Applications of Randomized Methods for Decomposing and Simulating from Large Covariance Matrices

Vahid Dehdari and Clayton V. Deutsch

Abstract Geostatistical modeling involves many variables and many locations. LU simulation is a popular method for generating realizations, but the covariance matrices that describe the relationships between all of the variables and locations are large and not necessarily amenable to direct decomposition, inversion or manipulation. This paper shows a method similar to LU simulation based on singular value decomposition of large covariance matrices for generating unconditional or conditional realizations using randomized methods. The application of randomized methods in generating realizations, by finding eigenvalues and eigenvectors of large covariance matrices is developed with examples. These methods use random sampling to identify a subspace that captures most of the information in a matrix by considering the dominant eigenvalues. Usually, not all eigenvalues have to be calculated; the fluctuations can be described almost completely by a few eigenvalues. The first k eigenvalues corresponds to a large amount of energy of the random field with the size of $n \times n$. For a dense input matrix, randomized algorithms require $O(nn\log(k))$ floating-point operations (flops) in contrast with $O(nnk)$ for classical algorithms. Usually the rank of the matrix is not known in advance. Error estimators and the adaptive randomized range finder make it possible to find a very good approximation of the exact SVD decomposition. Using this method, the approximate rank of the matrix can be estimated. The accuracy of the approximation can be estimated with no additional computational cost. When singular values decay slowly, power method can be used for increasing efficiency of the randomized method. Comparing to the original algorithm, the power method can significantly increase the accuracy of approximation.

V. Dehdari (✉) · C.V. Deutsch
Center for Computational Geostatistics, University of Alberta, 3-133 NREF Building, Edmonton, Alberta, Canada, T6G 2W2
e-mail: dehdari@ualberta.ca

C.V. Deutsch
e-mail: cdeutsch@ualberta.ca

P. Abrahamsen et al. (eds.), *Geostatistics Oslo 2012*,
Quantitative Geology and Geostatistics 17,
DOI 10.1007/978-94-007-4153-9_2, © Springer Science+Business Media Dordrecht 2012

1 Introduction

Sequential Gaussian simulation is a popular method for generating realizations. This method is based on recursive application of Bayes law. There are two problems related to this method. The cost of generating n realization is n times of cost of generating one realization. Generating one realization can be reasonably fast, but if 100 realizations are to be generated, it can be time consuming. Second, reproducing long range variogram structures using this method is sometimes difficult. LU simulation is another method that can be used for generating unconditional or conditional realizations [1, 2]. This method is based on decomposing the covariance matrix. This method reproduces covariance matrix and once covariance matrix decomposed, generating large number of realization can be done without significant cost. For this reason, covariance matrix should be defined which contains covariance between data to data C_{11}, data to unsampled nodes C_{12} and unsampled nodes to unsampled nodes C_{22}. This covariance matrix has the following form:

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}. \tag{1}$$

In this matrix $C_{12} = C_{21}^T$. If there are n data and N unsampled nodes, C_{11} is $n \times n$, C_{12} is $n \times N$ and C_{22} is $N \times N$. For generating realizations, the covariance matrix can be decomposed using Cholesky decomposition. Cholesky decomposition is a popular method when we have a symmetric and positive definite matrix. Using this method covariance matrix can be decomposed to the lower and upper triangular matrices such that $C = LU = LL^T$ where L and U are lower and upper triangular matrices respectively. Another popular method is eigenvalue-eigenvector decomposition. In Cholesky decomposition, unconditional realization can be generated by decomposing C_{22} and using $m_i = L_{22}z_i$ formula. In this formula, z_i is random vector from Gaussian distribution with zero mean and unit variance and L_{22} is lower triangular matrix in Cholesky decomposition. Also kriging can be used for generating conditional realizations [2]. There are two problems related to the LU decomposition:

1. Accumulated roundoff errors can cause imaginary values for poorly conditioned covariance matrix [8].
2. When the covariance matrix is very large, full decomposition needs a very large additional storage and CPU time. In this case, decomposition cannot be done.

But as an advantage, this method can reproduce the mean and covariance matrices very well.

As another approach, Davis proposed the following method for generating conditional realization in just one step [2]. In this method by decomposing the original covariance matrix which contains information about covariance among all of grids and data, conditional realizations can be generated easily. Assume

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}, \tag{2}$$

where y_1 is $n \times 1$ column vector of normal score of conditioning data, and y_2 is $N \times 1$ column vector of required simulation values. From this system of equations $z_1 = L_{11}^{-1} y_1$ can be found. Also z_2 is $N \times 1$ vector of independent normal score data from Gaussian distribution. By replacing this equation into the second equation vector of required simulation values can be found from the following equation:

$$y_2 = L_{21} L_{11}^{-1} y_1 + L_{22} z_2. \tag{3}$$

The eigenvalue-eigenvector decomposition (SVD or singular value decomposition), can be used to decompose the covariance matrix to the form

$$C = U \Lambda U^T = U \Lambda^{1/2} \Lambda^{1/2} U^T = \bigl(U \Lambda^{1/2} U^T\bigr)\bigl(U \Lambda^{1/2} U^T\bigr)^T, \tag{4}$$

where U is the matrix of eigenvectors—each column of U is one eigenvector of covariance matrix. Λ is a diagonal matrix of eigenvalues; each element on the diagonal is one of the eigenvalues. These eigenvalues are ordered so that the first one is the largest one and they decrease to zero. The number of non-zero eigenvalues is the rank of matrix. In contrast to LU decomposition, in this case $U \Lambda^{1/2} U^T$ is not triangular and previous approach (3) cannot be used for generating conditional realizations. Assume that:

$$U \Lambda^{1/2} U^T = \begin{bmatrix} A & B \\ C & D \end{bmatrix}. \tag{5}$$

In this case, if there are n data and N unsampled node, A is $n \times n$, B is $n \times N$, C is $N \times n$ and D is $N \times N$. Conditional simulation can be found from the following formula

$$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}. \tag{6}$$

The elements in z_1 can be found by the relation $A z_1 + B z_2 = y_1$. From this formula we obtain:

$$z_1 = A^{-1}(y_1 - B z_2). \tag{7}$$

Again z_2 is $N \times 1$ vector of independent normal score data from Gaussian distribution. As a result, conditional simulation can be found from the following formula:

$$y = \begin{bmatrix} y_1 \\ C A^{-1}(y_1 - B z_2) + D z_2 \end{bmatrix}. \tag{8}$$

In this case, the matrix A usually is much smaller than the size of covariance matrix and its inverse can be found easily. Usually Cholesky decomposition is faster than SVD decomposition, but as we mentioned before, if the dimension of covariance matrix is large, decomposing the covariance matrix would be impossible. To prevent this problem, this paper introduces a randomized low rank approximation method for decomposing the covariance matrix. The most important reason for using SVD decomposition instead of LU decomposition is to permit the use of randomized low rank approximation methods. Using this method, the SVD decomposition or inversion of large matrices can be done very fast. In the next section, the application of randomized methods for decomposing large matrices will be presented.

2 Randomized Low Rank Approximation

Randomized method allows designing provably accurate algorithms for very large or computationally expensive problems. This method has been investigated by many authors [5–7, 9, 10]. This method is very popular method in computer science for image processing. In this method, by randomized sampling of rows or columns of a matrix, a new matrix can be found that is much smaller in size than the original matrix and captures most of the action of that matrix in a subspace. Using this method, the original matrix can be restricted to the subspace and then decomposition can be done in a subspace using standard factorization methods such as QR or SVD decomposition. Assume that rank of a matrix is R. In this case this matrix has R eigenvalues greater than zero. Using the randomized low rank approximation method, the approximated SVD decomposition of this matrix can be found by considering the first k dominant eigenvalues. This is called rank-k approximation of this matrix. The energy of a system can be defined as a summation of the first k largest eigenvalues divided by summation of all of eigenvalues. Usually by considering 90–95 % of the energy of a system, a reasonable approximation of the matrix decomposition can be done without missing major variabilities. Random sampling from a matrix A with size $m \times m$ can be done by multiplying this matrix by another matrix Ω which has the size of $m \times k$ and has been drawn randomly from the Gaussian distribution. Then, instead of decomposing the original matrix, this matrix which has much smaller size than the original matrix should be decomposed. If $Y = A\Omega$, in this case matrix Y can be decomposed to the Q and R:

$$Y = QR = [\,Q_1 \quad Q_2\,]\begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1. \tag{9}$$

If Y has dimension of $m \times k$, R_1 is a $k \times k$ upper triangular matrix, Q_1 is $m \times k$, Q_2 is $m \times (m-k)$ and Q_1 and Q_2 both have orthogonal columns. Q_2 matrix is related to the null space of matrix Y. QR factorization of Y is equal to the QR factorization of A in the subspace which is approximate decomposition of matrix A in the full space. If SVD decomposition of A is needed, matrix B can be found such that $B = Q^T A$. Finally SVD decomposition of matrix A can be found from full SVD decomposition of matrix B which has much smaller size than the original matrix. The randomized method for decomposing matrix A with size of $m \times n$ as described in [5] is:

1. Draw an $m \times k$ Gaussian random matrix Ω.
2. Form the $m \times k$ sample matrix $Y = A\Omega$.
3. Form an $m \times k$ orthonormal matrix Q such that $Y = QR$.
4. Form the $k \times m$ matrix $B = Q^T A$.
5. Compute the SVD of the small matrix B: $B = \widehat{U}\Sigma V^T$.
6. Form the matrix $U = Q\widehat{U}$.

In terms of computation cost, assume that the first k eigenvalues of a dense matrix with the size of $m \times m$ should be approximated. In this case, randomized algorithms require $O(m\,m \log k)$ floating-point operations (flops) in contrast with $O(m\,m\,k)$ for classical algorithms. Also randomized methods usually required a constant number of passes over the original matrix instead of $O(k)$ passes for classical algorithms [5].

When singular values decay slowly, the algorithm should be modified to increase efficiency of the randomized method. For this purpose, the power method can be used. Assume that matrix A should be decomposed and its eigenvalues decay slowly. Comparing to the original algorithm, the power method can significantly increase the accuracy of approximation. For solving this problem, instead of decomposing matrix A, matrix B can be decomposed:

$$B = \left(AA^*\right)^q A, \tag{10}$$

where A^* is the conjugate transpose of matrix A and q is an integer which is called the power. Both matrices A and B have the same singular vectors, but singular values of matrix B decay much faster than matrix A [7, 10]. If the ith singular value shows with the notation σ_i then:

$$\sigma_i(B) = \left(\sigma_i(A)\right)^{2q+1}. \tag{11}$$

In some cases, if the first 10 000 eigenvalues represent 95 % of the energy of the random field with size of $10^6 \times 10^6$, using the power method entails that approximately the first 1 000 eigenvalues could represent 95 % of the energy of the random field. Due to the large size of the matrix A, finding $(AA^*)^q A$ is almost impossible, this expression typically can be evaluated via alternating application of A and A^*. In this case, after finding $Y = A\Omega$, QR decomposition of Y without significant computation cost can be found. Matrix Q is orthogonal and it is approximated QR decomposition of matrix A. Therefore $QQ^* = I$ and as a result $A \approx QQ^*A$. Matrix Q has much smaller size than matrix A. Then in an iterative procedure without any significant cost $(AA^*)^q A$ can be found.

The algorithm of randomized subspace power method as found in [5] is summarized by:

```
1. Draw an $n \times k$ Gaussian random matrix $\Omega$.
2. Form $Y_0 = A\Omega$ and compute its QR factorization $Y_0 = Q_0 R_0$
3. For $j = 1, 2, \ldots, q$ where $q$ is integer power
   Form $\widetilde{Y}_j = A^T Q_{j-1}$ and compute its QR factorization $\widetilde{Y}_j = \widetilde{Q}_j \widetilde{R}_j$
   Form $Y_j = A\widetilde{Q}_j$ and compute its QR factorization $Y_j = Q_j R_j$
   End
4. $Q = Q_q$
5. Form the $k \times n$ matrix $B = Q^T A$.
6. Compute the SVD of the small matrix $B$: $B = \widehat{U}\Sigma V^T$.
7. Form the matrix $U = Q\widehat{U}$.
```

This method is much more efficient than the original method and increases accuracy of approximation significantly. It works very well for problems where the singular values decay slowly. Usually $q = 2$ or 3 is a good choice. Larger q will not significantly improve estimation.

There are different challenges in using this method. First, before full decomposition of a matrix, the eigenvalues are unknown. As a result finding the total energy of

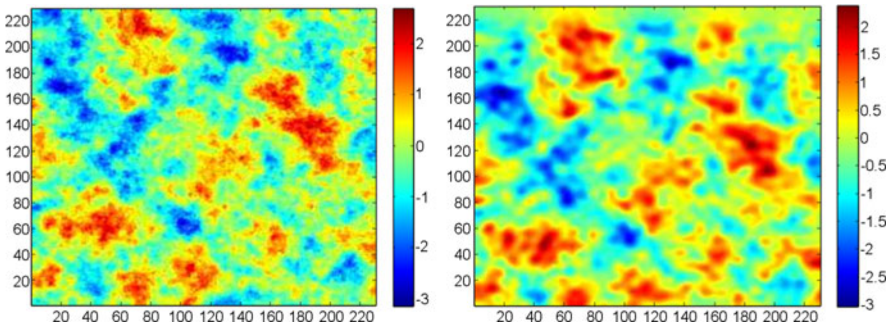

(a) Cholesky decomposition (b) Randomized SVD decomposition

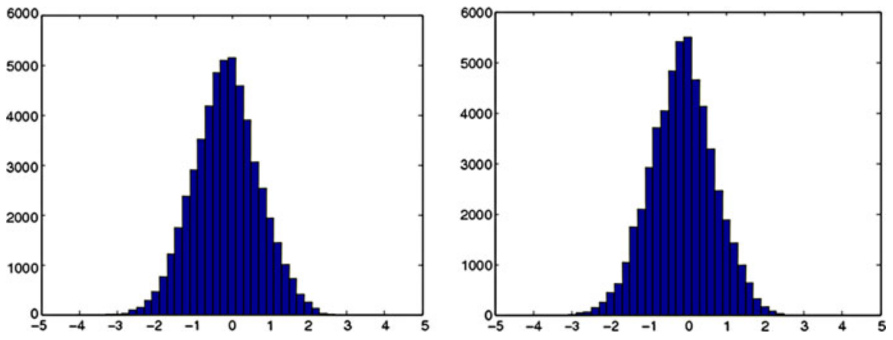
Fig. 1 Unconditional realization using exponential covariance

the system for computing the fraction of energy in the approximation is a problem. Second, the rank of the matrix is unknown in advance, so the number of eigenvalues for a good estimation is not known in advance. In the case of decomposing the covariance matrix, a solution of the first problem can be found easily. In this case, as an advantage of the covariance matrix, summation of eigenvalues is equal to the trace of covariance matrix, which is the summation of the elements on the diagonal. Because these values are equal to the variance, even before finding the covariance matrix, the total energy of the system can be easily found. Assume that the dimension of a covariance matrix is $50\,000 \times 50\,000$ and variance of the standard normal data is 1, in this case total energy of the system is equal to 50 000. After finding approximate decomposition, the summation of the eigenvalues can be found easily. In order to find the fraction of energy we used for simulation, this value can be divided by the total energy of the system.

Regarding the unknown rank of the matrix, the randomized range finder can be used. This method starts with an initial guess. Usually for a matrix with the size of $50\,000 \times 50\,000$, the first 1 000 eigenvalues are enough for approximating at least 90 % energy of the system. We can start with $k = 1\,000$ and by finding fraction energy of the system, if energy was not enough, the number of eigenvalues can be increased gradually to find a reasonable approximation. As another test, the accuracy of the approximation can be estimated with no additional computational cost [5]. This can be done by finding $l2$ norm of a matrix which is difference of original matrix and approximate matrix:

$$\left\| A - U\Sigma V^T \right\| = \left\| A - QQ^T A \right\| < \varepsilon. \tag{12}$$

The spectral norm of a matrix is the largest eigenvalue of that matrix. This is similar to the rank-1 approximation of that matrix which can be found very fast. For finding a good approximation, ε should be a small number. Using these methods, the accuracy of estimation can be estimated easily.

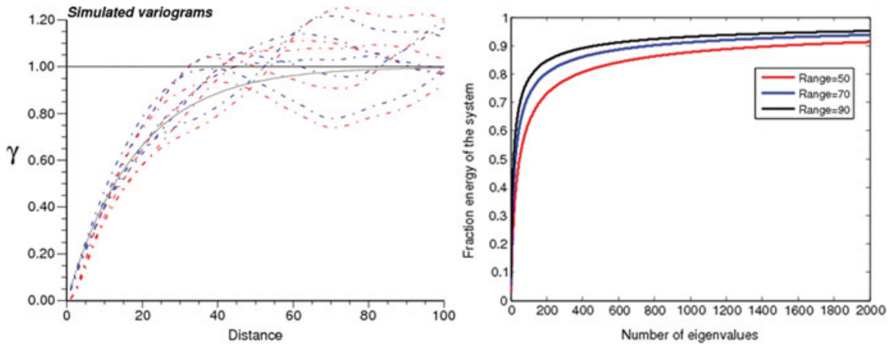


(a) Cholesky histogram (b) Randomized SVD histogram

Fig. 2 Histogram comparison using exponential covariance

3 Results and Discussion

The goal of this paper is to compare the efficiency of a randomized method with Cholesky decomposition, different synthetic examples with reasonable size have been considered. The dimension of the domain in these examples is 230×230 blocks. In this case, size of covariance matrix would be $52\,900 \times 52\,900$ and it has about 2.8 billion elements. This was the largest matrix that I could decompose with the LU simulation. Larger matrix can be created and decomposed using randomized methods, but because the goal of this paper is comparing the efficiency of these methods, this example is large enough for this reason. For the first example, a random field has been generated with variance 1 and also exponential covariance matrix. Using unconditional simulation and just by decomposing the C_{22} in the covariance matrix, one realization for both of LU and randomized SVD has been generated. In this example, correlation range is 60 blocks. This range is equal to practical range of covariance matrix (95 % of the sill). SVD used the first 2 000 eigenvalues for generating the result. In this case, full decomposition has 52 900 eigenvalues, but the first 2 000 eigenvalues equal to about 93 % energy of the system. In this case, the spectral norm is less than 0.01 which shows a good precision. Figure 1 shows result of each method. As Fig. 1(b) shows, randomized algorithm eliminated very short variabilities. It means that if correlation range is very small, very small eigenvalues should not be eliminated. In this case, more eigenvalues should be considered for storing 95 % energy of that system. In the limiting case, when correlation range is zero, none of eigenvalues are zero. In this case, covariance matrix would be a diagonal matrix and its decomposition using Cholesky, even if the matrix is very large, would be very fast. As a result, if range is very small (which is rare in the real cases) using Cholesky decomposition could be a better choice. What is important in the simulation is histogram and variogram reproduction. Figure 2 and Fig. 3(a) show histogram and variogram comparisons using both of methods.

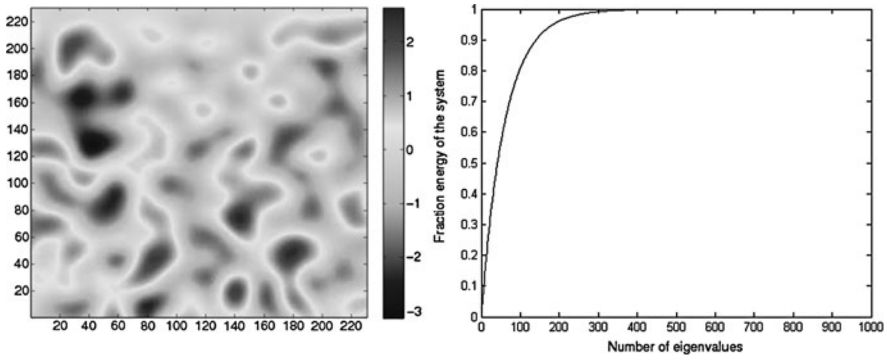


(a) Variogram reproduction (b) Energy fraction for SVD

Fig. 3 Variogram reproduction and energy fraction using exponential covariance. In (**a**) *red dashed lines* are variograms of simulated realizations using randomized SVD, *blue dashed lines* are variograms of simulated realizations using Cholesky decomposition and *gray solid line* is theoretical variogram. In (**b**) *colors* indicates correlation range in the covariance function

Usually histogram reproduction is not a big problem in the simulation. As you can see, Fig. 2 shows comparison between reproduced histograms using both methods. In this case both of histograms are normal and they are pretty the same.

In Fig. 3(a), five realizations generated for each method. Red dashed lines are variograms of simulated realizations using randomized SVD, blue dashed lines are variograms of simulated realizations using Cholesky decomposition and gray solid line is theoretical variogram. It seems that average of variograms in both methods can reproduce theoretical variogram. A small correlation range means the eigenvalues will decay slowly and by increasing the range, they will decay very fast. For smaller ranges, the power method can be used for increasing the decay rate. Using power method may increase computation time a little bit which is not significant, but it can solve problem related to the slow decay rate. As another solution, the number of desired eigenvalues can be increased. For example if 2 000 eigenvalues can give 0.95 energy of a system for range 70, 3 000 eigenvalues can give this amount of energy for range 40. The best results can be found by increasing the number of desired eigenvalues and also power simultaneously. Instead of selecting 3 000 eigenvalues with power equal to 2, 2 500 eigenvalues can be selected with power equal to 3. Figure 3(b) shows changing in the fraction energy of the system with changing the correlation range. Power of 3 has been selected for all of these figures, for this reason decreasing the range does not have a significant effect on the fraction of energy of the system using the first 2 000 eigenvalues. The first 200 eigenvalues with a range 90 gives about 85 % energy of the system, but with a range of 50 gives only about 70 % energy of the system. This method is very efficient in terms of matrix decomposition. Even for this small example, the run time of the Cholesky decomposition was more than 1.5 hours, but randomized SVD method decomposed the covariance matrix in 20 minutes for finding the first 2 000 eigenvalues. A small change in the number of desired eigenvalues does not change the computation time significantly. Unconditional realizations were generated for a random field with the



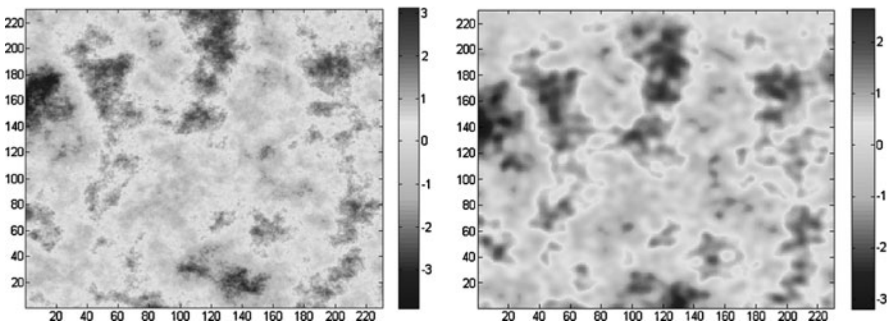
(a) Realization map (b) Eigenvalues decay rate

Fig. 4 Realization generated using Gaussian covariance

size of 230 × 230 blocks with the correlation range of 30 blocks using a Gaussian covariance matrix. Figure 4 shows the generated realization and the decay rate of the eigenvalues.

As you can see in Fig. 4(b), although a small range was selected for this case, the rate of eigenvalues decay is very fast. Only with the first 400 eigenvalues 99.99 % of the energy of the system has been stored. All other eigenvalues have very small values. They are smaller than the order of 10^{-4} and they do not have any significant effect on the estimations. In this case, run time was about 9 minutes compare to the 1.5 hours for Cholesky decomposition. This example shows that rate of eigenvalues decay in the Gaussian covariance is too fast. This finding confirms the results of [3] and [4] about rate of eigenvalues decay in different covariance matrices. They showed that eigenvalues of a Gaussian covariance matrix decay exponentially fast in contrast to the eigenvalues of an exponential covariance matrix which may decay slowly. Even by selecting a larger range, e.g. 65 blocks, the first 150 eigenvalues can show 99.99 % energy of the system. Full decomposition can give exactly the same realization. Suppose that there is a very large field with very large covariance matrix. In this case, using the first 1 000 eigenvalues, large number of realizations can be generated very fast. Nice thing about the LU or SVD simulation methods is that once covariance matrix decomposed, large number of realizations can be generated with a negligible computation time.

The last covariance function that we considered in this paper is spherical covariance function. As we considered different examples, we noticed that rate of eigenvalues decay for spherical covariance is not as fast as the Gaussian covariance, but it is faster than exponential covariance function. Again, same as the first example, one unconditional realization for a random field with the size of 230 × 230 blocks with the range of 50 blocks using the spherical covariance matrix generated. Figure 5 shows comparison between generated realizations using Cholesky and randomized SVD methods. In this case, run time was about 13 minutes compare to the 1.5 hours for Cholesky decomposition.



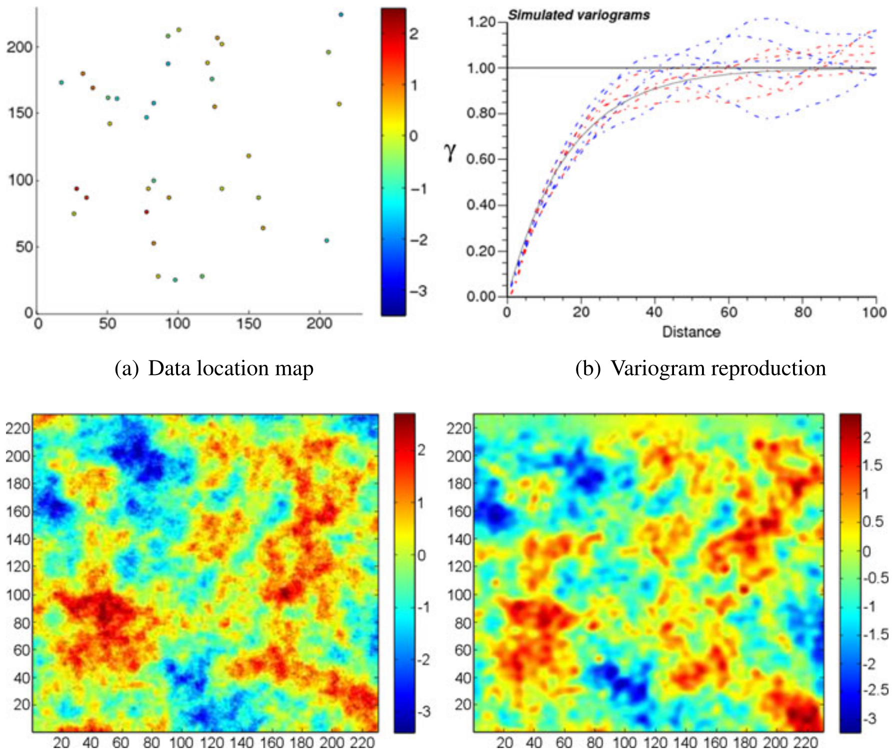
(a) Cholesky decomposition (b) Randomized SVD decomposition

Fig. 5 Realization generated using spherical covariance

The histogram and variogram reproduction for both the Gaussian and spherical covariance functions was satisfactory. The most difficult case is related to the exponential covariance function which causes eigenvalues to decay slower than other types. For the last example, a conditional simulation example has been considered. For this purpose, a synthetic model with the size of 230×230 blocks with the range of 50 blocks using the exponential covariance matrix generated. As we showed before, due to the small decay rate of eigenvalues, exponential covariance is the most challenging type of covariance function. Figure 6 shows location map of data, generated realizations using LU and SVD and also variogram reproduction.

Based on the Cholesky and SVD formulations, all of the input data can be reproduced after simulation. Also as you can see, for five different realizations the average of the variograms reproduce the input variogram, so covariance reproduction has been done satisfactorily. Again red dashed lines are variograms of simulated realizations using randomized SVD, blue dashed lines are variograms of simulated realizations using Cholesky decomposition and gray solid line is theoretical variogram. In this problem, SVD decomposition has been done using the first 3 000 eigenvalues and $q = 3$. In this case more than 95 % energy of the system has been found. Also spectral norm after decomposition was about 0.01.

The efficiency of this method has been considered with several examples. This method is very efficient when the size of the covariance matrix is very large. Once a matrix can be generated, decomposing it with this method is possible and the accuracy of the method can be made acceptable. The only problem is related to the storage of large matrices. Full decomposition of that matrix is another important problem. If the size of the matrix is very large, full decomposition of that matrix is almost impossible or at least takes several hours. In the case of using Cholesky decomposition, using double precision variables is necessary. As we mentioned before, roundoff error is one of the problems of this method, but in the case of SVD decomposition, there is no problem with roundoff error and the matrix does not need to be positive definite. Even for storing a larger matrix, single precision variables can be used to decrease the memory usage.



(a) Data location map (b) Variogram reproduction

(c) Cholesky decomposition (d) Randomized SVD decomposition

Fig. 6 Generating conditional simulation realization

4 Conclusions

In this paper, randomized SVD decomposition has been considered as an efficient method for decomposing covariance matrix and generating simulation realizations. This method is based on the randomized sampling of covariance matrix for finding a subspace which has much smaller size than the original matrix and captures most of the action of that matrix. Using this randomized low rank approximation method, approximated SVD decomposition of covariance matrix can be found by considering the first largest k eigenvalues. For testing efficiency of this method, different examples for different types of covariance functions have been considered. This method works very well for the cases that eigenvalues decay very fast. An example of this case is when the field correlation can be defined using Gaussian covariance function. For other cases like the spherical or exponential covariance functions that eigenvalues decay slower, using power method is a very good choice for solving this problem. The accuracy of results can be estimated by computing energy fraction or spectral norm. A good estimation should use about 95 % energy of the system and gives a small spectral norm. For assessing efficiency of this method, examples compared with Cholesky decomposition results. Comparisons show good efficiency of this method. Even for small examples, this method is much faster than Cholesky decomposition method. Using this method, very large covariance matrices can be decomposed which is impossible using Cholesky method. The only limitation of this method is related to storing a very large covariance matrix in the memory; the covariance matrix must be stored in machine memory.

References

1. Alabert F (1987) The practice of fast conditional simulations through the LU decomposition of the covariance matrix. Math Geol 19(5):369–386
2. Davis M (1987) Production of conditional simulations via the LU triangular decomposition of the covariance matrix. Math Geol 19(2):91–98
3. Efendiev Y, Hou T, Luo W (2007) Preconditioning Markov chain Monte Carlo simulations using coarse-scale models. SIAM J Sci Comput, 28(2):776
4. Frauenfelder P, Schwab C, Todor R (2005) Finite elements for elliptic problems with stochastic coefficients. Comput Methods Appl Mech Eng 194(2–5):205–228
5. Halko N, Martinsson P, Tropp J (2011) Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev 53:217
6. Liberty E, Woolfe F, Martinsson P, Rokhlin V, Tygert M (2007) Randomized algorithms for the low-rank approximation of matrices. Proc Natl Acad Sci USA 104(51):20167
7. Martinsson P, Szlam A, Tygert M (2010) Normalized power iterations for the computation of SVD. In: NIPS workshop on low-rank methods for large-scale machine learning, Vancouver
8. Oliver D, Reynolds A, Liu N (2008) Inverse theory for petroleum reservoir characterization and history matching. Cambridge University Press, Cambridge
9. Papadimitriou C, Raghavan P, Tamaki H, Vempala S (2000) Latent semantic indexing: a probabilistic analysis. J Comput Syst Sci 61(2):217–235
10. Rokhlin V, Szlam A, Tygert M (2009) A randomized algorithm for principal component analysis. SIAM J Matrix Anal Appl 31(3):1100–1124