Computational Methods for Differential Equations
http://cmde.tabrizu.ac.ir
Vol. 9, No. 1, 2021, pp. 159-179
DOI:10.22034/cmde.2020.30640.1453

A new Monte Carlo method for solving system of linear algebraic equations

Behrouz Fathi-Vajargah
Department of Statistics, Faculty of Mathematical Sciences,
University of Guilan, P.O. Box: 41335-19141, Rasht, Iran.
E-mail: fathi@guilan.ac.ir, behrouz.fathi@gmail.com

Zeinab Hassanzadeh*
Department of Applied Mathematics, Faculty of Mathematical Sciences,
University of Guilan, P.O. Box: 41335-19141, Rasht, Iran.
E-mail: hassanzadeh@phd.guilan.ac.ir, hassanzadeh.83@gmail.com

Abstract In this paper, we firstly study the employing of the Monte Carlo method for solving system of linear algebraic equations and then analyze on convergence of this method. We propound new results related to the convergence of the Monte Carlo method. Additionally, we introduce a new Monte Carlo algorithm with effective techniques. Finally, we compare the efficiency of new Monte Carlo algorithm with its old version in the numerical experiments.

Keywords. System of linear algebraic equations, Monte Carlo method, Transition probability matrix, Spectral radius, Ergodic Markov chain.

2010 Mathematics Subject Classification. 65L05, 34K06, 34K28.

1. Introduction

Solving a System of Linear Algebraic Equations (SLAE) is one of the substantial problems for researchers because of its widespread applications in engineering and sciences problems such as dynamical systems, circuit analysis, digital signal processing, financial or stochastic modelling and physical problems. The long-standing Monte Carlo (MC) method provides alternative option for us in selecting a solving method for SLAEs. Since today's need for large scale systems has incremented, hence researchers' tendency to this method has increased. This is because not only MC method has not the sequential nature but also this method has the parallelism capability on parallel processors in comparison to the modern deterministic linear solvers.

Exclusive advantages of MC method for solving SLAE and the authors' motivation for study in this area have been explained in [13]. Over the past years, we have observed extensive applications of the MC method in matrix computations by researchers. They improved the MC algorithms for solving SLAE and obtaining Matrix Inversion (MI) theoretically and numerically. In [11, 12], Fathi-Vajargah first introduced the hybrid MC algorithm for MI. He indicated that the MC approximation of

Received: 1 December 2018 ; Accepted: 19 December 2019.
∗ corresponding.

the MI can be combined with a deterministic refinement method to achieve higher precision and the results can be improved up to the exact inversion. Indeed, the flexibility of the MC method to reach the result with arbitrary precision is the main point of this method. Hybrid MC method more discussed to increase the speed of its computation with parallel computing by Alexandrov et al. in [1]. MC acceleration schemes combined with the original preconditioned Richardson iterative method in [4]. An enhanced hybrid method has been recently employed to construct an efficient stochastic preconditioner for solving SLAE and obtaining MI. Also, this preconditioner is based on Markov Chain Monte Carlo (MCMC) method to estimate a Sparse Approximate Inverse (SPAI) [2, 5, 8, 26]. Dimov et al. in [9] propounded a new technique for randomly selecting the indices of the iteration matrix, where the rate of convergence depends on the balancing of iteration matrix. Recently, Fathi-Vajargah et al. have exerted improvements on the hybrid MC method for matrix computations [13]. They have introduced a new version of the MC method for the real and complex fuzzy system of linear algebraic equations in [14].

In this paper, we study the background of MC method for solving SLAE. We more discuss on the convergence of this method and intend to present the applicable condition for convergence of MC method based on the type of transition probability matrix, especially for large scale matrices. In addition, we provide a new MC algorithm for SLAE, which is the main purpose of this work.

This article is organized as follows. After studying the necessary details of the MC method for SLAEs in Section 2, new results on the convergence of MC method are given in Section 3. We introduce a new MC algorithm in Section 4. Section 5 validates the convergence results and compares the efficiency of new MC algorithm with its old version in numerical examples. Finally, Section 6 ends this paper with conclusion.

2. Background of the MC method for solving SLAE

We consider the SLAE

$$Ax = b, \tag{2.1}$$

where $A \in \mathbb{R}^{n\times n}$ is the nonsingular matrix. Also, $b \in \mathbb{R}^n$ is the known vector so that every element of it satisfies $b_i \neq 0$ for all i. To start the MC method, we generally consider the splitting $A = M - N$ for the SLAE (2.1). Then we can recast (2.1) in the following iterative scheme

$$x^{(k+1)} = Tx^{(k)} + f, \quad k = 0, 1, \ldots, \tag{2.2}$$

in which $T = M^{-1}N$ and $f = M^{-1}b$. We know that the iterative relation (2.2) converges to the exact solution of the SLAE (2.1) regardless of the initial guess $x^{(0)}$ if and only if $\rho(T) < 1$ [24]. In the case $||T|| < 1$ for some matrix norm $||.||$, $I - T$ is nonsingular and the iteration (2.2) converges for any initial vector $x^{(0)}$ [24]. It is noted that the used matrix norm in this paper, is considered as the matrix ∞-norm.

Therefore, from (2.2) x can be written in the following form (Neumann series)

$$x = \sum_{k=0}^{\infty} T^k f. \tag{2.3}$$

In the early stages of the MC method, we construct an ergodic Markov chain $\gamma : r_0 \to r_1 \to \cdots \to r_k \to \cdots$, by random selecting the indices of T in order to sample the solution of (2.1). Realization of the stochastic trajectory γ is based on the transition probabilities, which makes the initial probability vector $p \in \mathbb{R}^n$ (starting point of Markov chain γ) and transition probability matrix $P \in \mathbb{R}^{n\times n}$ (transition from the current point to the next point) satisfying in the following conditions

$$\begin{aligned} &p_{ij} \geq 0, \quad \sum_{j=1}^{n} p_{ij} = 1, \quad if \quad t_{ij} \neq 0 \quad then \quad p_{ij} \neq 0, \\ &p_i \geq 0, \quad \sum_{i=1}^{n} p_i = 1, \quad if \quad h_i \neq 0 \quad then \quad p_i \neq 0, \end{aligned} \tag{2.4}$$

where the nonzero vector $h \in \mathbb{R}^n$ is considered to evaluate Euclidean inner product $< h, x >$. We restrict h to be e_i (i.e., ith standard basis vector) to reach our desired solution x_i. More details on the role of h have been given in [13]. The following unbiased estimator is defined for the stochastic trajectory γ

$$X(\gamma) = \sum_{m=0}^{\infty} W_m f_{r_m}, \tag{2.5}$$

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 the $< h, x >$, i.e., $E[X(\gamma)] =< h, x >$ [23]. It is usual to assume that $h = p = e_i$ to estimate the ith component of the solution vector. In this way, we get $E[X_i(\gamma)] = x_i$, which is proven by representing the single component of x in the following form

$$x_i = f_i + \sum_{m=1}^{\infty} \sum_{r_1=1}^{n} \sum_{r_2=1}^{n} \cdots \sum_{r_m=1}^{n} t_{ir_1} t_{r_1r_2} \cdots t_{r_{m-1}r_m} f_{r_m}. \tag{2.6}$$

Practically, the partial sum $X_i(\gamma_k) = \sum_{m=0}^{k} W_m f_{r_m}$ is applied on finite Markov chain $\gamma_k : r_0 \to r_1 \to \cdots \to r_k$, which its mathematical expectation tends to x_i by choosing k large enough. This truncation leads to systematic error, which is theoretically expressed as $|x_i - x_i^{(k)}| = |E[X_i(\gamma)] - E[X_i(\gamma_k)]| < \epsilon$, i.e., $\left| E[\sum_{m=0}^{\infty} W_m f_{r_m}] - E[\sum_{m=0}^{k} W_m f_{r_m}] \right| < \epsilon$, where $\epsilon > 0$ is the given parameter. Since $E[X_i(\gamma)] = E[\sum_{m=0}^{\infty} W_m f_{r_m}] < \infty$, effective stopping criterion for the continuity of Markov chain's length to create γ_k is given

as $|W_k| < \epsilon$. From $|W_k| < \epsilon$ the upper bound for k can be driven as the following way

$$|W_k| = |\frac{t_{r_0r_1}t_{r_1r_2}\dots t_{r_{k-1}r_k}}{p_{r_0r_1}p_{r_1r_2}\cdots p_{r_{k-1}r_k}}| \leq \frac{|t_{r_0r_1}t_{r_1r_2}\dots t_{r_{k-1}r_k}|}{\frac{|t_{r_0r_1}|}{||T||}\frac{|t_{r_1r_2}|}{||T||}\cdots\frac{|t_{r_{k-1}r_k}|}{||T||}} \leq ||T||^k < \epsilon.$$

Therefore, it can be deduced that

$$k \leq \frac{\log \epsilon}{\log(||T||)}. \tag{2.7}$$

It should be noted that the components of P are considered $p_{ij} = \frac{|t_{ij}|}{\sum_{j=1}^{n}|t_{ij}|}$, which introduces the Monte Carlo Almost Optimal (MAO) transition probability matrix [1]. Moreover, (2.7) is definite when $||T|| < 1$ [11, 23]. Another type of matrix P which has been applied by the authors is defined as $p_{ij} = \frac{1}{n}$. This type of P leads to the Uniform Monte Carlo (UM) method. Now, to approximate the real mean $E[X_i(\gamma_k)]$, N sample paths of Markov chains $r_0^{(s)} \to r_1^{(s)} \to \cdots \to r_k^{(s)}$, $s = 1, 2, \dots, N$ is simulated and then the sample mean of $X_i(\gamma_k^{(s)})$ is considered as its estimation (Strong Law of Large Numbers (SLLN) [15]). Therefore, $\overline{X}_i(\gamma_k) = \frac{1}{N}\sum_{s=1}^{N} X_i(\gamma_k^{(s)}) \approx x_i$, which leads to the statistical error. This error theoretically expressed as $|\overline{X}_i(\gamma_k) - E[X_i(\gamma_k)]| < \delta$, where δ is the given parameter. Generally, probable error is employed to estimate the statistical error. The following relation for a normal random variable X with mean μ and variance σ^2 [25]

$$p(|X - \mu| < 0.675\sigma) = 0.5 = p(|X - \mu| > 0.675\sigma),$$

is applied for $\overline{X}_i(\gamma_k)$. So by applying Central Limit Theorem (CLT) [15] it can be written that

$$p(|\overline{X}_i(\gamma_k) - E[X_i(\gamma_k)]| < r_N) \approx 0.5 \approx p(|\overline{X}_i(\gamma_k) - E[X_i(\gamma_k)]| > r_N),$$

in which $r_N = 0.6745\sqrt{\frac{var(X_i(\gamma))}{N}}$. By applying the precision $r_N \leq \delta$, the lower bound for N is obtained as $N \geq \frac{(0.6745)^2 var(X_i(\gamma))}{\delta^2}$. On the other, the upper bound for $var(X_i(\gamma))$ can be acquired in the following way

$$X_i(\gamma) = \sum_{m=0}^{\infty} W_m f_{r_m} = \sum_{m=0}^{\infty} \frac{t_{r_0r_1}t_{r_1r_2}\dots t_{r_{m-1}r_m}}{p_{r_0r_1}p_{r_1r_2}\cdots p_{r_{m-1}r_m}} f_{r_m}$$
$$\leq \sum_{m=0}^{\infty} \frac{t_{r_0r_1}t_{r_1r_2}\dots t_{r_{m-1}r_m}}{\frac{|t_{r_0r_1}|}{||T||}\frac{|t_{r_1r_2}|}{||T||}\cdots\frac{|t_{r_{m-1}r_m}|}{||T||}} f_{r_m} \leq \sum_{m=0}^{\infty} ||T||^m f_{r_m} \leq \frac{||f||}{1 - ||T||}.$$

Thereupon, $E[X_i^2(\gamma)] \leq \frac{||f||^2}{(1-||T||)^2}$ and then $var(X_i(\gamma)) \leq E[X_i^2(\gamma)] \leq \frac{||f||^2}{(1-||T||)^2}$. Consequently,

$$N \geq \frac{(0.6745)^2}{\delta^2} \frac{||f||^2}{(1-||T||)^2}. \tag{2.8}$$

It should be noted that similar to (2.7) P is considered MAO transition probability matrix and lower bound for N is obtained by assuming $||T|| < 1$. At the end of this section we represent the MC algorithm based on (2.7) and (2.8) (similar form of this algorithm can be found in [12, 23]).

Algorithm 1:

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

3. Analysis of convergence

In this section, we analyze convergence of the MC method for solving SLAE and conclude new results. In the early authors' research, the MC method was implemented

based on the condition $||T|| < 1$, i.e., Algorithm 1 [11, 23, 26]. Recently, Ji et al. have theoretically discussed convergence of the MC method in [21]. Based on [21], there is always a transition matrix P enabling the convergence of the MC method if $||T|| < 1$. Also, in the case $\rho(T) < 1$ and $||T|| \geq 1$, the MC method may or may not converge. Authors in [21] indicated the necessary and sufficient condition for the convergence of the MC method is

$$\rho(T^*) < 1, \tag{3.1}$$

where $t^*_{ij} = \frac{t^2_{ij}}{p_{ij}}$. For more details in effect of different cases of the iteration matrix T and probability transition matrix P on convergence, we refer to [21]. After that in [13], authors emphasize that validity of Eq. (3.1) should be checked for convergence regardless of the type of matrix P, MAO or UM transition probability matrix. For further studying, we refer to [13].

In this work, we intend to replace (3.1) with the simpler computing condition. For MAO transition matrix we have $t^*_{ij} = \frac{t^2_{ij}}{p_{ij}} = |t_{ij}| \sum_{k=1}^{n} |t_{ik}|$. By using Hadamard product, we can write T^*_{MAO} in the form $T^*_{MAO} = T^+ \circ T^{'}$, where the entries of nonnegative matrices T^+ and $T^{'}$ are $t^+_{ij} = |t_{ij}|$ and $t^{'}_{ij} = \sum_{k=1}^{n} |t_{ik}|$, respectively. For more details on the characteristics of the Hadamard product of matrices, we refer to [19]. Now, we apply some upper bounds for the spectral radius of the Hadamard product of nonnegative matrices. From [7] and [16], we have the following inequalities, respectively

$$\rho(A \circ B) \leq \max_i \{a_{ii}b_{ii} + t_i\rho(A) - t_i a_{ii}\}, \quad i = 1, 2, \ldots, n, \tag{3.2}$$

$$\begin{aligned} \rho(A \circ B) \leq \max_{i \neq j} \frac{1}{2} \{ & a_{ii}b_{ii} + a_{jj}b_{jj} + [(a_{ii}b_{ii} - a_{jj}b_{jj})^2 \\ & + 4t_i s_j(\rho(A) - a_{ii})(\rho(B) - b_{jj})]^{\frac{1}{2}} \}, \end{aligned} \tag{3.3}$$

where $A, B \geq 0$, $t_i = \max_{k \neq i} \{b_{ik}\}$, $s_j = \max_{k \neq j} \{a_{jk}\}$ and $i, j = 1, 2, \ldots, n$. By assuming $A = T^+$ and $B = T^{'}$, we have two cases for the matrix T^*_{MAO} related to upper bound (3.2). In first case, we assume that $T \geq 0$. Thereupon, we have

$$\begin{aligned} \rho(T^*_{MAO}) &\leq \max_i \left\{ (\sum_{j=1}^{n} t_{ij} - \sum_{j=1}^{n} t_{ij}) t_{ii} + \rho(T) \sum_{j=1}^{n} t_{ij} \right\} \\ &= \rho(T) \max_i \left\{ \sum_{j=1}^{n} t_{ij} \right\} < ||T||. \end{aligned} \tag{3.4}$$

As we observe by assuming

$$||T|| \leq 1, \tag{3.5}$$

the convergence condition $\rho(T^*_{MAO}) < 1$ is satisfied. In second case, we assume that the condition $T \geq 0$ is not satisfied then (3.2) can be computed in the following way

$$\begin{aligned}\rho(T^*_{MAO}) &\leq \max_i \left\{ (\sum_{j=1}^{n} |t_{ij}| - \sum_{j=1}^{n} |t_{ij}|)|t_{ii}| + \rho(T^+) \sum_{j=1}^{n} |t_{ij}| \right\} \\ &= \rho(T^+) \max_i \left\{ \sum_{j=1}^{n} |t_{ij}| \right\} = \rho(T^+)||T||. \end{aligned} \tag{3.6}$$

Therefore, by satisfying

$$\rho(T^+)||T|| < 1, \tag{3.7}$$

it is concluded that $\rho(T^*_{MAO}) < 1$, which leads to the convergence of the MC method. In continuation, we consider the upper bound (3.3) for T^*_{MAO} with the case $T \geq 0$. Therefore, we have

$$\begin{aligned}\rho(T^*_{MAO}) &< \frac{1}{2} \max_{i \neq j} \left\{ t_{ii} \sum_{k=1}^{n} t_{ik} + t_{jj} \sum_{k=1}^{n} t_{jk} + \left[(t_{ii} \sum_{k=1}^{n} t_{ik} - t_{jj} \sum_{k=1}^{n} t_{jk})^2 \right.\right. \\ &\quad \left.\left. + 4 \max_{k \neq j} \{t_{jk}\} \sum_{k=1}^{n} t_{ik}(1 - t_{ii})(\sum_{i,j=1}^{n} t_{ij} - \sum_{k=1}^{n} t_{jk}) \right]^{\frac{1}{2}} \right\} \\ &= \frac{1}{2} \max_{i \neq j} G_1(i,j), \quad i,j = 1,2,\ldots,n, \end{aligned} \tag{3.8}$$

where satisfies the condition

$$\max_{i \neq j} G_1(i,j) \leq 2, \tag{3.9}$$

guarantees convergence condition $\rho(T^*_{MAO}) < 1$. For the case that $T \geq 0$ is not satisfied, upper bound (3.3) can be computed as

$$\begin{aligned}\rho(T^*_{MAO}) &\leq \frac{1}{2} \max_{i \neq j} \left\{ |t_{ii}| \sum_{k=1}^{n} |t_{ik}| + |t_{jj}| \sum_{k=1}^{n} |t_{jk}| + \left[(|t_{ii}| \sum_{k=1}^{n} |t_{ik}| \right.\right. \\ &\quad - |t_{jj}| \sum_{k=1}^{n} |t_{jk}|)^2 + 4 \max_{k \neq j} \{|t_{jk}|\} \sum_{k=1}^{n} |t_{ik}|(\rho(T^+) - |t_{ii}|) \\ &\quad \left.\left. (\sum_{i,j=1}^{n} |t_{ij}| - \sum_{k=1}^{n} |t_{jk}|) \right]^{\frac{1}{2}} \right\} \\ &= \frac{1}{2} \max_{i \neq j} G_2(i,j), \quad i,j = 1,2,\ldots,n. \end{aligned} \tag{3.10}$$

It can be seen that if

$$\max_{i \neq j} G_2(i,j) < 2, \tag{3.11}$$

then $\rho(T^*_{MAO}) < 1$. Similarly, we iterate the same process for UM transition probability matrix. Because of $t^*_{ij} = \frac{t^2_{ij}}{p_{ij}} = nt^2_{ij}$, T^*_{UM} can be written in the form

$$T^*_{UM} = n(T^+ \circ T^+) = nT^{+\circ 2}, \tag{3.12}$$

where $T^{+\circ 2}$ denotes the Hadamard second power (entrywise second power) of T^+. By setting $A = B = T^+$, we can compute the upper bound (3.2) for the case $T \geq 0$ as follow

$$\begin{aligned}\rho(T^*_{UM}) &\leq n \max_i\{t^2_{ii} + \max_{k\neq i}\{t_{ik}\}(\rho(T) - t_{ii})\} \\ &< n \max_i\{t^2_{ii} + \max_{k\neq i}\{t_{ik}\}(1 - t_{ii})\} = nF_1(i).\end{aligned} \tag{3.13}$$

In the event that the following condition

$$F_1(i) \leq \frac{1}{n}, \tag{3.14}$$

implies $\rho(T^*_{UM}) < 1$. For the case $T \geq 0$ is not satisfied, upper bound (3.2) can be computed in the following way

$$\rho(T^*_{UM}) \leq n \max_i\{t^2_{ii} + \max_{k\neq i}\{|t_{ik}|\}(\rho(T^+) - |t_{ii}|)\} = nF_2(i). \tag{3.15}$$

If the condition

$$F_2(i) < \frac{1}{n}, \tag{3.16}$$

is satisfied then $\rho(T^*_{UM}) < 1$ and the MC method is convergent. For the upper bound (3.3) by considering $T \geq 0$, we have

$$\begin{aligned}\rho(T^*_{UM}) < \frac{n}{2} \max_{i\neq j} \Big\{ & t^2_{ii} + t^2_{jj} + [(t^2_{ii} - t^2_{jj})^2 + 4 \max_{k\neq i}\{t_{ik}\} \max_{k\neq j}\{t_{jk}\} \\ & (1 - t_{ii})(1 - t_{jj})]^{\frac{1}{2}} \Big\} = \frac{n}{2} \max_{i\neq j} H_1(i,j),\end{aligned} \tag{3.17}$$

where $i, j = 1, 2, \ldots, n$. We observe that by satisfying

$$\max_{i\neq j} H_1(i,j) \leq \frac{2}{n}, \tag{3.18}$$

the convergence condition $\rho(T^*_{UM}) < 1$ is valid. Also, in the case $T \geq 0$ is not satisfied, we get the following relations for T^*_{UM}

$$\begin{aligned}\rho(T^*_{UM}) &\leq \frac{n}{2} \max_{i\neq j} \Big\{ t^2_{ii} + t^2_{jj} + [(t^2_{ii} - t^2_{jj})^2 \\ &\quad + 4 \max_{k\neq i}\{|t_{ik}|\} \max_{k\neq j}\{|t_{jk}|\}(\rho(T^+) - |t_{ii}|)(\rho(T^+) - |t_{jj}|)]^{\frac{1}{2}} \Big\} \\ &= \frac{n}{2} \max_{i\neq j} H_2(i,j), \quad i, j = 1, 2, \ldots, n.\end{aligned} \tag{3.19}$$

Hence, by satisfying

$$\max_{i\neq j} H_2(i,j) < \frac{2}{n}, \tag{3.20}$$

we conclude that $\rho(T^*_{UM}) < 1$ and MC method is convergent with UM transition probability matrix. A lot of research has been made about the upper bound for the spectral radius of the Hadamard product of nonnegative matrices in [6, 10, 17, 20, 22]. Of all this, we invoke the sharper and less computational complexity upper bounds which are efficient for the large scale matrices and lead to validation of the convergence condition (3.1). Based on Eqs. (3.5-3.11) and (3.14-3.20) for MAO and UM transition matrices respectively, we can theoretically derive that $\rho(T^*_{MAO}) < 1$ is valid for a wider group of matrices than $\rho(T^*_{UM}) < 1$. This result is also shown in numerical results. Moreover, we indicate that the upper bound (3.2) is more efficient than (3.3) for MAO and UM matrices in numerical results.

4. A new MC method for solving SLAE

In this section, we present new MC algorithm based on the obtained results in the convergence analysis by employing effective techniques. We know that mentioned upper bound (2.7) for k is based on the assumption $||T|| < 1$, while MC method is convergent if (3.1) is satisfied (or more convenience, conditions (3.5-3.11), (3.14-3.20) are satisfied). Therefore, we consider the stopping condition $|W_k| < \epsilon$ for continuity the length of Markov chains. In this case, we get the unique length for each stochastic sample (Markov chain) for every component of solution vector x, i.e., $k_i^{(s)}$, $i = 1, 2, \ldots, n$, $s = 1, 2, \ldots, N$ are different.

On the other, lower bound (2.8) is obtained when $||T|| < 1$. We replace (2.8) with the following stopping condition for the number of stochastic samples to estimate the each component of the solution vector

$$\left| \frac{1}{s+1} \sum_{t=1}^{s+1} X_i^{(t)}(\gamma_{k_i^{(t)}}) - \frac{1}{s} \sum_{t=1}^{s} X_i^{(t)}(\gamma_{k_i^{(t)}}) \right| < \delta, \quad i = 1, 2, \ldots, n. \tag{4.1}$$

We consider s, which is satisfied in (4.1), as the number of required Markov chains for x_i, i.e., N_i. Indeed, we control the statistical error with the given parameter δ in this way. Therefore, the number of Markov chains N_i, $i = 1, 2, \ldots, n$ are distinctive per x_i, $i = 1, 2, \ldots, n$. Moreover, the required length of Markov chains can be represented as $k_i^{(s_i)}$, $s_i = 1, 2, \ldots, N_i$, $i = 1, 2, \ldots, n$, which are different from each other. Now, we display a new MC algorithm in the following way. At first we demonstrate the computing of estimator $X_i^{(s_i)}(\gamma_{k_i^{(s_i)}})$ in each Markov chain per component of x as the following function.

function $[X_i \quad k_i] = Markov(i, \epsilon)$, (*Markov is the name of function*)

Set $W_0 = 1$, $k_i = 0$, $point = i$, $X_i = W_0 f_i$

while $|W_k| \geq \epsilon$

Generate an r.v. nextpoint, distributed on ith row of matrix P as:

Set $nextpoint = 1$, $u = rand$

while $u > \sum_{j=1}^{nextpoint} p_{point,nextpoint}$

$nextpoint = nextpoint + 1$

end while

Set $k_i = k_i + 1$

if $t_{point,nextpoint} \neq 0$, (*this loop can be out of conditional in MAO*)

Compute $W_k = W_{k-1}\frac{t_{point,nextpoint}}{p_{point,nextpoint}}$, $X_i = X_i + W_k f_{nextpoint}$

end if

Set $point = nextpoint$

end while

end function

In continuation, we provide a new MC algorithm, included the function Markov, in Algorithm 2.

Algorithm 2:

(1) *Input matrix* T, *vector* f, *parameters* ϵ *and* δ

(2) *Compute* P *based on the type of probability transition matrix*

(3) **for** $i = 1$ *to* n

(4) *Set* $s = 1$, $Error = 1$

(5) *Compute* $[X_i^{(s)} \quad k_i^{(s)}] = Markov(i, \epsilon)$

(6) *Compute* $\overline{X}_0 = \frac{1}{s}X_i^{(s)}$

(7) *Set* $X_i = X_i^{(s)}$, $K_i(s) = k_i^{(s)}$(*sth component of vector* K_i *illustrates the length of sth Markov chain for* x_i)

(8) **while** $Error \geq \delta$

(9) *Set* $s = s + 1$

(10) *Compute* $[X_i^{(s)} \quad k_i^{(s)}] = Markov(i, \epsilon)$

(11) *Set* $X_i = X_i + X_i^{(s)}$

(12) *Set* $\overline{X}_1 = \frac{1}{s}X_i$

(13) *Compute* $Error = |\overline{X}_1 - \overline{X}_0|$

$$(i.e.,\ Error = \left|\frac{1}{s+1}\sum_{t=1}^{s+1} X_i^{(t)}(\gamma_{k_i^{(t)}}) - \frac{1}{s}\sum_{t=1}^{s} X_i^{(t)}(\gamma_{k_i^{(t)}})\right|)$$

(14) *Set* $\overline{X}_0 = \overline{X}_1$

(15) *Set* $K_i(s) = k_i^{(s)}$

(16) **end while**

(17) *Set* $N_i = s$

(18) *Set* $x_i = \overline{X}_1$ $(i.e.,\ x_i = \frac{1}{N_i}\sum_{t=1}^{N_i} X_i^{(t)}(\gamma_{k_i^{(t)}}))$

(19) **end for**

We can observe that the systematic and statistical error can be controlled with given parameters ϵ and δ, respectively in Algorithm 2 and the number of Markov chains per x_i, $i = 1, 2, \ldots, n$ is distinctive unlike Algorithm 1. Moreover, the length of Markov chain per statistical sample for each component of solution vector is unique, which is the prominent difference between the Algorithm 2 and Algorithm 1. By applying the stopping conditions $|W_k| < \epsilon$ and (4.1) for the length and number of Markov chains, we can obtain the optimum N_i and $k_i^{(t)}$, $i = 1, 2, \ldots, n$, $t = 1, 2, \ldots, N_i$. In this way, the complexity of computing is significantly reduced, which indicates the superiority of Algorithm 2 to Algorithm 1. Since the stopping conditions for continuity N_i and $k_i^{(t)}$ are implemented separately, so increasement any of them is stopped as soon as the related stop criteria of each one is satisfied. This feature improves the execution speed of Algorithm 2, while the fix value of k and N in rows 2 and 3 makes the Algorithm 1 be time consuming. These results have been illustrated in numerical experiments.

It is worthy to note that we can employ Algorithm 2 for finding MI, in general. For more details in this area, we refer to [13].

5. Numerical examples

In this section, we present performance of Algorithm 2 by some numerical examples. All computations are implemented in double precision using some MATLAB codes on a Pentium 5 PC, with a 3.20 GHz CPU and 4GB of RAM. We know that the superiority of MC method to deterministic linear solvers is apparent when implementing on parallel processors for large scale matrices. However, we intend to numerically verify the validity of convergence results and efficiency of new MC algorithm.

Example 5.1. *In this example, we numerically demonstrate the convergence behavior of MC method with MAO and UM transition probability matrices. Moreover, we investigate the proximity of the upper bounds (3.2) and (3.3) to real values of* $\rho(T^*_{MAO})$ *and* $\rho(T^*_{UM})$*. We also check the satisfying condition (3.5-3.11) and (3.14-3.20) based on the convergence condition (3.1). We consider the Symmetric Positive Definite (SPD) matrices 685 BUS, 494 BUS and 662 BUS of the order* $n = 685$, 494 *and* 662*, respectively as the coefficient matrix* A *in the SLAE (2.1). These matrices are selected from the well-known collection of the matrices in the Matrix Market website* [27].

We consider the splitting of SOR method for $\omega = 1$ and then we achieve the iteration matrix $T = (D - E)^{-1}F$, where D, $-E$ and $-F$ are the diagonal, strict lower part and strict upper part of A, respectively. In this way, we have $\rho(T) < 1$ [24]. We construct T^*_{MAO} related to the MAO transition probability matrix. According to the presented numerical results in Table 1, we have $||T|| = 1$ and $\rho(T^*_{MAO}) < 1$ for these matrices, which lead to the convergence of MC method with MAO transition matrix. Also, we observe that upper bound (3.4) is closer to real value of $\rho(T^*_{MAO})$ than upper bound (3.8). Moreover, satisfying condition (3.5) is in accordance with the validity of condition $\rho(T^*_{MAO}) < 1$. It should be noted that for these matrices with the mentioned splitting, we have $T \geq 0$. Therefore, we use the upper bound (3.4)

TABLE 1. Behavior of the convergence of MC method with MAO transition probability matrix related to the convergence condition (3.1) and proximity of the upper bounds (3.2) and (3.3) to $\rho(T^*_{MAO})$.

Matrix	$\|\|T\|\|$	$\rho(T)$	$\rho(T^*_{MAO})$	(3.4)	(3.5)	(3.8)	(3.9)
685 BUS	1.0000	0.9995	0.9994	1.0000	Satisfied	26.0999	Not satisfied
494 BUS	1.0000	0.9999	0.9999	1.0000	Satisfied	22.1402	Not satisfied
662 BUS	1.0000	0.9999	0.9999	1.0000	Satisfied	25.6716	Not satisfied
A $(n = 40)$	1.0018	0.9960	0.9968	1.0018	Satisfied	4.0082	Not satisfied
A $(n = 60)$	1.0018	0.9992	1.0007	1.0018	Not satisfied	4.9036	Not satisfied
A $(n = 70)$	1.0018	0.9999	1.0015	1.0018	Not satisfied	5.2919	Not satisfied
B $(n = 100)$	1.0000	0.9637	0.9637	1.0000	Satisfied	6.4760	Not satisfied
B $(n = 300)$	1.0000	0.9657	0.9656	1.0000	Satisfied	11.2984	Not satisfied
B $(n = 500)$	1.0000	0.9654	0.9662	1.0000	Satisfied	14.6071	Not satisfied

and (3.8) (i.e., first case of upper bounds (3.2) and (3.3) with MAO matrix). Now, we consider the tridiagonal matrix $A = tridiag(-0.55, 1.099, -0.55)$ of order n. From analysis of eigenvalues of tridiagonal matrices, we can derive that the eigenvalues of A is positive [18]. Since A is symmetric matrix, we can derive that A is the SPD matrix [18]. We employ the mentioned splitting (SOR splitting with $\omega = 1$) for A. Therefore, it can be derived that $\rho(T) < 1$. We observe similar convergence behavior for this matrix. For example in fifth row of Table 1 ($n = 60$) condition (3.5) is not satisfied and this is consistent with $\rho(T^*_{MAO}) = 1.0007$. Therefore, in this row of Table 1, MC method is not convergent. Additionally, upper bound (3.4) is closer to real value of $\rho(T^*_{MAO})$ than upper bound (3.8). In continuation of this example, we consider the toeplitz matrix $B = fivediag(-1, -2, 7, -1, -3)$ of order n. From [3], we know that B is M-matrix. By employing the iteration matrix $T = I - D^{-1}B$ we derive that $\rho(T) < 1$, where diagonal matrix D is considered as $D = diag(b_{11}, b_{22}, \ldots, b_{nn})$ [24]. We obtain similar numerical results for various values of n for matrix B in rows 7 to 9 of Table 1. Once again, we provide numerical results for the mentioned matrices with UM transition probability matrix in Table 2. In all rows of Table 2, the condition $\rho(T^*_{UM}) < 1$ and then (3.14) are not satisfied. Similar to Table 1, it is visible that upper bound (3.13) is close to the real value of $\rho(T^*_{UM})$ than upper bound (3.17). As it has been mentioned in Section 3, from shown numerical results in Table 1 and 2 we can deduce that the MC method with UM transition probability matrix is convergent for the group of very few matrices than MAO matrix. Consequently, UM transition probability matrix is practically set aside. Finally, we select the matrix DWB 512 from the Matrix Market website. By exerting the splitting of SOR method for $\omega = 1$, we achieve $\rho(T) < 1$. Related numerical results for DWB 512 with MAO and UM transition probability matrices are illustrated in Table 3 and 4, respectively. Because $T \geq 0$ is not satisfied, we apply the corresponding upper bounds (3.6), (3.10) and (3.15), (3.19) for MAO and UM matrices, respectively. For this matrix, we also attain the similar numerical results in Table 3 and 4 as Table 1 and 2.

TABLE 2. Behavior of the convergence of MC method with UM transition probability matrix related to the convergence condition (3.1) and proximity of the upper bounds (3.2) and (3.3) to $\rho(T^*_{UM})$.

Matrix	$\rho(T^*_{UM})$	Upper bound (3.13)	(3.14)	Upper bound (3.17)	(3.18)
685 BUS	641.9951	685	Not satisfied	685	Not satisfied
494 BUS	492.6666	494	Not satisfied	494	Not satisfied
662 BUS	651.4537	662	Not satisfied	662	Not satisfied
A $(n = 40)$	9.9776	18.6309	Not satisfied	20.0182	Not satisfied
A $(n = 60)$	15.0148	27.9464	Not satisfied	30.0273	Not satisfied
A $(n = 70)$	17.5294	32.6041	Not satisfied	35.0318	Not satisfied
B $(n = 100)$	26.2213	42.8571	Not satisfied	42.8571	Not satisfied
B $(n = 300)$	79.8163	128.5714	Not satisfied	128.5714	Not satisfied
B $(n = 500)$	131.2759	214.2857	Not satisfied	214.2857	Not satisfied

TABLE 3. Behavior of the convergence of MC method with MAO transition probability matrix related to the convergence condition (3.1) and proximity of the upper bounds (3.2) and (3.3) to $\rho(T^*_{MAO})$ for the matrix DWB 512.

Matrix	$\|\|T\|\|$	$\rho(T)$	$\rho(T^*_{MAO})$	(3.6)	(3.7)	(3.10)	(3.11)
DWB 512 (MAO)	0.2475	0.0857	0.0173	0.0212	Satisfied	0.3571	Satisfied

TABLE 4. Behavior of the convergence of MC method with UM transition probability matrix related to the convergence condition (3.1) and proximity of the upper bounds (3.2) and (3.3) to $\rho(T^*_{UM})$ for the matrix DWB 512.

Matrix	$\rho(T^*_{UM})$	Upper bound (3.15)	(3.16)	Upper bound (3.19)	(3.20)
DWB 512 (UM)	0.8498	7.2808	Not satisfied	7.3180	Not satisfied

Example 5.2. *We consider Initial Value Problem (IVP) of system of linear ODEs with non constant coefficient matrix in the form*

$$\begin{cases} y'(t) + A(t)y(t) = f(t), \\ y(t_0) = 0, \quad t \in [t_0, T], \end{cases} \tag{5.1}$$

where $A(t) = tridiag(-0.2t, 0.5t, -0.3t)$ *of order* n *and* $[t_0, T] = [0, 1]$. *The function* $f(t)$ *is computed such that the exact solution is given by*

$$y(t) = \left(\frac{t^2}{25}, \frac{t^3}{125}, \frac{t^4}{625}, \frac{t^5}{3125}, \ldots, \frac{t^2}{25}, \frac{t^3}{125}, \frac{t^4}{625}, \frac{t^5}{3125}\right)^T \in \mathbb{R}^n.$$

From analysis of eigenvalues of tridiagonal matrices, we can derive that all of eigenvalues of matrix $A(t)$ are nonzero over $(0, 1]$ and then $A(t)$ is nonsingular on this interval [18]. For the stepsize $h = 0.1$ and the equally spaced grid $\{t_0, t_1, \ldots, t_{10}\}$ for $t \in [0, 1]$, we apply the following forward Euler method

$$y'(t) \approx \frac{y(t_{n+1}) - y(t_n)}{h}, \quad n = 0, 1, \ldots, 9,$$

to approximate $y'(t)$ and discretize the IVP (5.1). Therefore, IVP (5.1) can be written in the following discretized form

$$\begin{cases} (I + 0.1A_{n+1})y_{n+1} = y_n + 0.1f(t_{n+1}), \\ y_0 = 0, \quad n = 0, 1, \ldots, 9, \end{cases} \tag{5.2}$$

where y_n is approximation for $y(t_n)$ and for brevity of notation $A(t_n)$ is denoted by A_n. Also, I is the identity matrix of order n. By setting $B(t) = (I+0.1A(t))$, we have $B(t) = tridiag(-0.02t, 1 + 0.05t, -0.03t)$. It is visible that $b_{ij}(t) \leq 0$ for $i \neq j$ and $b_{ii}(t) > 0$ on $[0, 1]$. Moreover, $B(t)$ is Strictly Diagonally Dominant (SDD) matrix on $[0, 1]$. Hence, it can be derived that $B(t)$ is M-matrix on $[0, 1]$ [3]. Now, we solve the SLAE $B_{n+1}y_{n+1} = y_n + 0.1f(t_{n+1})$ for $n = 0, 1, \ldots, 9$ by the MC method and approximate the value of solution vector function $y(t)$ in the grid points of interval $[0, 1]$. We know that considering the splitting $B(t) = D(t) - (D(t) - B(t))$ and iteration matrix $T(t) = I - D(t)^{-1}B(t)$ lead to $\rho(T(t)) < 1$ for $t \in [0, 1]$ [24]. This splitting is considered to start the Algorithm 1 and 2. It is evident that the first four components of the solution vector are repeated periodically. Therefore, we can apply the MC method to find the first four components which is the superiority of the MC method to the deterministic iterative methods. We apply the Algorithm 1 and 2 to solve SLAE (5.2) and numerically approximate the value of solution vector function $y(t)$ in the grid point t_{n+1}, $n = 0, 1, \ldots, 9$ for IVP (5.1). The numerical results of these algorithms is presented in Table 5, 6, 7, 8 and 9 for $n = 60$, 80, 800, 1200 and 2000, respectively. We compare the obtained numerical results of these algorithms. In these tables fifth column (MC solution) represents the first four components of solution vector function in the corresponding pint t_{n+1}. As we observe in Algorithm 2 the value of N is separate for each of the components while N is fix for all of the components (Eq. (2.8)). Moreover, N in Algorithm 2 is significantly less than Algorithm 1, which causes Algorithm 2 is implemented in so less time than Algorithm 1 (sixth column of the tables). We observe similar behavior of numerical results in Tables 6 and 7 for $n = 80$ and $n = 800$. In continuation of this example we high the order of IVP (5.1) and demonstrate the numerical results for $n = 1200$ and $n = 2000$ in Tables 8 and 9, respectively. In the Tables 8 and 9, we see that Algorithm 2 is implemented by so few N in a very good time, while Algorithm 1 is the working state for a long time because of the so many numbers of N. In these tables, the mark "–" indicates that Algorithm 1 is in the working state more than an hour. It is notable that the absolute error of every component for the obtained MC solution from Algorithm 1 and 2 are so close together, which verify the efficiency of Algorithm 2. The absolute error for the third and fourth component of $y(t)$ on $[0, 1]$ with $n = 80$ and $n = 800$ are illustrated in Figure 1 and 2, respectively.

6. Conclusion

In this study, after reviewing the MC method for solving SLAE, we have investigated the convergence of this method. We have provided simpler computing condition on the convergence of this method related to the type of transition probability matrix. Moreover, we have introduced Algorithm 2 based on the obtained convergence

FIGURE 1. The absolute error for the third component of $y(t)$ on $[0,1]$ for Algorithm 1 and 2 when $\epsilon = 10^{-6}$, $\delta = 10^{-4}$ and $n = 80$.

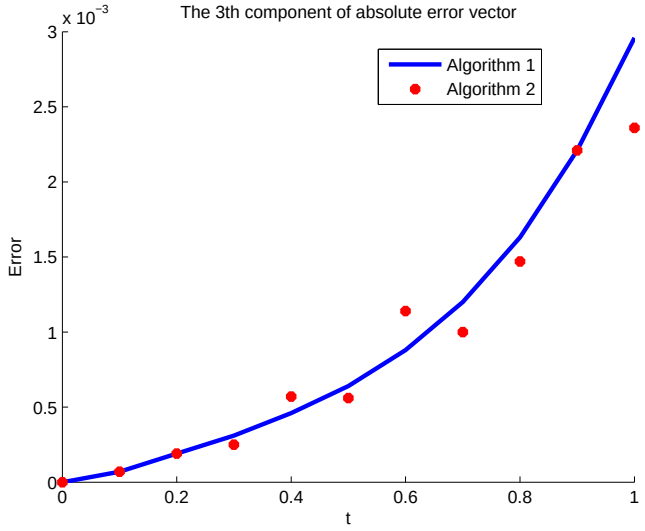

FIGURE 2. The absolute error for the fourth component of $y(t)$ on $[0,1]$ for Algorithm 1 and 2 when $\epsilon = \delta = 10^{-4}$ and $n = 800$.

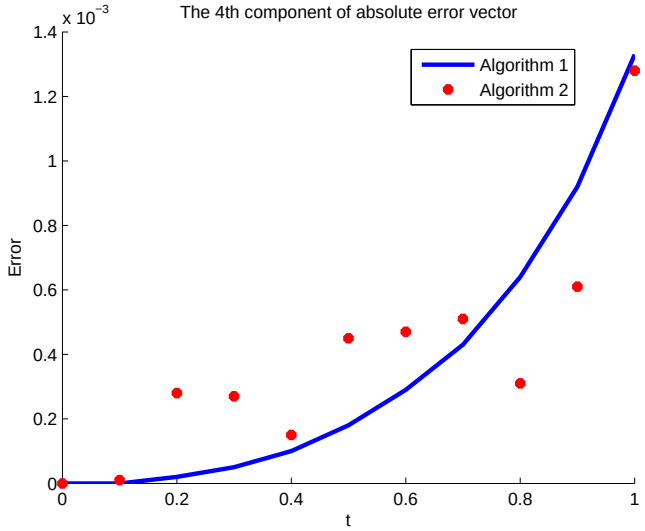

TABLE 5. Comparison between the obtained results of Algorithm 1 and 2 for IVP (5.1) of order $n = 60$.

Algorithm	ϵ	δ	t (grid point)	MC solution	time (s)	N
2	10^{-4}	10^{-3}	0.1	0.03350311 0.00222721 0.00011323 0.00000681	0.0094	2 2 2 6
1	10^{-4}	10^{-3}	0.1	0.27726833 0.08101776 0.02126389 0.00635916	0.0108	516
2	10^{-4}	10^{-4}	0.2	0.06734229 0.00599827 0.00049448 0.00030667	0.0100	2 2 2 2
1	10^{-4}	10^{-4}	0.2	0.06733833 0.00560034 0.00048545 0.00004144	2.4313	210267
2	10^{-4}	10^{-4}	0.4	0.13601618 0.02081601 0.00280037 −0.00114027	0.0111	2 2 3 2
1	10^{-4}	10^{-4}	0.4	0.13601309 0.01924847 0.00283503 0.00042525	10.4465	872823
2	10^{-4}	10^{-3}	0.6	0.20597949 0.04001173 0.00900468 0.00227076	0.0111	2 2 2 3
1	10^{-4}	10^{-3}	0.6	0.20601978 0.04229742 0.00888842 0.00186674	0.2438	20342
2	10^{-4}	10^{-3}	0.8	0.27726833 0.08101776 0.02126389 0.00635916	0.0108	2 2 3 6
1	10^{-4}	10^{-3}	0.8	0.27735807 0.07511885 0.02059928 0.00569094	0.4567	37387
2	10^{-4}	10^{-3}	0.9	0.31355990 0.09433112 0.02767167 0.00688560	0.0112	2 6 2 7
1	10^{-4}	10^{-3}	0.9	0.31352804 0.09518734 0.02920410 0.00902913	0.5749	48080

TABLE 6. Comparison between the obtained results of Algorithm 1 and 2 for IVP (5.1) of order $n = 80$.

Algorithm	ϵ	δ	t (grid point)	MC solution	time (s)	N
2	10^{-6}	10^{-4}	0.1	0.03350361 0.00224464 0.00011421 0.00007157	0.0151	2 2 2 2
1	10^{-6}	10^{-4}	0.1	0.03350331 0.00222849 0.00011353 0.00000520	0.6059	51565
2	10^{-4}	10^{-4}	0.3	0.10151775 0.01067183 0.00133478 0.00045360	0.0116	2 2 2 5
1	10^{-4}	10^{-4}	0.3	0.10150986 0.01126300 0.00131845 0.00015497	5.6938	482065
2	10^{-6}	10^{-4}	0.5	0.17085569 0.02989870 0.00519219 0.00087528	0.0119	2 21 4 19
1	10^{-6}	10^{-4}	0.5	0.17084967 0.02959232 0.00527774 0.00095653	16.9039	1388300
2	10^{-4}	10^{-4}	0.8	0.27742057 0.07541257 0.01955931 −0.00025879	0.0123	3 44 3 2
1	10^{-4}	10^{-4}	0.8	0.27735808 0.07508549 0.02059906 0.00569903	44.2700	3738700
2	10^{-6}	10^{-4}	1	0.35000512 0.12667018 0.04340228 0.01190943	0.0141	3 2 2 71
1	10^{-6}	10^{-4}	1	0.35003260 0.11780899 0.04000500 0.01367688	75.4980	6028569

TABLE 7. Comparison between the obtained results of Algorithm 1 and 2 for IVP (5.1) of order $n = 800$.

Algorithm	ϵ	δ	t (grid point)	MC solution	time (s)	N
2	10^{-5}	10^{-4}	0.1	0.03350361 0.00216154 0.00011421 0.00007155	0.2580	2 2 2 2
1	10^{-5}	10^{-4}	0.1	0.03350331 0.00222788 0.00011349 0.00000558	0.6263	51565
2	10^{-4}	10^{-5}	0.2	0.06733836 0.00533641 0.00046957 0.00011735	0.1766	2 2 2 21
1	10^{-4}	10^{-5}	0.2	0.06733833 0.00560091 0.00048552 0.00004131	5.0321	21026661
2	10^{-4}	10^{-4}	0.5	0.17088515 0.02854907 0.00516771 0.00123089	0.2091	2 7 4 15
1	10^{-4}	10^{-4}	0.5	0.17084958 0.02959330 0.00527709 0.00095824	35.2195	1388300
2	10^{-5}	10^{-3}	0.8	0.27726836 0.07357737 0.01949347 0.00307904	0.2161	2 4 2 6
1	10^{-5}	10^{-3}	0.8	0.27735950 0.07511707 0.02060666 0.00567846	0.9455	37387
2	10^{-3}	10^{-3}	1	0.35007300 0.11676348 0.04099375 0.01654690	0.1831	2 6 3 5
1	10^{-3}	10^{-3}	1	0.35003061 0.11779823 0.04000278 0.01367438	1.4823	60286

TABLE 8. Comparison between the obtained results of Algorithm 1 and 2 for IVP (5.1) of order $n = 1200$.

Algorithm	ϵ	δ	t (grid point)	MC solution	time (s)	N	Absolute error
2	10^{-4}	10^{-6}	0.1	0.03350336	0.3222	2	0.00017003
				0.00223339		74	0.00112228
				0.00011223		4	0.00007519
				0.00007154		2	0.00007031
1	10^{-4}	10^{-6}	0.1	–			
2	10^{-4}	10^{-4}	0.4	0.13600088	0.5603	2	0.00266755
				0.01932400		14	0.00154622
				0.00270790		2	0.00033753
				−0.00114399		2	0.00146004
1	10^{-4}	10^{-4}	0.4	–			
2	10^{-6}	10^{-3}	0.6	0.20607982	0.4973	2	0.00607982
				0.04003182		2	0.00003182
				0.00831568		2	0.00031568
				0.00418750		2	0.00258750
1	10^{-6}	10^{-3}	0.6	–			
2	10^{-3}	10^{-3}	0.7	0.24153779	0.4916	2	0.00820445
				0.05745789		5	0.00301345
				0.01422344		2	0.00151973
				0.00187393		5	0.00109026
1	10^{-3}	10^{-3}	0.7	–			
2	10^{-7}	10^{-3}	0.9	0.31340114	0.4982	2	0.01340114
				0.09030620		2	0.00030620
				0.02886121		3	0.00186121
				0.00575247		6	0.00234752
1	10^{-7}	10^{-3}	0.9	–			
2	10^{-5}	10^{-4}	1	0.35000311	0.5100	3	0.01666978
				0.11180743		2	0.00069632
				0.03868447		6	0.00164743
				0.01275208		69	0.00040640
1	10^{-5}	10^{-4}	1	–			

results and finally compared this new MC method with Algorithm 1 in the numerical examples.

ACKNOWLEDGMENT

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

TABLE 9. Comparison between the obtained results of Algorithm 1 and 2 for IVP (5.1) of order $n = 2000$.

Algorithm	ϵ	δ	t (grid point)	MC solution	time (s)	N	Absolute error
2	10^{-4}	10^{-6}	0.1	0.03350336	1.5329	2	0.00017003
				0.00220751		47	0.00109640
				0.00011342		5	0.00007638
				0.00000100		73	0.00000022
1	10^{-4}	10^{-6}	0.1	–			
2	10^{-4}	10^{-6}	0.2	0.06734229	1.5543	2	0.00016756
				0.00533613		2	0.00068916
				0.00046957		2	0.00051732
				−0.00035538		2	0.00013751
1	10^{-4}	10^{-6}	0.2	–			
2	10^{-4}	10^{-4}	0.5	0.17085555	1.5826	2	0.00418888
				0.03201514		2	0.00423737
				0.00503701		2	0.00040738
				0.00257172		2	0.00180011
1	10^{-4}	10^{-4}	0.5	–			
2	10^{-4}	10^{-4}	0.6	0.20602980	1.6145	2	0.00602980
				0.04184271		19	0.00184271
				0.00952614		2	0.00152614
				0.00420511		2	0.00260511
1	10^{-4}	10^{-4}	0.6	–			
2	10^{-6}	10^{-4}	0.7	0.24161663	1.9740	2	0.00828329
				0.05756886		34	0.00312441
				0.01499098		3	0.00228728
				0.00648553		2	0.00352133
1	10^{-6}	10^{-4}	0.7	–			
2	10^{-6}	10^{-4}	1	0.35000261	2.0127	2	0.01666928
				0.11756381		60	0.00645270
				0.04018043		25	0.00314339
				0.01367802		60	0.00133234
1	10^{-6}	10^{-4}	1	–			

REFERENCES

[1] V. Alexandrov, E. Atanassov, I. Dimov, S. Branford, A. Thandavan, and C. Weihrauch, *Parallel Hybrid Monte Carlo Algorithms for Matrix Computations*, Lecture Notes in Computer Science. Proceedings Part III. 3516: 744-752. 5th International Conference. Atlanta. GA. USA. (2005), 22–25.

[2] V. Alexandrov and O. A. Esquivel-Flores, *Towards Monte Carlo preconditioning approach and hybrid Monte Carlo algorithms for matrix computations*, Comput. Math. Appl, *70* (2015), 2709–2718.

[3] O. Axelsson, *Iterative solution methods*, Cambridge University Press, Cambridge, 1996.

[4] M. Benzi and D. Bertaccini, *Block preconditioning of real-valued iterative algorithms for complex linear systems*, IMA J. Numer. Anal, *28* (2008), 598–618.

[5] S. Branford, C. Sahin, A. Thandavan, C. Weihrauch, V. Alexandrov and I. T. Dimov, *Monte Carlo methods for matrix computations on the grid*, Future Gener. Comput. Syst, *24* (2008), 605–612.
[6] G. Cheng, *New bounds for eigenvalues of the Hadamard product and the Fan product of matrices*, Taiwanese Journal of Mathematics, *18* (2014), 305–312.
[7] G. Cheng and X. Rao, *Some inequalities for the spectral radius of the Hadamard product of two nonnegative matrices*, J. Math. Inequal, *7* (2013), 529–534.
[8] D. Davila, V. Alexandrov, and O. A. Esquivel-Flores, *On Monte Carlo hybrid methods for linear algebra*, 7th Workshop on latest advances in scalable algorithms for large-scale systems, Spain, 2016.
[9] I. Dimov, S. Maire, and J. M. Sellier, *A new walk on equations Monte Carlo method for solving systems of linear algebraic equations*, Appl. Math. Model., *39* (2015), 4494–4510.
[10] M. Fang, *Bounds on eigenvalues of the Hadamard product and the Fan product of matrices*, Linear Algebra Appl., *425* (2007), 7-15.
[11] B. Fathi-Vajargah, *Different stochastic algorithms to obtain matrix inversion*, Appl. Math. Comput., *189* (2007), 1841-1846.
[12] B. Fathi-Vajargah, *New advantages to obtain accurate matrix inversion*, Appl. Math. Comput., *189* (2007), 1798-1804.
[13] B. Fathi-Vajargah and Z. Hassanzadeh, *Improvements on the hybrid Monte Carlo algorithms for matrix computations*, Sādhanā, *44* (2018), 1–13.
[14] B. Fathi-Vajargah and Z. Hassanzadeh, *Monte Carlo method for the real and complex fuzzy system of linear algebraic equations*, Soft Comput., Online first published, 19 April 2019, Doi:10.1007/s00500-019-03960-1.
[15] C. M. Grinstead and J. L. Snell, *Introduction to probability*, Second edition, American Mathematical Society, 2003.
[16] Q. P. Guo, H. B. Li, and J. S. Leng, *Some new bounds for the Hadamard product and the Fan product of matrices*, Numer. Algorithms, *18* (2014), 1–15.
[17] Q. P. Guo, H. B. Li, and M. Y. Song, *New inequalities on eigenvalues of the Hadamard product and the Fan product of matrices*, J. Inequal. Appl., *2013* (2013), 421–433.
[18] R. A. Horn and C. R. Johnson, *Matrix analysis*, Cambridge U.P.l., 1985.
[19] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, New York, 1991.
[20] R. A. Horn and F. Zhang, *Bounds on the spectral radius of a Hadamard product of nonnegative or positive semidefinite matrices*, Electron. J. Linear Algebra, *20* (2010), 90–94.
[21] H. Ji, M. Mascagni and Y. Li, *Convergence analysis of Markov chain Monte Carlo linear solvers using Ulam-Von Neumann algorithm*, SIAM J. Numer. Anal., *51* (2013), 2107–2122.
[22] Q. Liu and G. Chen, *On two inequalities for the Hadamard product and the Fan product of matrices*, Linear Algebra Appl., *431* (2009), 974–984.
[23] R. Y. Rubinstein, *Simulation and the Monte Carlo method*, John Wiley and Sons, New York, 1981.
[24] Y. Saad, *Iterative methods for sparse linear systems*, SIAMY, Philadelphia, 2003.
[25] I. M. Sobol, *The Monte Carlo method*, The University of Chicago press, Chicago, 1975.
[26] J. Strassburga and V. Alexandrovb, *A Monte Carlo approach to sparse approximate inverse matrix computations*, Proc. Comput. Sci., *18* (2013), 2307–2316.
[27] Website: `http://math.nist.gov/MatrixMarket/matrices.html`