Dhaka Univ. J. Sci. 68(2): 117- 120, 2020 (July)

On Effectiveness of Decomposition Methods to Generate Multivariate Normal Variates: A Comparative Study

Syeda Fateha Akter and Anamul Haque Sajib

Department of Statistics, Dhaka University, Dhaka-1000, Dhaka, Bangladesh

(Received : 4 November 2019; Accepted : 4 October 2020)

Abstract

The multivariate normal density (MVN) is considered to be the underlying distribution of many observed samples in statistics for modelling purpose. Therefore, simulating sample from the MVN is required to verify the efficiency of the fitted model. Decomposition based approach is currently being used to simulate sample from MVN whose building block is Cholesky or eigen decomposition. Unfortunately, there is no concrete study in the literature so far regarding the efficient decomposition technique between these two[1]. In this paper, an attempt is made to determine the efficient decomposition technique between these two in the context of MVN generation through an extensive simulation study. From our simulation study, it is observed that in general the Cholesky decomposition is numerically faster than the eigen decomposition.

Keywords: Efficient decomposition methods, decomposition based MVN generation, Cholesky and eigen decomposition.

I. Introduction

Generating sample from MVN is required to serve many purposes in different fields. Most importantly, especially in Bayesian statistics simulating sample from MVN is required frequently. For example, suppose we have observed sample $\boldsymbol{X} = (x_{i1},\ x_{i2}, \ldots, x_{id}), i = 1, 2, \ldots, n$, from d dimensional normal density with mean vector, $\boldsymbol{\mu} = \boldsymbol{\mu_0}$ and variance covariance matrix, $\boldsymbol{\Sigma} = \boldsymbol{\Sigma_0}$, and we wish to estimate $\boldsymbol{\mu_0}$ and $\boldsymbol{\Sigma_0}$ in Bayesian approach. For simplicity, we consider $\boldsymbol{\Sigma_0}$, is known here and we need to estimate $\boldsymbol{\mu_0}$ only. Choosing d dimensional normal density with mean vector a_0 and variance covariance matrix B_0 (which are known) as a prior density for unknown mean vector $\boldsymbol{\mu}$ yields a posterior distribution of $\boldsymbol{\mu}$ (likelihood $\times$ prior distribution of $\boldsymbol{\mu}$), which is also d dimensional normal density. The parameters of the posterior density, d dimensional normal density, are the function of $\boldsymbol{X}$, a_0 and B_0. Point estimate of $\boldsymbol{\mu_0}$ can be obtained by taking mean of the generated sample (N draws) from the posterior density (d dimensional normal density).

Several statistical packages like *R*, *SAS*, *Stata* and *SPSS* offer generating sample from MVN through different functions. For example, *R* offers generating sample from MVN through 'mvrnorm' package under MASS library. The building block of all the statistical packages for generating sample from MVN is the decomposition method which uses either (i) eigen decomposition or (ii) Cholesky decomposition. Modern computing facility makes this task very simple, and using the state of art computing facility it is possible to generate 1 million draws from MVN even less than in one minute time. However, when this task is required in conjunction with other tasks then altogether computing time of the whole process could be an issue. For example, for the above problem if $\boldsymbol{\Sigma_0}$ also needs to be estimated along with $\boldsymbol{\mu_0}$, sampling from the two full conditional distributions of $\boldsymbol{\mu_0}$ and $\boldsymbol{\Sigma_0}$ is required to get the point estimates of $\boldsymbol{\mu_0}$ and $\boldsymbol{\Sigma_0}$ under MCMC framework. Therefore, simulating sample efficiently from a high-dimensional posterior density under MCMC framework demands efficient sampling from each of the full conditional density.

Efficient MVN generation under decomposition framework demands using an efficient decomposition method. In the context of efficient decomposition method, Ripley[1] pointed out that Cholseky decomposition might be faster than eigen decomposition but eigen decomposition is numerically stable. Unfortunately, there is no concrete study regarding these issues in the literature which motivated us to conduct this study. We limited our study only to investigate which decomposition method is faster and how much it is faster than other. Study on investigating numerical stability is still an open research problem, and we aim to investigate it in near future.

We organize the rest of the paper as follows: Section 2 introduces some important terminologies used in this paper. The Cholesky and eigen decomposition methods are discussed with illustrative examples in Section 3. Section 4 and Section 5 introduce the required simulation settings and numerical results obtained from simulation respectively. In the penultimate section, we present the discussions of these results which are followed by conclusion and future work presented in Section 6.

II. MVN and Related Terminologies

In this section, we introduce multivariate normal density and related terminologies required to generate from MVN. We here used the text book Johnson and Wichern[2] to prepare the following overview of related terminologies necessary to generate sample from MVN.

Multivariate Normal Distribution

The multivariate normal distribution is a generalization of univariate normal distribution to two or more variables. If the d-dimensional random vector, $\boldsymbol{X} = [X_1, X_2, X_3, \ldots, X_d]$ has mean vector, $\boldsymbol{\mu}$, and a symmetric positive definite covariance matrix, $\boldsymbol{\Sigma}$, then $\boldsymbol{X}$ is said to have a multivariate normal distribution with density function $f(\boldsymbol{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ as

*Author for correspondence. e-mail: *sajibstat@du.ac.bd*

$(2\pi)^{-d/2}|\boldsymbol{\Sigma}|^{-0.5}e^{-0.5[(\boldsymbol{X}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{X}-\boldsymbol{\mu})]}$,

with $-\infty < \mu_i < \infty$, $-\infty < X_i < \infty$ and $i = 1, 2\ ..., d$.

Symmetric Matrix

A square matrix M is said to be symmetric if it is equal to its transpose i.e., $M = M^T$. The entries of a symmetric matrix are symmetric with respect to the main diagonal. For example, matrix $M = \begin{pmatrix} 2 & 3 \\ 3 & 2 \end{pmatrix}$ is a symmetric matrix as $M = M^T$.

Positive Definite Matrix

A $n \times n$ symmetric real matrix M is said to be positive definite if $x^T Mx > 0$ for all non-zero x in $\mathbb{R}^n$. The properties of positive definite matrix are (i) $x^T Mx > 0$ for all $x \neq 0$, (ii) If a matrix M has full column rank (the n columns are linearly independent), then the product $x^T M$ is a positive definite matrix, (iii) A positive definite matrix M can be decomposed into a product of matrices, (iv) The eigenvalues of M are positive and (v) The determinants of principal submatrices (square submatrices beginning with row and column) are positive.

Eigen Decomposition of a Matrix

Let $\boldsymbol{\Sigma}$ be a d-dimensional positive definite square matrix and e_i's $(i = 1, 2, \ldots, n)$ are n linearly independent eigenvectors of $\boldsymbol{A}$ and λ_i's $(i = 1, 2, \ldots, n)$ are the corresponding eigenvalues. If $\boldsymbol{E}$ is the $n \times n$ matrix whose ith column is the eigenvector e_i and $\boldsymbol{\Lambda}$ is the diagonal matrix whose diagonal elements are the eigenvalues $(\boldsymbol{\Lambda_{ii}} = \lambda_i)$, then $\boldsymbol{\Sigma}$ can be factorized as, $\boldsymbol{\Sigma} = \boldsymbol{E\Lambda E^{-1}}$. Using the fundamental property of eigenvectors $\boldsymbol{\Sigma}v = \lambda v$ and substituting $\boldsymbol{E}$ by v in $\boldsymbol{\Sigma}v = \lambda v$, we can derive, $\boldsymbol{\Sigma} = \boldsymbol{E\Lambda E^{-1}}$.

III. Methods for Generating Multivariate Normal Random Variates via Decomposition

This section discusses decomposition based techniques, which are currently being used, for generating samples from MVN. In the current setting, usually two decompositions, namely eigen and Cholesky decompositions are used to generate samples from MVN.

Eigen Decomposition

Eigen decomposition is one of the most commonly used methods to generate multivariate normal random variates. To simulate $\boldsymbol{X} = [X_1, X_2, X_3, \ldots, X_d]$ from d-dimensional MVN with mean vector, $\boldsymbol{\mu}_{d\times 1}$, and covariance matrix, $\boldsymbol{\Sigma}_{d\times d}$, eigen decomposition decomposes $\boldsymbol{\Sigma} = \boldsymbol{E\Lambda E^{-1}}$, where $\boldsymbol{E} = [e_1, e_2, \ldots, e_d]$. The columns of $\boldsymbol{E}$ are the eigenvectors of $\boldsymbol{\Sigma}$ and $\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2, \ldots, \lambda_d)$ contains the corresponding d eigenvalues. Here '$diag$' means diagonal matrix. As $\boldsymbol{\Sigma}$ is symmetric, the eigen vectors are orthogonal $(\boldsymbol{E^T E} = \boldsymbol{I})$ and $\boldsymbol{E}$ is an orthonormal $(\boldsymbol{E^T} = \boldsymbol{E^{-1}})$ matrix, we have $\boldsymbol{\Sigma} = \boldsymbol{E\Lambda E^T}$.

Without loss of generality, assume that $\boldsymbol{X}$ is a zero mean vector with $\boldsymbol{\mu} = [0, 0, \ldots, 0]^T$ and define, $\widetilde{\boldsymbol{X}} = \boldsymbol{E^T X}$. The covariance of $\widetilde{\boldsymbol{X}}$ is $\mathbb{E}[\widetilde{\boldsymbol{X}}\widetilde{\boldsymbol{X}}^T] = \mathbb{E}[\boldsymbol{E^T XX^T E}] = \boldsymbol{E^T}\mathbb{E}[\boldsymbol{XX^T}]\boldsymbol{E} = \boldsymbol{E^T\Sigma E} = \boldsymbol{E^T E\Lambda E^T E} = \boldsymbol{\Lambda}$. The components of $\widetilde{\boldsymbol{X}}$ will be uncorrelated with each other since the covariance of $\widetilde{\boldsymbol{X}}$ is a diagonal matrix, $\boldsymbol{\Lambda}$. Bishop[3] showed that $\widetilde{\boldsymbol{X}} = \boldsymbol{E^T X} \sim N_d(\boldsymbol{0}, \boldsymbol{\Lambda})$, where $\boldsymbol{X} \sim N_d(\boldsymbol{0}, \boldsymbol{\Sigma})$. Since the random components of $\widetilde{\boldsymbol{X}}$ are uncorrelated, we can produce another random vector $\boldsymbol{W} \sim N_d(\boldsymbol{0}, \boldsymbol{I})$ whose components are independent and identically distributed (i.i.d), by normalizing the variance of each element of $\widetilde{\boldsymbol{X}}$. That means $\boldsymbol{W} = \boldsymbol{\Lambda}^{-1/2}\widetilde{\boldsymbol{X}} = \boldsymbol{\Lambda}^{-1/2}\boldsymbol{E^T X}$. By inverting this we get, $\boldsymbol{X} = \boldsymbol{E\Lambda}^{1/2}\boldsymbol{W}$. Finally, multivariate normal random variates along with mean vector, $\boldsymbol{\mu}$, and covariance matrix, $\boldsymbol{\Sigma}$, can be generated from the transformation $\boldsymbol{X} = \boldsymbol{\mu} + \boldsymbol{E\Lambda}^{1/2}\boldsymbol{W}$. Steps for generating random variates via eigen decomposition is discussed in Algorithm 1.

Cholesky Decomposition

To generate sample from MVN, another decomposition named Cholesky decomposition is also widely used. This decomposition is defined for a symmetric and positive definite matrix. In this decomposition method, the d-dimensional covariance matrix $\boldsymbol{\Sigma}$ ($\boldsymbol{\Sigma} = \sigma_{ij}$) is decomposed as, $\boldsymbol{\Sigma} = \boldsymbol{CC^T}$, where $\boldsymbol{C}$ is a unique lower triangular matrix of $\boldsymbol{\Sigma}$. Now, if Z_i's are independent $N(0,1)$ random variables then $\boldsymbol{CZ}$ is a multivariate normal[4] with covariance matrix $\boldsymbol{\Sigma}$.

Algorithm 1: Eigen decomposition to generate MVN

Input: $Z \sim N(0,1)$, mean vector, $\boldsymbol{\mu}_{1\times d}$ and covariance matrix, $\boldsymbol{\Sigma}_{d\times d}$.

Output: Simulated value of $\boldsymbol{X} \sim \text{MVN}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

1. Transform Z to $n \times d$ matrix $\boldsymbol{W}$.
2. Compute $\boldsymbol{E} = [e_1, e_2, \ldots, e_d]$ and $\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2, \ldots, \lambda_d)$.
3. Calculate $\boldsymbol{X} = \boldsymbol{\mu} + \boldsymbol{E\Lambda}^{1/2}\boldsymbol{W}$.

Matrix $\boldsymbol{C}$ can be obtained from the following formula :

$\boldsymbol{C}_{ij} = \frac{\sigma_{ij} - \sum_{k=1}^{j-1} \boldsymbol{C}_{ik}\boldsymbol{C}_{jk}}{\sqrt{\sigma_{jj} - \sum_{k=1}^{j-1} \boldsymbol{C}_{jk}^2}}$, where $\sum_{k=1}^{0} \boldsymbol{C}_{ik}\boldsymbol{C}_{jk} = 0$, $1 \leq j \leq i \leq n$. Finally, we get our desired $\boldsymbol{X}$ using, $\boldsymbol{X} = \boldsymbol{\mu} + \boldsymbol{CZ}$. The procedure for generating random variates via Cholesky decomposition is summarized in Algorithm 2.

Algorithm 2: Cholesky decomposition to generate MVN

Input: $Z \sim N(0,1)$, mean vector, $\boldsymbol{\mu}_{1\times d}$ and covariance matrix, $\boldsymbol{\Sigma}_{d\times d}$.

Output: Simulated value of $\boldsymbol{X} \sim \text{MVN}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

1. Transform Z to $n \times d$ matrix $\boldsymbol{W}$.
2. Compute $\boldsymbol{C}_{ij} = \frac{\sigma_{ij} - \sum_{k=1}^{j-1} \boldsymbol{C}_{ik}\boldsymbol{C}_{jk}}{\sqrt{\sigma_{jj} - \sum_{k=1}^{j-1} \boldsymbol{C}_{jk}^2}}$.
3. Calculate $\boldsymbol{X} = \boldsymbol{\mu} + \boldsymbol{CW}$.

There is another decomposition technique named singular value decomposition (SVD) in the literature, apart from eigen and Cholesky decompositions discussed earlier. SVD is a generalization of eigen decomposition of a square $(n \times n)$ normal matrix to any rectangular $(n \times m)$ matrix. But multivariate normal variates generation requires to decompose the covariance matrix, $\boldsymbol{\Sigma}$, which is indeed a square matrix.[5] Therefore, in our study we didn't consider singular value decomposition.

IV. Simulation Settings

In this section, we have presented the setup of our simulation study for the two methods that are considered in Section 3. Here all the computations are computed in *R* on a MacBook Air with an Intel (R) Core (TM) i7 processor running at 1.80 GHz. By using the eigen decomposition and Cholesky decomposition methods, samples of different size are generated for different values of d. We consider several combinations of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ to compare the performance of eigen and Cholesky decomposition methods. We choose mean vector, $\boldsymbol{\mu}$, arbitrarily (without help of any statistical packages) as choosing $\boldsymbol{\mu}$ is straightforward. However, choosing $\boldsymbol{\Sigma}$ arbitrarily is not straightforward like choosing mean vector as it needs to be a positive definite and invertible matrix.[6]

For example, choosing $\boldsymbol{\Sigma}$ arbitrarily may not be a positive definite and invertible matrix at the end. We need to consider a lot of $\boldsymbol{\Sigma}$ for different combinations and finding them by choosing arbitrarily is time consuming. For instance, for d = 100, arbitrarily choosing the elements of $\boldsymbol{\Sigma}$, 10,000 elements, is difficult and it may have ended up with non positive definite and non invertible matrix. Therefore, we use a *R* function "genPositiveDefMat" in *R* under "clusterGeneration" package to generate a positive definite matrix[6].

We have computed the CPU time consumed to execute the whole program using the "system.time" command and noticed that consumed CPU time varies one run to another run by 1 to 2 percent. We have also investigated the impact of choosing different $\boldsymbol{\Sigma}$ values by considering high and low variance and covariance components of $\boldsymbol{\Sigma}$ and different combinations of mean vector provided that n and d are fixed. We consider the following combinations of mean vector and covariance matrices,

$\boldsymbol{\mu}_1 = (120\ 540\ 470)^T$, $\boldsymbol{\mu}_2 = (0\ 0\ 0)^T$ and $\boldsymbol{\mu}_3 = (0.001\ 0.005\ 0.09)^T$.

$$\boldsymbol{\Sigma}_1 = \begin{pmatrix} 5 & 0.02 & 0.04 \\ 0.02 & 8 & 0.019 \\ 0.04 & 0.019 & 11 \end{pmatrix},$$

$$\boldsymbol{\Sigma}_2 = \begin{pmatrix} 0.3 & 0.007 & 0.036 \\ 0.007 & 0.1 & 0.09 \\ 0.036 & 0.09 & 0.8 \end{pmatrix} \text{ and}$$

$$\boldsymbol{\Sigma}_3 = \begin{pmatrix} 4.65 & -1.04 & -2.25 \\ -1.04 & 4.64 & -1.59 \\ -2.25 & -1.59 & 3.95 \end{pmatrix}.$$

Here, we have considered 9 combination of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. For instance, in $\boldsymbol{\Sigma}_1$, the variances of the variables are high but covariances among variables are low.

V. Numerical Results

In this section, we have presented our simulation results obtained for the two methods that are considered in Section 3. All the results presented here are produced using a random seed number and we have found that using different seed numbers produces similar kind of results. The performance of the two methods (computing time) due to choosing of different $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ values for a fixed n and d is presented in Table 1.

Table 1. Average computing time (in seconds) required to generate 10000 $N_3(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ random variates using different combinations of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$.

Mean	Covariance	Cholesky	Eigen
$\boldsymbol{\mu}_1$	$\boldsymbol{\Sigma}_1$	$0_{(\text{Approximately})}$	0.001
	$\boldsymbol{\Sigma}_2$	0.001	0.001
	$\boldsymbol{\Sigma}_3$	$0_{(\text{Approximately})}$	0.001
$\boldsymbol{\mu}_2$	$\boldsymbol{\Sigma}_1$	0.001	0.001
	$\boldsymbol{\Sigma}_2$	$0_{(\text{Approximately})}$	0.002
	$\boldsymbol{\Sigma}_3$	0.001	0.002
$\boldsymbol{\mu}_3$	$\boldsymbol{\Sigma}_1$	$0_{(\text{Approximately})}$	0.001
	$\boldsymbol{\Sigma}_2$	$0_{(\text{Approximately})}$	0.002
	$\boldsymbol{\Sigma}_3$	0.001	0.001

Table 2. Average computing time (in seconds) required to generate MVN random variates for different d and n values

Dimension (d)	Method	Sample Size (n)		
		500	10000	1000000
2	Eigen	0.001	0.002	0.003
	Cholesky	0.001	0.001	0.002
3	Eigen	0.001	0.002	0.036
	Cholesky	0.001	0.001	0.032
4	Eigen	0.001	0.002	0.056
	Cholesky	0.001	0.001	0.051
5	Eigen	0.001	0.002	0.080
	Cholesky	0.001	0.001	0.072
10	Eigen	0.001	0.003	0.237
	Cholesky	$0_{(\text{Approximately})}$	0.002	0.227
25	Eigen	0.002	0.009	1.403
	Cholesky	0.001	0.008	1.396
50	Eigen	0.005	0.038	4.428
	Cholesky	0.001	0.030	4.420
100	Eigen	0.015	0.122	14.384
	Cholesky	0.009	0.119	13.118

Table 2 shows the average required computing (CPU) time to generate samples for different values of n and d.

Again, average computing time required to generate MVN random variates for d = 2, 5, 25 and 50 using both eigen and Cholesky decomposition methods are shown in Figure 1. For each d value we have considered n = 500, 10000, 50000 and 100000.

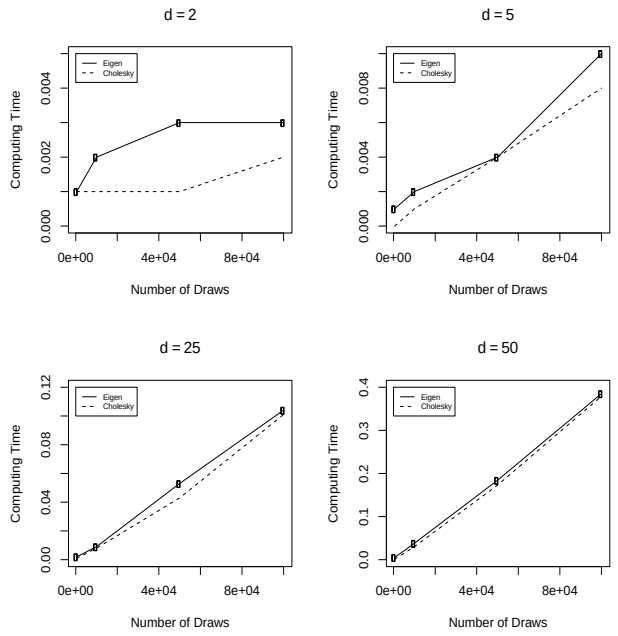

Fig. 1. Plot of average computing time (in seconds) required to generate MVN using both eigen and Cholesky decomposition methods.

VI. Discussion

In this section, we have discussed our simulation results of Section 5 obtained for the two methods that are considered in Section 3.

Table 1 shows the average computing time required to generate 10000 observations for d =3. From Table 1, we can see that the performance of the two methods (computing time) is not affected due to choosing of different $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ values for a fixed n and d. We have repeated this for other values of n and d and ended up with similar results (result is not shown here).

From Table 2, it is observed that Cholesky decomposition requires less computing time compared to eigen decomposition irrespective of n and d values. However, the required computing time difference of these two methods for generating sample for a particular n and d value is small. Therefore, one can use any decomposition method if the problem is solely related to MVN generation. But if MVN generation is required in conjunction with the other tasks, it is recommended to use Cholesky decomposition.

From Figure 1, it is observed that Cholesky decomposition method requires less time compared to eigen decomposition method irrespective of n and d values. However, margin of the difference of required computing time is small.

Finally, it can be concluded that both eigen and Cholesky decomposition methods can be used to generate MVN random variates. From our findings, it can also be stated that if the task is only data generation from MVN, then using Cholesky or eigen decomposition method is not an issue. But if generation from MVN is required in conjunction of other task, the method of Cholesky decomposition is numerically cheaper.

VII. Conclusion

This paper compares the performances of eigen and Cholesky decomposition method to generate MVN random variates, where performances are measured based on the required computing time. From our simulation study, we have seen that the performance of Cholesky decomposition method is better than eigen decomposition method. However, the required computing time difference of these two methods for generating sample for a particular n and d value is small. Therefore, one can use any decomposition method if the problem is solely related to MVN generation. But if MVN generation is required in conjunction with the other tasks, it is recommended to use Cholesky decomposition. As a future study one can investigate on the numerical stability of these two methods which is still an open research problem.

References

1. Ripley, B. D., 2009. Stochastic Simulation. John Wiley and Sons, **316**, 98.
2. Johnson, R. A., and D. W. Wichern, 2002. Applied Multivariate Statistical Analysis. Upper Saddle River, NJ: Prentice Hall, **5(8)**.
3. Bishop, C. M., 2006. Pattern Recognition and Machine Learning. Springer.
4. Wang, J., and M. R. Taaffe, 2015. Multivariate Mixtures of Normal Distributions: Properties, Random Vector Generation, Fitting, and as Models of Market Daily Changes. INFORMS Journal on Computing, **27(2)**, 193-203.
5. Abdi, H., 2007. The eigen-decomposition: Eigenvalues and Eigenvectors. Encyclopaedia of measurement and statistics, 304-308.
6. Joe, H., 2006. Generating Random Correlation Matrices Based on Partial Correlations. Journal of Multivariate Analysis, **97(10)**, 2177-2189.