Comparative Analysis of Normal Variate Generators: A Practitioner's Perspective

Sayo Ayodeji[*]

Abstract

The most often used non-uniform distribution in applications involving simulations is the Gaussian distribution, popularly referred to as the Normal distribution. This study examines performances of different Gaussian variate generators over small, moderate and large samples with respect to statistical accuracy of the parameter estimates produced and computational complexity. Results showed that in statistical accuracy, the Marsaglia-Bray's algorithm performed best in small, moderate and large samples with a maximum absolute error of 1.08; while in computational efficiency, the Box-Muller's algorithm took a bit longer to compute the normal variate Z compared to the other algorithms considered.

Keywords: normal distribution, data simulation, algorithm

1 Introduction

Simulation of data is an integral component of data analysis methodologies. As real datasets that satisfy all the required assumptions for a particular study are usually hard to obtain, synthetic datasets are employed in simulation studies to verify theoretical results, large sample properties of statistical methods, estimators, and test statistics. The most often used non-uniform distribution in simulation applications is the Normal (Gaussian) family.

Definition 1 *A continuous random variable X has the normal distribution (or is normally distributed) with mean μ and variance σ^2 if for $x \in X$ the probability density function $f(x)$ is written*

[*] Department of Mathematics, Obafemi Awolowo University, 220005, Ile-Ife, Nigeria. E-mail: idowu.sayo@yahoo.com

as

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} exp\Bigg(- \frac{1}{2}\bigg(\frac{x-\mu}{\sigma}\bigg)^2\Bigg), \qquad -\infty < x < \infty \tag{1}$$

This implies that X is normally distributed with mean μ and variance σ^2; $X \sim N(\mu, \sigma^2)$.

It is easy to establish that the moment generating function of $X \sim N(\mu, \sigma^2)$ is

$$M_x(t) = exp\Bigg(\mu t + \frac{1}{2}\sigma^2 t^2\Bigg). \tag{2}$$

So that the first four moments are given by

$$\begin{aligned}
E[X] &= \frac{dM_x(t)}{dt}\Bigg|_{t=0} = M_x^{'}(0) = \mu, & (3)\\
E\Big[X^2\Big] &= \frac{d^2M_x(t)}{dt^2}\Bigg|_{t=0} = M_x^{''}(0) = \mu^2 + \sigma^2, & (4)\\
E\Big[X^3\Big] &= \frac{d^3M_x(t)}{dt^3}\Bigg|_{t=0} = M_x^{'''}(0) = \mu^3 + 3\sigma^2\mu, \quad and & (5)\\
E\Big[X^4\Big] &= \frac{d^4M_x(t)}{dt^4}\Bigg|_{t=0} = M_x^{iv}(0) = \mu^4 + 6\mu^2\sigma^2 + 3\sigma^4. & (6)
\end{aligned}$$

Definition 2 *A continuous random variable Z has the standard normal distribution if for $z \in Z$ the probability density function $f(z)$ is given as*

$$f(z) = \frac{1}{\sqrt{2\pi}} exp\left(-\frac{1}{2}z^2\right), \qquad -\infty < z < \infty. \tag{7}$$

Variable $z = \frac{x-\mu}{\sigma} \sim N(0,1)$.

Univariate normality can be assessed by examining the skewness (β_1) and the kurtosis (β_2) coefficients. Skewness is the third standardized moment and is defined by

$$\beta_1 = \frac{\mu_3^{'}}{\Big(\mu_2^{'}\Big)^{\frac{3}{2}}}. \tag{8}$$

Kurtosis is the fourth standardized moment and is defined by

$$\beta_2 = \frac{\mu_4^{'}}{\left(\mu_2^{'}\right)^2}; \tag{9}$$

where μ_k is the kth moment about the mean.

For a normally distributed variable, $\beta_1 = 0$ and $\beta_2 = 3$. Typically, researchers adjust the kurtosis coefficient by subtracting 3, so that a normally distributed variable has kurtosis coefficient equal to zero. (Please, note that some texts refer to kurtosis after adjustment as excess kurtosis.) A non-zero kurtosis coefficient indicates a nonnormal distribution. A leptokurtic distribution, $\beta_2 > 3$, is taller and thinner than a normal distribution; it is denoted with a positive kurtosis coefficient. A platykurtic distribution, $\beta_2 < 3$, is shorter and wider than a normal distribution; accordingly, it corresponds to a negative kurtosis (Henson [1]). Similarly, a skewness coefficient of zero denotes a perfectly symmetrical distribution; $\beta_1 > 0$ indicates a positive skew while $\beta_1 < 0$ indicates a negative skew.

Algorithms to generate normal random numbers can be broadly classified into four categories; they include the acceptance-rejection, inversion based, Box-Muller and Wallace approaches. To select an appropriate algorithm for use may be fuzzy as compromises must be made among some basic desirable properties such as (i) the tolerance level of the variation between the parameter and the estimated values; (ii) the mathematical simplicity; and (iii) the computational efficiency of an algorithm. For instance, since no closed form expression is available for the normal cdf, the statistical accuracy of inversion based methods depends heavily on how closely the normal cumulative distribution function (cdf) can be approximated; however, these methods may exploit the symmetry of $F^{-1}(\cdot)$ to generate suitable normal variate without the need to implement trigonometric functions. On the other hand, due to advances in technology, trigonometric functions, such as sine and cosine, can be easily implemented relatively and accurately. Thus one may choose the Box-Muller's algorithm and implement efficiently, along with trigonometric functions, to increase the output sample rate. Further, due to conditional if-then-else assignment instructions, the output rate for the acceptance-rejection method is not constant. Wallace method generates normal variates by applying linear transformations to a pool of Gaussian samples. However, due to the inherent feedback in the method, unwanted correlations can occur between successive transformations (Lee [2]). Thus in a comparative analysis, no single simulation method may be declared the winner with respect to all desirable properties.

Some other forms of classification may be achieved by considering the mathematical simplicity of the algorithms. The derivations for most recent algorithms are mathematically elegant but complicated, in a practitioner's view. For instance, Ziggurat's algorithm, which uses the acceptance-rejection approach, has been adjugded the fastest in the literature (Doornik [3]). However, its derivations are complicated (Marsaglia and Tsang [4]). This explains why old methods such as the Box-Muller's and Marsaglia-Bray's algorithms remain practitioners' favourite despite their flaws. We also note that these two algorithms are usually employed in common statistical packages like, SAS, IMSL, SPSS, S-Plus, etc.

The present study designs experiment to examine the performance of commonly-used guassian variate simulators over varying sample sizes. Previously in the literature, (see Roy [5]) authors would generate a large sample size (say 1000) then compare the performance of normal variate simulators based on some metrics. However, this study acknowledges that in practice, simulation of data are required in various sizes - small, moderate and large samples. Thus a comparative study as described earlier does not provide detailed information on which of these guassian variate generators would provide good random numbers viz-a-viz the three cardinal properties earlier discussed. In this light, the present study examines the statistical accuracy of the parameter estimates generated by these simulators with respect to (i) sample sizes; and (ii) computational complexity.

One major contribution of the study is that it provides detailed information especially to practitioners on the performance level of normal variate generators for varying sample sizes. Such information is very vital as applications in the literature include samples in small, moderate and large sizes. Further, it updates the literature as the study reviews and compares old and recent commonly-used generators. That is, the Box-Muller's [6], Marsaglia-Bray's [7] and the recent Rao *et. al*'s [8] methods. And, lastly, it compares these methods with respect to their computational complexities. Thus a practitioner has access to comprehensive information on these generators and may easily choose the most suitable for application purposes.

The rest of the paper is organized as follows: Section 2 reviews the various methods for simulating normal variate. Section 3 describes the methodology of commonly-used algorithms. The next section presents the numerical experiment and observations therefrom. Section 5 contains the conclusion.

2 Literature Review

Algorithms for generating normal variate are mostly based on transformations from the uniform distribution. The first and oldest method uses the central limit theorem (CLT) on a uniformly distributed random variable U to provide a close approximation of normal random variate. That is, for a uniform random variable U in the range $(0,1)$, the mean and variance are given by 1 and $\frac{1}{12}$, respectively. Hence, the standard normal random variable Z may be approximated as (Wetheril [9])

$$Z = \frac{\displaystyle\sum_{i=1}^{n} U_i - \frac{n}{2}}{\sqrt{\frac{n}{12}}}. \tag{10}$$

for a sufficiently large sample size n. Further, choosing $n = 6$ leads to the simple form, $Z = \displaystyle\sum_{i=1}^{n} U_i - 6$. A major limitation of this method is that it requires n uniformly distributed variates to compute one normal variate! Hence, with the introduction of more computationally efficient methods, it has received less attention in recent times.

The Box-Muller algorithm was developed by Box and Muller [6]. The inputs to this algorithm are two independent uniformly-distributed random numbers U_1 and U_2. The outputs are two independent samples Z_1 and Z_2 with $N(0,1)$ distribution. The algorithm involves taking the product of the logarithm and the trigonometric functions. Note that this method requires only two random numbers from the uniform distribution as opposed to the central limit theorem method which requires n. However, a major drawback identified then was the computation of sine and cosine functions which used to be computationally expensive. Implementations of logarithm, square root, and trigonometric functions have been investigated extensively over the last three decades

(Ercegovac and Lang [10]).

Another class of generators, the acceptance-rejection, began with the Polar method of Marsaglia and Bray [4]. In contrast to the Box-Muler's, it avoided the computation of sine and cosine, thus it is more computationally efficient. However, due to conditional if-then-else statement involved, the output rate of this method is not constant. A notable member of this class is the Ziggurat's [4] algorithm.

The inversion based method transforms uniform random variate $U \in (0,1)$ into normal variate Z by approximating the inverse of the normal cummulative distribution function (CDF) as $Z = F^{-1}(U)$. Since there is no closed-form expression for $F^{-1}(\cdot)$, several approximations have been employed (Odeh and Evans [11]). The most recent of course is Rao *et.al* [8] who employed the logistic approximation of normal CDF given in Bowling *et al.* [12]. Bowling's attempt represent a very simple and the closest [12] approximation to the normal cdf, with the maximum error of less than 0.014% (See Figure 1).

Wallace [13] deviated from common practices: Without transforming uniform variates, it generated a sequence of standard normal variates by by applying linear transformations to a pool of normal samples. Lee [2] noted that owing to the inherent feedback in this method, unwanted correlations may occur between successive transformations using recurrence equation. A comprehensive list of various normal variate generating algorithms can be found in Johnson *et.al* [14].

3 Algorithms

This section presents algorithms for the commonly-used normal variate generators, namely, the Box-Muller's, Marsaglia-Bray's and Rao *et.al*'s algorithms. We note that a random variable from the standard normal distribution, $N(0,1)$ can easily be transformed so as to have $N(\mu, \sigma^2)$ distribution. Hence, the following discussion focusses on generating variates from $N(0,1)$.

3.1 The Box-Muller's Algorithm

Denote $U(0,1)$ the uniform distribution in the range $(0,1)$. The Box-Muller [6] algorithm proceeds as follows:

(i) generate two independent random numbers U_1 and U_2 from $U(0,1)$; and

(ii) return $Z = \sqrt{-2\log(U_1)\cos(2\pi U_2)}$ and $Z = \sqrt{-2\log(U_1)\sin(2\pi U_2)}$.

Of course, $Z \sim N(0,1)$.

3.2 The Marsaglia-Bray's Algorithm

The Marsaglia-Bray's [7] algorithm developed using the polar method also employs two uniformly distributed variates U_i, $i = 1,2$; however it does not require trigonometric functions for implementation:

(i) generate two independent random numbers U_1 and U_2 from $U(0,1)$; and

(ii) set $V_1 = 2U_1 - 1$, $V_2 = 2U_2 - 1$. $S = V_1^2 + V_2^2; \quad V_1, V_2 \sim U(-1,1)$;

(iii) if $S > 1$, go to step (i) otherwise, go to (iv);

(iv) return the independent standard normal variables $Z = \sqrt{\frac{-2\log(S)}{S}}V_1$ and $Z = \sqrt{\frac{-2\log(S)}{S}}V_2$.

3.3 Rao *et.al*'s Algorithm

Rao *et.al*'s [8] algorithm is based on inverse transform. It uses the logistic approximation of normal CDF given in Bowling *et al.* [12]: $F(z) \approx \frac{1}{1+e^{-1.702z}}$. The method improves on the earlier ones as it only requires a single random number U from $Uniform(0,1)$. The algorithm is described in what follows.

(i) generate U from $U(0,1)$; and

(ii) return $Z = \frac{-\log\left(\frac{1}{u}-1\right)}{1.702}$.

4 Numerical Experiment

A Normal distribution $N(\mu, \sigma^2)$ is completely defined by its parameters - mean (μ) and variance (σ^2). Further, univariate normality is usually assessed by examining the skewness (β_1) and kurtosis (β_2) coefficient values. Therefore, a measure of performance would naturally include the mean, variance, skewness and kurtosis obtained for each of the simulations under varying samples. Since the exact values of μ, σ^2, β_1 and β_2 for the normal distribution are known, that is, $\mu = 0$, $\sigma^2 = 1$, $\beta_1 = 0$ and $\beta_2 = 3$, the deviations of the parameter estimates (from theseexact values) obtained for each of the simulations can be computed and analyzed for differences.

For the three methods under consideration, gaussian random variates were generated using the algorithms described in section 3 for sample sizes (i) small - 10, 20, 30; (ii) moderate - 40, 50, 100; and (iii) large - 200, 500, 1000. Each trial is replicated 1000 times and the average values of the statistics $\hat{\mu}$, $\hat{\sigma}^2$, $\hat{\beta}_1$ and $\hat{\beta}_2$ recorded in Table 1. Parameter estimates were computed using Maple 12.

4.1 Observations from the Parameter Estimates

The absolute deviation (in percentage) of the estimates from the exact parameter values were displayed in **Figures 2** to **5**. We observe as follows:

(i) **Statistical accuracy:**

For μ, Marsaglia-Bray's algorithm reproduced $\mu = 0$ with the least error in small and large samples, while Box-Muller's and Rao *et.al*'s did better at moderate samples.

For σ^2, Rao *et.al*'s is the preferred choice in small, moderate and large samples followed by Marsaglia-Bray's, then Box-Muller's.

For β_1, the order of performance is Marsaglia-Bray's, Rao *et.al*'s and Marsaglia-Bray's in small, moderate and large samples, respectively.

For β_2, Marsaglia-Bray's consistently reproduced the exact value $\beta_2 = 3$ in all the sample sizes with maximum absolute error of 1.08. Rao *et.al*'s seemed to converge gradually to the true value in small samples however, it diverged in moderate and large samples with maximum absolute error of 1.24. Box-Muller's performed better than Rao *et.al*'s in moderate and large samples.

(ii) **Computational efficiency:**

While it took Maple relatively longer time to compute Z in Box-Muller's algorithm, the two remaining ones took the same length of time.

By and large, if accuracy and computational efficiency are the factors to consider, the Marsaglia-Bray's algorithm performed best in small, moderate and large samples.

A sample of data generated using this algorithm is presented in Figure 6.

5 Label of Tables and Figures

6 Conclusion

The performance of the normal variate generators were examined with respect to statistical accuracy and computational efficiency in small, moderate and large sample sizes. In particular, Box-Muller's, Marsaglia-Bray's and Rao *et.al*'s algorithms were compared. The study concluded that, by and large, the Marsaglia-Bray's algorithm performed better than the other two algorithms with absolute maximum errors of 0.12, 0.50, 0.38 and 1.08 for mean, variance, skewness and kurtosis, respectively. In addition, Marsaglia-Bray's and Rao *et.al*'s are a bit more computationally efficient than the Box-Muller's.

References

[1] Henson, R. (1999). Multivariate normality: What Is It and How Is It Assessed? In B. Thompson (Ed.), *Advances in Social Science Methodology,* 5, 193-211. Stamford, CT: JAI Press.

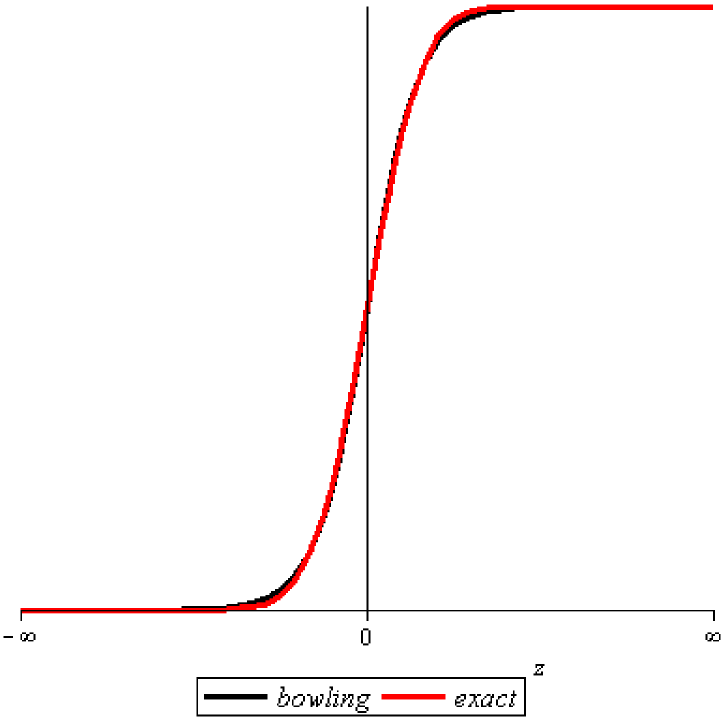

Figure 1: Bowling *et.al*'s Logistic Approximation to the Normal CDF

[2] Lee, D.,Luk, W., Villasenor, J. and Cheung, Y. (2005). A hardware Gaussian noise generator using theWallace method. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst,* 13(8), 911 - 920.

[3] Doornik, J. (2005). An improved ziggurat method to generate normal random samples. Working Paper, Department of Economics, University of Oxford.

[4] Marsaglia, G. and Tsang, W (2000). The ziggurat method for generating random variables. *Journal of Statistical Software,* 5, 1 - 7.

[5] Roy, R. (2002). Comparison of Different Techniques to Generate Normal Random Variables. Stochastic Signals and Systems (ECE 330:541), Rutgers, The State University of New Jersey.

[6] Box, G and Muller, M. (1958). A Note on the Generation of Random Normal Variates. *Annals of Mathematical Statistics,* 29, 610 - 611.

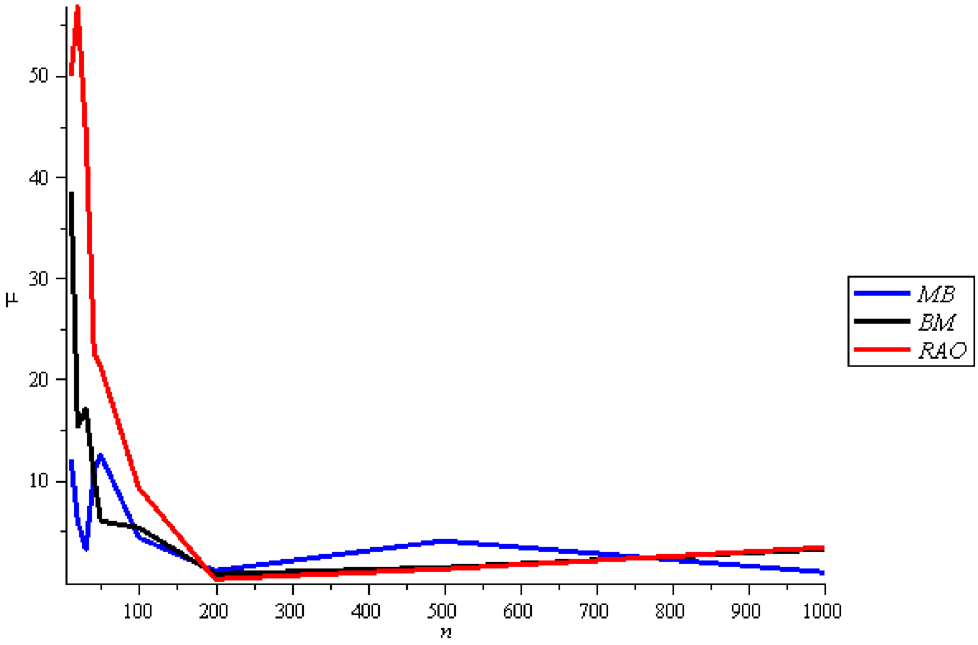

Figure 2: Plot of Absolute Deviation (in %) of $\hat{\mu}$ from μ by Algorithms

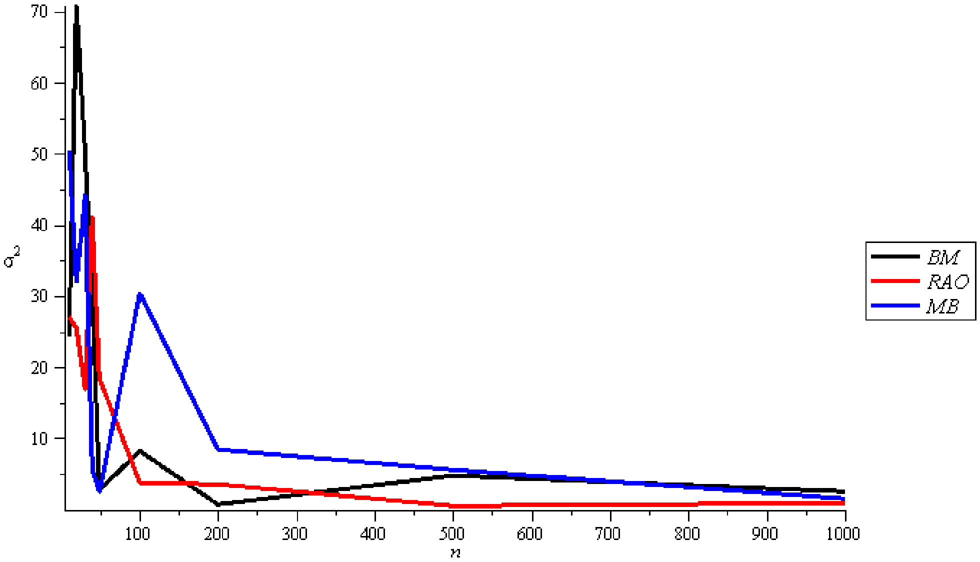

Figure 3: Plot of Absolute Deviation (in %) of $\hat{\sigma}^2$ from σ^2 by Algorithms

[7] Marsaglia, G. and Bray, T. (1964). A convenient method for generating normal variables. *SIAM*, 6, 260 - 264.

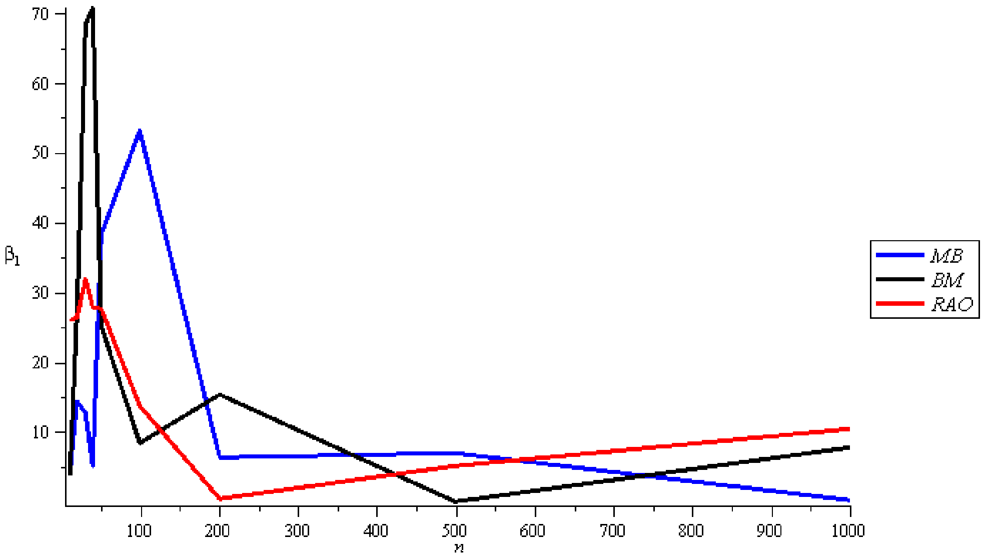

Figure 4: Plot of Absolute Deviation (in %) of $\hat{\beta}_1$ from β_1 by Algorithms

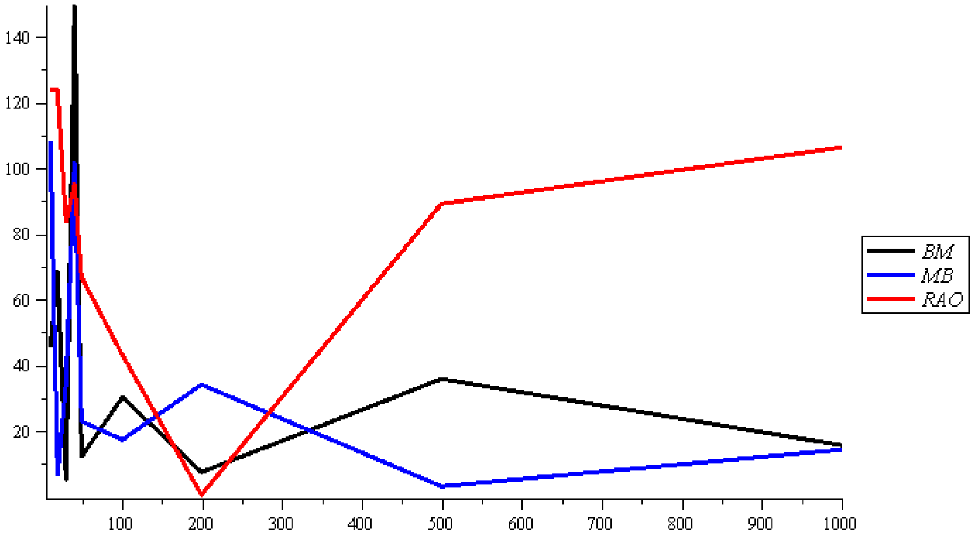

Figure 5: Plot of Absolute Deviation (in %) of $\hat{\beta}_2$ from β_2 by Algorithms

[8] Rao, K., Boiroju, N. and Reddy, M. (2011). Generation of Standard Normal Random Variates. *Indian J. Sci. Res.*, 2(4), 83- 85.

[9] Wetherill, G. (1965). An Approximation to the Inverse Normal Function Suitable for the Gen-

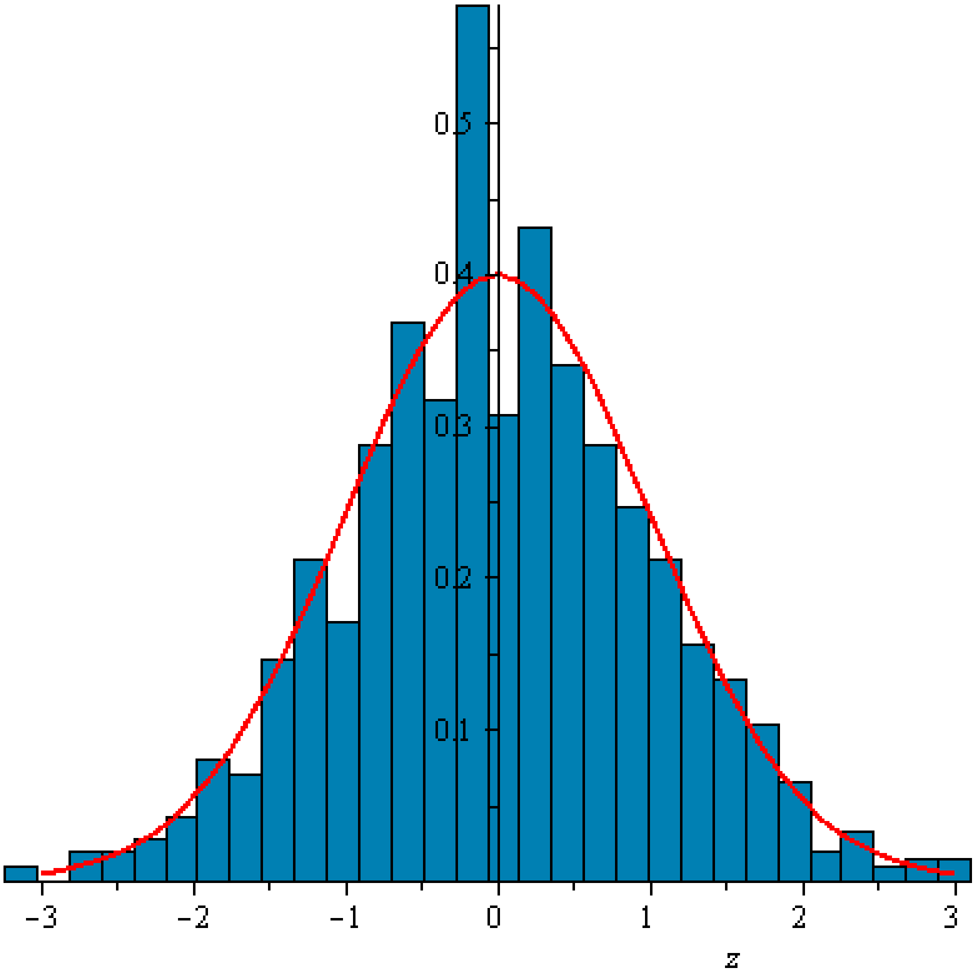

Figure 6: A histogram plot of $Z \sim N(0,1)$ and $n = 1000$ with a corresponding Normal curve superimposed on it.

eration of Random Normal Deviates on Electronic Computers. *Journal of the Royal Statistical Society, Series C,* 14, 201 - 205.

[10] Ercegovac, M. and Lang, T. (2004). Digital Arithmetic. San Mateo, CA: Morgan Kaufmann.

[11] Odeh, R, Evans, J. (1974). The percentage points of the normal distribution. *Appl. Stat.,* 23, 96 - 97.

[12] Bowling, X, Khasawneh, M., Kaewkuecool, S. and Cho, B. (2009). A Logistic Approximation to the Cumulative Normal Distribution. *Journal of Industrial Engineering and Management,*

2(1), 114 - 127.

[13] Wallace, C. (1996). Fast pseudorandom generators for normal and exponential variates. *ACM Transactions on Mathematical Software,* 22, 119 - 127.

[14] Johnson, N, Kotz, S. and Balakrishnan, N. (1995). *Continuous univariate distributions* (2nd ed.). Wiley, New York.

[15] Gibbons, J. (2003). *Nonparametric Statistical Inference* (4th ed.). Marcel Dekker, Inc. New York.

Table 1: Comparison of Parameter Estimates by Algorithms and Sample Sizes

Parameter	**n**	**BM**	**MB**	**Rao**
μ	10	0.3852679277	−0.1210553931	0.4994526626
	20	0.1530559464	0.05938518416	0.5685006210
	30	-0.1711577598	−0.03219852653	0.4423425123
	40	-0.1025890069	−0.1105125905	0.2241438738
	50	-0.06042986998	−0.1244057323	0.2116342935
	100	0.05341244793	0.04334166353	0.09259172339
	200	0.006810130982	−0.01134162105	−0.002962201220
	500	0.01461458568	−0.04042268704	−0.01316400225
	1000	-0.03155605161	−0.009887328423	−0.03362373709
σ^2	10	0.7583552986	0.4955298249	1.270017879
	20	0.2924112883	1.319749522	1.255646148
	30	0.4927864257	0.5588455569	1.169638420
	40	0.7029520909	0.9443556275	1.411535526
	50	0.9724703041	1.025347988	1.183864639
	100	0.9168788486	1.302479199	1.036933513
	200	1.006706325	1.084268791	1.036544876
	500	0.9527794149	0.9448396269	0.9943776765
	1000	1.024660083	1.015294830	1.009768230
β_1	10	−0.03638007471	0.03638007471	−0.2604077477
	20	0.2681373644	0.1442345186	−0.2633442454
	30	−0.6833597724	−0.1262205408	−0.3193734708
	40	−0.7084262298	−0.05183990151	−0.2783085868
	50	−0.2516039355	0.3822682571	−0.2756509750
	100	0.08443817482	−0.5312760633	−0.1379078623
	200	-0.1535864642	−0.06374827217	0.003636854938
	500	−0.0007671139	0.06904418311	0.05149050392
	1000	0.07745211045	−0.0015602177	0.1052810439
β_2	10	2.546301583	1.917140303	1.757473681
	20	2.315706749	3.065605848	1.759005157
	30	3.053434416	2.655525419	2.163981726
	40	4.495485168	1.983169263	2.046479304
	50	3.123586229	2.769640192	2.330969832
	100	2.696534384	3.175074447	2.568168816
	200	2.925932477	2.656824734	3.007444693
	500	2.642092057	3.032191831	3.892802974
	1000	2.843932783	3.143422786	4.063858257

BM: Box-Muller, MB: Marsaglia-Bray, Rao: Rao *et.al*