fractal and fractional

MDPI

Article

Fractal Tent Map with Application to Surrogate Testing

Ekaterina Kopets [1,*], Vyacheslav Rybin [1], Oleg Vasilchenko [1], Denis Butusov [2], Petr Fedoseev [2] and Artur Karimov [1,*]

[1] Youth Research Institute, Saint Petersburg Electrotechnical University "LETI", 197022 Saint Petersburg, Russia; vgrybin@etu.ru (V.R.); ovvasilchenko@stud.etu.ru (O.V.)
[2] Computer-Aided Design Department, Saint Petersburg Electrotechnical University "LETI", 197022 Saint Petersburg, Russia; dnbutusov@etu.ru (D.B.); psfedoseev@etu.ru (P.F.)
* Correspondence: eekopets@etu.ru (E.K.); aikarimov@etu.ru (A.K.)

Abstract: Discrete chaotic maps are a mathematical basis for many useful applications. One of the most common is chaos-based pseudorandom number generators (PRNGs), which should be computationally cheap and controllable and possess necessary statistical properties, such as mixing and diffusion. However, chaotic PRNGs have several known shortcomings, e.g., being prone to chaos degeneration, falling in short periods, and having a relatively narrow parameter range. Therefore, it is reasonable to design novel simple chaotic maps to overcome these drawbacks. In this study, we propose a novel fractal chaotic tent map, which is a generalization of the well-known tent map with a fractal function introduced into the right-hand side. We construct and investigate a PRNG based on the proposed map, showing its high level of randomness by applying the NIST statistical test suite. The application of the proposed PRNG to the task of generating surrogate data and a surrogate testing procedure is shown. The experimental results demonstrate that our approach possesses superior accuracy in surrogate testing across three distinct signal types—linear, chaotic, and biological signals—compared to the MATLAB built-in randn() function and PRNGs based on the logistic map and the conventional tent map. Along with surrogate testing, the proposed fractal tent map can be efficiently used in chaos-based communications and data encryption tasks.

Keywords: chaos; pseudorandom bit generation; surrogate testing; logistic map; tent map; NIST tests

check for updates

Citation: Kopets, E.; Rybin, V.; Vasilchenko, O.; Butusov, D.; Fedoseev, P.; Karimov, A. Fractal Tent Map with Application to Surrogate Testing. *Fractal Fract.* **2024**, *8*, 344. https://doi.org/10.3390/fractalfract8060344

Academic Editor: Esteban Tlelo-Cuautle

Received: 2 April 2024
Revised: 4 June 2024
Accepted: 5 June 2024
Published: 7 June 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Chaos theory presents a prosperous ground for investigation and has a great interdisciplinary influence. It finds applications in various fields of research and development, concerning both theory and practice, e.g., in physics, engineering, economics, and biology [1]. Chaotic maps naturally allow for generating pseudorandom sequences, so they are widely used in designing pseudorandom number generators (PRNGs) for various purposes: cryptography [2], secure communications [3], numerical simulations and statistical modeling, noise generation [4], and image encryption [5,6]. Unlike traditional PRNGs, such as the Mersenne Twister, which rely on number-theoretical methods, chaos-based PRNGs harness the inherent unpredictability of chaotic systems to produce number sequences with high complexity and better statistical properties [7].

In recent decades, a lot of studies have been carried out concerning PRNGs based on chaotic maps. For example, in [8], Rezk et al. proposed a reconfigurable FPGA-based chaotic PRNG based on the Lorenz and Lü chaotic systems. The authors of [9] proposed a hardware-based PRNG using the single skew tent map [10]. Investigations show the high applicability of the coupled skew tent map [11] and the cross-coupled skew tent map [12] to pseudorandom number generation. The well-known chaotic Bernoulli map was investigated in a work [13], where the authors present the results of running standardized tests and prove that the proposed PRNG based on the Bernoulli map has good statistical properties. In another article [14], Moysis et al. present a simple method for designing

a pseudorandom bit generator by extracting multiple bits per iteration from the decimal part of a chaotic map. This is carried out by selecting the decimal part of the state in each iteration and comparing each digit separately to a threshold value. In another study [15], Moysis et al. propose a modification of the classical logistic map by introducing fuzzy triangular numbers. The authors show higher complexity compared to the classic logistic map and showcase several phenomena, like antimonotonicity and crisis. Another interesting approach to implementing PRNGs is discussed in a recent study by Kvitko et al. [16], which describes a PRNG scheme based on the Courbage–Nekorkin neuron and its application to chaotic path planning. The authors of [10] show that coupling polynomial chaotic maps expands the number of chaos parameters. To further increase the randomness and uniformity, a PRNG based on the hyperchaotic map with microcontroller implementation was proposed by Murillo-Escobar et al. [17].

Many state-of-the-art PRNGs use modified chaotic maps to obtain necessary statistical properties. However, these systems face several pitfalls arising from a variety of factors. One of them is falling in short time periods [7]. To prevent this from happening, the authors of [7] decided to dynamically change the parameters of the chaotic system while increasing the randomness of the generated sequences. Another common shortcoming is a degeneration phenomenon, when the chaotic behavior of the underlying map deteriorates over time, leading to decreased randomness and increased predictability. Garcia-Bosque et al. [18] proposed a transform arithmetic to improve the properties of chaos-based PRNGs. Another drawback of chaos-based algorithms is that generating pseudorandom sequences using some popular approaches, e.g., using the least significant bits or digits of finite-precision numbers, is susceptible to rounding errors and discretization effects, as described in [19]. To mitigate this effect, the authors proposed an algorithm for the calculation of the maximum number of bits that are suitable for pseudorandom sequence generation from the binary representation of floating-point numbers. Another issue with PRNGs based on chaotic maps is having a narrow range of parameters suitable for exhibiting dynamical chaos, which reduces the keyspace in encryption tasks. To expand the keyspace of the designed system, Zhao et al. [20] combined a sequence based on quantum random walks with chaotic system outputs, creating a new system with high randomization, unpredictability, and uniform output distribution. The digital implementation of chaotic maps also has a negative influence on the properties of PRNGs. Many researchers apply their efforts to overcome the degradation of chaotic dynamics while being implemented in hardware. For example, Merah et al. [21] proposed a circuit design based on the logistic chaotic map, which enhanced chaotic map randomness and made the cycle length longer using a special perturbation mechanism.

The known solutions to the abovementioned problems, such as additional periodic perturbation or periodic parameter change, seem rather artificial and overcomplicate the algorithm. They also require introducing several excessive parameters, like perturbation frequency, which need to be handled with care, so they are definitely not a "magic pill" in this regard. Another known way to increase the randomness of the output sequence is using a more complicated right-hand-side function in the basic chaotic map. Fractal functions that possess inherently complex structures are good candidates for this role [22]. Apart from famous computationally complex variants like the Weierstrass function, several simpler fractal functions exist, for example, fractal piecewise linear functions. In this study, we introduce a fractal generalization of the tent map and apply it to the pseudorandom number generation and surrogate testing tasks. Noise generators (usually generating white noise) are often used as part of some larger system, such as a neural network or synthetic or surrogate data generators. It is very important to maintain a high quality of generated noise for the good operation of the entire system. Testing the signal with surrogate data is performed to check the hypothesis of signal nonlinearity. In our study, we explicitly show that the use of a PRNG as a noise generator improves the accuracy of the surrogate test for biological signal testing. The experimental work demonstrates that the designed

generator outperforms algorithms based on well-known simpler maps and, surprisingly, the MATLAB built-in pseudorandom number generator.

The main contributions of this study can be summarized as follows:

1. A novel fractal chaotic map is presented, being a generalization of the chaotic tent map with a fractal structure.
2. A pseudorandom number generator (PRNG) was developed based on the fractal tent map proposed in this paper. The developed generator successfully passes all NIST tests.
3. The developed PRNG was applied to the surrogate data testing task. The test results show that the use of the PRNG based on the fractal tent map shows better accuracy when testing data sequences for nonlinearity than the standard approach.

The rest of this paper is organized as follows. Section 2 contains a description of the materials and methods used in the study, including a description of the proposed chaotic fractal tent map. Section 3 describes the analysis results, including bifurcation diagrams of the proposed map and LLE plots. The results of NIST tests for the developed PRNG based on the proposed map are given, as well as an example of using the developed PRNG for surrogate testing. Finally, Section 4 concludes the paper.

2. Materials and Methods

In the context of dynamical systems, a map is a mathematical function that defines how a point in a space (e.g., Banach or Euclidean space) evolves over time. Chaotic maps exhibit dynamical chaos and can be characterized by their sensitivity to initial conditions, meaning that even small changes in the starting point can lead to vastly different outcomes, which is a hallmark of chaos theory. Discrete chaotic maps, also known as iterated maps, apply a function repeatedly to iterate points through discrete time. In this study, we introduce a novel chaotic fractal tent map and investigate its properties and possible applications to pseudorandom number generation. To compare the developed chaotic fractal map with existing solutions, we will also consider the well-known logistic map [23] and the tent map [24].

2.1. Logistic and Tent Maps

The logistic map is described by the following recurrent formula:

$$x_{n+1} = kx_n(1 - x_n), \tag{1}$$

where k is a parameter, and x is a state variable. The interval $[0, 1]$ is covered when $k = 4$.

The tent map can be described as follows:

$$x_{n+1} = f_\mu(x_n) = \begin{cases} \mu x_n, & if\ x_n < 0.5, \\ \mu(1 - x_n), & if\ x_n \geq 0.5, \end{cases} \tag{2}$$

where μ is a positive real constant, and x is a state variable. The interval $[0, 1]$ is covered when $\mu = 2$. Figure 1 shows the graphs of the logistic map function (on the left) and the tent map function (on the right).

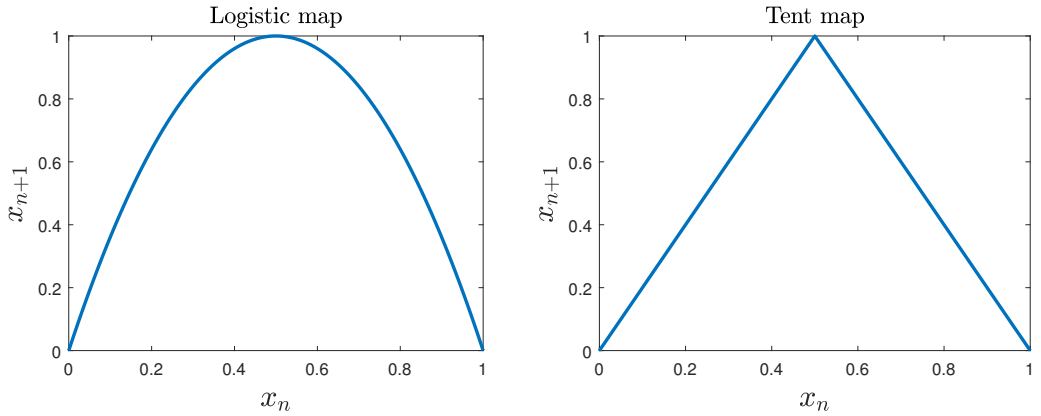


Figure 1. Graphs of the logistic map function (on the **left**) and the tent map function (on the **right**).

2.2. Proposed Fractal Tent Map

Let us propose the following modification of the tent map:

$$x_{n+1} = \mathcal{F}_f(x_n), \tag{3}$$

such that the two initial intervals $x_n < 0.5$ and $x_n \geq 0.5$ are iteratively divided into subintervals of lengths $K^i D$, where $i \in \mathbb{Z}_+$, $0 < K \leq 1$, and $0 < D \leq 0.5$. The parameter K stands for the contraction ratio of the interval after each division, D stands for the initial division, and i is an iteration counter, standing for the number of divisions. The construction of the map is illustrated in Figure 2. An iterative procedure for searching for the value of x_{n+1} is given in pseudo-code in Algorithm 1.

Algorithm 1: Evaluation of $x_{n+1} = \mathcal{F}_f(x_n)$.

input : x_n
output: x_{n+1}

if $x_n < 0.5$ **then**
  $l \leftarrow D$
  $m \leftarrow 0$
else
  $l \leftarrow 1$
  $m \leftarrow 1 - D$
while *True* **do**
  if $m < x_n \leq l$ **then**
    if $x_n < 0.5$ **then**
      $x_{n+1} \leftarrow \frac{x_n - m}{l - m}$
    else
      $x_{n+1} \leftarrow \frac{l - x_n}{l - m}$
    return x_{n+1}
  else
    $D \leftarrow KD$
    if $x_n < 0.5$ **then**
      $m \leftarrow l$
      $l \leftarrow l + D$
    else
      $l \leftarrow m$
      $m \leftarrow m - D$

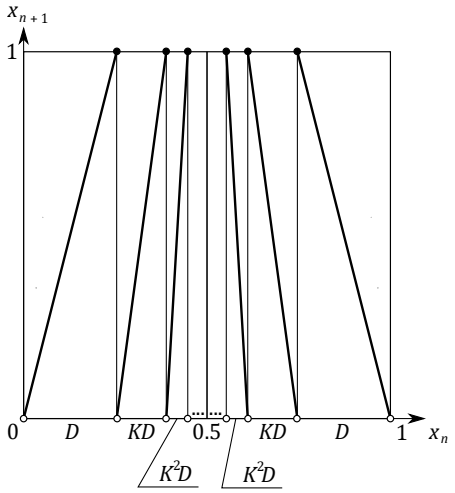


Figure 2. Fractal tent map construction.

In this algorithm, we first determine where x_n lies relative to the value 0.5. Then, the two variables m and l, playing the role of the left and right bounds, respectively, are used to search for an exact interval containing x_n, starting from one of the two side intervals of length D. The initial values of m and l are 0 and D, respectively, for the left subinterval, and $1 - D$ and 1, respectively, for the right subinterval. The iterative modification of these bounds belongs to a class of dichotomic search algorithms, like a renowned binary search. When the interval is found where $m < x_n \leq l$, the equation of a line is used to estimate the value of x_{n+1}.

This map can be considered a generalization of the tent map, and if

$$\begin{cases} K < 1, \\ \lim\limits_{N\to\infty} \sum\limits_{i=0}^{N} K^i D = \frac{1}{2}, \end{cases}$$

the system $\Big([0,1], \mathcal{F}_f(x)\Big)$ is a hyperbolic iterated function system with the attractor $\frac{1}{2}$, and $\mathcal{F}_f(x)$ becomes a fractal function [22].

Let us give some examples of the fractal and non-fractal behavior of $F_f(x)$. One may recall that the sum of a geometric progression written in terms of D and K for $K < 1$ is

$$\lim_{N\to\infty} S_N = \frac{D}{1-K}. \tag{4}$$

Therefore, the fractal behavior of $F_f(x)$ can be observed for any pair (D, K) satisfying

$$1 - K = 2D,$$

for example, $D = 0.25$, $K = 0.5$, or $D = \frac{1}{3}$, $K = \frac{1}{3}$. Correspondingly, the values $D = \frac{1}{3}$, $K = 1$, or $D = \frac{1}{7}$, $K = 1$, or $D = \frac{3}{4}$, $K = \frac{1}{5}$ give non-fractal behavior. Figure 3 illustrates phase portraits of the proposed fractal tent map for various values of (D, K) yielding non-fractal (upper row) and fractal (lower row) behavior.

By definition, a fractal function is a function whose graph has, in general, a nonintegral dimension [22,25]. To ensure that the function $\mathcal{F}_f$ is fractal, we estimate the Minkowski–Bouligand (box-counting) dimension of the proposed function's graph by the formula

$$\dim_{box} = \lim_{\varepsilon\to 0} \frac{\log N}{\log(1/\varepsilon)},$$

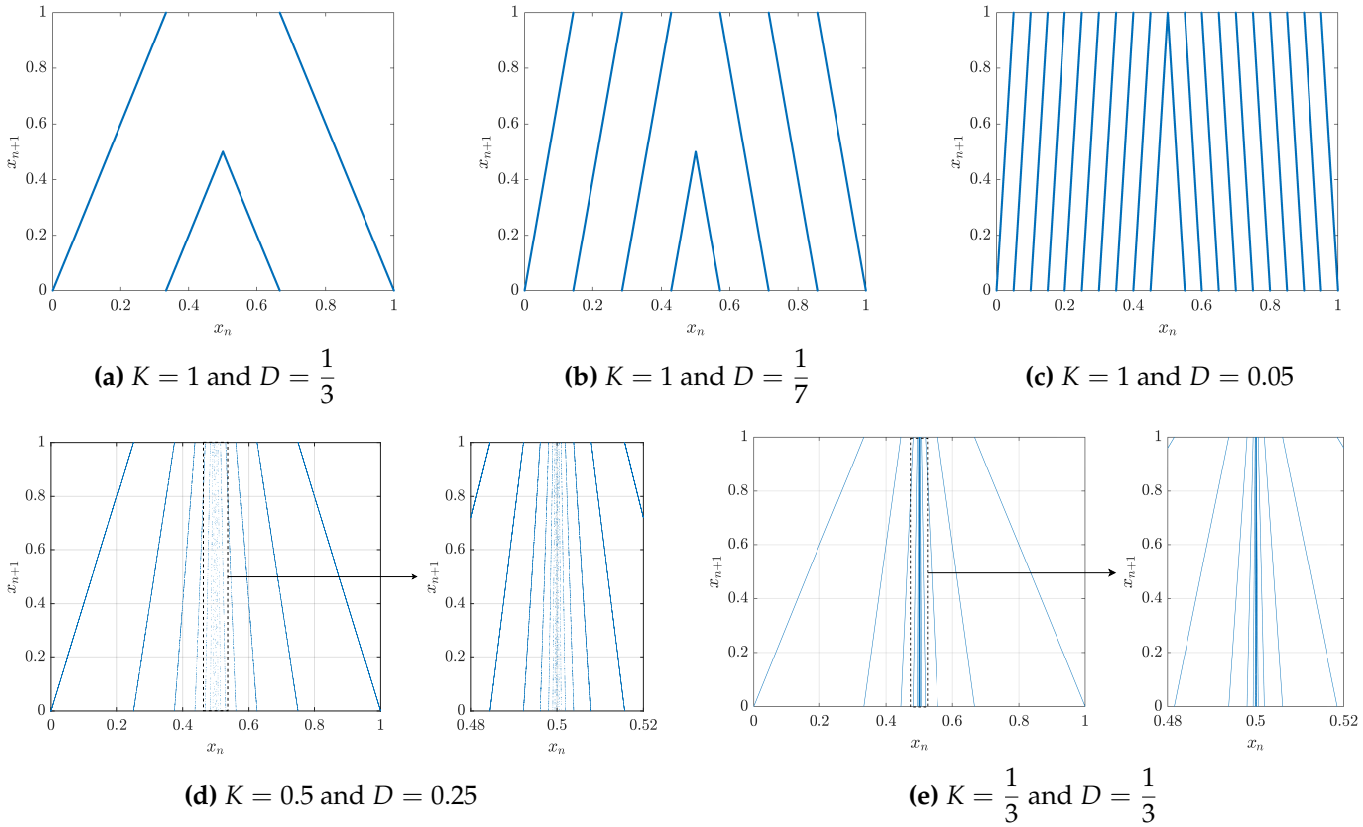
where N is the number of balls with the radius ε required to cover the function plot. For the parameters $D = 1/3$, $K = 1/3$, the numerical estimation gives the value

$$\dim_{box} = 1.24,$$

and for $D = 0.25$, $K = 0.5$, the estimated value is

$$\dim_{box} = 1.29,$$

which are both noninteger, which corresponds to the fractal.



(a) $K = 1$ and $D = \frac{1}{3}$ **(b)** $K = 1$ and $D = \frac{1}{7}$ **(c)** $K = 1$ and $D = 0.05$

(d) $K = 0.5$ and $D = 0.25$ **(e)** $K = \frac{1}{3}$ and $D = \frac{1}{3}$

Figure 3. Phase portraits of the fractal tent map in Equation (3) for various values of K and D. Panels (**a**–**c**) correspond to non-fractal behavior, and panels (**d**,**e**) correspond to fractal behavior. Decreasing D leads to smaller intervals, and decreasing K leads to more rapid interval length decay. Fractal behavior occurs if $1 - K = 2D$, $K \in (0, 1]$.

2.3. Surrogate Data Testing

Although PRNGs are typically used for cryptography purposes, they are also an excellent tool for generating noise, which can be used for simulating various processes. In this paper, we apply the proposed chaotic-map-based PRNG to surrogate data testing as a white noise generator.

The main idea of surrogate testing is to first formulate the null hypothesis H0 that the analyzed time series represents some linear process. Subsequently, surrogate datasets are generated that are consistent with the chosen null hypothesis, and finally, the test statistic for the original time series and the surrogate time series is calculated. The null hypothesis is rejected if the value of the test statistic of the original time series is significantly different from the value of the test statistic of the surrogate data.

Surrogate data testing is a popular method for testing the nonlinearity of time series. The null hypothesis is usually that the data were generated by a Gaussian linear stochastic process with constant coefficients [26].

Generation of surrogate data.The surrogate data are generated by a Gaussian linear stochastic process to have the same specified properties as the observed time series, such as the mean, variance, and Fourier spectrum. This can be achieved by applying a Fourier transform to the original signal and then randomly mixing the phases and applying the inverse Fourier transform. The output is a surrogate, which, in terms of the spectrum, dispersion, and average value, repeats the original one but is linear [27]. This approach is called Fourier-based surrogates and has many variations: the amplitude-adjusted Fourier transform (AAFT) method [28], iteratively refined surrogates (iAAFT) [29], and iterative multivariate surrogates (iAAFTn) [30]. Another way to create surrogate data is to use the autoregressive (AR) model and the moving-average (MA) model. Using the autocorrelation method (Yule–Walker method) [31], an AR model is matched to the input data so that their autocorrelation function coincides. As is known, the autocorrelation function is related to the spectrum by the Wiener–Khinchin theorem [32], which states that the power spectral density of a stationary random process in a broad sense is the Fourier transform of the corresponding autocorrelation function. Thus, surrogates generated using the AR process will also have the same spectrum as the original data.

Finding test statistics. To confirm or refute the null hypothesis about the linearity of the signal, the third moment of the function is calculated, which characterizes the asymmetry of the distribution of function values (x_i). The unbiased estimate of the third moment of the function is defined as

$$\mu = \frac{1}{N-1}\sum_{n=1}^{N}(x_n - x_{n-1})^3,$$

where N is the length of the source data. This test statistic is calculated for all surrogate data and the original signal.

Hypothesis testing. Next, a histogram of the distribution of test statistics for the surrogate data is constructed. Hypothesis H0 is considered confirmed if the value of the test statistic of the original signal is close to the distribution of the test statistic of the surrogate data, and it is rejected if the test statistic of the original signal differs from the values of the test statistics for the surrogate data.

3. Results

This section presents the results of the research concerning both the chaotic and statistical properties of the proposed chaotic map in comparison with the other maps and PRNGs.

3.1. Bifurcation Analysis

To construct a 1D bifurcation diagram, we use a modified version of the density-color bifurcation analysis tool proposed in [33]. Our modification consists of calculating the relative density of points for each parameter value (for each vertical line) separately. Also, histograms are calculated only over the entire range of values of the state variable. This approach allows us to mark the ranges of values of the state variable with greater contrast and more clearly identify orbits that are close to periodic. Bifurcation diagrams for the logistic and tent maps are shown in Figures 4 and 5, respectively.

The diagrams shown in Figures 4 and 5 without density coloring are often seen in the literature. In this study, they can serve as a reference in comparison with the proposed chaotic map.

For the chaotic fractal tent map, several bifurcation diagrams were constructed with the following parameters: for $K = 1$ and $D \in [0;1]$, for $K = 0.5$ and $D \in [0;1]$, and for $K \in [0;1]$ and $D = 0.25$, as given in Figure 6.

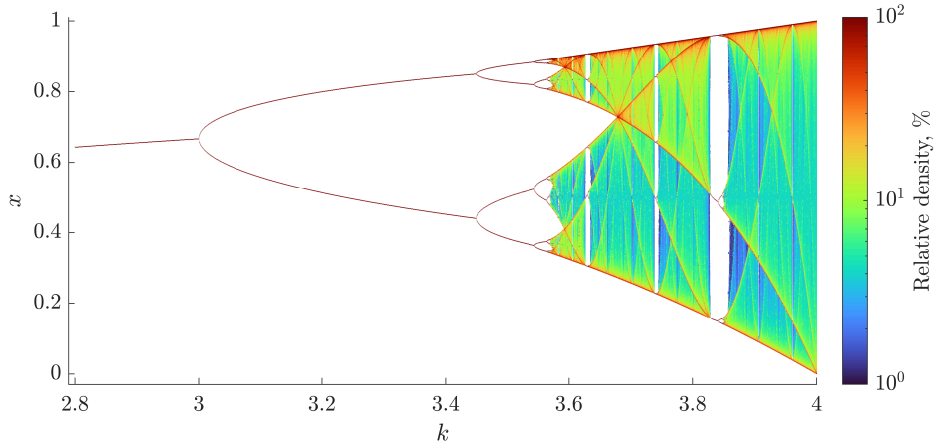


Figure 4. Relative density-based colored one-dimensional bifurcation diagram for the logistic map.

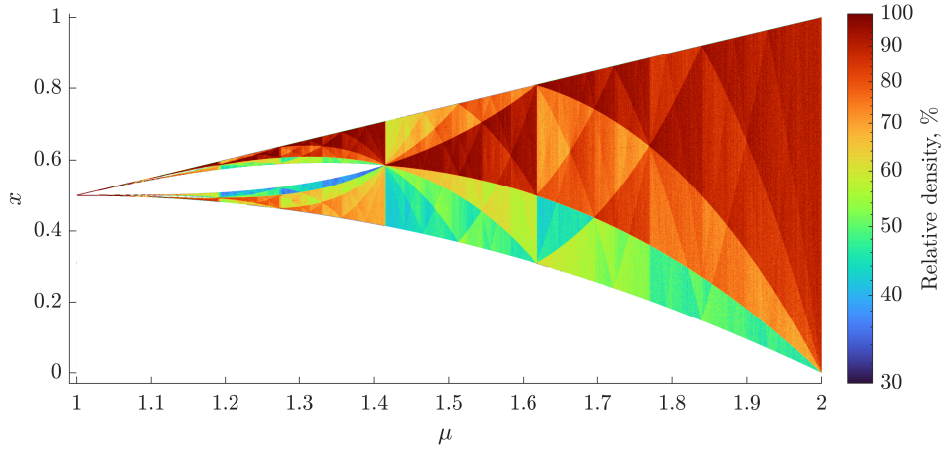
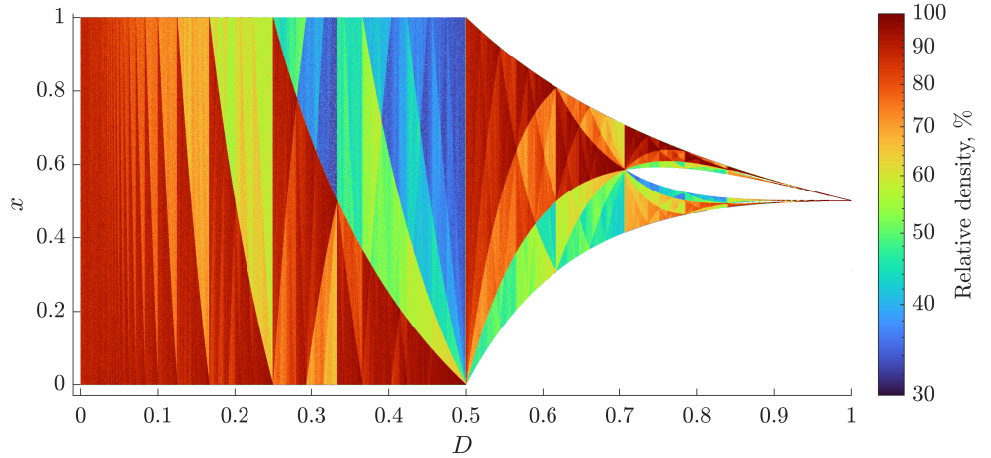
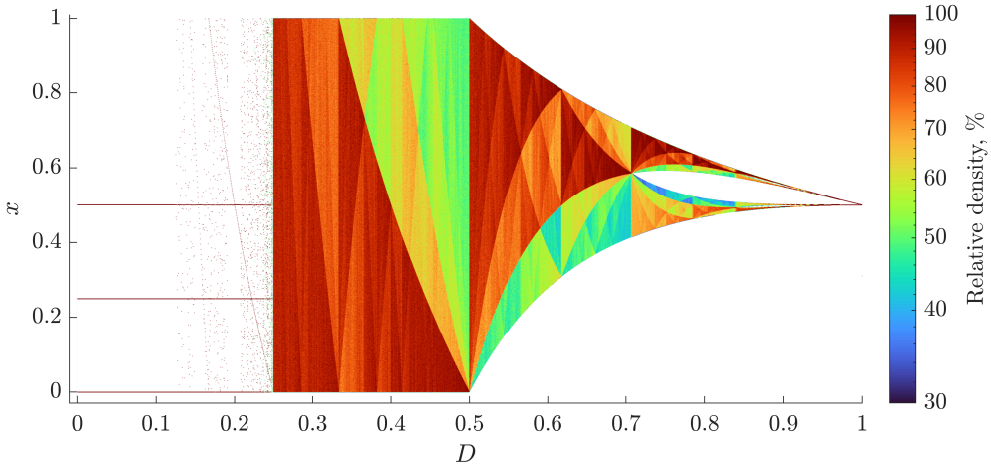


Figure 5. Relative density-based colored one-dimensional bifurcation diagram for the tent map.

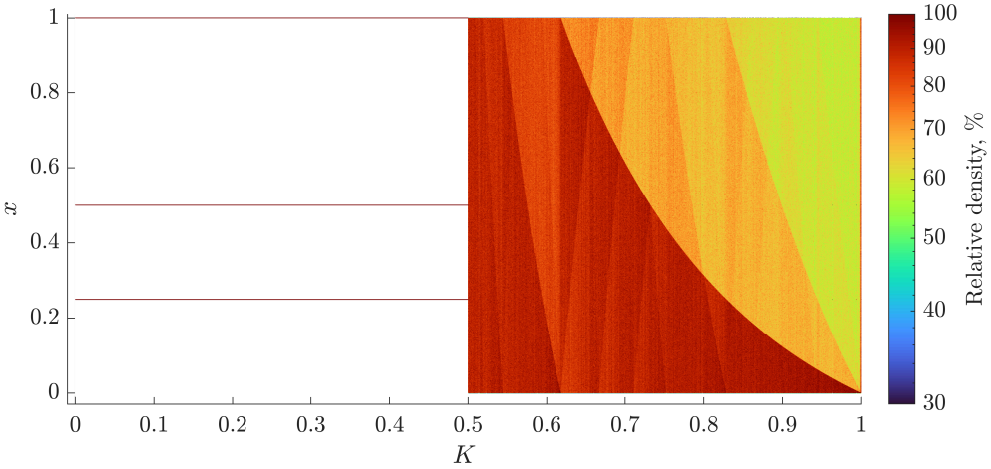
The analysis of these bifurcation diagrams reveals several chaotic properties inherent to the proposed map. From Figure 6a, one can clearly see that, in the case of $K = 1$, for each $D \leq 0.5$, the system's chaotic orbits cover the entire phase space, but it is only at the specific values $D = 1/2^i$, $i \in \mathbb{N}$, that the density of the orbit is uniform in the entire range $[0, 1]$ and reaches its maximum values. From Figure 6b, it is clear that the fractal behavior of the function $\mathcal{F}_f(x)$ occurring when $D = 0.25$ yields the greatest density possible, and for $D < 0.25$, the dynamics degrade. For values slightly greater than $D = 0.25$, the density remains high. In Figure 6c, we observe the same picture concerning the fractal behavior of the function $\mathcal{F}_f(x)$ near $K = 0.5$, which also yields the highest possible, and for $K < 0.5$, the dynamics also degrade. For values slightly greater than $K = 0.5$, the density remains high, as in the case of exploring the parameter D, which points out that a small perturbation of the system parameters toward increased values (e.g., related to rounding error) would not affect the dynamics, but a small perturbation of their values downward would prevent the system from exhibiting chaotic behavior, so this should be taken into account when implementing the chaotic fractal tent map in practical applications since the fractal behavior corresponds to the edge of chaotic dynamics.



(**a**) $K = 1$, $D \in [0; 1]$



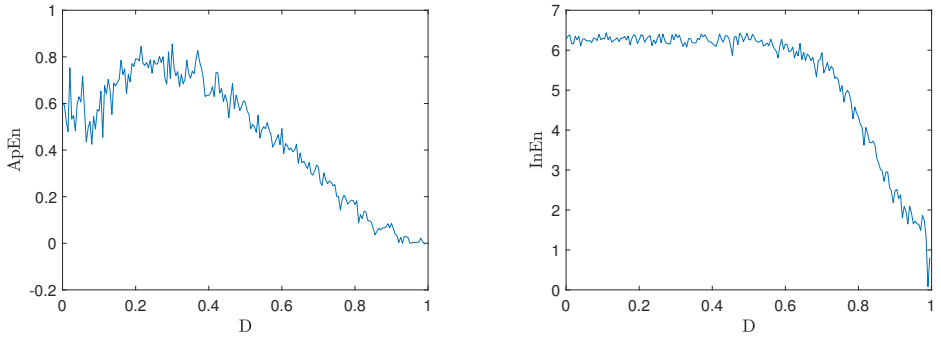
(**b**) $K = 0.5$, $D \in [0; 1]$



(**c**) $K \in [0; 1]$, $D = 0.25$

Figure 6. Relative density-based colored one-dimensional bifurcation diagram for the generalized tent map. Points $K = 1 - 2D$, where the condition of the fractal function is satisfied, relate to higher orbit densities. Nevertheless, non-fractal parameter combinations can also produce similar densities at some points.

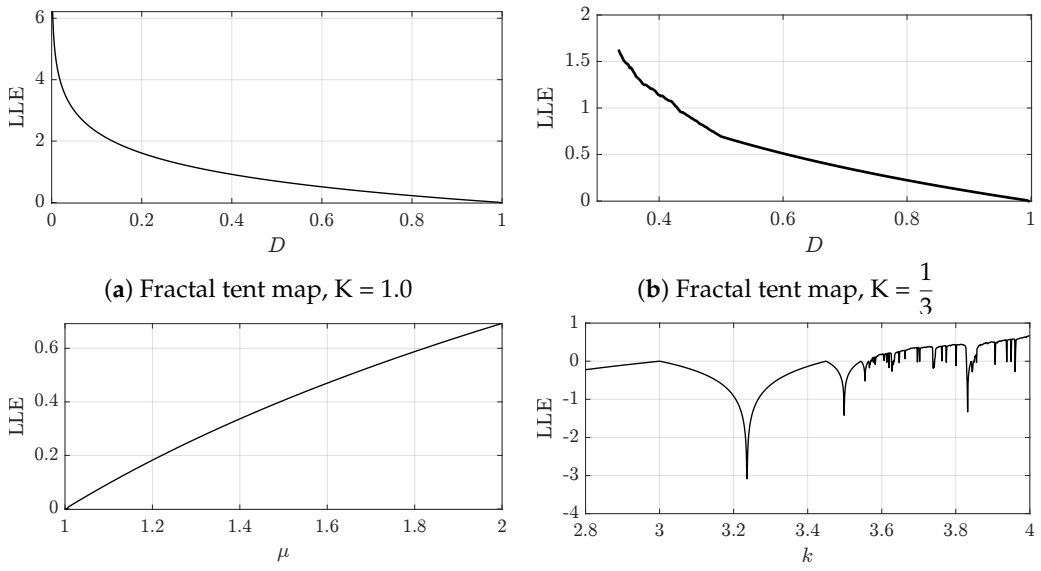
Figure 7 shows graphs of approximate and information entropy for the fractal tent map with the parameters $K = 1, D \in [0, 1]$ (Figure 6a).



(**a**) Approximate entropy (**b**) Information entropy

Figure 7. Approximate and information entropy for fractal tent map with parameter K = 1.0.

Figure 8 gives the values of the largest Lyapunov exponents (LLEs) of the considered chaotic maps. One can see an increase in the LLE in Figure 8a, corresponding to the fractal tent map when D tends to zero, which relates to greater mixing properties due to the growing number of linear fragments in the function $\mathcal{F}_f(x)$. Figure 8b, corresponding to the fractal tent map in the fractal regime, shows the same behavior. The two other panels illustrate LLEs depending on parameters for the logistic and tent maps. Notably, in the generalized tent map, high values of LLEs are much easier to obtain.



(**a**) Fractal tent map, K = 1.0 (**b**) Fractal tent map, K = $\frac{1}{3}$

(**c**) Tent map (**d**) Logistic map

Figure 8. Dependence of largest Lyapunov exponents on parameters of investigated chaotic maps.

To create a PRNG based on a chaotic map, we used the Box–Muller transformation, which allows one to transform a sample of random numbers from uniformly distributed to normally distributed. Thus, for two independent random samples x_1 and x_2 generated by a chaotic map, the Box–Muller transform will be

$$X = sin(2\pi x_1)\sqrt{-2ln(x_2)}$$

The selected parameters for chaotic maps are as follows: $k = 4$ for the logistic map; $\mu = 2$ for the tent map; $K = 1/3$ and $D = 1/3$ for the fractal tent map.

3.2. NIST Tests

To transform a generated value from the floating-point representation into binary, we used the method proposed in [34]. First, the generated value is incremented by 10^9, and

then it is rounded toward negative infinity; only the remainder from dividing this number by 2 is considered. The combination of values obtained in this way gives us a sequence suitable for testing with the NIST suite. NIST tests were performed for all studied PRNGs. Each of the PRNGs generated a sequence of 10^6 bits and was run 100 times. The significance level $\alpha = 0.01$. The results obtained are presented in Table 1. The null hypothesis is that the generated sequence is random. If the value of $P_{value} \geq \alpha$, the null hypothesis is accepted (i.e., the sequence is random). According to the NIST test package recommendations, the minimum pass rate for each statistical test, except for the random excursion (REVT) test, is 0.96. The minimum pass rate for the REVT test is 0.94. One can see that all the generators under study produce sequences with random characteristics for all cases considered.

Table 1. Results of NIST testing for PRNG based on different maps.

Statistic Tests	randn() Function		Logistic Map		Fractal Tent Map		Tent Map	
	P_{value}	*Pass Rate*	P_{value}	*Pass Rate*	P_{value}	*Pass Rate*	P_{value}	*Pass Rate*
FT	0.457	0.98	0.478	1	0.480	0.99	0.543	0.98
BFT	0.525	0.99	0.466	0.99	0.470	0.99	0.492	0.99
RT	0.513	1	0.515	1	0.543	1	0.527	0.99
LROT	0.502	1	0.505	0.99	0.568	0.98	0.487	1
MRT	0.498	0.98	0.517	0.98	0.464	0.99	0.482	1
SPT	0.462	1	0.504	0.99	0.542	0.99	0.521	0.99
NTMT	0.461	0.99	0.502	1	0.535	0.99	0.519	0.99
OTMT	0.393	0.97	0.529	0.97	0.512	1	0.474	1
MUST	0.474	1	0.473	1	0.499	1	0.490	1
LCT	0.459	0.99	0.546	0.98	0.536	1	0.497	0.99
ST0	0.475	0.98	0.541	1	0.514	1	0.520	1
ST1	0.465	1	0.535	0.99	0.475	1	0.553	1
AET	0.494	0.98	0.468	0.99	0.542	1	0.493	0.99
CSTF	0.457	0.98	0.444	1	0.498	0.99	0.524	0.98
CSTR	0.456	0.99	0.440	0.99	0.486	0.99	0.539	0.98
RET	0.505	0.99	0.501	0.98	0.537	0.99	0.454	0.98
REVT	0.470	0.99	0.543	0.98	0.504	0.99	0.488	0.95

A key-space analysis of the proposed schemes is performed in order to check the generation algorithms' security against attacks of a brute-force type. The condition that one is required to satisfy is an algorithm having more than 2^{100} different key values, which can be used to form streams of random data, that are different from each other [35,36]. By taking into account the usage of the double-precision floating-point data type, with mantissa being represented by 53 bits, one can achieve a rough estimation of viable key spaces for proposed schemes (Table 2):

Table 2. Key-space estimation for every proposed algorithm.

	Fractal Tent Map	Logistic Map	Tent Map
Key Space	2^{104}	2^{61}	2^{104}

Such key-space sizes were acquired by analyzing the different parameter values that can be used in order to generate data sequences. In the case of a logistic map, one might use various starting x_0 values in the range of [0; 1] and the k parameter in the range of roughly (3.55:4] as an initial parameter for outcome control; for the double-precision floating-point data type, this results in a potential key-space size of $2^{64} \times 2^9 = 2^{61}$. In the case of a tent map, one obtains a new parameter μ, which can potentially be chosen from anywhere in the range of (1:2), paired with an already-existing x_0, resulting in $2^{52} \times 2^{52} = 2^{104}$ potential key variants. As for the fractal tent map, one obtains the same result as for the tent map due to the parameter being available to be chosen from the range of (0:1) and parameter D being dependent on it, thus resulting in a key space of $2^{52} \times 2^{52} = 2^{104}$. The acquired

results for both the tent map and its fractal modification prove that they are both resistant to brute-force attacks, while the logistic map does not allow one to provide a sufficient number of various keys while using a representation of the double floating-point type.

In order to evaluate the PRNGs' bias resistance, one might use a Chi-square formula. Due to algorithms generating data in a binary format, one can calculate the Chi-square for datasets with 1 degree of freedom. In our case, we suggest the null hypothesis that there is no visible correlation between two datasets acquired for different generation parameters [37]. If the Chi-square C^2 value is less than the target value of 3.841, then one can state that the null hypothesis is correct and the data are unbiased (Table 3).

Table 3. Chi-squarecalculation for datasets acquired from PRNG based on different parameter values.

	Fractal Tent Map			**Logistic Map**			**Tent**		
	D = 1/2	**D = 1/3**	**Sum (Row)**	**k = 4**	**k = 3.8**	**Sum (Row)**	**u = 2×10^{-10}**	**u = 1.5**	**Sum (Row)**
Ones	500,765	1,000,509	1,501,274	500,316	999,435	1,499,751	500,141	1,000,064	1,500,205
Zeroes	499,235	999,491	1,498,726	499,684	1,000,565	1,500,249	499,859	999,936	1,499,795
Sum (Col)	1,000,000	2,000,000	3,000,000	1,000,000	2,000,000	3,000,000	1,000,000	2,000,000	3,000,000
Chi-Square Value	$2.3244476883238 \times 10^{-10}$			$5.320000146597924 \times 10^{-10}$			$9.688889069855805 \times 10^{-11}$		

According to the obtained results, one can state that the bias resistance of the proposed PRNGs might be considered high due to the Chi-square metric being significantly lower than that of a required value.

Figure 9 shows a comparison of Gaussian normal distributions and PRNG distributions of a logistic map, a tent map, and a fractal tent map. It can be seen that the studied PRNGs have normal or nearly normal distributions.

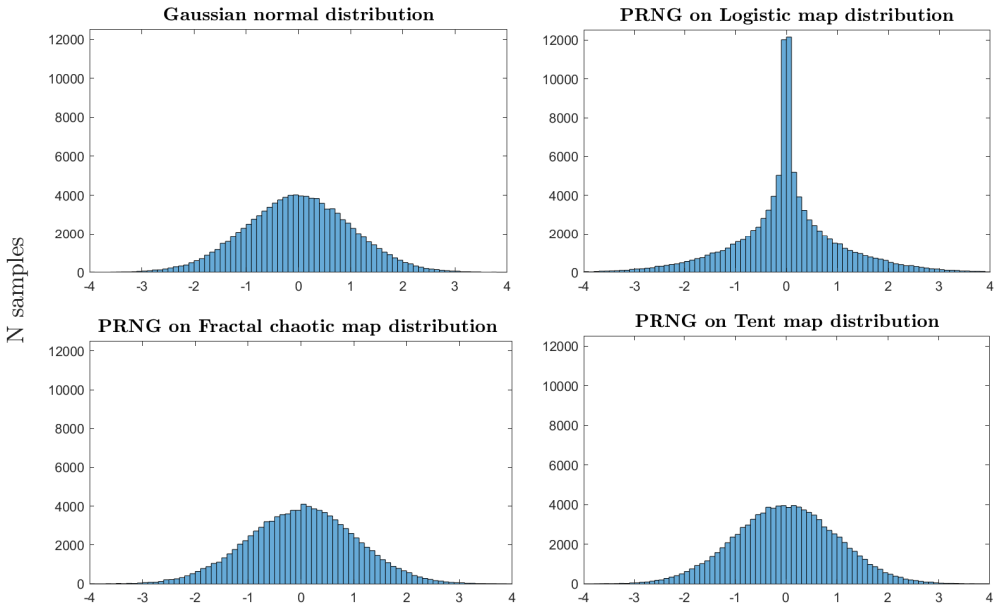
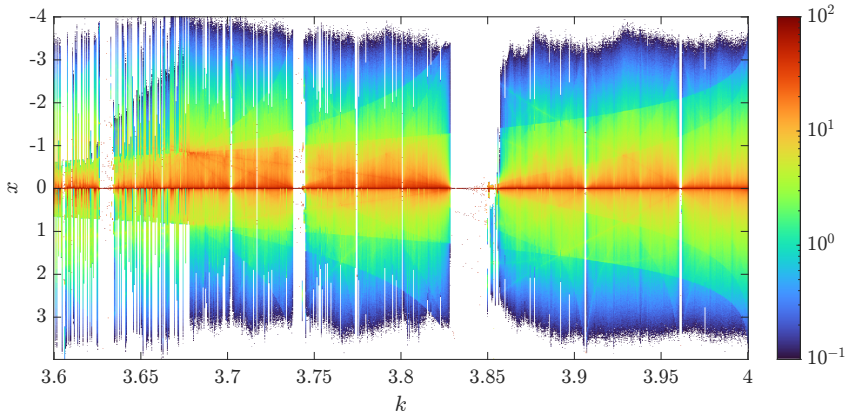
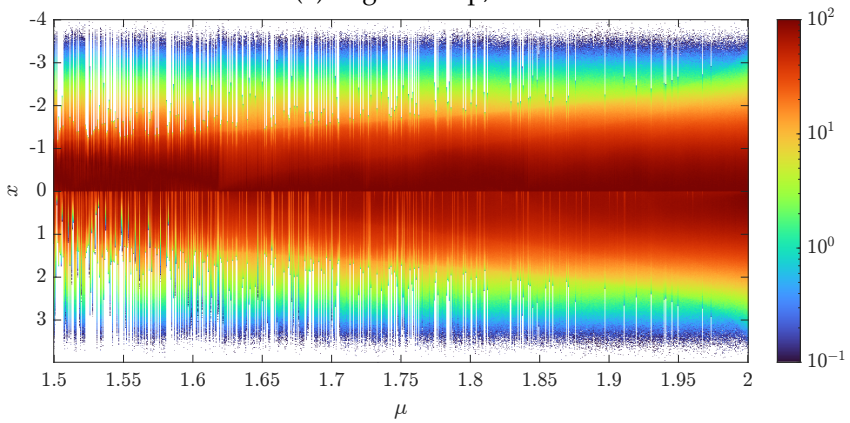


Figure 9. Comparison of distributions of sequences generated by PRNG based on logistic map, fractal chaotic map, and tent map with Gaussian normal distribution.

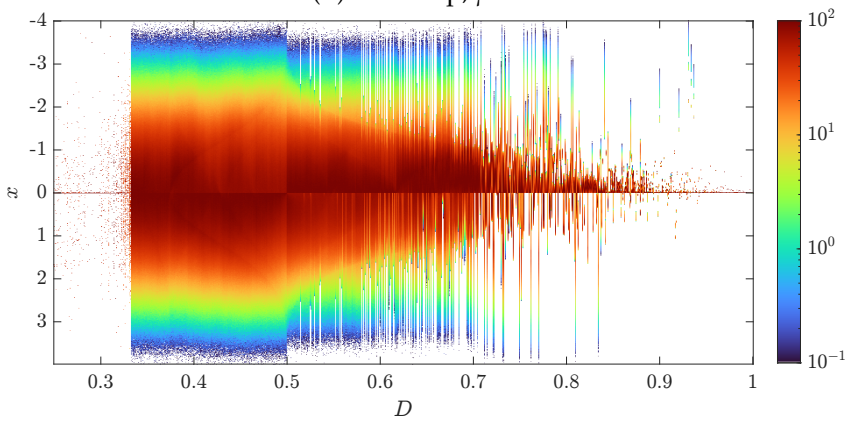
To evaluate the dependence between the distribution of PRNGs and different parameters of the underlying chaotic maps, we introduced the parametric histogram plot (Figure 10). In this plot, each parameter value of the chaotic map is represented by a colored vertical line. The color intensity corresponds to the histogram value. It should be noted that, for enhanced clarity, the histogram values are normalized for each parameter value in Figure 10).



(**a**) Logistic map, $k = 4$



(**b**) Tent map, $\mu = 2$



(**c**) Fractal tent map, $K = 1/3, D \in [0.25, 1]$

Figure 10. Relative density-based two-dimensional histograms showing the dependence of the PRNGs' distributions on map parameters.

Figure 10 illustrates that the PRNG distributions for the tent map and the generalized tent map are similar, closely resembling the normal distribution across a broad range of parameter values. However, there are noticeable differences in the distribution width. Conversely, the logistic map exhibits a distinct distribution pattern, with near-zero values occurring more frequently. These findings suggest that the characteristics of the PRNG are significantly influenced by the specific values of the chaotic map parameters, enabling precise adjustments to the PRNG's behavior.

3.3. Surrogate Testing Results

Let us recall that the null hypothesis was that the data being tested were generated by a linear stochastic process with a normal Gaussian distribution. Surrogate data were generated using an autoregressive process of the order p (AR(p) process), defined as follows:

$$X_t = \sum_{i=1}^{p} a_i X_{t-i} + \varepsilon_t,$$

where ε_t is a normally distributed random value (usually white noise), and a_i denotes the parameters of the autoregressive model. The order p of the AR process is determined using the minimum description length (MDL).

To conduct surrogate testing, we use three types of data:

1. A linear signal that is generated by the AR process.
2. A nonlinear chaotic signal generated by the logistic map (for $k = 4$).
3. A biological signal that is taken from a dataset of 81 recordings of sperm whale clicks. It is generally a linear signal, but quite noisy.

The results of surrogate testing are shown in Figures 11–13. The histogram shows the distribution of the test statistic for the surrogate data, and the vertical line shows the value of the test statistic for the signal under study. For example, for the first and third types of signals in Figures 11 and 13, we are examining a linear signal, so the test statistic of the signal under study is expected to fit into the distribution of the test statistic of the surrogate data. As we can see in Figures 11–13, in the exemplified runs of the surrogate testing algorithms, all tests give similar, correct results, but during different runs, the random number generator can spontaneously produce an incorrect distribution, which leads to a spurious test result.

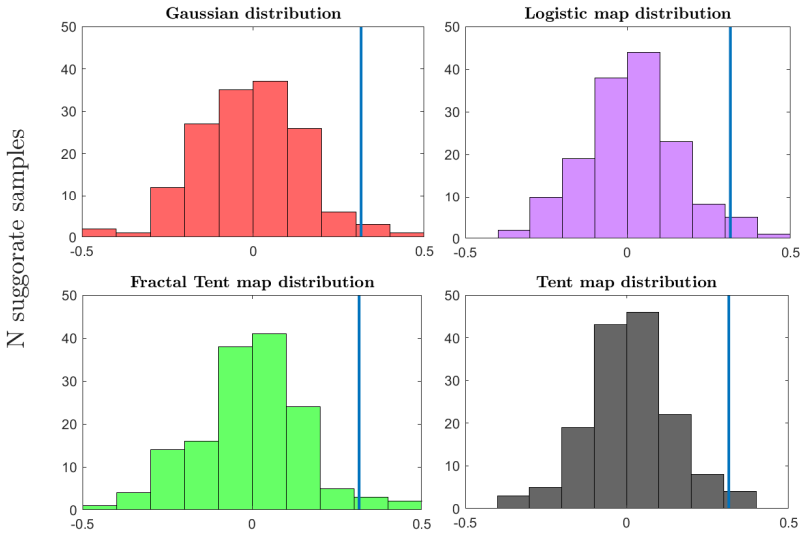


Figure 11. A histogram of distributionsof test statistics for a linear signal.

For each test, 150 surrogate datasets of length $N = 1000$ were generated, and each test for four PRNGs was conducted 100 times. Table 4 shows the results of checking the accuracy of determining the signal nonlinearity in terms of the true positive rate. For linear signals, the test statistics of the original signal should fall within the distribution boundary of the test statistics of the surrogate data, and vice versa for a nonlinear signal. As we can see, the best result was obtained when using both the fractal ($D = 1/3$, $K = 1/3$) and common variants of the tent map. Surprisingly, the worst result is achieved when using a standard MATLAB `randn()` function.

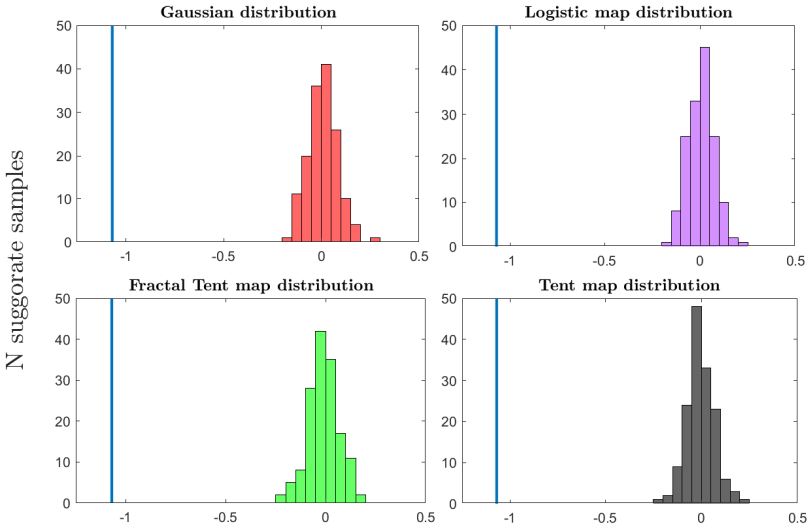


Figure 12. A histogram of distributions of test statistics for a nonlinear chaotic signal generated by the logistic map.

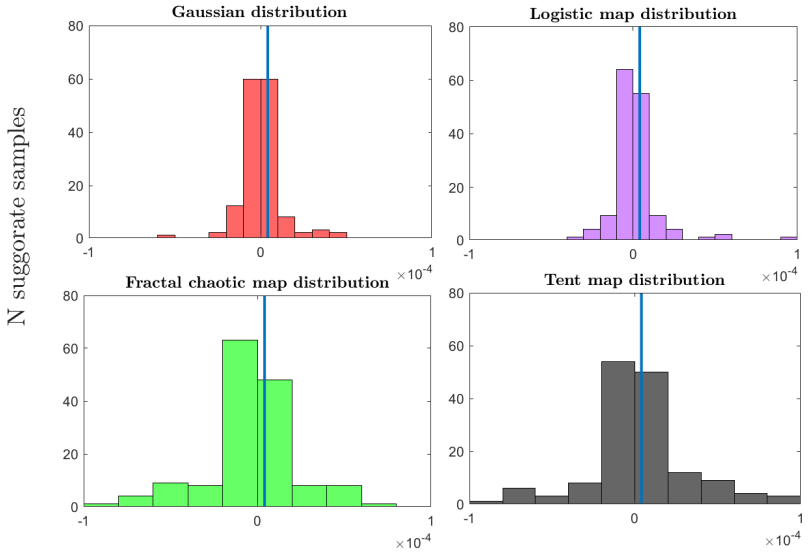


Figure 13. A histogram of distributions of test statistics for a biological signal.

Table 4. Accuracy of signal type determination based on various data.

Type of Signal	randn() Function	Logistic Map	Fractal Tent Map	Tent Map
Linear signal (AR)	0.99	1	1	1
Nonlinear signal (logistic map)	1	1	1	1
Biological signal (sperm whale)	0.93	0.96	1	1

4. Discussion and Conclusions

Designing new sophisticated chaotic maps is a common practice for avoiding problems associated with chaos degeneration and other issues that affect chaos-based pseudorandom number generators. In the current research, a hypothesis was confirmed that using a fractal function in a chaotic map can provide a more complex and less predictable behavior of the chaotic map. This results in better randomness of the pseudorandom sequence generated with the use of this map. We verified this statement using bifurcation diagrams and pseudorandom number distribution diagrams to evaluate the chaotic behavior of the proposed map, the uniformity of chaotic orbits, and the proximity of the generated pseudorandom

number distribution to the normal distribution after applying the Box–Muller transform. The results of NIST tests conducted for the proposed PRNG also show high randomness. The proposed generator occasionally outperforms the MATLAB built-in `randn()` function, as well as PRNGs based on the logistic map and the conventional tent map. The PRNG based on the proposed chaotic map can be used in stream encryption, chaos-based communications, and other real-world applications that require pseudorandom sequences with a high level of randomness. Using a chaotic-map-based PRNG for surrogate testing shows room for improving the accuracy of rejecting or confirming a nonlinearity hypothesis compared to a standard MATLAB function for generating normally distributed random values. Table 4 shows that the standard MATLAB function `randn()` makes more errors in determining a linear signal than the proposed PRNG. Errors have also been observed for the PRNG based on the logistic map when determining the linearity of a biological signal with noise.

The obtained results can inspire further studies of fractal chaotic maps and flows from both theoretical and applied points of view.

Author Contributions: Conceptualization, D.B. and A.K.; data curation, P.F. and A.K.; formal analysis, V.R.; funding acquisition, E.K.; investigation, V.R. and O.V.; methodology, D.B. and A.K.; project administration, E.K.; resources, V.R. and P.F.; software, O.V. and P.F.; supervision, D.B.; validation, E.K. and V.R.; visualization, V.R., O.V., and P.F.; writing—original draft, E.K. and O.V.; writing—review and editing, D.B. and A.K. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by the Russian Science Foundation (RSF), project 23-71-01084.

Data Availability Statement: Data are available from the authors on reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Biswas, H.R.; Hasan, M.M.; Bala, S.K. Chaos theory and its applications in our real life. *Barishal Univ. J. Part* **2018**, *1*, 123–140.
2. Alnajim, A.M.; Abou-Bakr, E.; Alruwisan, S.S.; Khan, S.; Elmanfaloty, R.A. Hybrid chaotic-based PRNG for secure cryptography applications. *Appl. Sci.* **2023**, *13*, 7768. [CrossRef]
3. Cao, W.; Cai, H.; Hua, Z. n-Dimensional Chaotic Map with application in secure communication. *Chaos Solitons Fractals* **2022**, *163*, 112519. [CrossRef]
4. Naik, R.B.; Singh, U. A review on applications of chaotic maps in pseudo-random number generators and encryption. *Ann. Data Sci.* **2024**, *11*, 25–50. [CrossRef]
5. Haliuk, S.; Krulikovskyi, O.; Vovchuk, D.; Corinto, F. Memristive structure-based chaotic system for prng. *Symmetry* **2022**, *14*, 68. [CrossRef]
6. Yu, F.; Li, L.; He, B.; Liu, L.; Qian, S.; Zhang, Z.; Shen, H.; Cai, S.; Li, Y. Pseudorandom number generator based on a 5D hyperchaotic four-wing memristive system and its FPGA implementation. *Eur. Phys. J. Spec. Top.* **2021**, *230*, 1763–1772. [CrossRef]
7. Fellet, M. Pseudorandom number generator uses true chaos to generate sequences with random behavior. *Scilight* **2018**, *2018*, 440001. [CrossRef]
8. Rezk, A.A.; Madian, A.H.; Radwan, A.G.; Soliman, A.M. Reconfigurable chaotic pseudo random number generator based on FPGA. *AEU-Int. J. Electron. Commun.* **2019**, *98*, 174–180. [CrossRef]
9. Elmanfaloty, R.A.; Abou-Bakr, E. Random property enhancement of a 1D chaotic PRNG with finite precision implementation. *Chaos Solitons Fractals* **2019**, *118*, 134–144. [CrossRef]
10. Kadir, A.; Hamdulla, A.; Guo, W.Q. Color image encryption using skew tent map and hyper chaotic system of 6th-order CNN. *Optik* **2014**, *125*, 1671–1675. [CrossRef]
11. Hasler, M.; Maistrenko, Y.L. An introduction to the synchronization of chaotic systems: Coupled skew tent maps. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **1997**, *44*, 856–866. [CrossRef]
12. Tan, Z.; Wu, Q. Study of linearly cross-coupled chaotic systems for a random bit generator. In Proceedings of the 2008 International Conference on Computational Intelligence and Security, Suzhou, China, 13–17 December 2008; IEEE: New York, NY, USA, 2008; Volume 2, pp. 267–272.
13. Saito, A.; Yamaguchi, A. Pseudorandom number generator based on the Bernoulli map on cubic algebraic integers. *Chaos Interdiscip. Nonlinear Sci.* **2018**, *28*, 103122. [CrossRef] [PubMed]
14. Moysis, L.; Tutueva, A.; Christos, K.; Butusov, D. A chaos based pseudo-random bit generator using multiple digits comparison. *Chaos Theory Appl.* **2020**, *2*, 58–68.

15. Moysis, L.; Volos, C.; Jafari, S.; Munoz-Pacheco, J.M.; Kengne, J.; Rajagopal, K.; Stouboulos, I. Modification of the logistic map using fuzzy numbers with application to pseudorandom number generation and image encryption. *Entropy* **2020**, *22*, 474. [CrossRef] [PubMed]
16. Kvitko, D.; Rybin, V.; Bayazitov, O.; Karimov, A.; Karimov, T.; Butusov, D. Chaotic Path-Planning Algorithm Based on Courbage–Nekorkin Artificial Neuron Model. *Mathematics* **2024**, *12*, 892. [CrossRef]
17. Murillo-Escobar, D.; Murillo-Escobar, M.Á.; Cruz-Hernández, C.; Arellano-Delgado, A.; López-Gutiérrez, R.M. Pseudorandom number generator based on novel 2D Hénon-Sine hyperchaotic map with microcontroller implementation. *Nonlinear Dyn.* **2023**, *111*, 6773–6789. [CrossRef] [PubMed]
18. Garcia-Bosque, M.; Pérez-Resa, A.; Sánchez-Azqueta, C.; Aldea, C.; Celma, S. Chaos-based bitwise dynamical pseudorandom number generator on FPGA. *IEEE Trans. Instrum. Meas.* **2018**, *68*, 291–293. [CrossRef]
19. Tutueva, A.V.; Karimov, T.I.; Moysis, L.; Nepomuceno, E.G.; Volos, C.; Butusov, D.N. Improving chaos-based pseudo-random generators in finite-precision arithmetic. *Nonlinear Dyn.* **2021**, *104*, 727–737. [CrossRef]
20. Zhao, W.; Chang, Z.; Ma, C.; Shen, Z. A pseudorandom number generator based on the chaotic map and quantum random walks. *Entropy* **2023**, *25*, 166. [CrossRef]
21. Hocine, M.; Lahcene, M.; Larbi, T.; Adda, A.P. A PRNG based on an improved chaotic map using a self-perturbation mechanism. *Rom. J. Inf. Technol. Autom. Control* **2023**, *33*, 47–60. [CrossRef]
22. Massopust, P.R. Fractal functions and their applications. *Chaos, Solitons Fractals* **1997**, *8*, 171–190. [CrossRef]
23. Lorenz, E.N. The problem of deducing the climate from the governing equations. *Tellus* **1964**, *16*, 1–11. [CrossRef]
24. Collet, P.; Eckmann, J.P. *Iterated Maps on the Interval as Dynamical Systems*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2009.
25. Wu, J.; Jin, X.; Mi, S.; Tang, J. An effective method to compute the box-counting dimension based on the mathematical definition and intervals. *Results Eng.* **2020**, *6*, 100106. [CrossRef]
26. Theiler, J.; Eubank, S.; Longtin, A.; Galdrikian, B.; Farmer, J.D. Testing for nonlinearity in time series: The method of surrogate data. *Phys. D Nonlinear Phenom.* **1992**, *58*, 77–94. [CrossRef]
27. Schreiber, T.; Schmitz, A. Surrogate time series. *Phys. D Nonlinear Phenom.* **2000**, *142*, 346–382. [CrossRef]
28. Theiler, J.; Prichard, D. Using 'surrogate surrogate data'to calibrate the actual rate of false positives in tests for nonlinearity in time series. *Fields Inst. Commun.* **1997**, *11*, 99.
29. Schreiber, T.; Schmitz, A. Improved surrogate data for nonlinearity tests. *Phys. Rev. Lett.* **1996**, *77*, 635. [CrossRef] [PubMed]
30. Prichard, D.; Theiler, J. Generating surrogate data for time series with several simultaneously measured variables. *Phys. Rev. Lett.* **1994**, *73*, 951. [CrossRef] [PubMed]
31. Stoica, P.; Friedlander, B.; Söderström, T. A high-order Yule-Walker method for estimation of the AR parameters of an ARMA model. *Syst. Control Lett.* **1988**, *11*, 99–105. [CrossRef]
32. Cohen, L. The generalization of the Wiener-Khinchin theorem. In Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181), Seattle, WA, USA, 12–15 May 1998; IEEE: New York, NY, USA, 1998; Volume 3, pp. 1577–1580.
33. Moysis, L.; Lawnik, M.; Volos, C. Density-Colored Bifurcation Diagrams—A Complementary Tool for Chaotic Map Analysis. *Int. J. Bifurc. Chaos* **2023**, *33*, 2330036. [CrossRef]
34. Tutueva, A.V.; Nepomuceno, E.G.; Karimov, A.I.; Andreev, V.S.; Butusov, D.N. Adaptive chaotic maps and their application to pseudo-random numbers generation. *Chaos Solitons Fractals* **2020**, *133*, 109615. [CrossRef]
35. Stinson, D.R. Classical Cryptography. In *Cryptography Theory and Practice*; CRC-Press: Boca Raton, FL, USA, 1995; pp. 1–20.
36. Alvarez, G.; Li, S. Some basic cryptographic requirements for chaos-based cryptosystems. *Int. J. Bifurc. Chaos* **2006**, *16*, 2129–2151. [CrossRef]
37. Pearson, K.X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonbly supposed to have arisen from random sampling. *Lond. Edinb. Dublin Philos. Mag. J. Sci.* **1900**, *50*, 157–175. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.