*ORIGINAL PAPER*

# COMPARISON BETWEEN SOME ALGORITHMS TO GENERATE DATA OF $N(\mu, \sigma^2)$ BY SIMULATION

FADHIL ABDUL ABBAS AL-ABIDI[1], HAMEED MAGEED HAMEED[1]

*Manuscript received: 03.12.2019; Accepted paper: 05.04.2020;*

*Published online: 30.06.2020.*

***Abstract.*** *In this paper, the theoretical foundations of algorithms is used to generate observations of a random variable follow Standard Normal Distribution N(0,1) these observations then turned into a general normal distribution* ***N(μ,σ²)****,using simulation experiments for each algorithm, measuring the efficiency of the algorithm with several statistical criteria under different samples size (small, medium, large). Each experiment has been repeated 10,000 times. Results demonstrated that Matlab method was more efficient than the rest of algorithm methods. Also, it showed that the ratio was greater by 34% and it censuses less running time.*

***Keywords:*** *Box-Muller, Polar, JB-Test, Algorithms to generate N(0,1),Simulation.*

## 1. INTRODUCTION

A normal distribution is most important probability distributions (discrete and continuous). This was mainly because most methods of statistical analysis assumed normal distribution case models of observations, and their limits are all field R(-∞,∞). Besides, most other statistical distributions approach to normal distribution, especially when in sample size is increased, probability function can be written as the following formula:

$$f\left(x,\mu,\sigma^2\right)=\frac{1}{\sqrt{2\pi\sigma}}\cdot e^{-\left(\frac{x-\mu}{2\sigma^2}\right)^2};\quad -\infty<x,\mu<\infty,\quad \sigma>0 \tag{1}$$

and can be drawn graphical as shown in Fig. 1.

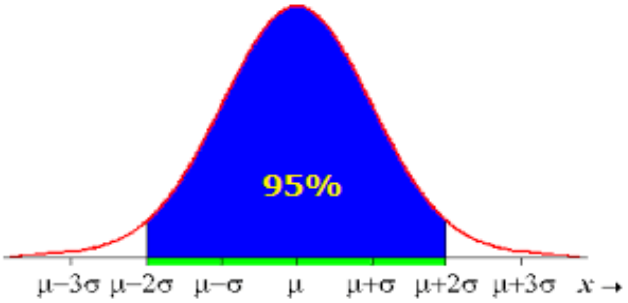


**Figure 1. curve graph of a normal distribution probability function.**

One of the first researchers who make comparison between these methods to generate a data distribution (GRNG) was Muller [1]. In this section, we review the most important

[1] Al Furat Al Awsat Technical University, IT Department, 54003 Najaf, Iraq. E-mail: abidy_fadhil@atu.edu.iq; abidy_fadhil@yahoo.com; inj.ham@atu.edu.iq.

researches that addressed generate spectators follow the normal distribution by more than one method. While there are some researches deal with only one method.

Kundu, D. used method for generating standard normal distribution of data transfer which follows lognormal distribution based on general exponential distribution function (GE). He compared with other methods, relying on comparative distance (K-S) and (P-value) [2].

Christian W. wrote algorithms for some methods of generating standard normal distribution [3]. His work does not involve any experimental work or a statistical comparison capability. Thomas, D. proved the importance of creating algorithms for generating random data follows a normal distribution [4], especially with the rapid and dramatic development in computer abilities. He mentioned several methods to generate observations of normal distribution. Variables were compared based on diagram views and how it recovered on mean values, with a focus on tail behavior for normal probability density function (pdf). Also, several criteria, including chi-square test and rapid delivery of computer simulation model, were used.

Rao et al. (2011) publicate researches on idea of generating data for a random variable follow a standard normal distribution N (0, 1) compared with the method proposed by them depends on Neural Networks, He tested randomize of data by ACF and Ljung-Box test. He used extracted proposed method did not differ from other methods results [5].

It would note that most of algorithms that have been taken up by research were depending on generating data for standard normal distribution N(0, 1) not general normal distribution N ($\mu$, $\sigma^2$), that does not depend on comparison tests between more Algorithms and simulation experiments, as well.

There are many algorithms to generate data for normal distribution **N($\mu$,$\sigma^2$)**, that require programming and in many steps; so research aims to determine the efficiency of algorithm with least steps and less time to implement resulting data, which represent the desired distribution best representation

## 2. MATERIALS AND METHODS

### A. *Algorithms to generate Data of Normal Distribution* [6-8]

More than one method is used to generate data distributed according to normal distribution. Most of these methods give N(0,1) and it can transformed to **N($\mu$,$\sigma^2$)** by:

$$y = mu + sd * N(0,1) \qquad (2)$$

However, recently Normal N($\mu$,$\sigma^2$) can be generated using differences software as Matlab or R-Package.

1. **Marsaglia's polar method -** The Marsaglia's polar method algorithm depends on the following steps:
   a. Generate two random numbers, say $U_1$ and $V_1$ from uniform distribution(-1,1) using any software, as Matlab application or R-package by command $U_1$=unifrnd(-1,1), $V_1$=unifrnd(-1,1).
   b. Return U=2$U_1$-1 ,V=2$V_1$-1 ; w=$U^2$+$V^2$ ; $z = -\sqrt{\frac{2Ln(w)}{w}}$
   c. If w<1 then y=w*z, we get y~N(0,1) … Go to a
   d. Otherwise go to a.

2. **Box-Muller method -** Algorithm of this method can be explained by the following steps:
   a. Generate two independent random numbers $U_1$ & $U_2$ form uniform distribution (0,1) using command $U_1$=unifrnd(0,1), $U_2$=unifrnd(0,1).
   b. Return $Z_1 = -\sqrt{2Ln(U_1)} \cdot cos(2\pi U_2)$ and $Z_2 = -\sqrt{2Ln(U_1)} \cdot sin(2\pi U_2)$. Then $Z_1$ and $Z_2$ each them N(0,1)

3. **Inverse CDF transformation method -** This method depends on making transformation to CDF for any distribution ; Let F(x)=pr(X≤ x) referees to distribution function of r.v. x and, F(x)=U and U is uniform(0,1); then its inverse is $x=F^{-1}(U)$ and x~N(0,1). Also we can now depend on software command, like norminv*(U)* or *icdf (U).*

4. **Matlab Codes Method -** In Matlab software there are exist two codes(commands):
   ***a.*** randn( ), which gives *N(0,1)* and can be written as :
   X=randn(n,p), where n is sample size and p is dimension.
   ***b.*** normrnd(mu,S,p,n) where :
   mu is mean of distribution and S is sigma(sd) of distribution
   this command gives **N(μ,σ$^2$)**.

5. **Generalized Exponential Distribution Method -** This method depends on algorithm given in the following steps:
   ***a.*** Generate U as a random number from uniform distribution (0,1).
   ***b.*** Generate $x = -Ln(1 - U^{0.0775})$
   ***c.*** To compute r.v. with distribution N(0,1) the following relation can be used:

$$z = \frac{Ln(x) - 1.084}{0.380}$$

**B. Statistical Criteria** [6, 9, 10]

Some criteria have been used by the researchers to compare between methods mentioned, as following:

1. **Bias square** - The bias for any parameter is calculated by

$$Bias = \left(\hat{\theta} - \theta_0\right)^2 \tag{3}$$

2. **Skewness and Kurtosis -** Skewness is a measure of symmetry on mean. If value near zero, the data are symmetry on mean. Yet, Kurtosis is a measure of how outlier-prone a distribution is the kurtosis of the normal distribution is 3 distributions that are more outlier-prone than the normal distribution have kurtosis greater than 3; distributions that are less outlier-prone have kurtosis less than 3.Skewness is computed utilizing as the following formula:

$$Sk = \frac{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^3}{nS^3} \tag{4}$$

while Kurtosis is computed from using:

$$Ku = \frac{\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^4}{nS^4} \tag{5}$$

In Matlab software, we can compute them (Sk,Ku) by the following formulas, respectively:

$$sk = skewness\left(x\right);\;\; Ku = kurtosis\left(x\right) \tag{6}$$

3. **Jarque-Bera Test (JB-Test) -** This test used to compare how asymmetry and Kurtosis measures outlier from values of Normal distribution. Hypothesis is written as:
   Ho: Data are following from symmetric normal distribution without outlier.
   and compute JB-TEST by following formula:

$$JB = \frac{n}{6}\left(Sk + \frac{1}{4}Ku\right) \tag{7}$$

JB has chi-square distribution, with degree freedom (2p), and reject H0 if JB>$\chi 2(2p,\alpha)$ or p-value <0.05 .

Also, in Matlab we can compute JB-test by the following formula:

$$\left[h \;\; p \;\; jbstats\right] = jbtest\left(x,\alpha\right) \tag{8}$$

where:

h: reject or accept indicator of Ho if:

$$\begin{cases} h = 1, reject\ Ho\ at\ the\ significant\ level \\ h = 0\ do\ not\ reject\ Ho\ (accept\ it) \end{cases}$$

P: is the smallest probability value to reject H0 and sometimes call (p-value) reject Ho if p<0.05 otherwise accept it.

Jbstats: returns the value of test statistic jbstats.

X: Data vector and α is probability error test (0.05, 0.01,…)

4. **Time speed -** A comparison among generation methods efficiency is done by run time for each method using following Matlab command: t0=cputime (), time=cputime ()-t0

## C. Experimental Side

The data source is an important stage in the procedure of scientific research. One of these sources is relying on building virtual models close to the real model for the purpose of drawing a sample depending on simulation method. This method has emerged in recent years as a result of development on computers Hardware and Software, making it easier for researchers to build models in all fields of engineering, applied sciences and Finical.

Dependent simulation Experiment carried out in this side on the algorithms, mentioned in section three, by generating a random variable follows a standard normal distribution N (0,1) then formula (2) is applied to get general Normal distribution N(μ,σ2). Default values for the parameters μ&σ were (2&3) respectively, this were used for different sample sizes (n=25, 50, 100) and replicated 10,000 times for each experiment.

Based on the comparison criteria's in mathematical formula mentioned in section four. All these accomplished by writing a matlab program which is described in the Appendix (B).

Overall results are tabulated for each algorithm described in Appendix (A). A summary of these results are shown in Tables 1-5.

## 3. RESULTS AND DISCUSSION

Simulation results are summarized and presented in Tables 1-3. Also, the success numbers for each algorithm and for each sample size are recorded in Table 4.

**Table 1.Algorithms generated for observation of sample size 25.**

| Method | | | skewness | kurtosis | JB_test | AIC | time |
|---|---|---|---|---|---|---|---|
| Polar | 1.992 (0.015)* | 2.999 (0.015) | -0.0354 | 2.000 | 0.441 | -61.910 | 24.72 |
| Box-Muller | 1.926 (0.016) | 2.970 (0.269) | -4.554 | 22.195 | <0.000 | 125.334 | 0.72 |
| Inverse CDF Transformation | 1.996 (0.014) | 2.997 (0.282) | 0.356 | 2.619 | 0.650 | -247.51 | 0.97 |
| Matlab function | 1.995 (0.015) | 2.994 (0.277) | 0.244 | 2.529 | 0.740 | -61.867 | 0.78 |
| Exponential-Normal | 1.990 (0.015) | 3.004 (0.289) | 0.349 | 2.62 | 0.630 | -61.933 | 0.87 |

**Note: the number (# ) in all tables represent bias-square*

**Table 2. Algorithms generated for observation of sample size 50 .**

| Method | | | skewness | kurtosis | JB_test | AIC | time |
|---|---|---|---|---|---|---|---|
| Polar | 1.996 (0.005) | 2.995 (0.068) | -0.275 | 2.190 | 0.226 | -124.784 | 48.3 |
| Box-Muller | 2.001 (0.004) | 3.008 (0.07) | 0.448 | 2.956 | 0.311 | 251.952 | 0.59 |
| Inverse CDF Transformation | 1.996 (0.015) | 2.997 (0.282) | 0.356 | 2.619 | 0.650 | -247.51 | 0.78 |
| Matlab function | 1.995 (0.004) | 3.005 (0.066) | 0.349 | 3.491 | 0.336 | -124.879 | 0.52 |
| Exponential-Normal | 1.988 (0.004) | 3.005 (0.07) | 0.115 | 2.597 | 0.786 | -124.929 | 0.78 |

**Table 3. Algorithms generated for observation of sample size 100.**

| Sample size | Time spent of each algorithm | | | | |
|---|---|---|---|---|---|
| | Polar | Box_Muller | Invers_CD Transformation | Matlab code | Exponential-Normal |
| 25 | 24.72 | 0.72 | 0.97 | 0.70 | 0.87 |
| 50 | 48.3 | 0.59 | 0.78 | 0.52 | 0.78 |
| 100 | 95.54 | 0.63 | 0.86 | 0.58 | 0.86 |
| Time Average | 56.19 | 0.65 | 0.87 | **0.60** | 0.84 |

**Table 4. Success number for each algorithms and sample size**

| Method | | | skewness | kurtosis | JB_test | AIC | time |
|---|---|---|---|---|---|---|---|
| Polar | 1.992 (0.001) | 2.999 (0.017) | 0.039 | 2.502 | 0.548 | -250.710 | 95.54 |
| Box-Muller | 2.007 (0.008) | 3.004 (0.016) | -0.181 | 2.624 | 0.515 | 503.785 | 0.63 |
| Inverse CDF Transformation | 1.992 (0.009) | 3.001 (0.019) | -0.024 | 2.835 | 0.940 | -107.44 | 0.86 |
| Matlab function | 1.995 (0.009) | 3.001 (0.017) | 0.022 | 2.868 | 0.963 | -250.709 | 0.58 |
| Exponential-Normal | 1.991 (0.009) | 3.007 (0.017) | -0.025 | 2.831 | 0.935 | -250.953 | 0.86 |

It is demonstrated that generation of normal distribution using Matlab function Normal distribution N(μ,σ2) achieved successfully 7 times out of 21 using any other statistical standard which means it surpassed by 34% Comparing to other algorithms. The average time spent on the implementation of Matlab algorithm was 0.60 second as shown in Table 5.

**Table 5. Average of time spent according to algorithms and sample size.**

| **Algorithm** | **Sample size** | | | **# of success in method** | **Ratio(#/21)** |
|---|---|---|---|---|---|
| | 25 | 50 | 100 | | |
| **Polar** | 2 | 1 | 1 | 4 | 0.19 |
| **Box-Muller** | - | 2 | 1 | 3 | 0.14 |
| **Inverse CDF Transformation** | 3 | 1 | - | 4 | 0.19 |
| **Matlab function** | 2 | 1 | 4 | 7 | **0.34** |
| **Exponential-Normal** | - | 2 | 1 | 3 | 0.14 |
| **#of success in sample** | 7 | 7 | 7 | **21** | |

Although, both Polar algorithm and Invers CDF algorithm have unequal success number, 4 times, with percentage 19%, the first was better because it spends less time to execute the algorithm in software, which was 0.65 second, the same results found for both Box-Muller and Exponential-Normal algorithms which were equal to 14% but Box-Muller algorithm was better because the time it takes for execution was 0.65 second.

## 4. CONCLUSION

Some conclusions can be extracted from this work, firstly the best algorithm method to generate data follow General Normal distribution $N(\mu,\sigma^2)$ is relying on MATLAB function based on statistical benchmarks as well as its execution time, also. Secondly convergence in the results for most of the methods used in this research or by other researchers showed that it fits generate N (0,1).

## REFERENCES

[1] Muller, M.E., *J. ACM*, **6**(3), 376, 1959.
[2] Kundu, D., Gupta, R.D., Manglick, A., *J. Modern Appl. Statist. Methods*, **5**, 266, 2006.
[3] Christian, W., *Hand-book on statistical distributions for experimentalists*, Particle Physics Group Fysikum, University of Stockholm, 2007.
[4] Thomas, D., Luk. W., Leong, P., Villasenor, J., *J. ACM Comput. Surv*., **39**(4), 11, 2007.
[5] Rao, K.R., Boiroju, N.K., Reddy, M.K., *Indian J. Sci. Res.*, **2**(4), 83, 2011.
[6] Hothorn, T., Everit, B., *A handbook of statistical Analysis using R*, 3rd Ed., CRC Press, New York, 2014.
[7] Martinez W., Angle, R. L., *Computational Statistics Handbook with Matlab*, 2nd Ed., CRC Press, New York, 2008.
[8] Seydel, R.U., Generating Random Numbers with Specified Distributions. In: *Tools for Computational Finance*, Universitext. Springer, London, 2012.
[9] Domanski, C., *Acta Univ. Lodziensis - Folia Oeconomica*, **235**, 75, 2010.
[10] Jarque, C.M., Bera, A.K., *Int. Statist. Rev.*, **55**(2), 163, 1987.