Improvements on the hybrid Monte Carlo algorithms for matrix computations

BEHROUZ FATHI-VAJARGAH¹ and ZEINAB HASSANZADEH^{2,*}

¹Department of Statistics, Faculty of Mathematical Sciences, University of Guilan,
P.O. Box 41335-19141, Rasht, Iran

²Department of Applied Mathematics, Faculty of Mathematical Sciences, University of Guilan,
P.O. Box 41335-19141, Rasht, Iran

e-mail: fathi@guilan.ac.ir; behrouz.fathi@gmail.com; hassanzadeh@phd.guilan.ac.ir;
hassanzadeh.83@gmail.com

MS received 30 October 2017; revised 5 February 2018; accepted 18 June 2018; published online 7 December 2018

Abstract. In this paper, we present some improvements on the Markov chain Monte Carlo and hybrid Markov chain Monte Carlo algorithms for matrix computations. We discuss the convergence of the Monte Carlo method using the Ulam–von Neumann approach related to selecting the transition probability matrix. Specifically, we show that if the norm of the iteration matrix T is less than 1 then the Monte Carlo Almost Optimal method is convergent. Moreover, we suggest a new technique to approximate the inverse of the strictly diagonally dominant matrix and we exert some modifications and corrections on the hybrid Monte Carlo algorithm to obtain the inverse matrix in general. Finally, numerical experiments are discussed to illustrate the efficiency of the theoretical results.

Keywords. System of linear algebraic equations; Markov chain Monte Carlo; convergence analysis; transition probability; matrix inversion.

1. Introduction

It is well known that solving a system of linear algebraic equations (SLAE) and obtaining the matrix inversion (MI) have an underlying role in engineering and sciences problems such as dynamical systems, circuit analysis, digital signal processing, financial or stochastic modelling and physical problems including Partial Differential Equations (PDEs). Because of the urgency in the recent years for big data problems and large scale systems, the Monte Carlo (MC) method has been widely applied in comparison with the modern deterministic linear solvers (e.g., Krylov subspace methods). Nowadays, the MC method is employed in many branches of medical areas and experimental sciences such as [1] and [2], respectively.

The authors' motivation for study in this work is the several advantages of the MC algorithms in using extremely large matrices. The MC method is based on random sampling of a certain variable, which is an unbiased estimator of the desired solution. They can be independently calculated. Therefore, MC algorithms are appropriate candidates for parallelization, which speeds up the solution-obtaining process. Moreover, this method has a computational complexity linear with the size of the matrix.

Therefore, the approximate solution can be obtained very efficiently for large matrices. Also, this method can quickly acquire low-accuracy rough approximate of the solution, which can be refined to enhance the accuracy of the obtained solution. Accordingly, the required components of the solution MC method, contrary to the deterministic iterative methods, can approximate one or some elements of the solution vector, independently.

MC method to estimate the solution of SLAE was first introduced by Ulam and von Neumann. Later, some developments were made in this context by the researchers. In recent years, Fathi-Vajargah in [3, 4] investigated the hybrid MC algorithm for MI and solving SLAE. He showed that the stochastic approximation of the MI can be efficiently combined with a deterministic refinement procedure to achieve higher precision. Alexandrov *et al* in [5] further discussed introduction of a hybrid MC method and tried to increase the speed of its computation by parallel computing. In [6], the authors analysed various types of MC acceleration schemes applied to the original preconditioned Richardson iterative method. Recently, an enhanced hybrid method based on constructing an efficient stochastic preconditioner to solve SLAE was presented. This preconditioner is based on Markov Chain Monte Carlo (MCMC) method to estimate a Sparse Approximate Inverse (SPAI) (see [7–10]). Dimov *et al* in [11] presented a new random

*For correspondence

walk on equations of the MC algorithm to solve SLAE, where the rate of convergence depends on the balancing of the matrix. A new approach of reusing random walks in the MC method for solving SLAE was considered in [12]. The key idea of this method is that during the MC sampling, the random walks generated to estimate one unknown element can also be reused to estimate the other elements in the solution vector. Okten in [13] presented a new MC estimator for solving SLAE and made a detailed investigation on the average complexity of the MC estimators, where several popular random variable generation techniques have been used.

In this work, we study the MC method for solving SLAE to provide more accurate results. We discuss on the convergence of this method and show that if $\|T\| < 1$ then the Monte Carlo Almost Optimal (MAO) method is convergent, where T is the iteration matrix. Also, $\|\cdot\|$ is considered as matrix ∞ -norm in this paper. In addition, we provide some corrections and modifications on the MC and hybrid MC methods for obtaining MI that are more efficient than in previous research works.

This paper is organized in the following way. After reviewing the MC method for SLAE, we concentrate on the convergence of this method in section 2. Enhancements on the hybrid MC method are presented to acquire MI in a special case and in the general case in section 3. Section 4 elucidates theoretical discussion by some numerical examples. Finally, section 5 ends this paper with a conclusion.

2. MC method for solving SLAE

In this section, we provide the background of the MC method for solving SLAE and study its convergence with new conclusions.

2.1 Some details on the background of the MC method for solving SLAE

We consider the SLAE

$$Ax = b, \quad (1)$$

where $A \in \mathbb{R}^{n \times n}$ is a nonsingular matrix and $b \in \mathbb{R}^n$ is a known vector. After substituting the splitting $A = M - N$, Eq. (1) can be recast in the following iterative scheme:

$$x^{(k+1)} = Tx^{(k)} + f, \quad k = 0, 1, \dots, \quad (2)$$

in which $T = M^{-1}N$ and $f = M^{-1}b$. Assuming $\rho(T) < 1$, it is derived that $I - T$ is nonsingular and the iterative relation (2) converges to the exact solution of (1) regardless of the initial guess x^0 . Conversely, if the iterations (2) converge for any x^0 then $\rho(T) < 1$. Let $\|T\| < 1$ for some matrix norm $\|\cdot\|$. Then $I - T$ is nonsingular and the iteration (2)

converges for any initial vector x^0 (for more details we refer [14]). Moreover, x can be written in terms of a power series in T (Neumann series):

$$x = \sum_{k=0}^{\infty} T^k f. \quad (3)$$

The basic idea of the MC algorithm is constructing a homogeneous discrete Markov chain $\gamma : r_0 \rightarrow r_1 \rightarrow \dots \rightarrow r_k \rightarrow \dots$, using random walks on the indices of T in order to sample the solution to (1) as developed via the Neumann series. Realization of random walks is based on transition probabilities, which make the initial probability vector $p \in \mathbb{R}^n$ (starting point of random walks) and transition probability matrix $P \in \mathbb{R}^{n \times n}$ (transition from the current point to the next point) to satisfy the following conditions:

$$\begin{aligned} p_{ij} &\geq 0, \quad \sum_{j=1}^n p_{ij} = 1, \quad \text{if } t_{ij} \neq 0 \quad \text{then } p_{ij} \neq 0, \\ p_i &\geq 0, \quad \sum_{i=1}^n p_i = 1, \quad \text{if } h_i \neq 0 \quad \text{then } p_i \neq 0, \end{aligned} \quad (4)$$

where the nonzero vector $h \in \mathbb{R}^n$ is considered to evaluate Euclidean inner product $\langle h, x \rangle$. Indeed, the role of h is restricted to the construction of the initial probability. We first follow the discussion for the arbitrary nonzero vector h . Then, we restrict h to be e_i (i.e., i th standard basis vector) to reach our desired solution. In continuation, the necessity of employing the vector h will be clarified. For the stochastic trajectory γ the following unbiased estimator is defined:

$$X(\gamma) = \sum_{m=0}^{\infty} W_m f_{r_m}, \quad (5)$$

where $W_m = W_{m-1} w_{r_{m-1}r_m}$, $w_{r_{m-1}r_m} = \frac{t_{r_{m-1}r_m}}{p_{r_{m-1}r_m}}$ and $W_0 = \frac{h_{r_0}}{p_{r_0}}$, so that mathematical expectation of $X(\gamma)$ is $\langle h, x \rangle$, i.e., $E[X(\gamma)] = \langle h, x \rangle$ ([11] and references therein). In this paper, we consider a discrete Markov chain without absorption state. We also suppose that $h = e_i$ and $p_i = 1$ to estimate the i th component of the solution. Representing the single component of x in the form

$$x_i = f_i + \sum_{m=1}^{\infty} \sum_{r_1=1}^n \sum_{r_2=1}^n \dots \sum_{r_m=1}^n t_{ir_1} t_{r_1 r_2} \dots t_{r_{m-1} r_m} f_{r_m}, \quad (6)$$

it can be proved that mathematical expectation of $X_i(\gamma) = \sum_{m=0}^{\infty} W_m f_{r_m}$ ($h = e_i, p_i = 1$) is the i th component of the solution, i.e., $E[X_i(\gamma)] = x_i$ [11]. In practical computation, we apply the partial sum $X_i(\gamma_k) = \sum_{m=0}^k W_m f_{r_m}$ on finite Markov chain $\gamma_k : r_0 \rightarrow r_1 \rightarrow \dots \rightarrow r_k$, whose mathematical expectation tends to x_i by choosing k large enough. This approximation leads to the systematic error. Since

$E[\sum_{m=0}^{\infty} W_m f_{r_m}] < \infty$, effective stopping criterion for the random walk to create the Markov chain γ_k is given as $|W_k| < \epsilon$, where $\epsilon > 0$ is the given parameter. As a result we can derive the upper bound for the length of Markov chain γ_k . From the systematic error $|W_k| < \epsilon$ and substituting the value of W_k in it, we have

$$|W_k| = \left| \frac{t_{r_0 r_1} t_{r_1 r_2} \dots t_{r_{k-1} r_k}}{p_{r_0 r_1} p_{r_1 r_2} \dots p_{r_{k-1} r_k}} \right| \leq \frac{|t_{r_0 r_1} t_{r_1 r_2} \dots t_{r_{k-1} r_k}|}{\frac{|t_{r_0 r_1}|}{\|T\|} \frac{|t_{r_1 r_2}|}{\|T\|} \dots \frac{|t_{r_{k-1} r_k}|}{\|T\|}} \leq \|T\|^k < \epsilon.$$

Hence $k < \frac{\log(\epsilon)}{\log \|T\|}$. It is noteworthy that the components of the transition probability matrix P , satisfying the transition conditions (4), are considered as $p_{ij} = \frac{|t_{ij}|}{\sum_{j=1}^n |t_{ij}|}$, which has been used in the MAO algorithm [11]. Assuming $\|T\| < 1$, it can be deduced that by decreasing $\|T\|$ the length of Markov chain declines and the convergence speed of the MC method increases. According to the Strong Law of Large Numbers (SLLN), we simulate N independent sample paths of the Markovian chain $r_0^{(s)} \rightarrow r_1^{(s)} \rightarrow \dots \rightarrow r_k^{(s)}$, $s = 1, 2, \dots, N$ and consider the sample mean (MC esti-

mation) of $X_i(\gamma_k^{(s)})$ to estimate the real mean $E[X_i(\gamma_k)]$. Therefore, we have $\bar{X}_i(\gamma_k) = \frac{1}{N} \sum_{s=1}^N X_i(\gamma_k^{(s)}) \approx x_i$. In this way we encounter the statistical error, which is theoretically expressed as $|\bar{X}_i(\gamma_k) - E[X_i(\gamma_k)]| < \delta$, where δ is the given parameter. From the practical point of view, probable error is employed to estimate the statistical error. Based on [15], by applying the Central Limit Theorem (CLT) for the random variable $\bar{X}_i(\gamma_k)$, we can obtain the probable error $r_N = 0.6745 \sqrt{\frac{\text{var}(X_i(\gamma))}{N}}$. Applying the precision $r_N \leq \delta$ and employing the upper bound for $\text{var}(X_i(\gamma))$, we can reach

$$N \geq \frac{(0.6745)^2}{\delta^2} \frac{\|f\|^2}{(1 - \|T\|)^2}. \quad (7)$$

As we observe, by decreasing $\|T\|$ the number of Markov chains declines and the MC estimator requires fewer samples. Moreover, obtained lower bound for N is independent of n , which is one of the advantages of the MC method for large matrices. At the end of this section we represent the MC method in the following algorithm format (unordered form is in [5, 11]).

Algorithm 1:

1. Input matrix T , vector f , parameters ϵ and δ
 2. Compute $N = \lceil \left(\frac{0.6745}{\delta} \right)^2 \frac{\|f\|^2}{(1 - \|T\|)^2} \rceil + 1$, ($\lceil x \rceil$ is the integer part of x)
 3. Compute P based on the type of probability transition matrix
 4. **for** $i = 1$ to n
 5. **for** $s = 1$ to N
 6. Set $W_0 = 1$, $k = 0$, $\text{point} = i$, $X_i^{(s)} = W_0 f_i$
 7. **while** $|W_k| \geq \epsilon$
 8. Generate an r.v. nextpoint , distributed on i th row of matrix P as:
 9. Set $\text{nextpoint} = 1$, $u = \text{rand}$
 10. **while** $u > p_{\text{point}, \text{nextpoint}}$
 11. $\text{nextpoint} = \text{nextpoint} + 1$
 12. **end while**
 13. Set $k = k + 1$
 14. **if** $t_{\text{point}, \text{nextpoint}} \neq 0$, (this loop can be out of conditional in MAO)
 15. Compute $W_k = W_{k-1} \frac{t_{\text{point}, \text{nextpoint}}}{p_{\text{point}, \text{nextpoint}}}$, $X_i^{(s)} = X_i^{(s)} + W_k f_{\text{nextpoint}}$
 16. **end if**
 17. Set $\text{point} = \text{nextpoint}$
 18. **end while**
 19. **end for**
 20. Compute $X_i(\gamma_k) = \frac{1}{N} \sum_{s=1}^N X_i^{(s)}$
 21. **end for**
-

2.2 Convergence analysis

It is well known that the necessary and sufficient condition for the Neumann series to convergence in the MC method is $\rho(T) < 1$ [14]. As indicated in Lemma 1, $\|T\| < 1$ is a stricter condition than $\rho(T) < 1$.

Lemma 1 ([14]) *For a nonsingular matrix $T \in \mathbb{C}^{n \times n}$, $\rho(T) \leq \|T\|$.*

According to the research of authors in previous papers [3–5, 8], if $\|T\| < 1$ then the MC method is convergent. In recent years, [16] (as the comprehensive reference) analysed the convergence of the MC method. Based on [16] condition $\|T\| < 1$ or the stronger condition $\rho(T) < 1$ (Lemma 1) is not the sufficient condition for the convergence of the MC method. Indeed, the assumption $\|T\| < 1$ (stronger than $\rho(T) < 1$) just provides the convergence of Neumann series. Not only is the structure of T a not sufficient criterion determining the convergence, but also the underlying transition probability matrix has an important role. An improper selection of the transition matrix may lead to divergence even though the condition $\|T\| < 1$ is valid. The necessary and sufficient condition for convergence of the MC method has been presented in Theorem 1.

Theorem 1 ([16]) *Given an $n \times n$ nonsingular matrix T such that $\rho(T) < 1$, a nonzero vector f and a transition matrix P , the necessary and sufficient condition for convergence of the MC linear solver using the Ulam–von Neumann algorithm is*

$$\rho(T^*) < 1, \quad (8)$$

where T^* is an $n \times n$ matrix such that $t_{ij}^* = \frac{(t_{ij})^2}{p_{ij}}$.

More results are presented in the convergence theorems in [16], which we review. If $\rho(T) < 1$ but $\|T\| \geq 1$, the MC method may or may not converge. A more precise version of “the MC method converges if $\|T\| < 1$ ” is “there is always a transition matrix P enabling the convergence of the MC method if $\|T\| < 1$ ”. In addition, if the row sum $\sum_{j=1}^n |t_{ij}| > 1$ for every row in T or more generally $\rho(T^+) \geq 1$, where $t_{ij}^+ = |t_{ij}|$, transition matrix leading to convergence of the MC does not exist.

In this paper, we emphasize that it cannot be surely claimed that the MAO and Uniform Monte Carlo (UM) transition probability matrices lead to the convergence. It should be reminded that in the UM method, matrix P is defined as $p_{ij} = \frac{1}{n}$. Consequently, regardless of the type of transition probability matrix, validity of Eq. (8) should be checked for convergence of the MC method.

Theorem 2 *Let T be an $n \times n$ nonsingular matrix such that $\|T\| < 1$. Then $\rho(T^*) < 1$, where the matrix T^* is defined in Theorem 1 and P is supposed to be the MAO transition probability matrix.*

Proof According to the definition of T^* , we have $t_{ij}^* = \frac{(t_{ij})^2}{p_{ij}} = |t_{ij}| \sum_{j=1}^n |t_{ij}|$. Since $\|T\| < 1$ it can be derived that

$$\sum_{j=1}^n |t_{ij}^*| = \left(\sum_{j=1}^n |t_{ij}| \right)^2 < 1, \quad i = 1, 2, \dots$$

Therefore, $\|T^*\| < 1$; applying Lemma 1, we can write $\rho(T^*) < 1$, which leads to convergence of MC method. $\square$

By a counterexample, we show that if $\|T\| \geq 1$ and $\rho(T) < 1$ the convergence condition (8) is not necessarily valid with the MAO and UM transition matrices. We can consider upper or lower triangular matrix T with $\|T\| \geq 1$ and diagonal elements, strictly less than 1, such that $\sum_{j=1}^n |t_{ij}| > 1$ is not valid for every row, such as

$$T = \begin{pmatrix} 0.75 & 0 & 0 & 0 \\ 0.2 & 0.5 & 0 & 0 \\ 0.34 & 0.2 & 0.25 & 0 \\ 1 & 4.2 & 2 & 0.34 \end{pmatrix}.$$

Using MATLAB software it can be seen that $\rho(T_{MAO}^*) = 2.5636$ and $\rho(T_{UM}^*) = 2.25$ while $\|T\| = 7.54$ and $\rho(T) = 0.75$.

Under the assumption $\|T\| < 1$ and consequently $\rho(T) < 1$ with the following counterexample, we can see the UM transition matrix does not satisfy conditions for stability. We consider the $n \times n$ matrix A in the form

$$a_{ij} = \begin{cases} a & \text{if } j = i, \\ -1 & \text{if } j = i + 1, \\ 2 & \text{if } j = i + 2, \\ -2 & \text{if } j = i - 1, \\ 1 & \text{if } j = i - 2, \\ 0 & \text{otherwise,} \end{cases}$$

and the diagonal component a is evaluated such that the matrix A is strictly diagonally dominant (SDD). Considering the splitting $A = D - B$, where $D = \text{diag}(a_{11}, \dots, a_{nn})$, the iteration matrix is $T = I - D^{-1}A$. It can be easily shown that $\|T\| < 1$. For $n = 9$ and $a = 7$, we can see that $\rho(T) = 0.7618$ and $\rho(T^*) = 1.6175$. In another case, $n = 15$ and $a = 10$, we obtain $\rho(T) = 0.5705$ and $\rho(T^*) = 1.4101$.

Continuing the convergence content, we pay attention on constructing the splitting $A = M - N$ so that $\rho(T) < 1$ for

convergence of Neumann series. Related to the type of matrix A , we can consider the splitting for matrix A such that $\rho(T) < 1$ and start the MC method. In the following, we mention the proper splitting for the SDD, M-matrix and Symmetric Positive Definite (SPD) matrices.

As mentioned for the SDD matrix A , applying $T = I - D^{-1}A$ is more preferable due to the simplicity of computing D^{-1} . Especially for the M-matrix A , we consider $T = I - D^{-1}A$ because $\rho(T) < 1$ (see Theorem 1.31 in [14]).

Now let us assume that A is an SPD matrix. Exerting the splitting $A = M - N$ on SLAE (1) such that $M^T + N$ is positive definite, we can deduce that $\rho(T) < 1$, where $T = M^{-1}N$ [14]. For the SPD matrix A , the splitting in the Gauss–Seidel iterative method leads to $\rho(T) < 1$, where $T = (D - E)^{-1}F$, D is the diagonal, $-E$ is the strictly lower part and $-F$ is the strictly upper part of A . It is also always assumed that the diagonal entries of A are all nonzero. If the matrix A is SPD, then for $0 < \omega < 2$, the splitting in the SOR iterative method will be convergent. In the SOR method, the iteration matrix is $T = (D - \omega E)^{-1}(\omega F + (1 - \omega)D)$. Eventually, for the SPD matrix A , we can employ the iteration matrix of the Richardson iterative method $T = I - \alpha A$, where α is a nonnegative scalar. In this case, the condition $0 < \alpha < \frac{2}{\lambda_{\max}}$ must be satisfied in order to satisfy $\rho(T) < 1$.

The most preferred iteration matrix is selected based on the structure of matrix A and rate of computational effort. In the general case, we can obtain an approximate inverse of the matrix A as preconditioner M and write (1) in the preconditioned form $MAx = Mb$. One of the least expensive methods to approximate M of large size is the MC method, which will be studied in the next section. After preconditioning, the preconditioned system requires only matrix-by-vector products to solve.

Moreover, in this paper we concentrate on affiliation of length of Markov chain (k) to $\rho(T^*)$; on decreasing $\rho(T^*)$ the length of Markov chains decreases. This result is illustrated in the numerical examples.

3. Modifications on the hybrid MC method to obtain accurate MI

Because of the importance of preconditioners theory, approximate inverse is a significant subject in solving linear systems using modern algorithms such as the Krylov subspace methods. Recently, the hybrid MC method was applied to approximate MI and its efficiency compared to those of the other numerical methods [7, 9, 10]. In this section, we provide some enhancements on the hybrid MC method for finding MI.

3.1 Preliminary MC method for MI

In this section, we describe an elementary version of the MC method to obtain MI. For any nonsingular matrix $A \in \mathbb{R}^{n \times n}$, it is obvious that

$$A(A^{-1})_{*j} = I_{*j}, \quad (9)$$

where j th column of the matrix A is denoted by double subscripts A_{*j} and I represents the $n \times n$ identity matrix. Assume that the matrix A can be split as $A = I - T$ so that $\|T\| < 1$. Writing (9) in the form

$$Ax = e_j, \quad (10)$$

and exerting the splitting $A = I - T$, we can apply the MC method to obtain j th column of A^{-1} ($j = 1, 2, \dots, n$). Since the vector f is replaced with e_j in the MC method (section 2.1), the estimator $X_i(\gamma_k) = \sum_{m=0}^k W_m f_{r_m}$ should be converted to the form $X_{ij}(\gamma_k) = \sum_{m|r_m=j}^k W_m$, where $(m|r_m=j)$ means that only W_m for which $r_m = j$ are included in the summation. Using the MC method n times repeatedly, the full inverse matrix is acquired, which can be independently implemented on parallel processors to attain high efficiency for large scale matrices. In Algorithm 2, we provide the ordered form of this method (unordered form is in [4, 5]).

Algorithm 2:

1. Input matrix T , parameters ϵ and δ
2. Compute $N = \lceil \frac{(0.6745)^2}{\delta^2(1-\|T\|^2)} \rceil + 1$ and probability transition matrix P
3. **for** $j = 1$ to n
4. **for** $i = 1$ to n
5. **for** $s = 1$ to N
6. Set $W_0 = 1, k = 0, point = i$
7. **if** $i = j$
8. Set $X_{ij}^{(s)} = W_0$
9. **end if**
10. **while** $|W_k| \geq \epsilon$
11. Generate an r.v. *nextpoint*, similar to Algorithm 1
12. Set $k = k + 1$
13. **if** $t_{point,nextpoint} \neq 0$, (this loop can be out of conditional in MAO)
14. Compute $W_k = W_{k-1} \frac{t_{point,nextpoint}}{p_{point,nextpoint}}$
15. **end if**
16. **if** $nextpoint = j$
17. Compute $X_{ij}^{(s)} = X_{ij}^{(s)} + W_k$
18. **end if**
19. Set $point = nextpoint$
20. **end while**
21. **end for**
22. Compute $X_{ij}(\gamma_k) = \frac{1}{N} \sum_{s=1}^N X_{ij}^{(s)}$ (ij th entry of matrix A^{-1})
23. **end for**
24. Set $A_{*j}^{-1} = X_{*j}(\gamma_k)$ (j th column of matrix A^{-1})
25. **end for**

The difference between Algorithms 1 and 2 is that Algorithm 2 includes Algorithm 1. In fact, in Algorithm 2 we use Algorithm 1 to find every column of the inverse matrix, which alone solves one SLAE using Algorithm 1. When applying Algorithm 2, related to the value of vector $f = e_j, j = 1, \dots, n$ the estimator $X_{ij}(\gamma_k)$ in the eighth and seventeenth rows of Algorithm 2 is different from its corresponding value for each component of the solution vector in the rows 6 and 15 of Algorithm 1. Since $f = e_j$ the number of Markov chains N for every entry of A^{-1} in row 2 of Algorithm 2 is different from its corresponding value in row 2 of Algorithm 1. We have two conditional loop in rows 7 and 16 of Algorithm 2 due to the assumption $f = e_j$ compared to Algorithm 1. The j th column of matrix A^{-1} is obtained in row 24 and the i th component of j th column is computed in row 22 of Algorithm 2. Generating the r.v. *nextpoint* in Algorithm 2 (row 11) is explained in the eighth to twelfth rows of Algorithm 1.

3.2 Changes on the MC method to find MI under SDD property

In this section, we study the MC method to approximate the inverse of SDD matrices. This method employs underlying splitting for the SDD matrix, which is efficient in terms of complexity of computing and assurance of the condition $\|T\| < 1$. Initially, we briefly describe the MC method to approximate MI of SDD matrix that has been used so far. Then we create a change on this method. Finding the inverse of the SDD matrix A by the previous MC method is in the following way [3, 5, 8]. After considering the splitting $A = D - (D - A)$ for the SDD matrix A and iteration matrix $T = I - D^{-1}A$, which leads to $\|T\| < 1$, $(I - T)^{-1}$ is approximated via Algorithm 2 and then A^{-1} is computed by matrix multiplication $(I - T)^{-1}D^{-1}$ (based on the old version of the MC method for MI of SDD matrices).

In this study, we present a new technique in Algorithm 3 to obtain the inverse of SDD matrix A , which leads to higher accuracy and requires less computational effort. Here, we consider $T = I - D^{-1}A$ and $f = D^{-1}e_j$; therefore, matrix multiplication is not needed to compute MI. Old version of Algorithm 3 was presented in [3].

Algorithm 3:

1. Input matrix T , parameters ϵ and δ
2. Compute $N = \lceil [(\frac{0.6745}{\delta})^2 (\frac{\min_i a_{ii}}{1-\|T\|})^2] + 1 \rceil$ and probability transition matrix P
3. **for** $j = 1$ to n
4. **for** $i = 1$ to n
5. **for** $s = 1$ to N
6. Set $W_0 = 1, k = 0, point = i$
7. **if** $i = j$
8. Set $X_{ij}^{(s)} = \frac{W_0}{a_{jj}}$
9. **end if**
10. **while** $|W_k| \geq \epsilon$
11. Generate an r.v. $nextpoint$, similar to Algorithm 1
12. Set $k = k + 1$
13. **if** $t_{point,nextpoint} \neq 0$, (this loop can be out of conditional in MAO)
14. Compute $W_k = W_{k-1} \frac{t_{point,nextpoint}}{p_{point,nextpoint}}$
15. **end if**
16. **if** $nextpoint = j$
17. Compute $X_{ij}^{(s)} = X_{ij}^{(s)} + \frac{W_k}{a_{jj}}$
18. **end if**
19. Set $point = nextpoint$
20. **end while**
21. **end for**
22. Compute $X_{ij}(\gamma_k) = \frac{1}{N} \sum_{s=1}^N X_{ij}^{(s)}$ (ij th entry of matrix A^{-1})
23. **end for**
24. Set $A_{*j}^{-1} = X_{*j}(\gamma_k)$ (j th column of matrix A^{-1})

Here, we describe the difference between Algorithm 3 and its unmodified version. In the unmodified version, first $(I - T)^{-1}$ is approximated by the MC method represented in Algorithm 2 because $\|T\| < 1$. Then A^{-1} is computed by matrix multiplication $(I - T)^{-1}D^{-1}$ as $T = I - D^{-1}A$. However, in Algorithm 3, the j th column of matrix A^{-1} is approximated by the MC method for solving SLAE on rows 4–22 of Algorithm 3, which is similar to Algorithm 1. In this way, we assume the iteration matrix $T = I - D^{-1}A$ and $f = D^{-1}e_j$ for $j = 1, 2, \dots, n$. The effect of these assumptions can be seen in rows 2, 8 and 17 of Algorithm 3. Since $f = D^{-1}e_j$ we have two conditional loop in rows 7 and 16 of Algorithm 3. Therefore, Algorithm 3 not only gives us the right approximation of the solution, but also reduces the accumulation of error and time spent compared with the unmodified version. In the rows 8 and 17 of Algorithm 3 related to the vector $f = D^{-1}e_j$, the estimator $X_{ij}(\gamma_k)$ is different from Algorithm 2 (rows 8 and 17). Moreover, in Algorithm 3 the number of Markov chains N is different from its corresponding value in Algorithm 2.

3.3 Corrections on the hybrid MC method to obtain MI in general

The reason for naming the MC method as hybrid is that this method is a combination of the MC method with the

deterministic procedures. In this section, we investigate on hybrid MC method for finding the MI in general and provide some modifications and corrections on it to increase efficiency.

We suppose that $A \in \mathbb{R}^{n \times n}$ is an arbitrary nonsingular matrix in (10). It is comfortable to indicate that the matrix $B = A + k\|A\|I$ is SDD for the scalar $k > 1$. In the special case $k = 1$, matrix B can be SDD when the row that determines the value of infinity norm of matrix A includes the positive diagonal entry. The approach to approximate inverse of the nonsingular matrix A in general, investigated by many authors in [4, 5, 7, 8], is the hybrid MC method, which is implemented in multiple stages. In the first step, the inverse of matrix B is approximated via the old version of Algorithm 3 and then the precision of the approximate inverse is elevated by the iterative filter process. The filter procedure is started by setting $B_0 = B^{-1}$ (approximate B^{-1} obtained from the old version of Algorithm 3). Then it is continued in succession as $B_{i+1} = B_i(I + R_i)$ and $R_i = I - BB_i$, which, if it is convergent, will continue to the validity of stopping condition $\|R_i\| < \gamma$, where γ is the given parameter. Therefore the initial approximate inverse B_0 is refined to get B_i ($i = 1, 2, \dots$) with high accuracy associated to the underlying stopping condition in filter approach. In the final step we set $A_n^{-1} = B_n$ (B_n obtained from the filter process) and $A^{-1} = A_0^{-1}$; A_0^{-1} is

the approximated inverse of A from the following backward-correction procedure [4]:

$$A_{i-1}^{-1} = A_i^{-1} + \frac{A_i^{-1} S_i A_i^{-1}}{1 - \text{trace}(A_i^{-1} S_i)}, \quad i = n, n-1, \dots, 1. \quad (11)$$

In Eq. (11), $S = \sum_{i=1}^n S_i$ is a diagonal matrix with the same diagonal entries $k\|A\|$, i.e., $S = k\|A\|I$ and S_i is a matrix with only one non-zero component $s_{ii} = k\|A\|$.

In this work, we exert three amendments in the mentioned hybrid MC method. First of all, we concentrate on the convergence of the iterative filter process. Secondly, we add one conditional step in the backward iterative relation (11) to prevent the failure of the MC algorithm when $1 - \text{trace}(A_i^{-1} S_i)$ is close to zero in i th stage. Finally, since the approximate inverse of the SDD matrix B , obtained from Algorithm 3, has a relatively proper accuracy, we employ the filter procedure after applying the process (11).

Lemma 2 *Let A_0 be an initial approximate of A^{-1} and R_0 be a residual matrix such that $R_0 = I - AA_0$. Successive iteration*

$$A_{i+1} = A_i(I + R_i), \quad i = 1, 2, \dots, \quad (12)$$

where $R_i = I - AA_i$ is convergent (i.e., residual matrix R_i converges to zero) if and only if $\rho(R_0) < 1$.

Proof By invoking Eq. (12) and induction, we can rewrite R_i in the following way:

$$\begin{aligned} R_i &= I - AA_i = I - AA_{i-1}(I + R_{i-1}) = I - (I - R_{i-1})(I + R_{i-1}) \\ &= R_{i-1}^2 = (I - AA_{i-1})^2 = (I - AA_{i-2}(I + R_{i-2}))^2 = R_{i-2}^2 \\ &= \dots = R_0^{2^i}. \end{aligned}$$

On the other hand $\lim_{i \rightarrow \infty} A_i = A^{-1} \lim_{i \rightarrow \infty} (I - R_0^{2^i})$, where $\lim_{i \rightarrow \infty} R_0^{2^i} = 0$ if and only if $\rho(R_0) < 1$ [14], which completes the proof. $\square$

As a result, we check the spectral radius of initial residual matrix $\rho(R_0)$ instead of $\|R_0\|$, considered in [4, 5],

for the convergence of the filter procedure. Because of the stronger condition of Lemma 2, the filter method may be convergent for $\|R_0\| \geq 1$, which is shown in numerical examples.

Contrary to the apparent computational complexity of (11), the calculation of this relation is simpler than that shown. Based on the structure of matrix S_i we apply simplified form of (11) in Algorithm 4. In continuation, we remind the following basic Theorem 3 related to the recurrence relation (11).

Theorem 3 ([17]) *Let S be an $n \times n$ diagonal matrix given by $S = \sum_{i=1}^n S_i$. Then Eq. (11) provides*

$$1 - \text{trace}(A_i^{-1} S_i) \neq 0, \quad i = n, n-1, \dots, 1. \quad (13)$$

Theorem 3 essentially expresses that the recurrence relation (11) achieves the desired solution successfully if $1 - \text{trace}(A_i^{-1} S_i)$ is nonzero for every i th stage. In practice even if $1 - \text{trace}(A_i^{-1} S_i)$ is close to zero, (11) will not implemented successfully due to the accumulation of error. Thence, to avoid the failure of the hybrid MC method, we add a conditional loop to this method in row 9 of Algorithm 4. According to row 9, if (13) is satisfied in the i th step then the recurrence relation (11) is executed (rows 10–15 of Algorithm 4); otherwise, the current step is shifted to the next step in row 17 and recurrence relation (11) is implemented in the next step (rows 10–15). We propound the improved hybrid MC method, considered with mentioned three corrections, in Algorithm 4. Unmodified version of Algorithm 4 was presented in [3–5, 7, 8]. In Algorithm 4, the SDD matrix B is constructed from the arbitrary non-singular matrix A in row 2. The iteration matrix T is computed in row 3 and Algorithm 3 is applied to obtain B^{-1} in row 4. We employ the recurrence relation (11) to approximate A^{-1} from B^{-1} in rows 5–18. Simplified form of Eq. (11) is represented in row 15. In the conditional loop (row 9), Theorem 3 is exerted. In rows 19–27 the iterative filter method is employed to acquire high accuracy. Finally, row 28 is assigned to the desired output algorithm.

Algorithm 4:

1. Input matrix A , parameters γ and k
2. Compute $B = A + k\|A\|I$ (construct SDD matrix)
3. Compute $T = I - D^{-1}B$ ($D = \text{diag}(b_{11}, b_{22}, \dots, b_{nn})$)
4. Compute B^{-1} by invoking Algorithm 3
5. Compute A_0 (approximate inverse of A) from B^{-1} pursuant to (11) as:
6. Set $t = n$ and $A_t = B^{-1}$
7. **while** $t \geq 1$
8. Compute $s = 1 - k\|A\|A_t(t, t)$
9. **if** $s \neq 0$
10. **for** $i = 1$ to n
11. **for** $j = 1$ to n
12. Compute $G(i, j) = k\|A\|A_t(i, t)A_t(t, j)$
13. **end for**
14. **end for**
15. Compute $A_{t-1} = A_t + \frac{1}{s}G$
16. **end if**
17. Set $t = t - 1$
18. **end while**
19. Refine A_0 by the filter process as the following steps:
20. Compute $R_0 = I - AA_0$
21. Set $\text{iter} = 0$ (iteration number of filter process)
22. **if** $\rho(R_0) < 1$
23. **while** $\|R_0\| \geq \gamma$
24. Set $A_0 = A_0(I + R_0)$ and $R_0 = I - AA_0$
25. Set $\text{iter} = \text{iter} + 1$
26. **end while**
27. **end if**
28. Set $A^{-1} = A_0$

4. Numerical examples

In this section, we check some numerical examples to validate the theoretical results. All numerical experiments were computed in double precision using some MATLAB codes on a Pentium 4 PC, with a 2.10 GHz CPU and 4 GB of RAM. It is evident that the MC method for matrices that are not large is expensive and time-consuming process than

deterministic methods. Efficiency and superiority of the MC method are apparent when implementing on parallel processors for large-scale matrices. However, we try to verify the corrections made on the MC algorithms on the following simple examples.

Example 1 In this example, we intend to illustrate the dependence of k , N , $\rho(T_{MAO}^*)$ and $\rho(T_{UM}^*)$ to $\|T\|$ as mentioned in sections 2.1 and 2.2. We suppose a matrix

Table 1. MC method to solve SLAE with the coefficient M -matrix A for different values of a and $\epsilon, \delta = 0.001$, $n = 30$.

a	$\ T\ $	N	k_1	k_n	k_{mean}	$\rho(T)$	$\rho(T_{MAO}^*)$	$\rho(T_{UM}^*)$	Error
5	0.6	1820001	9	13	13	0.5628	0.3375	4.7754	5.0687×10^{-4}
10	0.3	752143	4	6	6	0.2814	0.0844	1.1938	2.1870×10^{-4}
15	0.2	619306	4	5	5	0.1876	0.0375	0.5306	1.4351×10^{-4}
20	0.15	568357	3	4	4	0.1407	0.0211	0.2985	7.5937×10^{-5}
25	0.12	541488	3	4	4	0.1126	0.0135	0.1910	4.1371×10^{-5}
30	0.1	524904	3	3	4	0.0938	0.0094	0.1326	5.9274×10^{-5}
35	0.0857	513653	3	3	3	0.0804	0.0069	0.0975	5.3978×10^{-5}
40	0.0750	505519	3	3	3	0.0703	0.0053	0.0746	3.1641×10^{-5}
50	0.06	494548	3	3	3	0.0563	0.0034	0.0478	1.2960×10^{-5}
100	0.03	473957	2	2	2	0.0281	8.4379×10^{-4}	0.0119	1.2310×10^{-5}

$A \in \mathbb{R}^{n \times n}$, with $a_{ij} \leq 0$ for $i \neq j$ and $a_{ii} > 0$. We know that if A is SDD matrix, then A is the M-matrix [18]. In this example, we let $A = \text{tridiag}(-2, a, -1)$ for different values of n and a , so that A is an SDD matrix. We employ the MC method to solve SLAE $Ax = b$, where the vector b is computed such that the exact solution is supposed to be a vector all of whose elements are one. Obtained numerical results are shown in table 1 for the matrix A of order 30×30 with various values of a . In this table, k_1 and k_n represent the required length of Markov chain based on associated stopping condition for the first and n th component. Moreover, k_{mean} indicates the average of the required length of Markov chain for n entries. As we observe from table 1, by increasing a other parameters are reduced such as k_{mean} , N , $\rho(T_{MAO}^*)$ and $\|T\|$. Hence, the convergence of the MC method occurs more quickly. In these results, $Error = \|x - x_{exact}\|$ declines on increasing a , where x and x_{exact} are the approximate solution from the MAO and exact solution, respectively. In continuation, we obtain similar behaviour of numerical results for the matrix A of order 50×50 with various values of a in table 2.

Because of $\|T_{MAO}^*\|$ dependence on $\|T\|$, in practice the MAO method is used more than UM. In a clearer explanation, on decreasing $\|T\|$, $\|T_{MAO}^*\|$ is reduced and thus $\rho(T_{MAO}^*)$ reduces. Therefore, based on Theorem 4.2 in [12] the variance of the estimator $X(\gamma)$ converges to a finite value with smaller length of Markov chain. Consequently, the convergence of the MAO method is accelerated. However, this dependence is not seen in the UM method. On the other hand, from (7), on decreasing $\|T\|$, the number of Markov chains N is reduced. Hence, the speed of the convergence of the MC method increases. As a result, by reducing the number and length of Markov chains, the MC method converges rapidly.

Example 2 In this example, we employ the modified MC method (Algorithm 3) and its old version to obtain approximate inverse of the SDD matrices and compare their

performance. We consider the following nonsingular SDD matrices A , B and C . We suppose that $Error_1$ and $Error_2$ are the infinity norm of the residual matrix for the modified MC method and old version of it, respectively. For the matrix A we have $Error_1 = 0.0021$ and $Error_2 = 0.0117$. Also, $Error_1 = 0.0033$ and $Error_2 = 0.0293$ were calculated for the matrix B . Finally, for the matrix C , $Error_1 = 0.0013$ and $Error_2 = 0.0115$ were computed. In continuation of this example, we assume the matrix A , defined in section 2.2, to be of higher order. We consider different values of n and a such that the matrix A is an SDD. In figure 1 the results of $Error_1$ and $Error_2$ are illustrated for various values of diagonal element a and δ , $\epsilon = 0.01$, $n = 30$. The results of $Error_1$ and $Error_2$ are shown in figure 2 for different values of diagonal element a and δ , $\epsilon = 0.001$, $n = 50$.

Figure 1. Infinity norm of the residual matrix of the modified MC method and its old version to obtain MI for δ , $\epsilon = 0.01$ and $n = 30$.

Table 2. MC method to solve SLAE with the coefficient M-matrix A for different values of a and ϵ , $\delta = 0.001$, $n = 50$.

a	$\ T\ $	N	k_1	k_n	k_{mean}	$\rho(T)$	$\rho(T_{MAO}^*)$	$\rho(T_{UM}^*)$	$Error$
5	0.6	1820001	9	13	14	0.5646	0.3387	7.9848	5.0678×10^{-4}
10	0.3	752143	5	6	6	0.2823	0.0847	1.9962	2.1870×10^{-4}
15	0.2	619306	4	5	5	0.1882	0.0376	0.8872	1.4345×10^{-4}
20	0.15	568357	4	4	4	0.1412	0.0212	0.4990	7.5937×10^{-5}
25	0.12	541488	3	4	4	0.1129	0.0135	0.3194	4.1319×10^{-5}
30	0.1	524904	3	3	4	0.0941	0.0094	0.2218	5.9269×10^{-5}
35	0.0857	513653	3	3	3	0.0807	0.0069	0.1630	5.3978×10^{-5}
40	0.0750	505519	3	3	3	0.0706	0.0053	0.1248	3.1641×10^{-5}
45	0.0667	499366	3	3	3	0.0627	0.0042	0.0986	1.9753×10^{-5}
50	0.06	494548	3	3	3	0.0565	0.0034	0.0798	1.2960×10^{-5}
100	0.03	473957	2	2	2	0.0282	8.4683×10^{-4}	0.0200	1.170×10^{-5}

Figure 2. Infinity norm of the residual matrix of the modified MC method and its old version to obtain MI for $\delta, \epsilon = 0.001$ and $n = 50$.

$$A = \begin{pmatrix} 3 & 1 & 0.1 & 0 & -1 \\ 0 & 4 & 2.1 & 1 & 0.04 \\ 1 & -0.1 & 3 & 0.2 & 1 \\ 1 & 0.2 & -0.3 & 3 & 0 \\ 0 & 1 & -1 & 1.1 & 5 \end{pmatrix},$$

$$B = \begin{pmatrix} 4 & 1 & 2 & 0.1 & 0 \\ 1 & 5 & 0.01 & -1 & 1 \\ 0 & -0.3 & 3 & 1 & -1 \\ 1 & 0.1 & -0.2 & -3 & 0 \\ 0 & 1 & -1 & -0.2 & 4 \end{pmatrix},$$

$$C = \begin{pmatrix} 2 & 0 & 0.01 & 0.1 & -1 \\ 0 & 4 & 1 & -0.3 & 1 \\ 1 & -0.3 & 5 & 1 & 0 \\ 0 & 1 & -0.04 & 4 & -1 \\ 0 & -1 & -0.1 & 1 & 3 \end{pmatrix}.$$

As we observe from the numerical results of this example, the modified MC method to obtain inverse of

the SDD matrix (Algorithm 3) is more accurate than its old version (which was presented in [3]).

Example 3 In this example, we apply the hybrid MC method and its modified version (Algorithm 4) to obtain approximate MI in general and compare their performance and numerical results. We consider the following nonsingular matrices:

$$A = \begin{pmatrix} 1 & -1 & 0 & 2 & 0.1 \\ 2.1 & 3 & 1 & -1 & 0 \\ 0 & 1 & -2 & 0.1 & 2.1 \\ 1.1 & 1.2 & 2 & 0 & -1 \\ 1 & 0 & 2 & -1 & -2 \end{pmatrix},$$

$$B = \begin{pmatrix} 0.1 & 2 & -1 & 3 & 4 \\ 0 & 1 & 2 & -3 & 0.01 \\ 0.02 & 3 & -1 & 2 & 0 \\ 0 & 1 & 2 & 3 & 0.03 \\ 1 & -1 & -2 & 3 & 0 \end{pmatrix}.$$

We execute Algorithm 4 for various values of parameters δ, ϵ, γ and k as shown in table 3, for matrix A . For all values of parameter k , the convergence condition $\rho(T^*) < 1$ is valid to find the inverse of the SDD matrix $A + k\|A\|I$ (obtained from the general nonsingular matrix A) by the MAO method. In table 3, we set $Error_1 = \|I - A'A\|$, where A' is the approximate inverse obtained from the backward-correction procedure (11) (regardless of Theorem 3 and the conditional loop in row 9 of Algorithm 4). Likewise, $Error_2$ and $Error_3$ are the infinity norm of the residual matrix for the modified version of the backward-correction procedure (11) (presented in Algorithm 4 based on Theorem 3) and the filter method, respectively. In table 3, “Iter” represents the iteration number of the filter approach; on decreasing $\rho(R_0)$ it also declines. From table 3 we observe that modified version of (11) in Algorithm 4 has been successfully implemented in places where backward-correction procedure (11) failed, e.g., for $k = 2, 10, 50$. Also, the filter method converges only in places where the condition $\rho(R_0) < 1$ is satisfied, which

Table 3. Comparison between the hybrid MC method and its modified version for the matrix A .

k	δ	ϵ	γ	$\ T\ $	$\rho(T)$	$\rho(T^*)$	$Error_1$	$Error_2$	$\rho(R_0)$	Iter	$Error_3$
2	0.1	0.1	10^{-3}	0.3732	0.2384	8.14×10^{-2}	Fail	2.6883	0.9645	9	0.0017
5	10^{-2}	10^{-2}	10^{-3}	0.1493	0.0889	1.24×10^{-2}	17.9731	17.9731	0.9477	8	0.0213
8	10^{-3}	10^{-3}	10^{-5}	0.0933	0.0547	4.8×10^{-3}	4.6870	4.6870	0.7095	7	0
10	10^{-2}	10^{-2}	10^{-3}	0.0746	0.0435	3.1×10^{-3}	Fail	19.1497	1.0122	Fail	Fail
15	10^{-3}	10^{-3}	10^{-4}	0.0498	0.0288	1.4×10^{-3}	12.5541	12.5541	0.7337	7	0
20	10^{-2}	10^{-2}	10^{-3}	0.0373	0.0215	7.6792×10^{-4}	18.4211	18.4211	0.9635	9	0.0017
25	10^{-4}	10^{-4}	10^{-5}	0.0299	0.0172	4.9139×10^{-4}	7.7328	7.7328	0.4238	5	0
50	0.1	0.1	10^{-3}	0.0149	0.0085	1.2283×10^{-4}	Fail	17.9689	1.7719	Fail	Fail

Table 4. Comparison between the hybrid MC method and its modified version for the matrix B .

k	δ	ϵ	γ	$\ T\ $	$\rho(T)$	$\rho(T^*)$	$Error_1$	$Error_2$	$\rho(R_0)$	$Iter$	$Error_3$
2	10^{-3}	10^{-3}	10^{-4}	0.4926	0.1636	4.67×10^{-2}	Fail	0.1952	0.0627	3	4.3152×10^{-3}
5	0.1	0.1	10^{-2}	0.1976	0.0659	7.6×10^{-3}	Fail	6.0119	0.9873	10	63×10^{-4}
7	10^{-3}	10^{-3}	10^{-5}	0.1412	0.0471	3.9×10^{-3}	0.5065	0.5065	0.0959	4	1.6993×10^{-4}
9	10^{-3}	10^{-3}	10^{-4}	0.1099	0.0367	2.4×10^{-3}	0.5529	0.5529	0.5529	4	1.6980×10^{-4}
10	10^{-3}	10^{-3}	10^{-4}	0.0989	0.0330	1.9×10^{-3}	0.5607	0.5607	0.2587	4	4.3334×10^{-5}
15	10^{-3}	10^{-3}	10^{-4}	0.0660	0.0221	8.5960×10^{-4}	3.3225	3.3225	0.6662	6	1.6998×10^{-4}
20	10^{-3}	10^{-3}	10^{-4}	0.0495	0.0165	4.8434×10^{-4}	0.5940	0.5940	0.1077	4	1.6990×10^{-4}
30	10^{-3}	10^{-3}	10^{-4}	0.0330	0.0110	2.1563×10^{-4}	1.5815	1.5815	0.5734	6	1.6993×10^{-4}
35	10^{-4}	10^{-4}	10^{-5}	0.0283	0.0095	1.5850×10^{-4}	0.2898	0.2898	0.0444	4	1.6993×10^{-4}
40	10^{-3}	10^{-3}	10^{-4}	0.0247	0.0083	1.2140×10^{-4}	Fail	3.5420	1.4062	Fail	Fail
50	10^{-2}	10^{-2}	10^{-3}	0.0198	0.0066	7.7737×10^{-5}	Fail	13.3746	1.5426	Fail	Fail
100	10^{-2}	10^{-2}	10^{-4}	0.0099	0.0033	1.9455×10^{-5}	Fail	12.0044	2.0439	Fail	Fail

Table 5. Comparison between the hybrid MC method and its modified version for some matrices of the Matrix Market website.

Matrix	k	ϵ	δ	γ	$Error_1$	$Error_2$	$\rho(R_0)$	$Iter$	$Error_3$
ARC130	2	10^{-3}	10^{-3}	10^{-3}	20.3930	20.3930	0.2471	5	5.9884×10^{-4}
ARC130	10	10^{-3}	10^{-3}	10^{-3}	25.8090	25.8090	0.0720	5	6.6818×10^{-11}
ARC130	20	10^{-3}	10^{-3}	10^{-3}	22.9850	22.9850	0.1131	5	8.6854×10^{-9}
BFW62A	2	10^{-3}	10^{-3}	10^{-3}	7.1307	7.1307	0.4147	5	1.3880×10^{-5}
BFW62A	7	10^{-3}	10^{-3}	10^{-3}	34.4526	34.4526	1.3049	Fail	Fail
BP200	5	10^{-3}	10^{-3}	10^{-3}	Fail	14.8683	1.0044	Fail	Fail
BP200	30	10^{-3}	10^{-3}	10^{-3}	Fail	14.8582	1.0007	Fail	Fail
BP200	49	10^{-3}	10^{-3}	10^{-3}	Fail	14.8574	1.0004	Fail	Fail
DWB512	5	10^{-2}	10^{-2}	10^{-3}	0.0939	0.0939	0.0248	3	2.3612×10^{-6}
DWB512	20	0.1	0.1	10^{-3}	1.6461	1.6461	0.3020	5	2.6199×10^{-8}
FS1831	10	10^{-3}	10^{-3}	10^{-3}	45.5682	45.5682	0.7262	7	3.0554×10^{-7}
FS1831	20	10^{-3}	10^{-3}	10^{-3}	72.1273	72.1273	0.7303	7	2.4970×10^{-7}

numerically validates Lemma 2. For example, in the second row of table 3 ($k = 5$) we have $\rho(R_0) = 0.9477$ while $\|R_0\| = 2.7800$, which leads to convergence of the filter method and better approximate of A^{-1} .

For the matrix B , the corresponding results are shown in table 4. The behaviour of the numerical results in table 4 is similar to that in table 3. We observe that by properly selecting the parameters δ , ϵ and γ , the accuracy of the obtained MI can be controlled.

It should be noted that due to the random nature of the MC method, each time the codes are executed, the initial approximations in (11) will also be different.

Example 4 In this example, we consider some square fairly large matrices taken from the well-known collection of the matrices in the Matrix Market website [19]. We apply the hybrid MC method and its modified version (Algorithm 4) to obtain approximate MI in general and compare their numerical results in table 5. We employ the

nonsingular real matrices ARC130, BFW62A, BP 200, DWB512 and FS 1831 of order $n = 130, 62, 822, 512$ and 183, respectively. In table 5, $Error_1$, $Error_2$, $Error_3$, $\rho(R_0)$ and “Iter” are defined as in Example 3. Similar to Example 3 we observe that the modified hybrid MC method has been successfully implemented in places where its unmodified version failed, e.g., matrix BP 200. Moreover, the iterative filter method failed in cases where $\rho(R_0) < 1$ is not valid (even in bigger and very close to 1 case), e.g., the ninth row of table 5, which numerically verify Lemma 2.

5. Conclusion

We first studied the MC method for solving SLAE with new results. Later, we investigated the MC and hybrid MC method with some modifications for obtaining MI. Under some specific conditions based on the transition probability matrix, we analysed the convergence of the MC method. One important point that we have concluded is that the

transition probability matrix of the MAO and UM methods does not necessarily lead to convergence, and the necessary and sufficient condition should be checked for the convergence of the MC method. Numerical results show that the modified versions of the MC and hybrid MC methods are superior to their unmodified form.

Acknowledgements

The authors would like to thank the reviewer(s) and the corresponding editor, Prof. Manoj Kumar Tiwari, for their valuable comments and suggestions in the significant improvement of the manuscript.

References

- [1] Drovandi C C, Pettitt N A and Henderson D A 2014 Marginal reversible jump Markov chain Monte Carlo with application to motor unit number estimation. *Comput. Statist. Data Anal.* 72: 128–146
- [2] Veselinović M A, Velimorović D, Kaličanin B, Toropova A, Toropov A and Veselinović J 2017 Prediction of gas chromatographic retention indices based on Monte Carlo method. *Talanta* 168: 257–262
- [3] Fathi-Vajargah B 2007 Different stochastic algorithms to obtain matrix inversion. *Appl. Math. Comput.* 189: 1841–1846
- [4] Fathi-Vajargah B 2007 New advantages to obtain accurate matrix inversion. *Appl. Math. Comput.* 189: 1798–1804
- [5] Alexandrov V, Atanassov E, Dimov I, Branford S, Thandavan A and Weihrauch C 2005 Parallel hybrid Monte Carlo algorithms for matrix computations. In: *Proceedings Part III of the 5th International Conference*, Atlanta, GA, USA, May 22–25, Lecture Notes in Computer Science 3516, pp. 744–752
- [6] Benzi M, Evans T M, Hamilton S P, Pasini M L and Slattery S R 2016 Analysis of Monte Carlo accelerated iterative methods for sparse linear systems. *Numer. Linear Algebra Appl.* 24: 1–28
- [7] Alexandrov V and Esquivel-Flores O A 2015 Towards Monte Carlo preconditioning approach and hybrid Monte Carlo algorithms for matrix computations. *Comput. Math. Appl.* 70: 2709–2718
- [8] Branford S, Sahin C, Thandavan A, Weihrauch C, Alexandrov V and Dimov I T 2008 Monte Carlo methods for matrix computations on the grid. *Future Gener. Comput. Syst.* 24: 605–612
- [9] Davila D, Alexandrov V and Esquivel-Flores O A 2016 On Monte Carlo hybrid methods for linear algebra. In: *Proceedings of the 7th Workshop on Latest Advances in Scalable Algorithms for Large-scale Systems*, Spain
- [10] Strassburga J and Alexandrov V 2013 A Monte Carlo approach to sparse approximate inverse matrix computations. *Proc. Comput. Sci.* 18: 2307–2316
- [11] Dimov I, Maire S and Sellier J M 2015 A new walk on equations Monte Carlo method for solving systems of linear algebraic equations. *Appl. Math. Model.* 39: 4494–4510
- [12] Ji H and Li Y 2012 Reusing random walks in Monte Carlo methods for linear systems. *Proc. Comput. Sci.* 9: 383–392
- [13] Okten G 2005 Solving linear equations by Monte Carlo simulations. *SIAM J. Sci. Comput.* 27: 511–531
- [14] Saad Y 2003 *Iterative methods for sparse linear systems*. Philadelphia: SIAM
- [15] Sobol I M 1975 *The Monte Carlo method*. Chicago: The University of Chicago Press
- [16] Ji H, Mascagni M and Li Y 2013 Convergence analysis of Markov chain Monte Carlo linear solvers using Ulam–Von Neumann algorithm. *SIAM J. Numer. Anal.* 51: 2107–2122
- [17] Sen S K and Krishnamurthy E V 1974 Rank-Augmented LU-Algorithm for computing generalized matrix inverses. *IEEE Trans. Comput.* 23: 199–201
- [18] Axelsson O 1996 *Iterative solution methods*. Cambridge: Cambridge University Press
- [19] Website: <http://math.nist.gov/MatrixMarket/matrices.html>