electronics

MDPI

Article

Pseudorandom Number Generator (PRNG) Design Using Hyper-Chaotic Modified Robust Logistic Map (HC-MRLM)

Muhammad Irfan [1], Asim Ali [2], Muhammad Asif Khan [1,*], Muhammad Ehatisham-ul-Haq [1], Syed Nasir Mehmood Shah [3], Abdul Saboor [4,*] and Waqar Ahmad [1]

1 Department of Computer Engineering, University of Engineering and Technology, Taxila 47050, Pakistan; irfanbabar42@gmail.com (M.I.); ehatishamuet@gmail.com (M.E.-u.-H.); waqar.ahmad@uettaxila.edu.pk (W.A.)
2 Department of Computer Science, University of Wah, Wah Cantt 47040, Pakistan; asim.ali@uow.edu.pk
3 Department of Computer Sciences KICSIT, Institute of Space Technology, P.O. Box 2750, Islamabad 44000, Pakistan; nasirsyed.utp@gmail.com
4 Department of Research and Technology Simbal Tech, Sommerburg str. 157, 45149 Essen, Germany
* Correspondence: masif.khan@uettaxila.edu.pk (M.A.K.); abdulsaboor11@outlook.com (A.S.)

Received: 1 November 2019; Accepted: 1 January 2020; Published: 6 January 2020

Abstract: Robust chaotic systems, due to their inherent properties of mixing, ergodicity, and larger chaotic parameter space, constitute a perfect candidate for cryptography. This paper reports a novel method to generate random numbers using modified robust logistic map (MRLM). The non-smooth probability distribution function of robust logistic map (RLM) trajectories gives an un-even binary distribution in randomness test. To overcome this disadvantage in RLM, control of chaos (CoC) is proposed for smooth probability distribution function of RLM. For testing the proposed design, cryptographic random numbers generated by MRLM were vetted with National Institute of Standards and Technology statistical test suite (NIST 800-22). The results showed that proposed MRLM generates cryptographically secure random numbers (CSPRNG).

Keywords: chaotic logistic map; robust chaos; key scheduling; PRNG; lyapunov exponent

1. Introduction

Random number sequences have immense impact on numerous applications, such as signal processing [1,2], stochastic simulations [3,4], spread spectrums [5,6], gaming [7–9], statistics [10], captcha [11,12], machine learning [13,14], and cryptography etc. The random number generators that pertain to excellent statistical properties are considered critical for robust cryptographic applications [15]. The random numbers are categorized into two types: (1) true random number generator (TRNG), and (2) pseudorandom number generator (PRNG). TRNGs are generally based on physical and natural phenomenon and are non-deterministic, such as quantum random process, photon noise, frequency jitter in the oscillator, thermal noise, human brain signals, and free-running oscillator [16–18]. The generated sequences can be passed through the sampling and post-processing techniques to enhance the randomness. TRNG properties reveal that the truly generated random sequences must be non-reproducible, unpredictable, and statistically unbiased [19]. On the other hand, PRNGs are based on mathematical functions stem from initial condition to generate deterministic sequences over a long period. These PRNGs possess good statistical properties, such as repeatability, reproducibility, and fast execution time. A special type of PRNG, desirable for cryptography, is cryptographically secure PRNG (CSPRNG). A CSPRNG is unpredictable and computationally infeasible to generate the prior data bits.

CSPRNGs are used in cryptographic mechanisms to provide cryptographic services of encryption, digital signature, hashing algorithms, and key generation.

Shannon, in his famous secrecy theory [20], stated that a source to design primitive cryptography must have the properties like sensitive dependence on initial condition, mixing and ergodicity to change, iteratively, position, and value of plaintext known as confusion and diffusion, respectively. Over the last decade, the use of the chaos in cryptography has gained increased attention due to the properties of mixing, ergodicity, sensitive dependence on initial condition, and deterministic dynamics. Chaotic maps are simple mathematical functions that exhibit chaotic behavior. Chaotic maps are iterated using an initial seed that gives nonlinear trajectories, which are mapped to binary sequences in order to generate highly unpredictable random numbers that are employed in cryptography [21–26].

In recent years, researchers have implemented PRNGs based on chaotic maps in both hardware and software form. There are several drawbacks when these maps are used in cryptographic systems such as the discontinuity of ranges, non-uniform distribution, periodicity in chaotic range, and a small key space [27]. Ahmed et al. [28] proposed a reconfigurable hardware-based chaotic PRNG. In their proposed system, Lorenz [29] and Lü [30] used chaotic systems, which generated four three-dimensional chaotic attractors with the distribution of one and three attractors respectively. Hereby, multiplexing and shifting schemes are used for reconfiguration in real-time. The proposed method is synthesized on a field-programmable array (FPGA) and tested successfully using the National Institute of Standard and Technology (NIST) 800-22 test suite. Only 1.4% of the FPGA's slices were utilized with the operating frequency of 78 MHz Random robustness strongly depends upon the data type, fixed-point, and floating-point notation. Floating-point notation has a lower data rate and inefficient resource utilization. Rania and Ehab [31] proposed a hardware-based PRNG using single skew tent map [32], coupled skew tent map [33], and cross-coupled skew tent map [34]. They examined the fixed-point notation and its consequence on the periodicity and statistical possessions of the random sequences. The authors reduced the fixed-point fraction length and expand the dependency on control parameters. The generated sequences required randomness for cryptography, by passing the NIST 800-22 tests. Koyuncu and Özcerit [35], presented the implementation of Sundarapandian—Pehlivan chaotic system [36] to design on Xilinx Virtex-6 with 58.76 Mbit/s data rate. NIST 800-22 and FIPS 140-1 tests were successful for the generated sequences. Avaroğlu et al. [37] presented a hybrid scheme to generate pseudo-random numbers. Sprott 94 G chaotic system is used as an additional input to improve the security of raw PRNG, which is generated from the proposed system. This additional input element is implemented on Virtex-6 FPGA. A seed value is randomly selected, and the Sprott 94 G is used to generate the key and fed to AES to generate the state values, which are XORed with the additional input to increase unpredictability. NIST 800-22 generated the successful result in the generated sequence. Murrilo-Escobar et al. [38] proposed a novel method to generate random numbers by modifying the logistic map. They presented successful statistical tests such as NIST 800-22 and TestU01. Moreover, the authors presented 2128 possible secret keys for a seed value to verify the sensitivity of key at bit-level. Wang et al. [39] proposed a piecewise logistic method to generate random numbers by modifying the logistic map and enhance the range of control parameters with successful statistical test NIST 800-22. Avaroğlu et al. [40] proposed a novel technique for generating TRNG. The ring oscillator is used as an entropy source and the logistic map is used to generate high-quality random numbers at the post-processing stage. Altera FPGA board is used to implement the proposed scheme. Successful results are presented using TESTU01 and NIST 800-22. François et al. [41], proposed a method to generate random numbers by mixing three chaotic maps with the initial vectors. The positions and indexes are calculated using chaotic function and linear congruence, which is used for the permutation. NIST 800-22 verified that the generated sequences are cryptographically secure.

Robust chaotic map [1] possesses positive Lyapunov exponent however they might result in cryptographically unsecure random numbers. Thus, CoC might represent a valid solution to generate CSPRNG. The larger parameter space of robust chaotic maps can be beneficial to fulfill cryptographic mechanisms. In this paper, we used a robust chaotic map with positive Lyapunov exponent, however

the distribution of this map is non-uniform. The non-uniform distribution of the opted map is the cause of the failure of NIST 800-22 tests that require equal probability of '0' and '1' in a binary sequence. Hence, the control of chaos is proposed which styles the probability of 0's and 1's equally likely. The uniform region of output is selected to produce CSPRNG by applying the scaling operation followed by modulo shifting operation. The threshold value is chosen carefully and considered critical in mapping the output value to either zero or one. The probability for the occurrence of each value of the set {0,1} remains equally likely. To validate the proposed PRNG design, we apply the NIST 800-22 test suite, which passes all the tests successfully. Hence, the proposed CSPRNG can be used effectively in different cryptographic applications including key generation, image encryption, and watermarking.

This paper is organized into following sections. Section 2 discusses the robust logistic map's bifurcation diagram, Lyapunov exponent, and ergodic behavior. The proposed scheme of PRNG design using RLM and NIST 800-22 tests evaluation is presented in Section 3. In Section 4, the performance analysis of proposed CSPRNG is explained for correlation, key-space and key sensitivity analysis. Finally, the conclusion of this paper is provided in Section 5.

2. The Robust Chaotic Logistic Map

The chaotic logistic map is one-dimensional. One-dimensional maps are considered simple because they are based on single mathematical equation. The chaotic logistic map, due to its simple structure, is generally studied to generate a PRNG for cryptographic services [42–45]. Equation (1) defines the one-dimensional logistic map.

$$x_{i+1} = \gamma x_i(1 - x_i), \tag{1}$$

where γ is a control parameter $\gamma \in [0,4]$ while $x_i \in (0,1)$ is initial value. The map is chaotic when control parameter $\gamma \in [3.57,4]$. When $\gamma = 4$, the map's output $x_{n+1} \in (0,1)$ covers the complete phase, which is evident in Figure 1. The chaotic logistic map has positive Lyapunov for the region $\gamma \in [3.57,4]$ in Figure 2 that represents the chaotic behavior. The Lyapunov exponent measures the quantitative divergence of orbits which confirms the chaotic behavior.

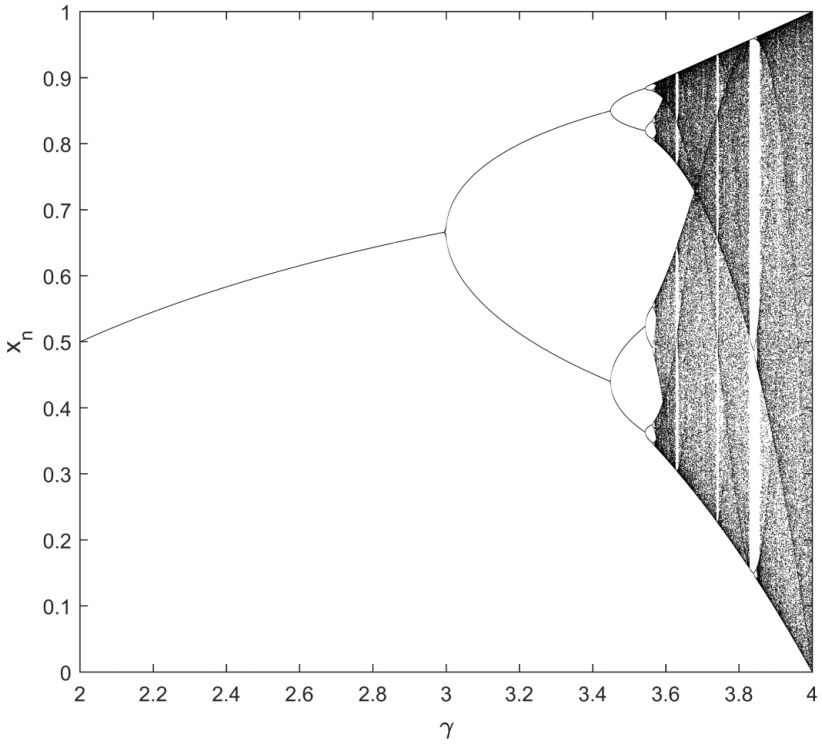


Figure 1. Bifurcation diagram for the Logistic Map.

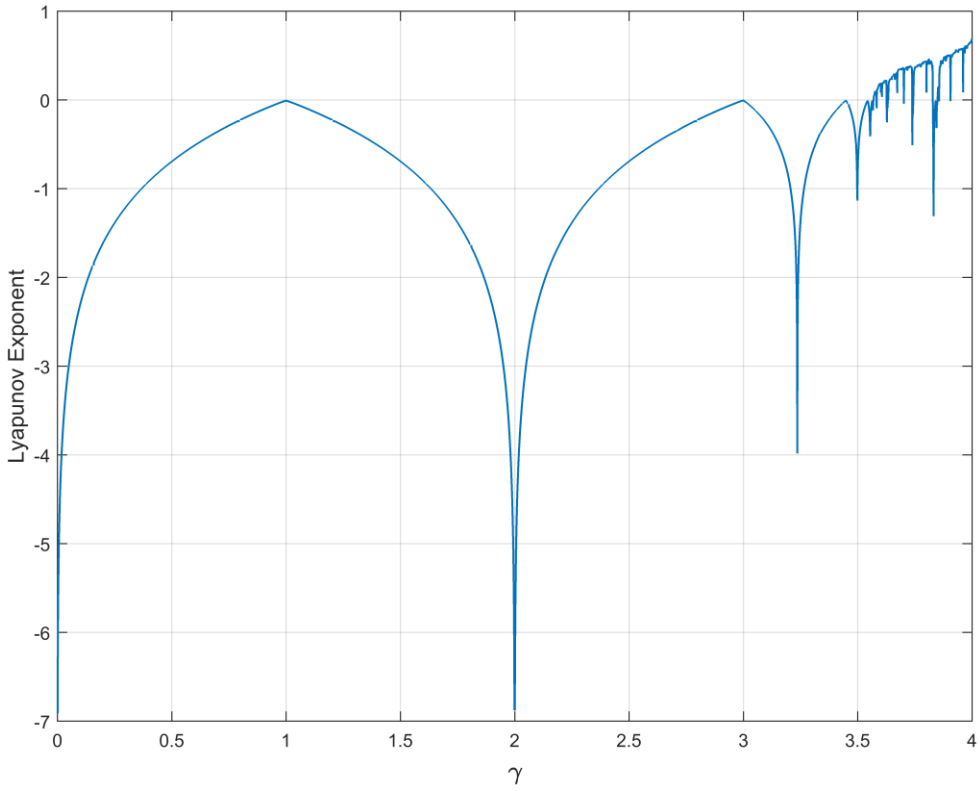


Figure 2. Lyapunov Exponent of Logistic Map.

Chen et al. [1], proposed an RLM which is an improved form of the logistic map, they enhanced the control parameter space for $\gamma \in [0,4] \rightarrow \gamma \in [0,31]$. Equation (2) defines the modified map.

$$L(\gamma, x) = \begin{cases} \gamma x(1-x)(mod\ 1), & x \in I_{ext} \\ \frac{\gamma x(1-x)(mod\ 1)}{\frac{\gamma}{4}\ (mod\ 1)}, & x \in I_{int} \end{cases}, \tag{2}$$

In Equation (2) above, $I_{ext} \in (0,1) \backslash I_{int}$, where $I_{int} \in [\eta_1, \eta_2]$, $\eta_1 = 1/2 - \sqrt{(1/4 - \lfloor \gamma/4 \rfloor / \gamma)}$ and $\eta_2 = 1/2 + \sqrt{(1/4 - \lfloor \gamma/4 \rfloor / \gamma)}$. Figure 3 measures the correlation between trajectories that are generated using the logistic map and RLM with an arbitrary initial condition. The x-axis shows input x_n and y-axis shows the map's output x_{n+1}. Based on the comparison, RLM and logistic map generates completely different trajectories at various values of γ. We have changed the initial conditions and results became same and maps are highly de-correlated. The inherent chaotic behavior of the RLM is discussed below:

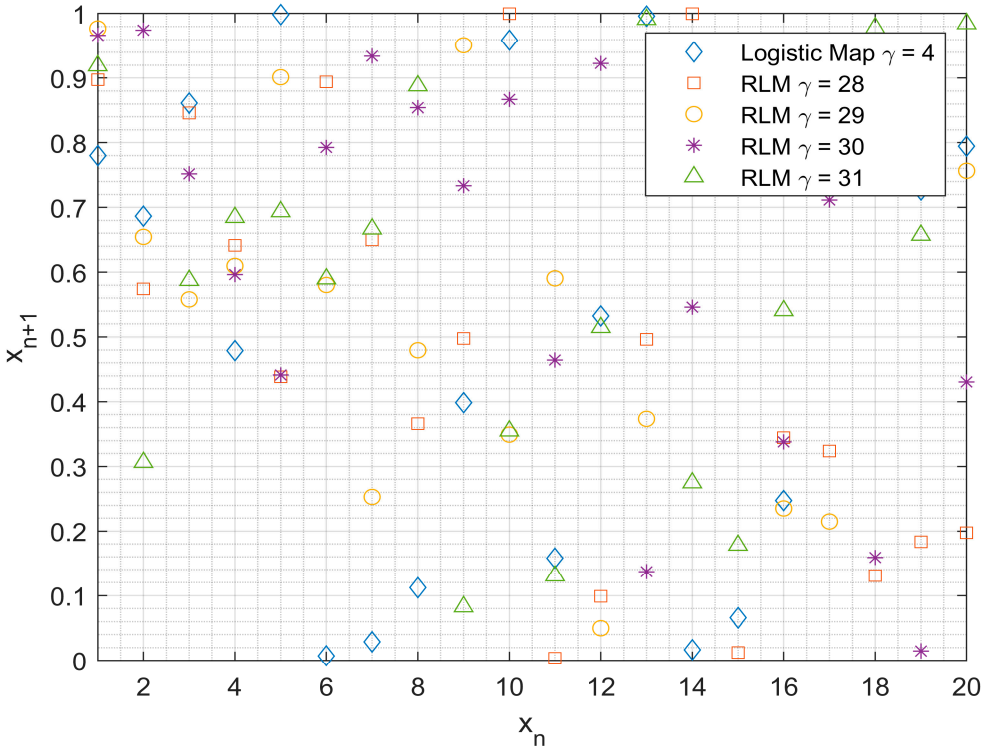
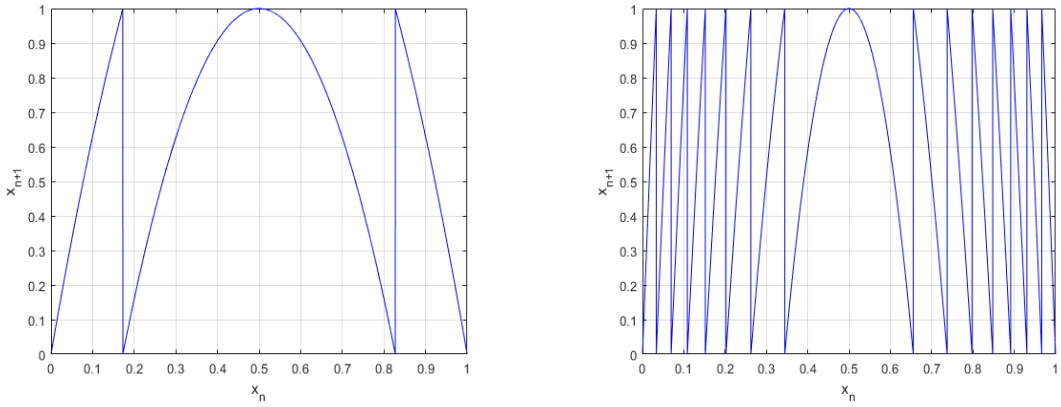


Figure 3. Correlation between trajectories of robust chaotic logistic map and chaotic logistic map.

2.1. Ergodicity

Ergodicity is an essential and closely related to the mixing property of chaotic system. In an ergodic system, orbit of every initial point from its definition interval leads to cover the complete phase space after a large number of iterations. For example, if one picks arbitrary initial condition x_0 and another number x_1 in phase space then trajectory of x_0 eventually visit x_1 during the course of iterations sometime or the other. The orbits having nature of visiting complete phase are called ergodic. Hence when iterated, the ergodic orbits are mixed throughout the whole interval.

By iterating Equation (2), the orbits of different arbitrary chosen inputs provide ergodic behavior and cover the complete phase space (0,1) when $\gamma \geq 4$ (Figure 4). It has good mixing properties when $\gamma \in [3.9,31]$ (Figure 5).



(**a**) (**b**)

Figure 4. Mapping of x_n vs. x_{n+1} (**a**) $\gamma = 7$ (**b**) $\gamma = 31$.

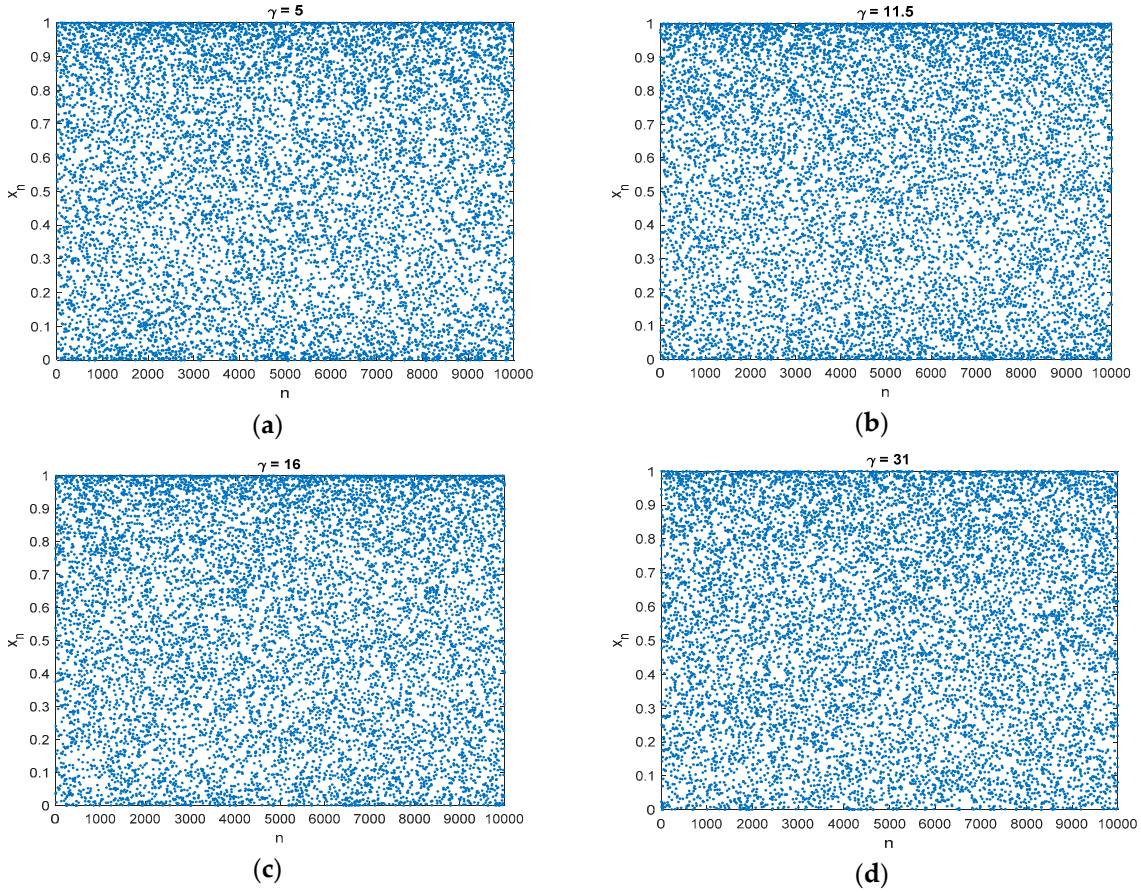


Figure 5. State Distribution of RLM (**a**) $\gamma = 5$ (**b**) $\gamma = 11.5$ (**c**) $\gamma = 16$ (**d**) $\gamma = 31$.

2.2. Bifurcation Diagram

The bifurcation diagram shows the mapping of orbits. It gives an inherent behavior of the chaotic system with the change of control parameters. The logistic map and its variants demonstrate period doubling bifurcation as shown in Figures 1 and 6. With the change in control parameter, chaotic obits switch their behavior to stable/unstable that results in period doubling bifurcation. All stable orbits mean they are covering the complete phase space, and hence the system is chaotic. The bifurcation diagram of RLM in Figure 6 shows chaotic behavior at $\gamma \geq 4$ and this range keep growing till $\gamma = 31$.

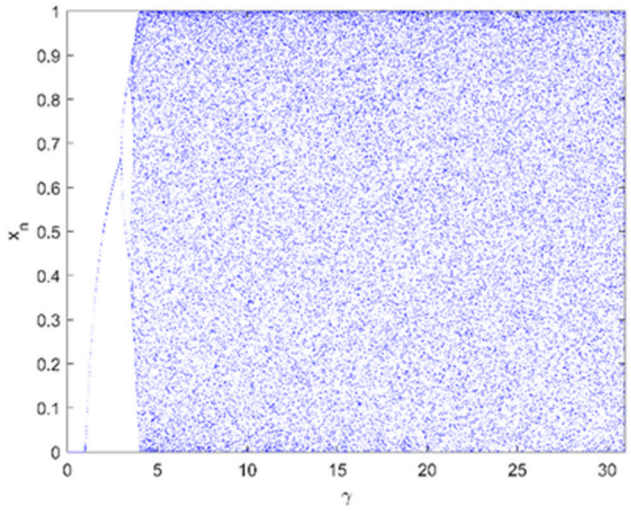


Figure 6. Bifurcation diagram of robust logistic map.

2.3. Lyapunov Exponent

The Lyapunov exponent is a quantitative measure of chaos and orbital divergence, by which the chaotic orbit of the logistic map is verified. A positive value of Lyapunov exponent indicates chaotic behavior and orbital divergence. The Lyapunov exponents of RLM for $\gamma = 0$ to 16 is shown in Figure 7. The Lyapunov exponent for RLM is positive for $\gamma \geq 4$.

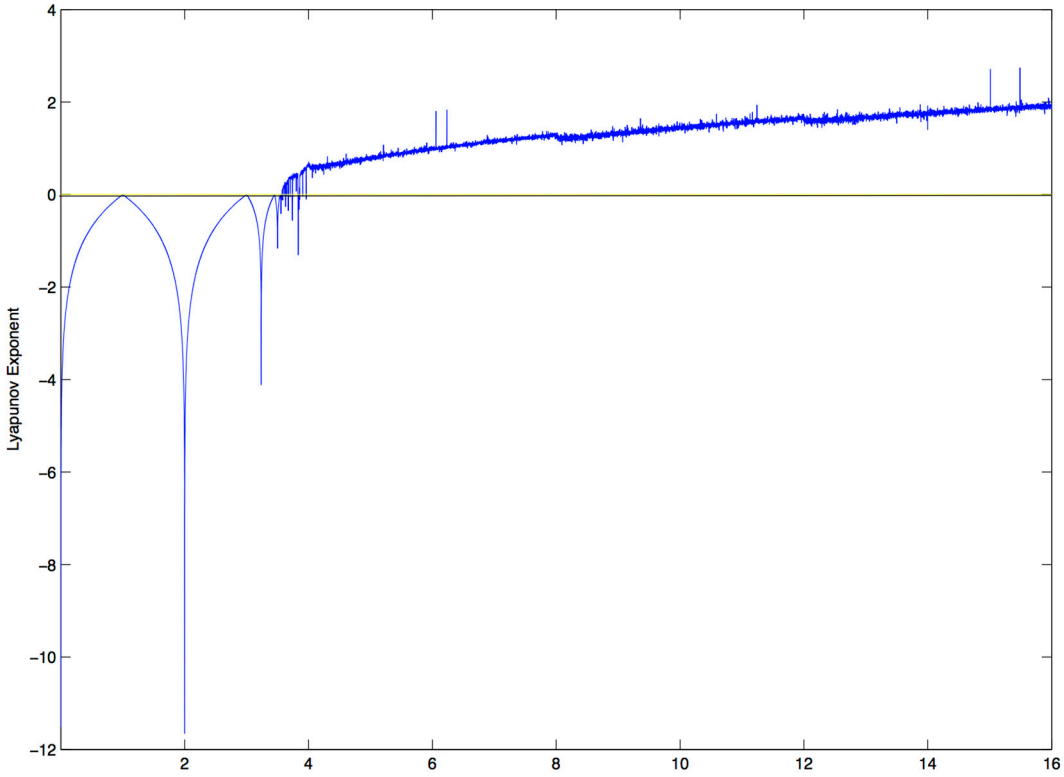


Figure 7. Lyapunov Exponent of robust logistic map.

3. Proposed Methodology

The objective of this study is to investigate RLM for CSPRNG. Initially, we show that NIST 800-22 test suite PRNG for RLM is unsecure. Therefore, in this study, the design of CSPRNG is presented. The NIST 800-22 test of PRNG using CSPRNG validates the effectiveness of proposed methodology. The next subsection will cover both methodologies in detail.

3.1. Parameter Selection

The initial conditions and control parameters are chosen arbitrarily for the maps. Small perturbation on initial conditions and control parameters gives us completely different trajectories that are used to measure the key sensitivity and key space analysis. Small perturbations on control value investigate the behavior of bifurcations of the maps as shown in Figure 6. The logistic map bifurcations diagram in Figure 1 show that all trajectories are stable and covers complete phase space when $\gamma = 4$. Based on the results in Figure 6, RLM is chaotic and covers the complete phase space, and all trajectories are stable for $\gamma \in [4,31]$.

3.2. PRNG Using RLM

In this study, PRNs generated using RLM are verified using NIST test suit. The PRNGs are crucially important, because the achieved randomness directly affects the security of encrypted applications. PRNGs having uniform probability distribution of binary sequences are desirable in cryptography. The chaotic trajectories generated using RLM having infinite real number values in range $\in (0,1]$.

However, Figure 8 validated the non-uniform distribution of RLM trajectories. Therefore, efficient mapping is required from real to binary domain that gives random bits stream of '0' and '1' with uniform probability distribution. One of the methods is to threshold the real domain to generate binary sequence. We choose a median value as a threshold $\mathbb{T} = 0.5$. The performance of the proposed RNG is evaluated with NIST-800-22 statistical test suite [46], which includes 15 different tests. A bit stream of length 1 million (1M) is required for NIST-800-22 statistical tests. The flow graph of unsecure random bit generator (RBG) is shown in Figure 9, and the steps for generating a random bit stream using RLM PRNG are as follows:

Step-1: Choose an arbitrary chosen initial condition $x_0 \in (0,1)$ and control parameter $\gamma \in [4,31]$ as an input to iterate RLM for generating output $x_n \in [0,1]$.

Step-2: The RLM is iterated 1M times for generating N random floating-point values $\in [0,1]$.

Step-3: For real to binary domain mapping, thresholding is applied to the floating-point numbers. Threshold value of $\mathbb{T} = 0.5$ is chosen.

Step-4: Each generated output of RLM is checked where it falls. The binary value of '1' is chosen if $x_i \geq \mathbb{T}$ or '0' otherwise.

Step-5: Stop iterating RLM once the bit stream of 1M is generated.

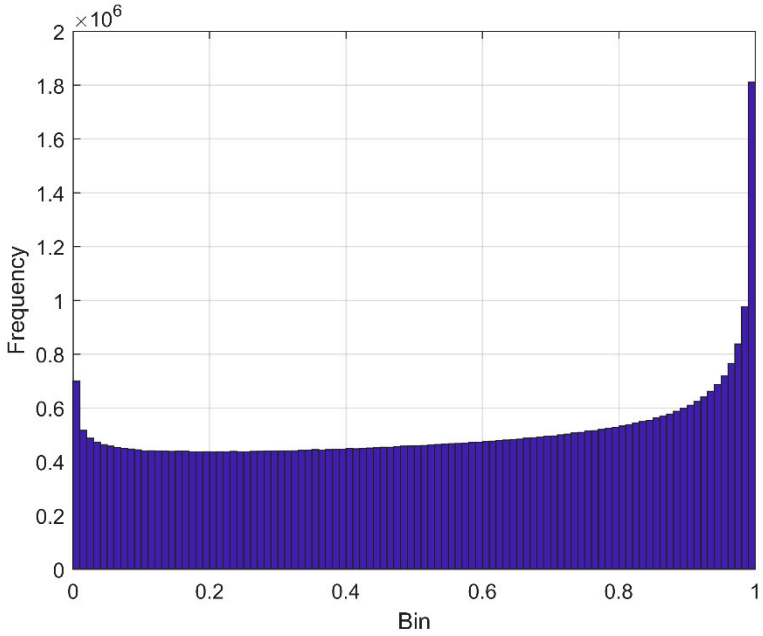


Figure 8. Histogram of RLM.

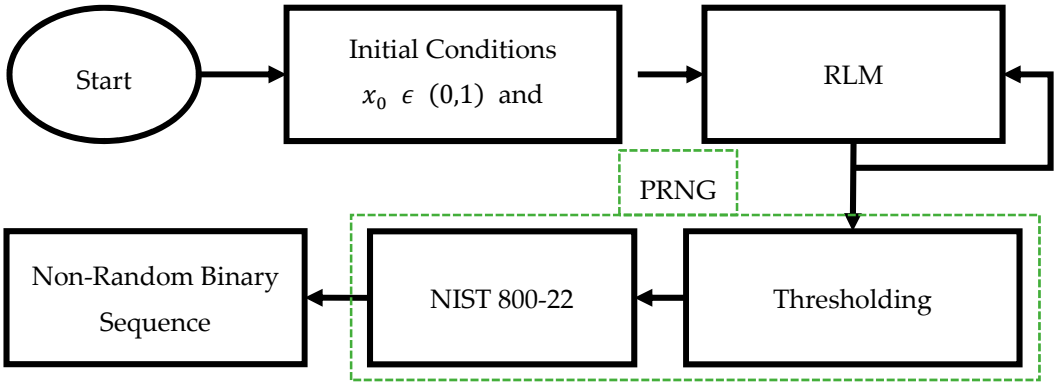


Figure 9. Unsecure RBG using RLM.

The generated bit stream using above mention steps is tested using NIST 800-22 test. Based on the results in Table 1, standard NIST test on RLM's PRNG is failed. This is because the non-uniform probability distribution nature of RLM trajectories. Thus, PRNGs using RLM are not suitable to be used in cryptography. The next subsection presents the proposed CSPRNG design using modified RLM.

Table 1. Randomness test of robust logistic map using national institute of standards and technology (NIST) 800-22 test suit.

NIST Statistical Test	*p*-Value	Status
Frequency (monobit)	0.00	Failed
Block Frequency	0.00	Failed
Cumulative Sum	0.00 (Forward) 0.00 (Reverse)	Failed
Longest Run	0.00	Failed
Runs	0.00	Failed
Rank	0.755843	Passed
Non-overlapping Template Matchings	0.00	Failed
Discrete Fourier Transform	0.00	Failed
Overlapping Template Matchings	0.00	Failed
Universal Statistical	0.00	Failed
Serial	0.00	Failed
Random Excursions Variant	0.00	Failed
Random Excursions	0.00	Failed
Approximate Entropy	0.00	Failed
Linear Complexity	0.589895	Passed

3.3. Modified RLM (MRLM) Aided by CoC

PRNG to be used in any cryptographic application requires passing all the NIST tests. The Lyapunov exponent of RLM is positive but NIST test is failed. During mapping from real to decimal domain, maybe the inherent randomness of chaotic map is reduced. Therefore, in this study CoC is proposed where small perturbations are applied to the input parameters to generate RLM chaotic trajectories with uniform probability distribution function. For the CoC of RLM, the output region $0.1 \leq x_i \leq 0.6$ is desirable where the output of RLM is uniform as shown in Figure 10. The flow graph of proposed CSPRNG using Modified RLM (**M**RLM) is give in Figures 11 and 12. The pseudocode of the proposed method is also presented in Algorithm 1.

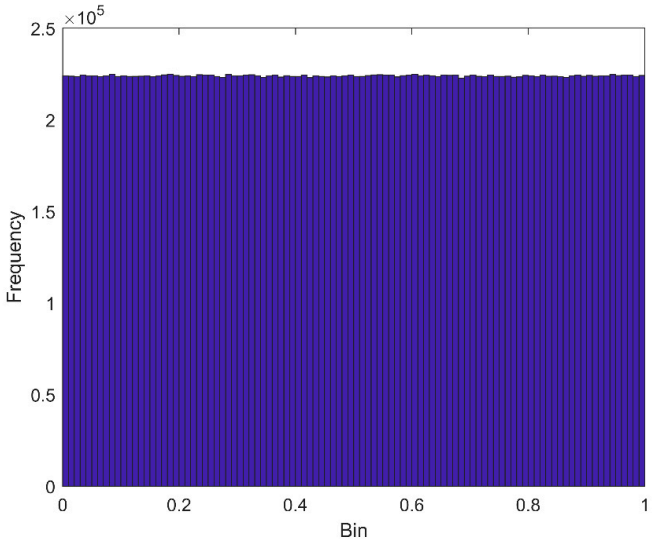


Figure 10. Histogram of modified robust chaotic logistic map aided CoC.

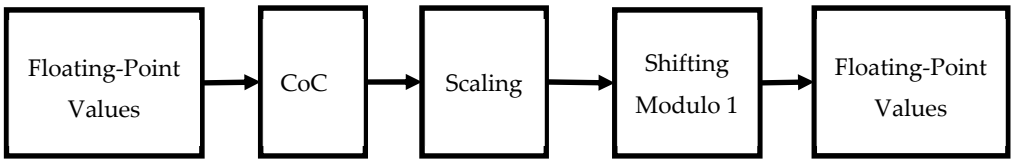


Figure 11. Controlling of Chaos (CoC) in RLM.

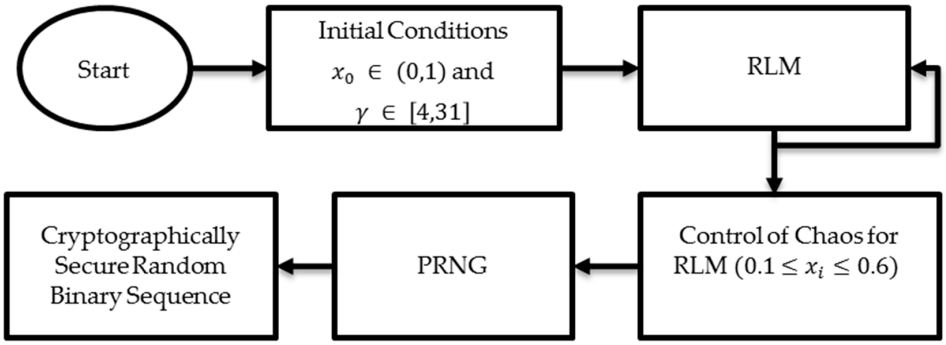


Figure 12. Secure RBG by CoC.

Algorithm 1: Generation of CSPRNG.

Pseudocode: Generation of Optimized Pseudo-random binary Sequence
Output: Pseudo-random binary sequence
Procedure:
initial seed value of $x_0 \in (0,1)$ and $\gamma \in [4,31]$.

$$\eta_1 \leftarrow \left\lceil \frac{1}{2} - \sqrt{\frac{1}{4} - \frac{\lfloor \frac{\gamma}{4} \rfloor}{\gamma}} \right\rceil$$
$$\eta_2 \leftarrow \left\lfloor \frac{1}{2} + \sqrt{\frac{1}{4} - \frac{\lfloor \frac{\gamma}{4} \rfloor}{\gamma}} \right\rfloor$$

for $i = 1 : N$ **do**
if $\gamma \leq 4$

$$x_{i+1} \leftarrow [\gamma x_i (1 - x_i)]$$

else
if $x_i \geq \eta_1$ and $x_i \leq \eta_2$

$$x_{i+1} \leftarrow \frac{\gamma x(1-x)(mod\ 1)}{\frac{\gamma}{4}\ (mod\ 1)}$$

else

$$x_{i+1} \leftarrow \gamma x(1-x)(mod\ 1)$$

end if
end if
if $x_i \geq 0.1$ and $x_i \leq 0.6$

$$y_j \leftarrow x_i$$
$$y_j \leftarrow y_i \times 10^{10}\ mod\ 1$$

if $y_j \geq \mathbb{T}$

$$y_j = 1$$

else

$$y_j = 0$$

end if

$$j \leftarrow j + 1$$

else
discard x_i
end if
end for
y_j is the resultant binary sequence

The detail of steps involved in generating CSPRNG using proposed MRLM is given below:

Step-1: Choose an arbitrary initial condition $x_0 \in (0,1)$ and control parameter $\gamma \in [4,31]$.

Step-2: Calculate $I_{int} \in [\eta_1, \eta_2]$, where $\eta_1 = \left[\frac{1}{2} - \sqrt{\frac{1}{4} - \frac{\lfloor \frac{\gamma}{4} \rfloor}{\gamma}}\right]$ and $\eta_2 = \left[\frac{1}{2} + \sqrt{\frac{1}{4} - \frac{\lfloor \frac{\gamma}{4} \rfloor}{\gamma}}\right]$.

Step-3: Iterate RLM using (2) N times, and generate N floating-point numbers in the range $\in [0,1]$.

Step-4: Apply CoC to choose the output region within the range $\in [0.1, 0.6]$ by carefully small perturbations are applied to the input to filter out the desired output sequence.

Step-5: Scale the output region $\in [0.1, 0.6]$ by a factor of 10^{10}.

Step-6: Apply modulo 1 operation on scaled region to generate output value $\in (0,1)$.

Step-7: Apply threshold to the resultant floating values such that each random value x_i is mapped to '1' if $x_i \geq \mathbb{T}$, else it is mapped to '0'. In this regard, choose a threshold value such that the probability distribution of 0's and 1's is almost equal. In our case, we choose $\mathbb{T} = 0.5$ for CoC to generate cryptographically secure pseudorandom numbers.

Step-8: Stop iterating the map when 1M bits stream is generated.

The generated bits stream is then tested using NIST suit. Based on the results in Table 2, the proposed MRLM passed all the NIST tests with chosen $\gamma = 31$ and $x_0 = 0.78$.

Table 2. Randomness NIST test for MRLM.

NIST Statistical Test	*p*-Value	Status
Frequency (monobit)	0.870382	Passed
Block Frequency	0.63719	Passed
Cumulative Sum	0.540598 (Forward) 0.687661 (Reverse)	Passed
Longest Run	0.289731	Passed
Runs	0.635262	Passed
Rank	0.890846	Passed
Non-overlapping Template Matchings	0.813509	Passed
Discrete Fourier Transform	0.902994	Passed
Overlapping Template Matchings	0.711692	Passed
Universal Statistical	0.939078	Passed
Serial	0.757126 0.422466	Passed
Random Excursions Variant	0.756827	Passed
Random Excursions	0.999976	Passed
Approximate Entropy	0.651855	Passed
Linear Complexity	0.579334	Passed

4. Performance Analysis

The proposed method is tested for the well-established performance parameters of correlation, key space and key sensitivity analysis. The details of these properties are explained in the next subsection.

4.1. Key Space Analysis

The key space of secret keys or seed values must be larger than 2^{100} to resist against the brute force attack [47]. The proposed CSPRNG using MRLM is based on the two parameters of initial condition $x_n \, \epsilon \, (0,1)$ and control parameter of $\gamma \, \epsilon \, [3.9, 31]$. The precision of double floating-point is 10^{-16} stated in the IEEE floating-point standard [48]. Therefore, x can be any of 10^{16} values. Similarly, γ can be any value in the range $(31 - 3.9) \times 10^{16} = 2.71 \times 10^{17}$. Therefore, the key-space of proposed PRNG is $2.71 \times 10^{16} \times 10^{16} \approx 2^{111}$. The given range is used to compare the key space of proposed method with recently proposed methods is given in Table 3. Based on the results, the proposed scheme has a

larger key space compared to recently proposed schemes. The proposed method satisfies the key space requirement and resists the brute force attack.

Table 3. Comparison analysis of key space.

Study	Control Parameter	Initial Condition	Key Space
Proposed work	$\gamma \in [4, 31]$	$x \in (0, 1)$	$(31-4) \times 10^{16} \times 10^{16} \approx 2^{111}$
Luyao et al. [49]	$\gamma \in [3.5699, 4]$	$x \in (0, 1)$	$(4-3.5699) \times 10^{16} \times 10^{16} \approx 2^{104}$
Wang et al. [39]	$\gamma \in (0, 4]$	$x \in (0, 1)$	$(4-0) \times 10^{16} \times 10^{16} = 4 \times 10^{32} \approx 2^{108}$
Murrilo-Escobar et al. [38]	$\gamma \in [3.999, 4]$	$x \in (0, 1)$	$(4-3.999) \times 10^{16} \times 10^{16} = 10^{30} \approx 2^{99}$
Behnia et al. [50]	$\gamma \in [0, 1]$	$x \in (0, 1)$	$(1-0) \times 10^{16} \times 10^{16} = 10^{32} \approx 2^{106}$

4.2. Correlation Analysis

The correlation coefficient is used to measure the dependence and statistical relationship among random variables. The correlation coefficient between the two sequences is given as:

$$cov(x, y) = E\{(x - E(x))(y - E(y))\}, \tag{3}$$

$$r_{xy} = \frac{cov(x, y)}{\sqrt{D(x)}\sqrt{D(y)}}, \tag{4}$$

where x and y are two random variable sequences $E(x) = \frac{1}{M}\sum_{i=1}^{M} x_i$, $D(x) = \frac{1}{M}\sum_{i=1}^{M}(x_i - E(x))^2$. The MRLM is iterated 1000 times by ignoring initial transient. To measure correlation using (2) and (3), small perturbations of $(\gamma + \Delta)$ are applied to control parameter to generate MRLM trajectories. The chosen value of $\Delta = \frac{271}{2^{58}}$ is required that generate uncorrelated MRLM trajectories. The correlation coefficient falls in the range $\in$ [−0.0875,0.0915] as shown in Figure 13. In other words, there is no correlation between the generated sequences.

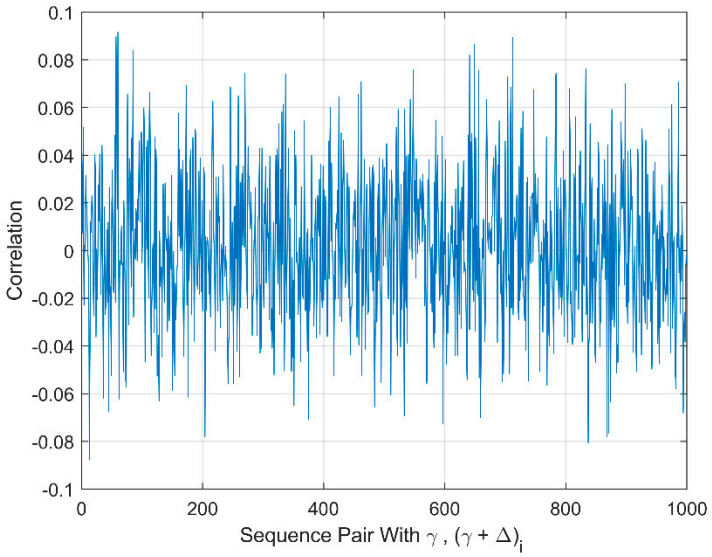


Figure 13. Correlation analysis of MRLM trajectories with the slight change in control parameter.

4.3. Key Sensitivity Analysis

Essentially, a cryptographic algorithm is required to be highly sensitive to small change in the key [47]. In this study, to measure key sensitivity between the keys, arbitrary keys of length 128-bits are generated with the change of one bit at least significant bit positions. A method for initial condition mapping to generate CSPRNG is given in Figure 14. Key sensitivity analysis measures small change in keys can generate highly uncorrelated CSPRNGs. The steps involved for initial condition mapping is given below:

Step-1: Choose an arbitrary 128-bit hexadecimal key

Step-2: Split the key into sixteen bytes. To spread the effect of small change at least significant bit over complete key and CSPRNGs, sixteen bytes are XORed together to generate an 8-bit number.

Step-3: Convert the 8-bit binary number into decimal.

Step-4: Divide the decimal value by 256 to generate real value $x_0 \in (0,1)$.

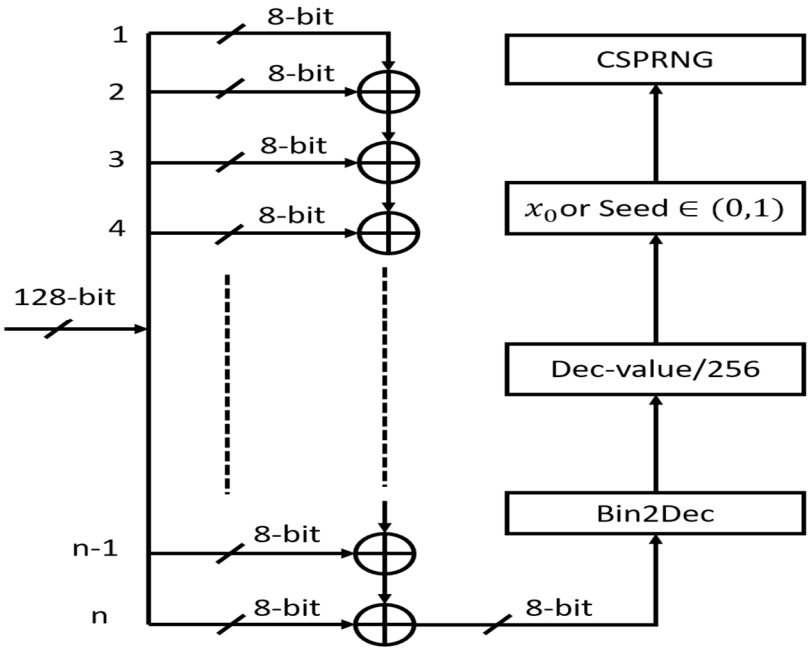


Figure 14. Initial Condition mapping for MRLM.

CSPRNGs are generated by iterating MRLM with varying mapped x_0 and arbitrary chosen fixed control parameter $\gamma = 31$. Arbitrary chosen keys with Single bit change at least significant bit are given in Table 4, generate highly CSPRNGs of 1000 bits are given in. Based on results in Figure 15, CSPRNGs are apparently uncorrelated and highly sensitive to small change in the key. The correlation between the generated sequences falls in the range $\in (-0.0245, 0.0138)$.

Table 4. Arbitrary Keys involved in sensitivity analysis.

Keys	Secret Key	Marker
Key1	AEDCBA09876543211234567890ABCDEF	□
Key2	AEDCBA09876543211234567890ABCDEE	○
Key3	AEDCBA09876543211234567890ABCDED	★
Key4	AEDCBA09876543211234567890ABCDEC	Δ
Key5	AEDCBA09876543211234567890ABCDEB	★
Key6	AEDCBA09876543211234567890ABCDEA	✧

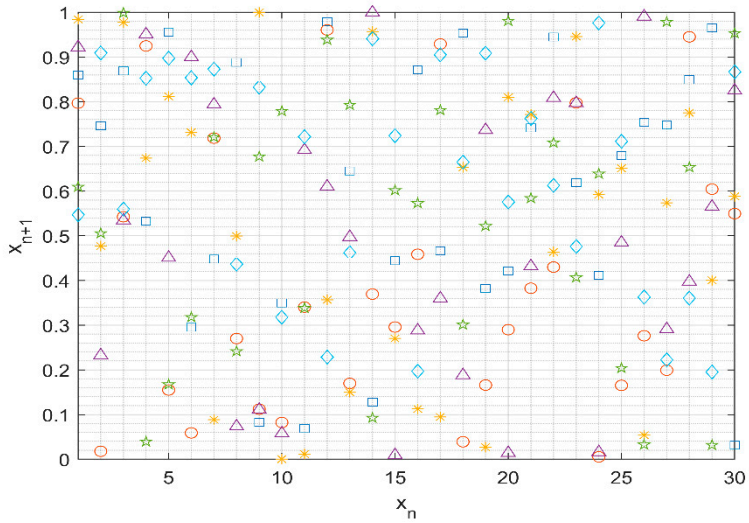


Figure 15. Correlation analysis between trajectories using proposed method.

4.4. CoC on Chaotic Map

Essentially, the probability distribution function of any given chaotic map describes its behavior. Smooth probability distribution function is required in chaotic map to generate CSPRNG. Therefore, in Table 5 four different chaotic maps such as circle [51], iterative [52], tent [53], and singer [54] maps are chosen with non-smooth probability distribution function. NIST STS test is performed on these maps with and without applying proposed CoC. It is apparent in Table 4 that typical chaotic maps used for comparison pass all NIST STS tests using the proposed CoC.

In any given chaotic map, changing the control parameters beyond the given limit move the fix points to infinity hence chaotic behavior is vanished. Therefore, modifications are proposed in maps to keep the chaotic fix points stable and within the range $\in (0, 1)$ [1,53]. Chaotic maps are based on mathematical equations. Modified chaotic maps for larger parameter space having variable internal region that is shrink and expand with the change in control parameters. The change in internal region is described mathematically based on chaotic map's equation aided scaling and modulo operations. In [53] author modified chaotic tent map (MCTM) that enlarge the parameter space. The internal region is mathematically derived and aided by modulo and scaling operations. In [1] chaotic logistic map's parameter space is expanded with derived internal region. The internal region required deriving carefully for chaotic behavior and entirely based on map's equation. To analyze internal region selection, two different use cases are presented. Case 1 is measuring the NIST STS of MCTM and RLM with their respective internal region equations. Case 2 is measuring NIST STS by swapping the internal region equations. Based on the results in Table 6, it is evident that the wrongly derived internal region entails a failed NIST STS test.

Table 5. Comparison with different Chaotic Maps.

Chaotic Maps	**Circle**		**Iterative**		**Tent**		**Singer**	
	Equation	**IC**	**Equation**	**IC**	**Equation**	**IC**	**Equation**	**IC**
	$x_{n+1} = \mathbf{mod}(x_n + b - (\frac{a}{2\pi}) \sin(2\pi x_n), 1)$	x_0=0.7 a=0.5 b=0.2	$x_{n+1} = \sin(\frac{a\pi}{x_n})$	x_0=0.5 a=0.7	$x_{n+1} = \begin{cases} mod(\mu x_n, 1), & x_n < 0.5 \\ mod(\mu(1-x_n), 1), & 0.5 \leq x_n \end{cases}$	x_0= 0.5 μ= 1.99	$x_{n+1} = \mu(7.86x_n - 23.31x_n^2 + 28.75x_n^3 + 13.302875x_n^4)$	x_0= 0.7 μ=1.07
	NIST STS		**NIST STS**		**NIST STS**		**NIST STS**	
	Without CoC	**With CoC**	**Without CoC**	**With CoC**	**Without CoC**	**With CoC**	**Without CoC**	**With CoC**
Frequency (monobit)	×	✓	✓	✓	×	✓	×	✓
Block Frequency	×	✓	×	✓	✓	✓	×	✓
Cumulative Sum	×	✓ (R) ✓ (F)	✓ (R) ✓ (F)	✓ (R) ✓ (F)	×	✓ (R) ✓ (F)	×	✓ (R) ✓ (F)
Longest Run	×	✓	×	✓	✓	✓	×	✓
Runs	×	✓	×	✓	×	✓	×	✓
Rank	×	✓	✓	✓	✓	✓	×	✓
Non-overlapping Template Matchings	×	✓	×	✓	×	✓	×	✓
Discrete Fourier Transform	×	✓	×	✓	×	✓	×	✓
Overlapping Template Matchings	×	✓	×	✓	×	✓	×	✓
Universal Statistical	×	✓	×	✓	×	✓	×	✓
Serial	× ×	✓	× ×	✓	× ✓	✓ ✓	× ×	✓ ✓
Random Excursions Variant	×	✓	✓	✓	×	✓	×	✓
Random Excursions	×	✓	✓	✓	×	✓	×	✓
Approximate Entropy	×	✓		✓	×	✓	×	✓
Linear Complexity	×	✓	✓	✓	✓	✓	✓	✓

Table 6. Comparison with region equations.

NIST Statistical Test				
	Initial Parameter: $x_0 = 0.23$, $\mu = 5.95$			
	Case 1		**Case 2**	
TESTS	**MRLM**	**MTM**	**MRLM**	**MCTM**
Frequency (monobit)	✓	✓	✓	×
Block Frequency	✓	✓	✓	×
Cumulative Sum	✓ (R) ✓ (F)	✓ (R) ✓ (F)	✓ (R) ✓ (F)	×(R) ×(F)
Longest Run	✓	✓	✓	×
Runs	✓	✓	×	×
Rank	✓	✓	✓	✓
Non-overlapping Template Matchings	✓	✓	✓	×
Discrete Fourier Transform	✓	✓	✓	✓
Overlapping Template Matchings	✓	✓	✓	×
Universal Statistical	✓	✓	✓	✓
Serial	✓ ✓	✓ ✓	× ✓	× ✓
Random Excursions Variant	✓	✓	✓	×
Random Excursions	✓	✓	×	×
Approximate Entropy	✓	✓	×	×
Linear Complexity	✓	✓	✓	✓

5. Conclusions

In this study, modifying RLM using the proposed CoC generates cryptographically secure pseudorandom numbers. The RLM has large chaotic parameter space and possesses a positive Lyapunov exponent, but it has a non-uniform probability distribution of trajectories that makes it undesirable for cryptography. Hence, CoC in RLM gives a uniform distribution of the output sequence. The CSPRNG proposed design is vetted using NIST 800-22 tests. The correlation, key-sensitivity and key-space statistical analysis show that large parameter space of RLM gives sufficiently large key length and key space to resist all known attacks. The proposed modified RLM (MRLM) has the potential to be used in various cryptographic applications including image, telemedicine, electronic payment, computation, text source, personal information, biometrics, and military among others.

Author Contributions: All authors contributed in the design problem and the proposed methodology. Simulations were performed by M.I. and M.E.-u.-H. Initial draft was written by A.A., M.E.-u.-H. The simulations were verified by M.A.K. The conceptualization and supervised by M.A.K. and S.N.M.S. Final Draft was proofread and edited by M.A.K., W.A. and A.S. All authors have read and agreed to the published version of the manuscript.

Acknowledgments: I would like to thank department of computer engineering, university of engineering and technology, Taxila, Pakistan for their support and facilitation.

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

References

1. Chen, S.-L.; Chang, S.-M.; Lin, W.-W.; Hwang, T. Digital secure-communication using robust hyper-chaotic systems. *Int. J. Bifurc. Chaos* **2008**, *18*, 3325–3339. [CrossRef]
2. Eisencraft, M.; Evangelista, J.V.C.; Costa, R.A.; Fontes, R.T.; Candido, R.; Chaves, D.P.B.; Pimentel, C.; Silva, M.T.M. *New Trends in Chaos-Based Communications and Signal Processing*; Springer: Berlin, Germany, 2019.
3. Ghauch, Z.G.; Aitharaju, V.; Rodgers, W.R.; Pasupuletti, P.; Dereims, A.; Ghanem, R.G. Integrated stochastic analysis of fiber composites manufacturing using adapted polynomial chaos expansions. *Compos. Part A Appl. Sci. Manuf.* **2019**. [CrossRef]

4. Waqas, A.; Melati, D.; Manfredi, P.; Melloni, A. Stochastic process design kits for photonic circuits based on polynomial chaos augmented macro-modelling. *Opt. Express* **2018**. [CrossRef]
5. Bai, C.; Ren, H.P.; Baptista, M.S.; Grebogi, C. Digital underwater communication with chaos. *Commun. Nonlinear Sci. Numer. Simul.* **2019**. [CrossRef]
6. Soujeri, E.; Kaddoum, G.; Herceg, M. Design of an initial condition-index chaos shift keying modulation. *Electron. Lett.* **2018**. [CrossRef]
7. Nomura, H.; Temsiririrkkul, S.; Ikeda, K. Generation of "Natural" Pseudorandom Numbers from the Standard Player's View. Available online: https://ci.nii.ac.jp/naid/120006675914/en (accessed on 1 November 2019).
8. Berg, D.A.; Luciano, R.A., Jr.; Saffari, A. Central Random Number Generation for Gaming System. U.S. Patent 5,779,545, 14 July 1998.
9. Meoni, F. Casino Random Number Card Covering Game. U.S. Patent 5,700,009, 23 December 1997.
10. Hull, T.E.; Dobell, A.R. Random number generators. *SIAM Rev.* **1962**, *4*, 230–254. [CrossRef]
11. Singh, V.; Pal, P. Survey of different types of CAPTCHA. *Int. J. Comput. Sci. Inf. Technol.* **2014**, *5*, 2242–2245.
12. Pawar, T.S.; Sawant, R.G.; Bothe, P.S.; Chopade, S.A. A Survey on Login Authentication System using Captcha as Graphical Password Technquies. *Int. J. Innov. Res. Comput. Commun. Eng.* **2015**. [CrossRef]
13. Tiwari, A.K. Introduction to machine learning. In *Ubiquitous Machine Learning and Its Applications*; IGI Global: Dehradun, India, 2017.
14. Seeger, M. Gaussian processes for machine learning. *Int. J. Neural Syst.* **2004**, *14*, 69–106. [CrossRef]
15. Corrigan-Gibbs, H.; Mu, W.; Boneh, D.; Ford, B. Ensuring high-quality randomness in cryptographic key generation. In Proceedings of the ACM Conference on Computer and Communications Security, Berlin, Germany, 4–8 November 2013; pp. 685–696.
16. Bucci, M.; Germani, L.; Luzzi, R.; Tommasino, P.; Triftlettv, A.; Varanonuov, M. A high speed truly IC random number source for Smart Card microcontrollers. In Proceedings of the IEEE International Conference on Electronics, Circuits, and Systems, Rodos, Greece, 6 August 2002.
17. Pétrie, C.S.; Alvin Connelly, J. A noise-based ic random number generator for applications in Cryptography. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **2000**. [CrossRef]
18. Qi, B.; Chi, Y.-M.; Lo, H.-K.; Qian, L. High-speed quantum random number generation by measuring phase noise of a single-mode laser. *Opt. Lett.* **2010**. [CrossRef] [PubMed]
19. Schindler, W.; Killmann, W. Evaluation criteria for true (Physical) random number generators used in cryptographic applications. In Proceedings of the 4th International Workshop, Redwood Shores, CA, USA, 13–15 August 2002.
20. Shannon, C.E. Communication Theory of Secrecy Systems. *Bell Syst. Tech. J.* **1949**, *28*, 656–715. [CrossRef]
21. Patidar, V.R.; Sud, K.K. A novel pseudo random bit generator based on chaotic standard map and its testing. *Electron. J. Theor. Phys.* **2009**, *6*, 327–344.
22. Pellicer-Lostao, C.; López-Ruiz, R. Pseudo-random bit generation based on 2D chaotic maps of logistic type and its applications in chaotic cryptography. In Proceedings of the Computational Science and Its Applications—ICCSA 2008, Perugia, Italy, 30 June–3 July 2008.
23. Shatheesh Sam, I.; Devaraj, P.; Bhuvaneswaran, R.S. Transformed logistic block cipher scheme for image encryption. In *Communications in Computer and Information Science*; Springer: Berlin, Germany, 2011.
24. Kocarev, L.; Jakimoski, G. Logistic map as a block encryption algorithm. *Phys. Lett. Sect. A Gen. Solid State Phys.* **2001**. [CrossRef]
25. Yang, H.; Wong, K.W.; Liao, X.; Wang, Y.; Yang, D. One-way hash function construction based on chaotic map network. *Chaos Solitons Fractals* **2009**. [CrossRef]
26. Jhansi Rani, P.; Rao, M.S.; Durga Bhavani, S. Design of secure chaotic hash function based on logistic and tent maps. In *Communications in Computer and Information Science*; Springer: Berlin, Germany, 2011.
27. Arroyo, D.; Amigó, J.M.; Alvarez, G.; Aplicada, F. On the inadequacy of unimodal maps for cryptographic applications. In *XI Reunión Española Sobre Criptología y Seguridad de la Información (XI RECSI)*; Available online: https://pdfs.semanticscholar.org/7b96/a768e2eeaf86bbe789cfc14f7ece29b97d96.pdf (accessed on 1 November 2019).
28. 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]

29. Lorenz, E.N. Deterministic nonperiodic flow. In *Universality in Chaos*, 2nd ed.; Taylor and Francis: New York, NY, USA, 2017; pp. 367–378.
30. Lü, J.; Chen, G. A new chaotic attractor coined. *Int. J. Bifurc. Chaos* **2002**. [CrossRef]
31. Elmanfaloty, R.A.; Abou-Bakr, E. Random property enhancement of a 1D chaotic PRNG with finite precision implementation. *Chaos Solitons Fractals* **2019**. [CrossRef]
32. Kadir, A.; Hamdulla, A.; Guo, W.Q. Color image encryption using skew tent map and hyper chaotic system of 6th-order CNN. *Optik (Stuttg)* **2014**. [CrossRef]
33. 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]
34. 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, CIS 2008, Suzhou, China, 13–17 December 2008.
35. Koyuncu, İ.; Turan Özcerit, A. The design and realization of a new high speed FPGA-based chaotic true random number generator. *Comput. Electr. Eng.* **2017**, *58*, 203–214. [CrossRef]
36. Sundarapandian, V.; Pehlivan, I. Analysis, control, synchronization, and circuit design of a novel chaotic system. *Math. Comput. Model.* **2012**, *55*, 1904–1915. [CrossRef]
37. Avaroğlu, E.; Tuncer, T.; Özer, A.B.; Türk, M. A new method for hybrid pseudo random number generator. *Inf. MIDEM* **2014**, *44*, 303–311.
38. Murillo-Escobar, M.A.; Cruz-Hernández, C.; Cardoza-Avendaño, L.; Méndez-Ramírez, R. A novel pseudorandom number generator based on pseudorandomly enhanced logistic map. *Nonlinear Dyn.* **2017**, *87*, 407–425. [CrossRef]
39. Wang, Y.; Liu, Z.; Ma, J.; He, H. A pseudorandom number generator based on piecewise logistic map. *Nonlinear Dyn.* **2016**, *83*, 2373–2391. [CrossRef]
40. Avaroğlu, E.; Tuncer, T.; Özer, A.B.; Ergen, B.; Türk, M. A novel chaos-based post-processing for TRNG. *Nonlinear Dyn.* **2015**, *81*, 189–199. [CrossRef]
41. François, M.; Defour, D.; Negre, C. A fast chaos-based pseudo-random bit generator using binary64 floating-point arithmetic. *Informatica* **2014**, *38*, 115–124.
42. Rostami, M.J.; Shahba, A.; Saryazdi, S.; Nezamabadi-pour, H. A novel parallel image encryption with chaotic windows based on logistic map. *Comput. Electr. Eng.* **2017**, *62*, 384–400. [CrossRef]
43. Pareek, N.K.; Patidar, V.; Sud, K.K. Image encryption using chaotic logistic map. *Image Vis. Comput.* **2006**, *24*, 926–934. [CrossRef]
44. Patidar, V.; Pareek, N.K.; Sud, K.K. A new substitution-diffusion based image cipher using chaotic standard and logistic maps. *Commun. Nonlinear Sci. Numer. Simul.* **2009**, *14*, 3056–3075. [CrossRef]
45. Murillo-Escobar, M.A.; Cruz-Hernández, C.; Abundiz-Pérez, F.; López-Gutiérrez, R.M.; Acosta Del Campo, O.R. A RGB image encryption algorithm based on total plain image characteristics and chaos. *Signal Process.* **2015**, *109*, 119–131. [CrossRef]
46. Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Barker, E.; Leigh, S.; Levenson, M.; Vangel, M.; Banks, D.; Heckert, A.; et al. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications—Special Pub. 800-22—Rev. 1*; NIST Spec. Publ. 800-22; Natl. Inst. Stand. Technol. (NIST): Gaithersburg, MD, USA, 2010.
47. Alvarez, G.; Li, S. Some basic cryptographic requirements for chaos-based cryptosystems. *Int. J. Bifurc. Chaos* **2006**, *16*, 2129–2151. [CrossRef]
48. Kahan, W. IEEE Standard 754 for Binary Floating-Point Arithmetic. *Lect. Notes Status IEEE* **1996**, *754*, 11.
49. Wang, L.; Cheng, H. Pseudo-Random Number Generator Based on Logistic Chaotic System. *Entropy* **2019**, *21*, 960. [CrossRef]
50. Behnia, S.; Akhavan, A.; Akhshani, A.; Samsudin, A. A novel dynamic model of pseudo random number generator. *J. Comput. Appl. Math.* **2011**, *235*, 3455–3463. [CrossRef]
51. Li-Jiang, Y.; Tian-Lun, C. Application of chaos in genetic algorithms. *Commun. Theor. Phys.* **2002**, *38*, 168. [CrossRef]
52. Guo, Z.; Cheng, B.; Ye, M.; Cao, B. Self-adaptive chaos differential evolution. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Berlin, Germany, 2006.

53. Khan, M.A.; Jeoti, V. On the enlargement of robust region of chaotic tent map for the use in key based substitution-box (S-Box). *J. Comput. Sci.* **2015**, *11*, 517–525. [CrossRef]
54. Simon, D. Biogeography-based optimization. *IEEE Trans. Evol. Comput.* **2008**, *12*, 702–713. [CrossRef]