MIDDLE EAST TECHNICAL UNIVERSITY
DEPARTMENT OF STATISTICS

A New Method to Solve Singularity Problems in Covariance Matrices

Ezgi Ayyıldız, Vilda Purutçuoğlu, Ernst Wit

METU-STAT-Technical Report-2012- 3

April, 2012

DEPARTMENT OF STATISTICS
MIDDLE EAST TECHNICAL UNIVERSITY
ANKARA 06531 – TURKEY

TECHNICAL REPORT

A new method to solve singularity problems in covariance matrices

Ezgi Ayyıldız[*], Vilda Purutçuoğlu Gazi[†], Ernst Wit[‡]

Abstract

Due to the dependency structure, most of the covariance matrix faces with the singularity problem. There are several methods to overcome this challenge. The most well-known ones are the eigen-value, singular value, and cholesky decompositions. In this paper, we develop a new method to deal with the singularity problem while preserving the covariance structure of the original matrix and compare our alternative solution with other methods. For analysis we generate various covariance matrices that have different dimensions and dependency structures and compare the CPU times of each approach.

1 Introduction

When dealing with the high dimensional covariance-variance matrix, we typically face with the singularity problem which causes the inverse and

[*]Middle East Technical University, Department of Statistics, Turkey. E-mail:ezgi.ayyildiz@metu.edu.tr

[†]Middle East Technical University, Department of Statistics, Turkey. E-mail: vpurutcu@metu.edu.tr

[‡]University of Groningen, Institute of Mathematics and Computing Science, The Netherlands. E-mail: e.c.wit@rug.nl

square of the underlying matrix intractable. There are a number of methods to unravel this challenge. The most well-known ones can be listed as the eigen-value decomposition, singular value decomposition, and cholesky decomposition. In this study, we suggest an alternative method for these major approaches to solve the singularity problem. We perform our suggested method in different covariance matrices with distinct dimensions and singularity proportions and we compare our results with the other well-known methods in terms of computational time. For comparison we use Monte Carlo methods. In Section 2 we initially explain each method explicitly. Then in Section 3 we present our comparative analyzes and interpret the outputs. Finally in Section 4, we summarize the outcomes and discuss the future works.

2 Methods

2.1 Eigen-value decomposition

Let A be a symmetric matrix and U be a matrix that includes the set of eigenvectors of A. Finally the diagonal matrix Λ stores the eigenvalues of A in the diagonal elements. Hence the following equation can be written for the given terms.

$$AU = \Lambda U. \tag{1}$$

Hereby

$$A = U\Lambda U^{-1}. \tag{2}$$

The square root of matrix A can be calculated by taking the square root of the diagonal matrix Λ via

$$A^{1/2} = U\Lambda^{1/2}U^{-1}. \tag{3}$$

If a matrix is positive semi-definite, then the eigen-value decomposition of this matrix always exists and the associated eigenvalues are always positive or null. More details can be found in Appendix. The correlation, covariance, and cross-product matrices can be given as examples of such types of matrices [1]. In this decomposition, the structure of the matrix can disappear if the original matrix is singular [2, 5].

2.2 Singular value decomposition

Let A be an $(m \times n)$ rectangular matrix which can be written as a product of three matrices such as an $(m \times m)$ orthogonal matrix U, an $(m \times n)$ diagonal matrix Λ, and the transpose of an $(n \times n)$ orthogonal matrix V. Λ is a matrix containing singular values in diagonal elements with descending order.

Accordingly the singular value decomposition (SVD) of a matrix A can be presented in terms of U, Λ, V by

$$A = U\Lambda V^T \tag{4}$$

where the columns of U are orthogonal eigenvectors of AA^T and the columns of V are orthogonal eigenvectors of A^TA. Thus the square root of matrix A can be obtained via

$$A^{1/2} = U\Lambda^{1/2}V^T. \tag{5}$$

Similar to the eigen-value decomposition, the structure of the original matrix A can change if it becomes nonsingular [2, 5].

2.3 Cholesky decomposition

If A denotes a symmetric positive definite matrix, the Cholesky decomposition is an upper triangular matrix U with strictly positive diagonal entries such that

$$A = U^T U. \tag{6}$$

Here the matrix U is called the square root of A. But if the matrix A is positive semi-definite, i.e. some eigenvalues are zero, a numerical tolerance is used in the decomposition of A. Finally the Cholesky decomposition, similar to other alternatives, it cannot preserve the original structure of the matrix when the singularity problem is solved [2, 5].

2.4 New method

We suggest a new updating regime to solve the singularity problem of the covariance matrix V which enables us to protect the covariance structure of V by working on the low dimensional matrix. The steps of this regime are listed as follows [3, 4].

1. By checking the columns of V from left to right, each linearly dependent column V is identified.

2. The dependent columns S, totally $\mid S \mid$, are described as the linear combination of independent columns.

3. A new $(N- \mid S \mid) \times (N- \mid S \mid)$ dimensional covariance matrix V^* is defined by eliminating $\mid S \mid$ dependent columns and rows from $V_{N \times N}$.

3 Application

In order to compare the performance of each method mentioned above, we use simulated data. Hereby, we generate different covariance matrices which are symmetric and singular. In the matrix generation, firstly independent columns are produced one by one from left to right. Then these columns are used to construct the linearly dependent columns. To obtain the symmetric matrix from the underlying singular matrix, the covariance matrix is calculated. These matrices are generated under two different scenarios. In the first plan, we generate matrices whose 80 % of the columns are independent and the remaining 20 % of columns are linearly dependent. Then in the second scenario, we use matrices whose 80 % of columns are linearly dependent, and the remainings are taken as independent. Under both scenarios, we use the matrices that have different dimensions, namely, 50 by 50, 100 by 100, 400 by 400, 500 by 500, and 750 by 750.

In this study, all the computational works are carried out in the R programming language version 2.10.0 and our codes are executed on Core 2 Duo 2.0 GHz processor. To compare the performance of four different methods, we check the CPU (Central Processing Unit) times by using two schemes. In the first scheme to estimate the time of each method, we iterate the MCMC algorithm 1000 times for each matrix structure with distinct dimensions and take the mean of the underlying Monte Carlo outputs. However because of the time restriction and computational inefficiency, we iterate the MCMC algorithm 250 times for the matrix dimension 750 by 750.

Table 1 and Table 2 show CPU of matrices with 3 different sizes under the first scenario. Although the new method has more computational steps, it is seen that there is no difference in the computational time of all methods under low, like (50×50), and moderately high dimensional-matrices

Table 1: Comparison of CPU time for eigen-value, singular value, and cholesky decompositions with new method based on different dimensional matrices under the first scheme and the first scenario.

	Dimensions			
Methods	(50×50)	(100×100)	(400×400)	(500×500)
Eigen-value decomposition	0	0	0	0.001
Singular value decomposition	0	0	0	0.001
Cholesky decomposition	0	0	0	0
Our method	0	0	0	0.318

Table 2: Comparison of CPU time for eigen-value, singular value, and cholesky decompositions with new method based on different dimensional matrices under the first scheme and the second scenario.

	Dimensions			
Methods	(50×50)	(100×100)	(400×400)	(500×500)
Eigen-value decomposition	0	0	0	0.007
Singular value decomposition	0	0	0	0.007
Cholesky decomposition	0	0	0	0.002
Our method	0	0	0	0.041

such as the matrices under (100×100) and (400×400) dimensions. Whereas if we consider very high dimensions, like (500×500) or (750×750), the new method slightly looses more time with respect to other methods under both scenarios. For instance the CPU times for the 750 by 750 matrix under the first scenario with 250 iterations are 0.008, 0.008, 0.002, and 2.147 for the eigen-value, singular value, cholesky, and our new method, respectively. Similarly the CPU times for this matrix-dimension under the second scenario are 0.014, 0.014, 0.003, and 0.785 for the same order of decomposition methods. On the other hand when we increase the number of dependent columns for the same dimensional matrix, it is seen that all of the methods own shorter times. However the computational time of our method decreases more sharply. This result can be interpreted that our new method can be preferable, specifically, when the singularity of the matrix becomes a severe challenge.

Whereas as the second scheme for the comparison of time demand, we report the complete CPU at the end of 1000 Monte Carlo runs. The results are presented in Table **??** and Table **??**. From the listed values, it is seen that under the first scenario, i.e. when the dependency is not seriously observed in the matrices, our suggested algorithm uses almost the same computational times for low and moderately high dimensional matrices. But it becomes less efficient while the dimension increases (Table **??**). On the other hand when the dependency problem is seriously observed, then our algorithm becomes the most computationally efficient, in particular, under moderately and very high dimensional matrices (Table 1). These findings indicate that our method can be evaluated as a promising alternative approach to deal with the serious singularity problems under different types of matrices with less computational demand.

Table 3: Comparison of total CPU time of 1000 MCMC runs for eigen-value, singular value, and cholesky decompositions with new method based on different dimensional matrices under the second scheme and the first scenario.

	Dimensions			
Methods	(50×50)	(100×100)	(400×400)	(500×500)
Eigen-value decomposition	0.05	0.03	34.86	27.28
Singular value decomposition	0.03	0.02	13.11	30.72
Cholesky decomposition	0.01	0.03	15.81	27.11
Our method	0.01	0.02	37.68	265.74

Table 4: Comparison of total CPU time of 1000 MCMC runs for eigen-value, singular value, and cholesky decompositions with new method based on different dimensional matrices under the second scheme and the second scenario.

	Dimensions			
Methods	(50×50)	(100×100)	(400×400)	(500×500)
Eigen-value decomposition	0.03	0.03	2.85	34.6
Singular value decomposition	0.02	0.05	1.47	29.35
Cholesky decomposition	0.03	0.04	0.09	23.48
Our method	0.05	0.05	0.94	9.44

4 Conclusion and Discussion

In this study, we have developed a new method to deal with the singularity problem of covariance matrix. We have explained the most well-known methods used to overcome this challenge such as the eigen-value, singular value, and cholesky decompositions. Moreover we have generated different covariance matrices with distinct dimensions and dependency structures. Then we have implemented these matrices in order to check the performance of both our new method and the methods mentioned above. Here we have compared the results according to their computational times.

From the results we have observed that although all methods have the same mean of CPU times for low and moderately high dimensional matrices, in the long run, it is seen that our novel algorithm is computationally efficient for highly singular matrices under different dimensions. We consider that its efficiency comes from its ability to detect the dependency structures in the matrices and keeping those relationships if they are necessary in further calculations. Indeed it is found that since it enables us to keep the same structure of the original observation as it saves the linear relationship between each column, it can be more effectively used in simulation study as well [3]. Because it enables us to regenerate eliminated column by using the actual linear information of the initial singular matrix.

5 Appendix

If a symmetric matrix A is positive semi-definite, then all its eigenvalues are nonnegative.

Let $e_1, e_2, ..., e_k$ be normalized eigenvectors of A. Given a column vector U,

we can write

$$U = \alpha_1 e_1 + \alpha_2 e_2 + ... + \alpha_k e_k. \tag{7}$$

Hereby

$$AU = \alpha_1 \lambda_1 e_1 + \alpha_2 \lambda_2 e_2 + ... + \alpha_k \lambda_k e_k \tag{8}$$

where λ_i associates to e_i. Then, since $e_i^T e_j = 0$ for $i \neq j$ and $e_i^T e_j = 1$ for $i = j$, the following equation can be written as

$$U^T AU = \alpha_1^2 \lambda_1 + \alpha_2^2 \lambda_2 + ... + \alpha_k^2 \lambda_k. \tag{9}$$

Hence the quantity in (9) is nonnegative if all λ_i's are nonnegative.

References

[1] Healy, M. J. R., 1986. Matrices for statistics. Oxford, Oxford University Press.

[2] Johnson, R. A. and Wichern, D. W., 2002. Applied multivariate statisticsl analysis. Upper Saddle River, N.J., Prentice Hall.

[3] Purutçuoğlu, V. and Wit, E., 2006. Exact and approximate stochastic simulation of the MAPK pathway and comparisons of simulations' results. Journal of Integrative Bioinformatics, 3, 231-243.

[4] Purutçuoğlu, V. and Wit, E., 2008. Bayesian inference of the complex MAPK pathway under the structural dependency. Journal of Statistical Research, 6, 1-17.

[5] Rencher, A.C., 2002. Methods of multivariate analysis. New York, John Wiley and Sons.