representing real numbers in a computer, the floating-point data type established by the IEEE standard, is often used [20]. Most of the chaos-based generators in a floating-point implementation can be categorized into two groups depending on the underlying way of conversion of finite-precision numbers into pseudo-random bits or bytes. The first set includes algorithms involving the multiplication of numbers defining chaotic trajectories by a sufficiently large constant. The sequence numbers are taken modulo 2, 256, or other numbers, according to the algorithm [13,21–26]. Then the obtained result is converted to the byte format. Another technique implies bits extraction from the mantissa of the binary representation of floating-point numbers [16–19]. Thus, in many chaos-based PRNGs, to obtain output sequences the last digits or least bits of finite-precision numbers are commonly used.

It is known that the floating-point representation allows performing calculations in a larger range of values than fixed-point numbers [20]. However, this circumstance is fraught with pitfalls. In comparison with fixed-point arithmetic, not all real numbers can be represented exactly in the form of floating-point numbers while the same number of bits for representation is used. For example, the real number 0.1 can be approximately read as 0.100000000000000006 using the 64-bit floating-point data type according to the IEEE standard [20]. The imperfection of the floating-point representation combined with the accumulation of round-off errors even in simple arithmetic operations can distort the final result. In extreme cases, a loss of significance, also known as *catastrophic cancellation* is occurred [27]. Moreover, there is another source of error while generating pseudo-random sequences from chaotic orbits described by ordinary differential equations (ODE). Since such systems are computationally difficult or impossible to solve analytically, numerical integration is traditionally used. The continuous mathematical model is replaced by an approximate finite-difference scheme, thereby introducing the discretization effects into the results [28,29]. Combined with rounding errors of floating-point calculations, this leads to uncertain mixing of the least bits of numbers obtained through the ODE system simulation. This fact often causes the opposite problem—the degradation of chaotic dynamics [30–32].

Here the question arises: what is the real source of pseudo-random properties of sequences generated by the mentioned algorithms from numerical chaotic orbits? As it will be shown in this paper, even the 32 least significant bits of the mantissa of periodic signals generated with the finite-precision arithmetic can possess pseudo-random properties. Therefore, some of the known algorithms may possibly be based on numerical and discretization effects, and not on the specific features of chaotic systems. In this case, it is of interest to thoroughly investigate the binary representation of such floating-point numbers to determine the maximum number of bits or digits that can be used as the chaos-based PRNG output to obtain reliable results. For most of described generators, the technique for evaluating this value is unknown. High-performance methods allow obtaining 32 output bits at a single iteration of the chaotic system simulation [16,18]. However, the question is, can the performance of such algorithms be increased and how does it depend on the simulated system and the floating-point representation.

The key novelty of the reported study consists of two main advances. First, we explicitly show that periodic signals generated with finite precision can lead to pseudo-random properties of the least significant bits of numbers. This effect is also aggravated by the discretization method. We illustrate that in the case of chaotic systems it is possible to obtain pseudo-random sequences regardless of the oscillations mode. Moreover, we develop the technique for estimating the maximum number of pseudo-random bits that can be extracted from the binary representations of chaotic systems described by ODEs. We also propose a simple algorithm based on the error estimation between two points of the investigated system obtained using a pair of different finite-difference schemes. To increase the performance of existing chaos-based generators we determine this value for all state variables of the system. The obtained sequences pass all statistical tests that reveal the pseudo-random properties of examined sequences. Using the proposed approach allows getting 3-5 times more pseudo-random bits per calculation iteration than known algorithms.

The rest of the paper is organized as follows. In Sect. 2, the proposed algorithm is described and examined using non-chaotic systems. Then, in Sect. 3, two samples of chaos-based generators are investigated. Finally, some conclusions and discussions are given in Sect. 4.

2 Pseudo-random bit generation from floating-point numbers

Following IEEE 754-2008 standard [20], the binary representation of the floating-point data type consists of a sign, an exponent and the mantissa. In our study, we investigated the *double*-precision floating-point format that comprises 1 sign bit, 11 exponent bits, and 52 mantissa bits. To explicitly show the effects of numerical discretization and rounding, we considered the following ODE system

$$\begin{cases} \dot{x} = y \\ \dot{y} = -x \end{cases} \tag{1}$$

with initial conditions x_0 and y_0. To obtain a finite-difference model of harmonic oscillator (1) we applied the first-order Euler–Cromer symplectic integration method that is able to preserve the qualitative properties of the original Hamiltonian system [33]. The system (1) possesses the periodic solution as is shown in Fig. 1.

To generate bit sequences from the solution of ODE system we followed the idea of Francois et al. [16]. We extracted the least 32 mantissa bits of variable x of system (1) simulated over $N = 10^6/32$ iterations

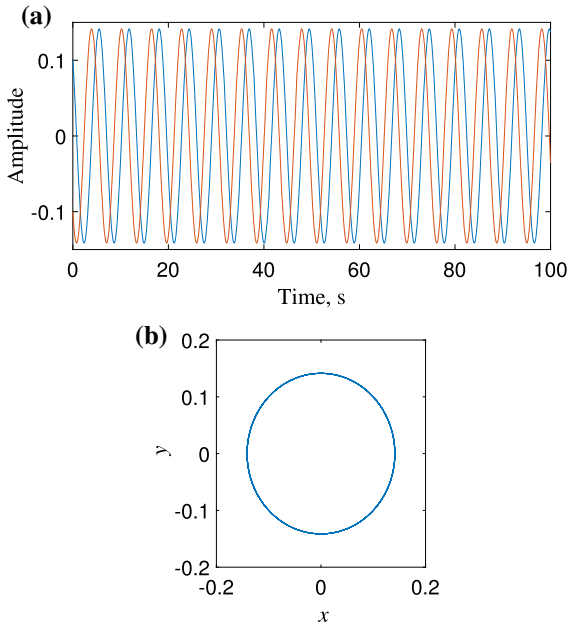

Fig. 1 **a** Time domain and **b** phase trajectory of system (1) simulated with $x_0 = 0.1$, $y_0 = -0.1$

with the integration step $h = 0.01$. The final simulation time was 312.5 seconds. The obtained bits were combined into the output sequence of length 10^6. We generated 1000 sequences for different x_0 values distributed uniformly in the interval [0; 1] while y_0 was constant.

To investigate the properties of obtained sequences, in this and further experiments we used the statistical testing suite proposed by NIST [34]. Each of the fifteen NIST tests is aimed at searching for certain patterns that are typical for non-random sequences. The results are a set of probability values p_{values} that illustrate how the studied sequences are close to random. If p_{values} is not lower than the significance level α, then the test is passed, and the sequence is taken as pseudo-random. To examine the generator, for each test one considers the number of passed sequences, as well as the distribution of p_{values}. In all experiments, we generated 10^3 sequences of length 10^6 and set α equal to 0.01. Thus, according to the NIST tests strategies [34], proportion of sequences passed each test must belong to the interval [0.980561; 0.999439] and the p_{value} calculated for distribution of p_{values} in the Pearson's χ^2 test (i.e., a p_{value} of the p_{values}) must be greater than or equal to 0.0001.

The results of the NIST statistical testing for sequences obtained from the linear system (1) are shown in the left column of Table 1. As one can see, despite that the simulated system is non-chaotic, the generated sequences passed all tests successfully. It can be assumed that the main source of error is the first-order discretization method. To prove this assumption, we repeated the experiment, replacing the ODE solver based on Euler–Cromer integration with the fourth-order Runge–Kutta method, which is default implemented in most modern simulation software [35]. The integration step was equal to 0.01. The results are presented in Table 2. Failure of the Run test shows that in studied sequences the series of 0 and 1 are alternated too slowly, as in periodic sequences. The results of the Approximate Entropy test pointed out that the distribution of m-bit words is not close to uniform. Moreover, failure of the Cumulative sums and Random Excursions tests indicates the uneven distribution of 1 at the beginning of sequences, as well as various subsequences. Thus, the experimental results are consistent with our hypothesis that one of the significant sources of bits mixing is the low-order numerical integration method.

Table 1 The results of NIST statistical testing with $\alpha = 0.01$ for sequences obtained from system (1) simulated using Euler–Cromer integration

Statistical test	Ratio	p_{value}	Result
Frequency	0.118120	0.988	Success
Block frequency	0.118812	0.985	Success
Runs	0.846338	0.987	Success
Longest run of ones	0.000546	0.982	Success
Rank	0.098920	0.992	Success
Discrete Fourier	0.657933	0.990	Success
Non-overlapping match	0.380407	0.986	Success
Overlapping math	0.801865	0.990	Success
Universal statistical	0.095426	0.991	Success
Linear complexity	0.045088	0.986	Success
Serial	0.053969	0.988	Success
Approximate entropy	0.548314	0.993	Success
Cumulative sums	0.375313	0.987	Success
Random excursions	0.406499	0.991	Success
Random excursions var	0.106246	0.987	Success

Table 2 The results of NIST statistical testing with $\alpha = 0.01$ for sequences obtained from system (1) simulated using Runge–Kutta integration

Statistical test	Ratio	p_{value}	Result
Frequency	0.134172	0.979	Fail
Block frequency	0.020269	0.987	Success
Runs	0.408275	0.991	Success
Longest run of ones	0.000269	0.983	Success
Rank	0.111389	0.988	Success
Discrete Fourier	0.125927	0.989	Success
Non-overlapping match	0.653773	0.991	Success
Overlapping math	0.518106	0.990	Success
Universal statistical	0.450297	0.989	Success
Linear complexity	0.361938	0.985	Success
Serial	0.299736	0.988	Success
Approximate entropy	0.0133805	0.977	Fail
Cumulative sums	0.064015	0.963	Fail
Random excursions	0.000586	0.963	Fail
Random excursions var	0.915317	0.975	Fail

If some bits of numbers generated by finite-difference schemes of periodic systems possess pseudo-random properties, then it is of interest to determine their maximum number. We propose the algorithm based on the difference between the binary representation of two solutions obtained by different integration schemes from the same initial point. It consists of the following steps:

1. Choose the set of initial conditions size N for the studied ODE system. Based on the results of a large number of experiments, we recommend choosing at least 1000 values that are distributed over the entire interval of state variables changes.
2. Get two various finite-difference schemes for numerical simulation. One can use two schemes obtained by different discrete operators of equal order.
3. Calculate one integration step for all initial conditions using two chosen finite-difference schemes.
4. Compare the binary representations of all state variables and find a vector of indices b of first bits from which they become different.
5. Calculate the mean values of b as $B_j = \text{mean}\lfloor b_{i,j} \rfloor$ for the entire set of initial conditions where is varied from 1 to N and j is the state variable index.

If we subtract the obtained values B from 64, we obtain the number of bits for each variable that are suitable for use as the PRNG output.

On step 2 we highly recommend using a pair of semi-explicit methods, since this is the simplest way to get two different finite-difference schemes with similar arithmetic operations. Using integration with different approximation methods, for example, explicit and implicit Euler methods, will reduce the accuracy of the proposed estimation. Moreover, using a pair of semi-explicit methods allows one to estimate the maximum number of bits for extraction even in the case when equations described the chaotic system include trigonometric and other mathematical functions, the implementation of which is not strictly standardized [20]. At the moment, it is impossible to predict theoretically the rounding error propagation of calculations results with such functions, especially if the absolute values of numbers with which the operations are performed are close [36]. Thus, we propose to use an empirical approach based on two semi-implicit models. In the case of stiff systems, extrapolation methods of a higher order can be used with semi-explicit integration as the basic method.

Table 3 The results of NIST statistical testing with $\alpha = 0.01$ for sequences obtained from system (1), 39 bits case

Statistical test	Ratio	p_{value}	Result
Frequency	0.230755	0.991	Success
Block frequency	0.540204	0.992	Success
Runs	0.021554	0.992	Success
Longest run of ones	0.042212	0.989	Success
Rank	0.428095	0.989	Success
Discrete Fourier	0.334538	0.988	Success
Non-overlapping match	0.415422	0.993	Success
Overlapping math	0.735908	0.992	Success
Universal statistical	0.878618	0.989	Success
Linear complexity	0.765632	0.995	Success
Serial	0.071620	0.990	Success
Approximate entropy	0.029011	0.988	Success
Cumulative sums	0.370262	0.992	Success
Random excursions	0.310049	0.989	Success
Random excursions var	0.016602	0.988	Success

Table 4 The results of NIST statistical testing with $\alpha = 0.01$ for sequences obtained from system (1), 40 bits case

Statistical test	Ratio	p_{value}	Result
Frequency	0.093157	0.975	Fail
Block frequency	0.699313	0.990	Success
Runs	0.883171	0.991	Success
Longest run of ones	0.059358	0.989	Success
Rank	0.038062	0.988	Success
Discrete Fourier	0.680755	0.989	Success
Non-overlapping match	0.651693	0.990	Success
Overlapping math	0.047673	0.989	Success
Universal statistical	0.233162	0.991	Success
Linear complexity	0.328297	0.981	Success
Serial	0.092041	0.990	Success
Approximate entropy success	0.036592	0.986	Success
Cumulative sums	0.504219	0.982	Success
Random excursions	0.030806	0.967	Fail
Random excursions var	0.556460	0.976	Fail

To test the proposed algorithm, we examined system (1) and obtained the pair of finite-difference schemes using Euler–Cromer integration:

$$\begin{cases} x_{n+1} = x_n + hy_n \\ y_{n+1} = y_n - hx_{n+1} \end{cases} \tag{2}$$

$$\begin{cases} y_{n+1} = y_n - hx_n \\ x_{n+1} = x_n + hy_{n+1} \end{cases} \tag{3}$$

where h is the integration step.

Using both state variables to generate pseudo-random sequences from system (1), can cause the failure of Serial and Overlapping templates tests, since the right-hand side functions differ only in a sign. Therefore, at stage 2 of the proposed algorithm, we considered the single variable x. We performed all the steps and got the number 39. Then we repeated the simulation of the investigated system with the scheme (2) for different initial values $N = 10^6/39$ times, obtained a subarray of length 10^6 and performed the NIST tests (Table 3). The obtained sequences passed all the tests. Then we increased the number of extracted bits to 40 and repeated the statistical testing (Table 4). One can note that this led to a change in the distribution of 0 and 1 bits in sequences and the Frequency test was failed. Moreover, the sequences did not pass the Random Excursions tests.

Let us consider the proposed algorithm to generate pseudo-random bits from chaotic systems.

3 Chaos-based pseudo-random number generation

We have chosen two chaotic systems of different dimensions as sample systems. In both cases, we used the first-order integration methods with $h = 0.001$. Since the obtained finite-difference schemes in computational costs are comparable to chaotic discrete maps, the proposed PRNGs will also possess a high speed in terms of computing in a single iteration as their counterparts based on discrete-time chaotic systems.

3.1 Rossler system

As the first sample nonlinear system, we choose the well-known Rossler oscillator [37], described as:

$$\begin{cases} \dot{x} = -y - z \\ \dot{y} = x + ay \\ \dot{z} = b + z(x - c) \end{cases} \tag{4}$$

where a, b and c are parameters. Following the bifurcation diagram (Fig. 2), we chose two values of c cor-

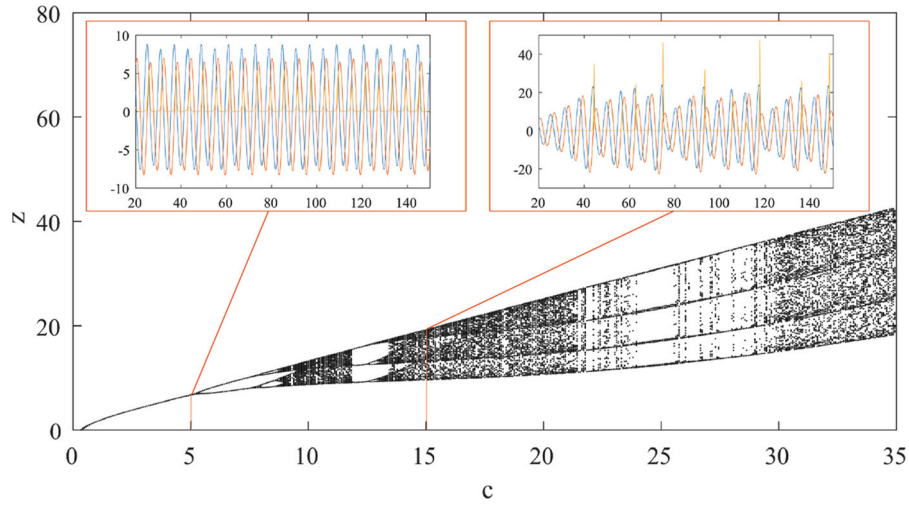

Fig. 2 Bifurcation diagram for the Rossler system for varying c

responding to chaotic behavior ($c = 15$) and harmonic oscillations ($c = 5$) while a and b were equal to 0.1.

In the proposed algorithm we used two semi-explicit systems of equations obtained using the pair of Euler–Cromer methods:

$$\begin{cases} x_{n+1} = x_n + h(-y_n - z_n) \\ y_{n+1} = y_n + h(x_{n+1} + ay_n) \\ z_{n+1} = z_n + h(b + z_n(x_{n+1} - c)) \end{cases} \quad (5)$$

$$\begin{cases} z_{n+1} = z_n + h(b + z_n(x_n - c)) \\ y_{n+1} = y_n + h(x_n + ay_n) \\ x_{n+1} = x_n + h(-y_{n+1} - z_{n+1}) \end{cases} \quad (6)$$

3.2 5D hyperchaotic Sprott B system

The ODE system proposed by Ojoniyi et. al [38]

$$\begin{cases} \dot{x} = yz - v \\ \dot{y} = x - y - u \\ \dot{z} = 1 - xy \\ \dot{u} = ax + y \\ \dot{v} = x \end{cases} \quad (7)$$

has more arithmetic operations than the Rossler system. It can be assumed that the number of bits that are suitable for pseudo-random generation is greater than in the case of the three-dimensional system (4) without the use of special techniques for errors reducing [39].

We performed our experiments with $a = 0.01$ and $a = -0.9$ obtained from the bifurcation diagram presented in Fig. 3. With $a = 0.01$ system (7) generates chaotic oscillations. The value $a = -0.9$ corresponds to the non-chaotic mode.

To generate bits, applying the Euler–Cromer integration we used the following finite-difference schemes

$$\begin{cases} x_{n+1} = x_n + h(y_n z_n - v_n) \\ y_{n+1} = y_n + h(x_{n+1} - y_n - u_n) \\ z_{n+1} = z_n + h(1 - x_{n+1} y_{n+1}) \\ u_{n+1} = u_n + h(ax_{n+1} + y_{n+1}) \\ v_{n+1} = x_n + hx_{n+1} \end{cases} \quad (8)$$

In our algorithm, as the second scheme, we chose the following equations

$$\begin{cases} v_{n+1} = v_n + hx_n \\ u_{n+1} = u_n + h(ax_n + y_n) \\ z_{n+1} = x_n + h(1 - x_n y_n) \\ y_{n+1} = y_n + h(x_n - y_n - u_{n+1}) \\ x_{n+1} = x_n + h(y_{n+1} z_{n+1} - v_{n+1}) \end{cases} \quad (9)$$

Let us consider the investigation results for sequences obtained by generators based on the two described chaotic systems.

3.3 Experimental results

We applied the proposed algorithm for 1000 different initial conditions of the Rossler model and the hyperchaotic Sprott B system using pairs of finite-difference

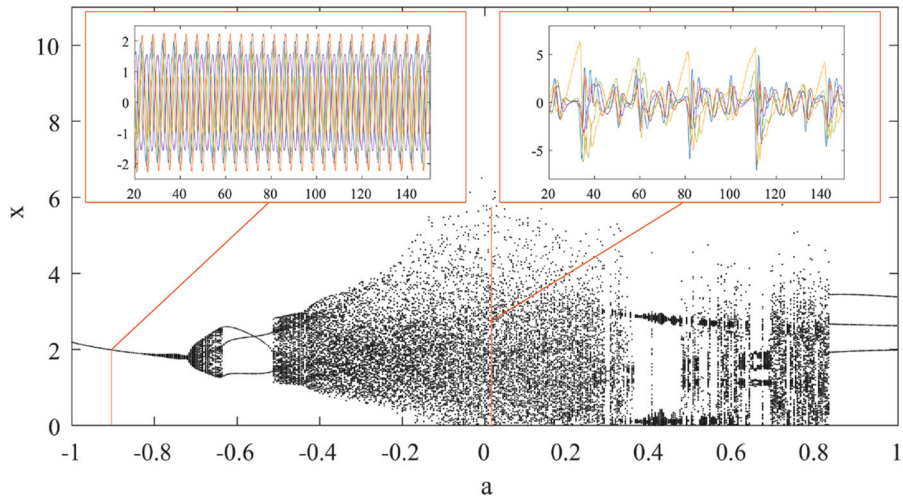

Fig. 3 Bifurcation diagram for the 5D hyperchaotic Sprott B system for varying a

Table 5 The results of evaluating the pseudo-random properties of the least bits of numbers generated by the Rossler system

Parameter	Indices (B)	Total extracted bits
$c = 15$	$B_x = 27, B_y = 22, B_z = 33$	110
$c = 5$	$B_x = 28, B_y = 22, B_z = 33$	109

Table 6 The results of evaluating the pseudo-random properties of the least bits of numbers generated by the Sprott B system

Parameter	Indices (B)	Total extracted bits
$a = 0.01$	$B_x = 30, B_y = 28, B_z = 31, B_u = 29, B_v = 24$	178
$a = -0.9$	$B_x = 30, B_y = 31, B_z = 31, B_u = 28, B_v = 24$	176

schemes (5), (6) and (8), (9) to estimate the maximum number of bits with pseudo-random properties. To show that the main source of mixing of the least bits is the discretization and inaccuracy of operations with finite numbers, we studied the harmonic and chaotic modes of the sample cases. We completed all the steps of the proposed algorithm and got the indices presented in Tables 5, 6 for both systems, respectively. The rightmost column shows the sum of bits that can be used for generation, obtained by subtracting each value of vector B from 64. As one can see, in both cases we obtained similar estimates for chaotic and non-chaotic oscillations. The small difference can be explained by the features of floating-point calculations, which imply the normalization of numbers before performing an arithmetic operation. Since in the normalized form of the taken parameter values possess different exponents and mantissae, then the calculation result may be rounded differently. Hence, the number of significant bits of the mantissa affected by the changes can increase or decrease.

We tested the sequences obtained by extracting the least significant mantissa bits of numbers of numerical orbits calculated using (5) and (8) for both systems, respectively. The results for $\alpha = 0.01$ are shown in Tables 7, 8, 9 and 10 and for $\alpha = 0.005$—in "Appendices A and B". As one can see, all tests were successfully passed. It can be noted that, in comparison with the case of the linear system, the number of pseudo-random bits extracted from each variable increased insignificantly. Moreover, the sequences obtained from the non-chaotic oscillations successfully passed random tests too. This confirms our assumption about the large degree of influence of discretization effects and rounding errors. Moreover, the performance of existing chaos-based pseudo-random generators, such as algorithms described in [16,18], can be significantly increased with the proposed estimation technique. We can improve the generation speed by more than five times using high-dimensional chaotic systems simulated by methods of the first or second order of accuracy. This enhances the advantages of chaos-based stream ciphers in comparison with traditional encryption schemes, especially in the case of multimedia processing [40].

Table 7 The results of NIST statistical testing with $\alpha = 0.01$ for sequences obtained from the Rossler system, chaotic mode ($c = 15$)

Statistical test	Ratio	p_{value}	Result
Frequency	0.013991	0.992	Success
Block frequency	0.544254	0.986	Success
Runs	0.650830	0.989	Success
Longest run of ones	0.015444	0.989	Success
Rank	0.060912	0.988	Success
Discrete Fourier	0.028056	0.988	Success
Non-overlapping match	0.270265	0.989	Success
Overlapping math	0.620465	0.987	Success
Universal statistical	0.242986	0.993	Success
Linear complexity	0.015816	0.985	Success
Serial	0.794391	0.993	Success
Approximate entropy	0.129620	0.988	Success
Cumulative sums	0.660012	0.990	Success
Random excursions	0.082513	0.985	Success
Random excursions var	0.123038	0.984	Success

Table 8 The results of NIST statistical testing with $\alpha = 0.01$ for sequences obtained from the Rossler system, harmonic mode ($c = 5$)

Statistical test	Ratio	p_{value}	Result
Frequency	0.057510	0.993	Success
Block frequency	0.753844	0.994	Success
Runs	0.739918	0.988	Success
Longest run of ones	0.116065	0.984	Success
Rank	0.228367	0.991	Success
Discrete Fourier	0.037320	0.985	Success
Non-overlapping match	0.266235	0.992	Success
Overlapping math	0.163513	0.989	Success
Universal statistical	0.666245	0.990	Success
Linear complexity	0.448424	0.981	Success
Serial	0.181557	0.988	Success
Approximate entropy	0.233162	0.987	Success
Cumulative sums	0.518106	0.992	Success
Random excursions	0.931185	0.990	Success
Random excursions var	0.926487	0.993	Success

Table 9 The results of NIST statistical testing with $\alpha = 0.01$ for sequences obtained from the hyperchaotic Sprott B system, chaotic mode ($a = 0.01$)

Statistical test	Ratio	p_{value}	Result
Frequency	0.259616	0.991	Success
Block frequency	0.308561	0.991	Success
Runs	0.404728	0.993	Success
Longest run of ones	0.195864	0.988	Success
Rank	0.036352	0.991	Success
Discrete Fourier	0.463512	0.986	Success
Non-overlapping match	0.145326	0.991	Success
Overlapping math	0.135720	0.994	Success
Universal statistical	0.996335	0.992	Success
Linear complexity	0.459717	0.982	Success
Serial	0.875539	0.996	Success
Approximate entropy	0.711601	0.991	Success
Cumulative sums	0.399442	0.990	Success
Random excursions	0.046269	0.990	Success
Random excursions var	0.406499	0.984	Success

Table 10 The results of NIST statistical testing with $\alpha = 0.01$ for sequences obtained from the hyperchaotic Sprott B system, harmonic mode ($a = -0.9$)

Statistical test	Ratio	p_{value}	Result
Frequency	0.520102	0.995	Success
Block frequency	0.087162	0.991	Success
Runs	0.869278	0.987	Success
Longest run of ones	0.024688	0.988	Success
Rank	0.221317	0.990	Success
Discrete Fourier	0.488534	0.982	Success
Non-overlapping match	0.165340	0.992	Success
Overlapping math	0.144504	0.984	Success
Universal statistical	0.072066	0.989	Success
Linear complexity	0.933472	0.990	Success
Serial	0.492436	0.990	Success
Approximate entropy	0.138069	0.995	Success
Cumulative sums	0.832561	0.992	Success
Random excursions	0.741918	0.987	Success
Random excursions var	0.055714	0.985	Success

4 Discussion

The obtained results pointed out that one should be very careful while simulating chaotic systems using a discrete computer with finite-precision numbers. On the one hand, rounding and discretization errors lead to the pseudo-random properties of the least bits even in the case of linear systems. However, when chaotic

systems are considered, this causes dynamics degradation and the periodicity that reduces the security of chaos-based cryptosystems [41–43]. Moreover, the discretization effects can lead to the fact that the chaotic system model gains new properties that the continuous prototype does not possess [28]. Therefore, electronic circuits with chaotic behavior seem to be a more reliable source of random numbers [44]. It can be assumed that for finite-precision numbers obtained from the analog-to-digital converter, the generation method based on extracting the least bits from the fractional part of the number will yield weakly correlated sequences with pseudo-random properties. The topic of our further studies will be the investigation of such approaches.

5 Conclusion

In this paper we considered pseudo-random bits generation using the double-precision floating-point data type. We have shown that several bits of the mantissa of numbers obtained using numerical integration for solving linear differential equations have pseudo-random properties according to the NIST suite. We proposed the novel algorithm to calculate the maximum number of bits that are suitable for pseudo-random sequences generation from the binary representation of floating-point numbers. We applied this technique in a comparative study of linear oscillator and chaotic ODE systems as a source of pseudo-randomness. It was found that in both cases sequences derived from the least bits of the mantissa of floating-point numbers successfully passed NIST statistical tests. We explicitly showed that using the proposed approach, it is possible to increase the performance of chaos-based pseudo-random number generators.

Acknowledgements We thank the anonymous reviewers for their careful reading of our manuscript and their many insightful comments and suggestions.

Funding This study was supported by the Grant of the Russian Science Foundation (RSF), Project 20-79-10334.

Data availability statement Not applicable.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix A: The results of NIST statistical testing with $\alpha = 0.005$ for sequences obtained from the Rossler system

See Tables 11 and 12.

Table 11 chaotic mode ($c = 15$)

Statistical test	Ratio	p_{value}	Result
Frequency	0.993	0.769527	Success
Block frequency	0.997	0.350485	Success
Runs	0.999	0.939005	Success
Longest run of ones	0.993	0.751866	Success
Rank	0.995	0.126658	Success
Discrete Fourier	0.994	0.278461	Success
Non-overlapping match	0.991	0.288249	Success
Overlapping math	0.995	0.851383	Success
Universal statistical	0.996	0.848027	Success
Linear complexity	0.991	0.015707	Success
Serial	0.996	0.632955	Success
Approximate entropy	0.991	0.031848	Success
Cumulative sums	0.992	0.257004	Success
Random excursions	0.990	0.948298	Success
Random excursions var	0.990	0.097743	Success

Table 12 harmonic mode ($c = 5$)

Statistical test	Ratio	p_{value}	Result
Frequency	0.996	0.798139	Success
Block frequency	0.996	0.155499	Success
Runs	0.989	0.394195	Success
Longest run of ones	0.995	0.344048	Success
Rank	0.992	0.068571	Success
Discrete Fourier	0.990	0.248014	Success
Non-overlapping match	0.998	0.179584	Success
Overlapping math	0.997	0.973718	Success
Universal statistical	0.993	0.366918	Success
Linear complexity	0.990	0.149495	Success
Serial	0.992	0.244236	Success
Approximate entropy	0.992	0.011709	Success
Cumulative sums	0.996	0.618385	Success
Random excursions	0.991	0.544254	Success
Random excursions var	0.991	0.216713	Success

Appendix B: The results of NIST statistical testing with $\alpha = 0.005$ for sequences obtained from the hyperchaotic Sprott B system

See Tables 13 and 14.

Table 13 chaotic mode ($a = 0.01$)

Statistical test	Ratio	p_{value}	Result
Frequency	0.994	0.689019	Success
Block frequency	0.993	0.363593	Success
Runs	0.999	0.143686	Success
Longest run of ones	0.993	0.126658	Success
Rank	0.994	0.039587	Success
Discrete Fourier	0.991	0.428095	Success
Non-overlapping match	0.995	0.342451	Success
Overlapping math	0.998	0.239266	Success
Universal statistical	0.991	0.063217	Success
Linear complexity	0.990	0.246750	Success
Serial	0.996	0.402962	Success
Approximate entropy	0.991	0.220159	Success
Cumulative sums	0.994	0.566688	Success
Random excursions	0.994	0.079051	Success
Random excursions Var	0.996	0.568739	Success

Table 14 harmonic mode ($a = -0.9$)

Statistical test	Ratio	p_{value}	Result
Frequency	0.998	0.717714	Success
Block frequency	0.994	0.202268	Success
Runs	0.996	0.070299	Success
Longest run of ones	0.995	0.260930	Success
Rank	0.995	0.235589	Success
Discrete Fourier	0.993	0.221317	Success
Non-overlapping match	0.993	0.184549	Success
Overlapping math	0.996	0.599693	Success
Universal statistical	0.989	0.415422	Success
Linear complexity	0.990	0.240501	Success
Serial	0.993	0.123038	Success
Approximate entropy	0.993	0.131879	Success
Cumulative sums	0.998	0.846338	Success
Random excursions	0.993	0.281232	Success
Random excursions Var	0.990	0.410055	Success

References

1. Irani, B.Y., Ayubi, P., Jabalkandi, F.A., Valandar, M.Y., Barani, M.J.: Digital image scrambling based on a new one-dimensional coupled sine map. Nonlinear Dyn. **97**(4), 2693–2721 (2019)
2. Lambić, D., Nikolić, M.: Pseudo-random number generator based on discrete-space chaotic map. Nonlinear Dyn. **90**(1), 223–232 (2017)
3. Karimov, T., Nepomuceno, E.G., Druzhina, O., Karimov, A., Butusov, D.: Chaotic oscillators as inductive sensors: theory and practice. Sensors **19**(19), 4314 (2019)
4. Parastesh, F., Jafari, S., Azarnoush, H., Hatef, B., Namazi, H., Dudkowski, D.: Chimera in a network of memristor-based Hopfield neural network. Eur. Phys. J. Spec. Top. **228**(10), 2023–2033 (2019)
5. Moysis, L., Petavratzis, E., Volos, C., Nistazakis, H., Stouboulos, I.: A chaotic path planning generator based on logistic map and modulo tactics. Rob. Auton. Syst. **124**, 103377 (2020)
6. Datcu, O., Macovei, C., Hobincu, R.: Chaos based cryptographic pseudo-random number generator template with dynamic state change. Appl. Sci. **10**(2), 451 (2020)
7. Kuiate, G.F., Rajagopal, K., Kingni, S.T., Tamba, V.K., Jafari, S.: Autonomous Van der Pol-Duffing snap oscillator: analysis, synchronization and applications to real-time image encryption. Int. J. Dyn. Control **6**(3), 1008–1022 (2018)
8. Dmitriev, A.S., Mokhseni, T.I., Teran, K.S.: Differentially coherent information transmission based on chaotic radio pulses. J. Commun. Technol. Electron. **63**(10), 1183–1190 (2018)
9. Öztürk, I., Kılıç, R.: A novel method for producing pseudo random numbers from differential equation-based chaotic systems. Nonlinear Dyn. **80**(3), 1147–1157 (2015)
10. Ye, G., Jiao, K., Huishan, W., Pan, C., Huang, X.: An asymmetric image encryption algorithm based on a fractional-order chaotic system and the rsa public-key cryptosystem. Int. J. Bifurc. Chaos **30**(15), 2050233 (2020)
11. Ye, G., Pan, C., Huang, X., Mei, Q.: An efficient pixel-level chaotic image encryption algorithm. Nonlinear Dyn. **94**(1), 745–756 (2018)
12. Hasimoto-Beltrán, R., Mota-García, E.: Real-time secure multimedia communication system based on chaos theory. In: Pacific-Rim Conference on Multimedia, pp. 441–445. Springer (2007)
13. Ye, H.-S., Zhou, N.-R., Gong, L.-H.: Multi-image compression-encryption scheme based on quaternion discrete fractional Hartley transform and improved pixel adaptive diffusion. Signal Processing **175**, 107652 (2020)
14. Kanso, A., Smaoui, N.: Irregularly decimated chaotic map (s) for binary digits generations. Int. J. Bifurc. Chaos **19**(04), 1169–1183 (2009)
15. Palacios-Luengas, L., Pichardo-Méndez, J.L., Díaz-Méndez, J.A., Rodríguez-Santos, F., Vázquez-Medina, R.: PRNG based on skew tent map. Arab. J. Sci. Eng. **44**(4), 3817–3830 (2019)
16. François, M., Defour, D., Negre, C.: A fast chaos-based pseudo-random bit generator using binary64 floating-point arithmetic. Informatica **38**, 115–124 (2014)

17. Flores-Vergara, A., García-Guerrero, E.E., Inzunza-González, E., López-Bonilla, O.R., Rodríguez-Orozco, E., Cárdenas-Valdez, J.R., Tlelo-Cuautle, E.: Implementing a chaotic cryptosystem in a 64-bit embedded system by using multiple-precision arithmetic. Nonlinear Dyn. **96**(1), 497–516 (2019)
18. Lui, O.Y., Yuen, C.H., Wong, K.W.: A pseudo-random number generator employing multiple Renyi maps. Int. J. Mod. Phys. C **24**(11), 1350079 (2013)
19. Alawida, M., Samsudin, A., Teh, J.: Enhanced digital chaotic maps based on bit reversal with applications in random bit generators. Inf. Sci. **512**, 1155–1169 (2020)
20. Institute of Electrical and Electronics Engineers. 754-2008-IEEE standard for floating-point arithmetic. IEEE (2008)
21. Lambić, D.: Security analysis and improvement of the pseudo-random number generator based on quantum chaotic map. Nonlinear Dyn. **94**(2), 1117–1126 (2018)
22. Kwok, H.S., Tang, W.K.: A fast image encryption system based on chaotic maps with finite precision representation. Chaos Solitons Fractals **32**(4), 1518–1529 (2007)
23. Akhshani, A., Akhavan, A., Mobaraki, A., Lim, S.C., Hassan, Z.: Pseudo random number generator based on quantum chaotic map. Commun. Nonlinear Sci. Numer. Simul. **19**(1), 101–111 (2014)
24. Liu, L., Miao, S., Cheng, M., Gao, X.: A pseudorandom bit generator based on new multi-delayed Chebyshev map. Inf. Process. Lett. **116**(11), 674–681 (2016)
25. García-Martínez, M., Ontañón-García, L., Campos-Cantón, E., Čelikovskỳ, S.: Hyperchaotic encryption based on multi-scroll piecewise linear systems. Appl. Math. Comput. **270**, 413–424 (2015)
26. Wu, X., Wang, D., Kurths, J., Kan, H.: A novel lossless color image encryption scheme using 2D DWT and 6D hyperchaotic system. Inf. Sci. **349**, 137–153 (2016)
27. Cuyt, A., Verdonk, B., Becuwe, S., Kuterna, P.: A remarkable example of catastrophic cancellation unraveled. Computing **66**(3), 309–320 (2001)
28. Butusov, D., Karimov, A., Tutueva, A., Kaplun, D., Nepomuceno, E.G.: The effects of Padé numerical integration in simulation of conservative chaotic systems. Entropy **21**(4), 362 (2019)
29. Karimov, T.I., Butusov, D.N., Pesterev, D.O., Predtechenskii, D.V., Tedoradze, R.S.: Quasi-chaotic mode detection and prevention in digital chaos generators. In: 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), pp. 303–307. IEEE (2018)
30. Li, S., Chen, G., Mou, X.: On the dynamical degradation of digital piecewise linear chaotic maps. Int. J. Bifurc. Chaos **15**(10), 3119–3151 (2005)
31. Lv-Chen, C., Yu-Ling, L., Sen-Hui, Q., Jun-Xiu, L.: A perturbation method to the tent map based on Lyapunov exponent and its application. Chin. Phys. B **24**(10), 100501 (2015)
32. Teh, J.S., Alawida, M., Ho, J.J.: Unkeyed hash function based on chaotic sponge construction and fixed-point arithmetic. Nonlinear Dyn. **100**, 713–729 (2020)
33. Hairer, E., Lubich, C., Wanner, G.: Geometric numerical integration illustrated by the Störmer–Verlet method. Acta Numer. **12**, 399–450 (2003)
34. Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E.: A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, Booz-Allen and Hamilton Inc. McLean (2001)
35. Alcin, M.: The Runge Kutta-4 based 4D hyperchaotic system design for secure communication applications. Chaos Theory Appl. **2**(1), 23–30 (2020)
36. Goldberg, D.: What every computer scientist should know about floating-point arithmetic. ACM Comput. Surv. (CSUR) **23**(1), 5–48 (1991)
37. Rössler, O.E.: An equation for continuous chaos. Phys. Lett. A **57**(5), 397–398 (1976)
38. Ojoniyi, O.S., Njah, A.N.: A 5D hyperchaotic Sprott B system with coexisting hidden attractors. Chaos Solitons Fractals **87**, 172–181 (2016)
39. Mann, W.R.: Mean value methods in iteration. Proc. Am. Math. Soc. **4**(3), 506–510 (1953)
40. Su, Z., Zhang, G., Jiang, J.: Multimedia security: a survey of chaos-based encryption technology. In: Multimedia—A Multidisciplinary Approach to Complex Issues, pp. 99–124. InTech (2012)
41. Liu, B., Xiang, H., Liu, L.: Reducing the dynamical degradation of digital chaotic maps with time-delay linear feedback and parameter perturbation. Math. Probl. Eng. (2020). https://doi.org/10.1155/2020/4926937
42. Fan, C., Ding, Q.: Effects of limited computational precision on the discrete chaotic sequences and the design of related solutions. Complexity (2019). https://doi.org/10.1155/2019/3510985
43. Fan, C., Ding, Q.: Analysing the dynamics of digital chaotic maps via a new period search algorithm. Nonlinear Dyn. **97**(1), 831–841 (2019)
44. Adiyaman, Y., Emiroglu, S., Ucar, M.K., Yildiz, M.: Dynamical analysis, electronic circuit design and control application of a different chaotic system. Chaos Theory Appl. **2**(1), 8–14 (2020)