Menemui Matematik
(Discovering Mathematics)

journal homepage: https://myjms.mohe.gov.my/index.php/dismath/

Analysis Of Modified Linear Congruential Generator for The Randomness Property

Balqis Mohd Zamulud[1] and Aniza Abd. Ghani[2*]
[1,2]*Department of Mathematics and Statistics, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor*

[1]balqismz24@gmail.com, [2]aniza@upm.edu.my
*Corresponding author

Received: 10 August 2024
Accepted: 1 November 2024

ABSTRACT

Linear Congruential Generators (LCGs) are fundamental tools in the generation of pseudo-random numbers, playing a crucial role in various computational applications. This study delves into the analysis and enhancement of LCG through the modification of utilizing inverse modular multiplication designed to address the limitations. This research paper is done to study the modified LCG by generating the sequence, determine the randomness property of modified LCG by utilizing the NIST Statistical Test Suite and comparing the randomness of the sequence generated by the standard LCG with the modified LCG using the empirical results obtained from the statistical tests. Investigation and analysis of randomness are discussed to show that the modified LCG is a better generator to generate the keystream sequence. Finding shows that the standard LCG sequences failed the randomness test while the modified LCG passed the test. Therefore, it can be concluded that the generated sequence using the modified LCG is more random than sequence generated using the standard LCG.

Keywords: Modified Linear Congruential Generator, Inverse modular multiplication, Pseudo-Random Number Generation, NIST Statistical Test

INTRODUCTION

A Random Number Generator (RNG) generates truly random numbers. True randomness requires gathering entropy from natural sources like atmospheric noise or quantum phenomena. Games like bingo, card games and lotteries rely on true randomness for fairness. Meanwhile, a Pseudo-Random Number Generator (PRNG) is an algorithm based on mathematical formulas. It is not truly random because it relies on an initial value called a seed, which determines the output. The output can be replicated by using the same seed, undermining the goal of randomness. However, with a carefully chosen seed, a PRNG can still generate numbers that appear random. It is crucial for the seed to be unpredictable to ensure that others cannot reproduce the PRNG's sequence and that the seed cannot be predicted from the generated numbers. Statistical tests are useful as it is a first step in analysing whether a generator is appropriate for a specific cryptographic application.

The linear congruential generator (LCG) was published by Thomson, 1958 and Rotenberg, 1960.

$$x_{n+1} \equiv (ax_n + c)(mod\ m) \quad (1)$$

m - modulus, $m > 0$,

a - multiplier, $0 < a < m$,
c - growth, $0 \leq c < m$,
x_0- starting value (seed), $0 \leq x_0 < m$

Modification for linear congruential method by utilizing the inverse element of multiplication were introduced by Irawadi, Prayanti, et al. (2022). The process initiates by employing the original LCG formula (1). The pivotal modification is then introduced by incorporating the inverse element into the modulo m of the LCG calculation. For each x_i in the LCG sequence, the generated number is decided based on the greatest common divisor (gcd). When gcd(x_i, m) = 1, signifying that x_i and m are coprime, the inverse element, x^{-1} is chosen as the number generated. Conversely, when gcd(x_i, m) ≠ 1, indicating a lack of coprimality, the value of x_i remains the same. This decision-making process aims to selectively leverage the inverse element to enhance the randomness and statistical properties of the LCG sequence.

The NIST test suite, developed by the National Institute of Standards and Technology, is a collection of statistical tests used to assess the quality and randomness of binary sequences generated by random or pseudo-random number generators. The NIST test suite comprises fifteen tests that have been proven to be useful in the examination and evaluation of binary sequences generated by random and pseudo-random number generators.

NUMERICAL EXPERIMENTS

The proposed LCG by Irawadi et al. (2022) utilizes inverse elements of modulo multiplication to generate more random numbers than the standard LCG. The proposed modifications aim to enhance the generator's randomness properties and address potential limitations associated with the standard LCG. The process initiates by employing the original LCG formula (1), which encompasses the standard parameters such as modulus (m), multiplier (a), increment (c), and the seed (x_0). The pivotal modification is then introduced by incorporating the inverse element into the modulo m of the LCG calculation. For each x_i in the LCG sequence, the generated number is decided based on the greatest common divisor (gcd). When gcd(x_i, m) = 1, signifying that x_i and m are coprime, the inverse element, x_i^{-1} is chosen as the number generated. Conversely, when gcd(x_i, m) ≠ 1, indicating a lack of coprimality, the value of x_i remains the same. This decision-making process aims to selectively leverage the inverse element to enhance the randomness and statistical properties of the LCG sequence. For testing, a = 71, c = 59 and $m = 2^{256}$ were chosen. The evaluation of the Linear Congruential Generator (LCG) parameters a = 71, $m = 2^{256}$, and c = 59 involves an examination of their fundamental properties. The LCG serves as a deterministic method for generating pseudo-random numbers. Then, the randomness of the sequence will be tested using The NIST test suite comprises Monobit Test, Frequency Test within a Block, Longest Run Test, Runs Test and Cumulative Sums (CUSUM) Test, Discrete Fourier Transform Test, Approximate Entropy Test and Serial Test.

For each applied test, a conclusion can be made that accepts or rejects the null hypothesis. The normal distribution or χ^2 is the reference distribution for the most of the NIST Statistical Test Suite. The reference distribution is typically used to transform an observed test statistic into a *P*-value since *P*-values are simpler to interpret. The *P*-value shows the likelihood that a perfect random number generator would have generated a less random sequence than the tested sequence. It is common to set α in a smaller value such as 0.01. Therefore, to conclude the random number generator is a good generator, the *P*-value must be greater or equal than 0.01

RESULT AND DISCUSSION

The LCGs uses different seeds to check the randomness properties. Comparison was done on both LCGs so that conclusion can be made where the modified LCG are better producing random number than the standard LCG. However, there are few restrictions and constraint in testing both generators as follows:

1. Length of bit streams: 1024
2. Choosing odd integer as the parameter since even integer will lead to same number generation as the standard LCG for the modified LCG.

To observe the randomness of the sequence for both generators, the NIST Statistical Test Suite were used where only eight statistical tests were selected due to limitation in bit stream generation. The results were categorized to three parts based on the seed value which are when the seed is prime, even and odd. A sample size of two for each category of seed were tested using the NIST test. The parameter chosen which are a = 71, c = 59 and $m = 2^{256}$ were kept constant and only substituting the initial value (seed), x_0 to start the sequence of random number. The Frequency Test consist of calculating the n^{th} partial sum for the values $x_i = \{-1, +1\}$; i.e., the sum of the first n values of x_i is represented as S_n.

If the Initial Value, x_0 is a Random Prime Number

Using Maple software, two random prime numbers were chosen as initial value, x_0 for the generation of random sequence of both generators which are 1303 and 28283839.

```
                FREQUENCY TEST
        -----------------------------------------
        COMPUTATIONAL INFORMATION:
        -----------------------------------------
        (a) The nth partial sum = 86
        (b) S_n/n               = 0.083984
        -----------------------------------------
FAILURE     p_value = 0.007199
```

Figure 1: *P*-value result from the Frequency Test by NIST Statistical Test if x_0 = 1303 using standard LCG

```
                FREQUENCY TEST
        -----------------------------------------
        COMPUTATIONAL INFORMATION:
        -----------------------------------------
        (a) The nth partial sum = 4
        (b) S_n/n               = 0.003906
        -----------------------------------------
SUCCESS     p_value = 0.900524
```

Figure 2: *P*-value result from the Frequency Test by NIST Statistical Test if x_0= 1303 using modified LCG

Table 1: Result from NIST Statistical Test if x_0 = 1303 using modified LCG

No.	Statistical Test	*P*-value	Proportion	Result
1	Frequency (Monobit)	0.900524	1.000	success
2	Frequency within a Block	0.715390	1.000	success
3	Runs	0.802207	1.000	success
4	Longest Run of Ones	1.000000	1.000	success
5	Discrete Fourier (Spectral)	0.194366	1.000	success
6	Serial	0.269397	1.000	success
7	Approximate Entropy	0.254922	1.000	success
8	Cumulative Sums (CUSUM)	0.378538	1.000	success

Based on Figure 1, generator using the standard LCG method with initial value of 1303 fails the Frequency Test. Note that, if it fails the Frequency Test, then is also fails another statistical test. Meanwhile, the random number generated using the modified LCG passed the Frequency Test which means the sequence proven to be random.

```
                FREQUENCY TEST
        ----------------------------------------
        COMPUTATIONAL INFORMATION:
        ----------------------------------------
        (a) The nth partial sum = 86
        (b) S_n/n               = 0.083984
        ----------------------------------------
FAILURE     p_value = 0.007199
```

Figure 3: *P*-value result from the Frequency Test by NIST Statistical Test if x_0 = 28283839 using standard LCG

```
                FREQUENCY TEST
        ----------------------------------------
        COMPUTATIONAL INFORMATION:
        ----------------------------------------
        (a) The nth partial sum = 42
        (b) S_n/n               = 0.041016
        ----------------------------------------
SUCCESS     p_value = 0.189351
```

Figure 4: *P*-value result from the Frequency Test by NIST Statistical Test if x_0 = 28283839 using modified LCG

Table 2: Result from NIST Statistical Test if x_0= 28283839 using modified LCG

No.	Statistical Test	*P*-value	Proportion	Result
1	Frequency (Monobit)	0.189351	1.000	success
2	Frequency within a Block	0.209932	1.000	success
3	Runs	0.893488	1.000	success
4	Longest Run of Ones	1.000000	1.000	success
5	Discrete Fourier (Spectral)	0.598041	1.000	success
6	Serial	0.601003	1.000	success
7	Approximate Entropy	0.495879	1.000	success
8	Cumulative Sums (CUSUM)	0.221981	1.000	success

Based on Figure 3, it is shown that the initial value of 28283839 using the standard LCG fails the Frequency Test, while using the modified LCG passes the Frequency Test. It is once again proven that random number generated using the modified LCG produces a good sequence of random number.

If the Initial Value, x_0 is a Random Even Number

In this section, two random even numbers were chosen as the initial value, x_0, for the generation of random sequences for both generators, which are 37570 and 89868.

```
                FREQUENCY TEST
      -----------------------------------------
      COMPUTATIONAL INFORMATION:
      -----------------------------------------
      (a) The nth partial sum = -96
      (b) S_n/n               = -0.093750
      -----------------------------------------
FAILURE     p_value = 0.002700
```

Figure 5: *P*-value result from the Frequency Test by NIST Statistical Test if x_0 = 37570 using standard LCG

```
                FREQUENCY TEST
      -----------------------------------------
      COMPUTATIONAL INFORMATION:
      -----------------------------------------
      (a) The nth partial sum = 40
      (b) S_n/n               = 0.039063
      -----------------------------------------
SUCCESS     p_value = 0.211300
```

Figure 6: *P*-value result from the Frequency Test by NIST Statistical Test if x_0= 37570 using modified LCG

Table 3: Result from NIST Statistical Test if x_0 = 37570 using modified LCG

No.	Statistical Test	*P*-value	Proportion	Result
1	Frequency (Monobit)	0.211300	1.000	success
2	Frequency within a Block	0.655643	1.000	success
3	Runs	0.265697	1.000	success
4	Longest Run of Ones	1.000000	1.000	success
5	Discrete Fourier (Spectral)	0.935354	1.000	success
6	Serial	0.578803	1.000	success
7	Approximate Entropy	0.548913	1.000	success
8	Cumulative Sums (CUSUM)	0.357948	1.000	success

According to Figure 5, the initial value of 37570, when generated through the standard Linear Congruential Generator (LCG), does not meet the criteria of the Frequency Test. Conversely, in Figure 6, when employing the modified LCG, the generated sequence successfully satisfies the Frequency Test. This reaffirms the notion that random numbers generated through the modified LCG exhibit a good random sequence.

```
                FREQUENCY TEST
        ---------------------------------------------
        COMPUTATIONAL INFORMATION:
        ---------------------------------------------
        (a) The nth partial sum = 114
        (b) S_n/n               = 0.111328
        ---------------------------------------------
FAILURE      p_value = 0.000367
```

Figure 7: *P*-value result from the Frequency Test by NIST Statistical Test if x_0 = 89868 using standard LCG

```
                FREQUENCY TEST
        ---------------------------------------------
        COMPUTATIONAL INFORMATION:
        ---------------------------------------------
        (a) The nth partial sum = -44
        (b) S_n/n               = -0.042969
        ---------------------------------------------
SUCCESS      p_value = 0.169131
```

Figure 8: *P*-value result from the Frequency Test by NIST Statistical Test if x_0 = 89868 using modified LCG

Table 4: Result from NIST Statistical Test if x_0 = 89868 using modified LCG

No.	Statistical Test	P-value	Proportion	Result
1	Frequency (Monobit)	0.169131	1.000	success
2	Frequency within a Block	0.962101	1.000	success
3	Runs	0.528969	1.000	success
4	Longest Run of Ones	1.000000	1.000	success
5	Discrete Fourier (Spectral)	0.776497	1.000	success
6	Serial	0.773031	1.000	success
7	Approximate Entropy	0.116442	1.000	success
8	Cumulative Sums (CUSUM)	0.078320	1.000	success

In accordance with the data presented in Figure 7, the initial value of even integer of 89868, when produced using the standard Linear Congruential Generator (LCG), fails to conform to the requirements stipulated by the Frequency Test. In contrast, as depicted in Figure 8, the utilization of the modified LCG results in a generated sequence that effectively fulfills the criteria outlined in the Frequency Test. This observation serves to underscore the proposition that random numbers generated via the modified LCG display a commendable adherence to a truly random sequence.

If the Initial Value, x_0 is a Random Odd Number

Using Maple software, two random odd number were chosen as initial value, x_0 for the generation of random sequence of both generators which are 4044145 and 21123143.

```
                FREQUENCY TEST
        ------------------------------------------
        COMPUTATIONAL INFORMATION:
        ------------------------------------------
        (a) The nth partial sum = 86
        (b) S_n/n               = 0.083984
        ------------------------------------------
FAILURE     p_value = 0.007199
```

Figure 9: *P*-value result from the Frequency Test by NIST Statistical Test if x_0 = 4044145 using standard LCG

```
                FREQUENCY TEST
        ------------------------------------------
        COMPUTATIONAL INFORMATION:
        ------------------------------------------
        (a) The nth partial sum = -22
        (b) S_n/n               = -0.021484
        ------------------------------------------
SUCCESS     p_value = 0.491768
```

Figure 10: *P*-value result from the Frequency Test by NIST Statistical Test if

x_0 = 4044145 using modified LCG

Table 5: Result from NIST Statistical Test if x_0 = 4044145 using modified LCG

No.	Statistical Test	P-value	Proportion	Result
1	Frequency (Monobit)	0.491768	1.000	success
2	Frequency within a Block	0.550051	1.000	success
3	Runs	0.281137	1.000	success
4	Longest Run of Ones	1.000000	1.000	success
5	Discrete Fourier (Spectral)	0.745603	1.000	success
6	Serial	0.858893	1.000	success
7	Approximate Entropy	0.493258	1.000	success
8	Cumulative Sums (CUSUM)	0.600895	1.000	success

From the result, it is shown that the generated sequence using the standard LCG with odd integer as initial value also fails the Frequency Test. Meanwhile, by utilizing the modified LCG for generation of sequence with odd integer as initial value, passes the Frequency Test. Thus, the generated sequence also passes the rest eight statistical tests.

```
                FREQUENCY TEST
      ----------------------------------------
      COMPUTATIONAL INFORMATION:
      ----------------------------------------
      (a) The nth partial sum = 86
      (b) S_n/n               = 0.083984
      ----------------------------------------
FAILURE     p_value = 0.007199
```

Figure 11: *P*-value result from the Frequency Test by NIST Statistical Test if x_0 = 21123143 using standard LCG

```
                FREQUENCY TEST
      ----------------------------------------
      COMPUTATIONAL INFORMATION:
      ----------------------------------------
      (a) The nth partial sum = 28
      (b) S_n/n               = 0.027344
      ----------------------------------------
SUCCESS     p_value = 0.381574
```

Figure 12: *P*-value result from the Frequency Test by NIST Statistical Test if x_0= 21123143 using modified LCG

Table 6: Result from NIST Statistical Test if x_0 = 21123143 using modified LCG

No.	Statistical Test	P-value	Proportion	Result
1	Frequency (Monobit)	0.381574	1.000	success
2	Frequency within a Block	0.849326	1.000	success
3	Runs	0.725336	1.000	success
4	Longest Run of Ones	1.000000	1.000	success
5	Discrete Fourier (Spectral)	0.465392	1.000	success
6	Serial	0.838181	1.000	success
7	Approximate Entropy	0.727226	1.000	success
8	Cumulative Sums (CUSUM)	0.629223	1.000	success

The outcomes demonstrate that the sequence generated through the standard Linear Congruential Generator (LCG), initialized with an odd integer, does not meet the criteria set by the Frequency Test. Conversely, when employing the modified LCG for sequence generation with an odd integer as the initial value, the generated sequence successfully satisfies the Frequency Test. Consequently, the generated sequence also successfully passes the remaining eight statistical tests. Based on both results, it is clear to conclude that the modified LCG shows better characteristics of random sequence than the standard LCG.

In accordance with all the results illustrated above, the *P*-value for Longest Run of Ones Test remain the same which is 1.00. This can be explained by the length of each block, *M* where usually the test code will be pre-set to accommodate three values for *M* in aligned with the following values of sequence length, *n*:

Table 7: Minimum value for *n* for each *M*

Minimum *n*	***M***
128	8
6272	128
750000	10^4

```
      LONGEST RUNS OF ONES TEST
---------------------------------------------
COMPUTATIONAL INFORMATION:
---------------------------------------------
(a) N (# of substrings)  = 0
(b) M (Substring Length) = 10000
(c) Chi^2                = -1.#IND00
---------------------------------------------
      F R E Q U E N C Y
---------------------------------------------
<=10  11  12  13  14  15 >=16 P-value  Assignment
   0   0   0   0   0   0    0 1.000000  SUCCESS
```

Figure 13: *P*-value result for the Longest Run of Ones Test by NIST Statistical Test if x_0 = 1303 using modified LCG

Meanwhile for the NIST Statistical Test Suite that was used did not follow the value that should have been used while testing the sequences for the Longest Run of Ones Test as it only set the value of M to 10^4 which only accommodate $n \geq 750000$ while the sequence length that was tested is $n = 1024$ which means the test should set $M = 8$ for the Longest Run of Ones Test. The value for the number of blocks, N which is selected in accordance with the value M is 0 and the test statistic used for the reference distribution is $\chi^2 = -1$. Therefore, as shown in Figure 13, the P-value for the Longest Run of Ones Test is invalid for all the sample tested previously. This does not dismiss other statistical tests result obtained in determining the randomness of the sequences.

CONCLUSION

In summary, this research explored a modified Linear Congruential Generator (LCG) proposed by Irawadi et al. (2022) for its ability to generate more random sequences compared to the standard LCG. Using Maple Software, sequences were generated for both LCGs, achieving the main objective. Statistical tests, including the NIST suite, confirmed the modified LCG's sequences were more random, as evidenced by a *P*-value exceeding 0.01 in the Frequency Test.

Results showed the modified LCG outperformed the standard LCG in randomness. Even with different initial seed characteristics, the modified LCG consistently passed all tests. Future research could focus on optimal LCG parameters for practical cryptography, selecting initial values, and testing larger size of binary sequences for the remaining NIST Statistical Tests.

REFERENCES

Bassham III, L. E., Rukhin, A. L., Soto, J., Nechvatal, J. R., Smid, M. E., Barker, E. B., Leigh, S. D., Levenson, M., Vangel, M., Banks, D. L., et al. (2010). *Sp 800-22 rev. 1a.* A Statistical Test Suite For Random And Pseudorandom Number Generators For Cryptographic Applications. National Institute of Standards & Technology.

Elizalde-Canales, F. A., Rivas-Cambero, I. D., Rebolledo-Herrera, L. F., and CamachoBello, C. J. (2019). Pseudo-random bit generator using chaotic seed for cryptographic algorithm in data protection of electric power consumption. *International Journal of Electrical and Computer Engineering*, **9(2)**:1399.

Fran¸cois, M., Grosges, T., Barchiesi, D., and Erra, R. (2014). Pseudo-random number generator based on mixing of three chaotic maps. *Communications in Nonlinear Science and Numerical Simulation*, **19(4)**:887–895.

Hegadi, R. and Patil, A. P. (2020). A statistical analysis on in-built pseudo random number generators using nist test suite. In *2020 5th International Conference on Computing, Communication and Security (ICCCS)*, 1–6. IEEE.

Irawadi, S., Prayanti, B. D. A., et al. (2022). Modify linear congruent generator algorithms using inverse elements of modulo multiplication for randomizing exams. In *2022 4th International Conference on Cybernetics and Intelligent System (ICORIS)*, 1–3. IEEE.

Kadir, R. and Maarof, M. A. (2009). A comparative statistical analysis of pseudorandom bit sequences. In *2009 Fifth International Conference on Information Assurance and Security*, volume **2**: 91–94. IEEE.

Katti, R. S. and Kavasseri, R. G. (2008). Secure pseudo-random bit sequence generation using coupled linear congruential generators. In *2008 IEEE International Symposium on Circuits and Systems*, 2929–2932. IEEE.

Katti, R. S., Kavasseri, R. G., and Sai, V. (2010). Pseudorandom bit generation using coupled congruential generators. *IEEE Transactions on Circuits and Systems II: Express Briefs*, **57(3)**:203–207.

Katti, R. S. and Srinivasan, S. K. (2009). Efficient hardware implementation of a new pseudo-random bit sequence generator. In *2009 IEEE International Symposium on Circuits and Systems*, 1393–1396. IEEE.

Kelsey, J., Schneier, B., Wagner, D., and Hall, C. (1998). Cryptanalytic attacks on pseudorandom number generators. In *Fast Software Encryption: 5th International Workshop, FSE'98 Paris, France, March 23–25, 1998 Proceedings 5*, 168–188. Springer.

Origines, D. V., Sison, A. M., and Medina, R. P. (2019). A novel pseudo-random number generator algorithm based on entropy source epoch timestamp. In *2019 International Conference on Information and Communications Technology (ICOIACT)*, 50–55. IEEE.

Rotenberg, A. (1960). A new pseudo-random number generator. *J. ACM*, **7**:75–77.

Thomson, W. (1958). A modified congruence method of generating pseudo-random numbers. *The Computer Journal*, **1(2)**:83–83.