Monte Carlo Algorithms for Solving Linear Systems

Behrouz Fathi Vajargah,

Department of Mathematics,

Islamic Azad University, Rasht Branch, IRAN.

fathi@guilan.ac.ir

Abstract

This paper presents two different Monte Carlo algorithms for obtaining the solution of vector x in linear system x=Ax+f, where A is a given non singular matrix and f is a known vector with the same size as A. In this paper, we consider two individual transition probabilities for our Monte Carlo computations. Considering different statistical nature for generating the random walk to select the nonzero entries of the coefficient matrix A, are caused different accuracy and computational times for our employed algorithms.

We first make Monte Carlo estimator for the elements of the solution vector x, then we determine the Monte Carlo estimation of the solution vector x based on its unbiased estimator. Finally, we present the computational results obtained by two different transition probabilities.

Key Words: Monte Carlo, Markov Chain, Linear System, Transition Probabilities, Random Walk.

Introduction

One of the most important problems in the numerical linear algebra is solving the linear systems. When we want to have accurate results of the linear systems, we prefer to use the classical method such as Gauss and Gauss-Jordan methods. But, when the dimension of coefficient matrix in the linear systems increases, the classical methods obtain the solution so slow. Then we prefer to use the iterative methods such as Jacobi,

SOR , … [2]. Here we want to introduce a stochastic-numerical method which is called Monte Carlo method as an alternative for the iterative methods.

The Monte Carlo method can obtain the solution of the linear systems based on a non singular coefficient matrix. In this method we first establish an unbiased estimator for *jth* element of the solution *x* in the given linear system *x=Ax+f*. Then we simulate *N* different random paths on the matrix *A* and employ them in our corresponding Monte Carlo solution.

Monte Carlo computations in general and especially finding the solution of the given linear system are based on generating the random numbers. Then we have the stochastic natures in the computations of Monte Carlo methods.

Monte Carlo method for solving SLAE

Consider solving SLAE of the form *Bx=f*, where *B* is a square non-singular matrix of the size *n*, and $f = (f_1, f_2,..., f_n)^t$ are known. We want to obtain the solution of the above linear system by Monte Carlo method. If we consider *I-A=B*, where *I* is an identity matrix with size *n*, then this system can be converted to:

$$x=Ax+f. \tag{1}$$

Let

$$\max_{1\le i\le n}\sum_{j=1}^{n}\left|a_{ij}\right|<1.. \tag{2}$$

Under this assumption we can solve (1) by applying the recursive equation:

$$x^{(k+1)} = Ax^{(k)} + f. \qquad k = 0,1,2,... \tag{3}$$

Suppose $x^{(0)} \equiv 0$ and $A^0 \equiv I$, we have

$$x^{(k+1)} = (I + A + ... + A^{k-1} + A^k) f$$

$$= \sum_{m=0}^{k} A^m f. \tag{4}$$

Taking the limit:

$$\lim_{k\to\infty} x^{(k)} = \lim_{k\to\infty} \sum_{m=0}^{k} A^m f = (I - A)^{-1} f = B^{-1} f = x, \tag{5}$$

it means that, the exact solution is available for (3) when *k* is sufficiently large. The j^{th} coordinate of the vector x^{k+1} is equal to

$$x_j^{(k+1)} = f_j + \sum_{i_1} a_{ji} f_{i_1} + \sum_{i_2}\sum_{i_1} a_{ji_1} a_{i_1 i_2} f_{i_2} + ... + \sum_{i_k}...\sum_{i_2}\sum_{i_1} a_{ji_1} a_{i_1 i_2}...a_{i_{k-1} i_k} f_{i_k}. \qquad (6)$$

We consider the definition of the inner product of two vectors $h, x \in R^n$ by:

$$< h, x >= h^t x = h_1 x_1 + h_2 x_2 + ... + h_n x_n, \qquad (7)$$

where h is a known vector and x is the exact solution of (1). If we set $h = (0,0,...,0,\underset{\underset{j}{\smile}}{1},0,...,0)$ then from (7) we obtain the j^{th} element of the vector *x i.e.* x_j. We want to consider a Markov chain to obtain the estimate of solution *x* by Monte Carlo method.

The probability functions p_{i_0} and p_{ij} (initial distribution and the transition probability from state *i* to state *j* of the Markov chain) are acceptable for h_i and a_{ij}, respectively if:

1. $p_{i_0} > 0, \quad if \quad h_{i_0} \neq 0 \quad i_0 = 1,2,...,n$
2. $P_{ij} > 0, \quad if \quad a_{ij} \neq 0, \quad i, j = 1,2,...,n, \qquad (8)$

Under condition (2) we first consider the estimation $< h, x^{(k+1)} >$, which it approximates $< h, x >$, since (5) is valid. Let *k* be given integer and let us simulate the Markov chain:

$$i_0 \to i_1 \to i_2 \to ... \to i_k,$$

where $i_0, i_1, ..., i_k \in S = \{1,2,...,n\}$. For this sample path we define:

$$W_m = \frac{a_{i_0 i_1} a_{i_1 i_2} ... a_{i_{m-1} i_m}}{p_{i_0 i_1} p_{i_1 i_2} ... p_{i_{m-1} i_m}} \qquad m = 0,1,2,...,k, \qquad (9)$$

and

$$W_m = W_{m-1} \frac{a_{i_{m-1} i_m}}{p_{i_{m-1} i_m}}, \quad W_0 \equiv 1. \qquad (10)$$

We define the random variable $T_k(h)$ by:

$$T_k(h) = \frac{h_{i_0}}{p_{i_0}} \sum_{m=0}^{k} W_m f_{i_m}. \qquad (11)$$

Under the above conditions, we have

$$E[T_k(h)] = \sum_{i_0=1}^{n}\sum_{i_1=1}^{n}\ldots\sum_{i_k=1}^{n} T_k(h)\, p_{i_0}\, p_{i_0 i_1}\, p_{i_1 i_2}\ldots p_{i_{k-1} i_k}$$

$$= \sum_{i_0=0}^{n}\ldots\sum_{i_k=1}^{n}\left(\frac{h_{i_0}}{p_{i_0}}\sum_{m=0}^{k} W_m\, f_{i_m}\right) p_{i_0}\, p_{i_0 i_1}\, p_{i_1 i_2}\ldots p_{i_{k-1} i_k}$$

$$= \sum_{i_0=1}^{n}\ldots\sum_{i_k=1}^{n}\frac{h_{i_0}}{p_{i_0}}\sum_{m=0}^{k}\left(\frac{a_{i_0 i_1} a_{i_1 i_1}\ldots a_{i_{m-1} i_m}}{p_{i_0 i_1}\, p_{i_1 i_2}\ldots p_{i_{m-1} i_m}}\right) f_{i_m}\, p_{i_0}\, p_{i_0 i_1}\, p_{i_1 i_2}\ldots p_{i_{m-1} i_m}\, p_{i_m i_{m+1}}\ldots p_{i_{k-1} i_k}$$

$$= \sum_{m=0}^{k}\sum_{i_0=1}^{n}\sum_{i_1=1}^{n}\ldots\sum_{i_m=1}^{n}\sum_{i_{m+1}=1}^{n}\ldots\sum_{i_k=1}^{n} h_{i_0}\, a_{i_0 i_1}\, a_{i_1 i_2}\ldots a_{i_{m-1} i_m}\, f_{i_m}\, p_{i_m i_{m+1}}\, p_{i_{m+1} i_{m+2}}\ldots p_{i_{k-1} i_k} \qquad (12)$$

Since for all $i=1,2,\ldots,n$, $\sum_{j=1}^{n} p_{ij} = 1$ then $\sum_{i_{m+1}=1}^{n} p_{i_m i_{m+1}} = 1, \ldots, \sum_{i_k=1}^{n} p_{i_{k-1} i_k} = 1$, then we have:

$$E[T_k(h)] = \sum_{m=0}^{k}\sum_{i_0=1}^{n}\sum_{i_1=1}^{n}\ldots\sum_{i_m=1}^{n} h_{i_0}\, a_{i_0 i_1}\, a_{i_1 i_2}\ldots a_{i_{m-1} i_m}\, f_{i_m}\,. \qquad (13)$$

If we note to the following product of order $m \geq 2$ of matrix $\boldsymbol{A}$:

$$A^2 = A \times A = [\sum_{i_1=1}^{n} a_{i_0 i_1}\, a_{i_1 i_2}\,]_{i_1=1}^{n}, \ldots,$$

$$A^m = \underbrace{A \times \ldots \times A}_{m\_times} = [\sum_{i_0=1}^{n}\sum_{i_1=1}^{n}\ldots\sum_{i_{m-1}=1}^{n} a_{i_0 i_1}\, a_{i_1 i_2}\ldots a_{i_{m-1}}\, a_{i_m}\,]_{i_0, i_{m-1}=1}^{n}. \qquad (14)$$

Thus,

$$E[T_k(h)] = < h, \sum_{m=1}^{k} A^m f > = < h, x^{(k+1)} > \qquad (15)$$

i.e. $T_k(h)$ is an unbiased estimator of the inner product $< h, x^{(k+1)} >$. With regards to the above discussion we proved the following theorem.

Theorem: Introduced $T_k(h)$ in (11) is an unbiased estimator for $< h, x^{(k+1)} >$.

Generally, to approximate $< h, x^{(k+1)} >$ we simulate *N* random paths [7]

$$i_0^{(s)} \to i_1^{(s)} \to \ldots \to i_k^{(s)}, \qquad s = 1,2,\ldots, N. \qquad (16)$$

In (17) each path we may realize a different estimation and therefor we consider the sample mean of the estimations for obtaining better estimation of the parameter:

$$\hat{\Theta}_k = \frac{1}{N}\sum_{s=1}^{N} T_k^{(s)}(h) \approx < h, x^{(k+1)} >. \qquad (17)$$

If we consider $h^t = (0,0,...,\underset{\underset{i}{\smile}}{1},0,...,0) = e(i)^t$, then we have

$$\hat{\Theta}_k = \frac{1}{N}\sum_{s=1}^{N} T_k^{(s)}(h) \approx x_i, \tag{18}$$

Where $\hat{\Theta}_k$ in (18), is the Monte Carlo estimation for the i^{th} element of the vector *x*. Here, we obtained only one element of the vector solution *x*. We can obtain the other elements of the solution vector *x* in the same way, it only needs to be changed the vector *h*. This is another point of the Monte Carlo method that with a simple variation in selection of *h* we can get an arbitrary element of *x*. This is in contrast with the iterative methods, where all elements of the vector *x* are obtained in each step.

$\hat{\Theta}_k$ is called the Monte Carlo estimation of the $< h, x^{(k+1)} >$. It is clear that

$$\lim_{k\to\infty} E(T_k(h)) = E(T_\infty(h)) = \lim_{k\to\infty} < h, x^{(k+1)} >|_{h=e(i)} = x_i . \tag{19}$$

It can be easily proved that this Monte Carlo estimator is convergent to its exact solution, i.e.

$$\hat{\Theta}_k \to < h, x > \quad in \quad \left\{ \begin{array}{c} (q.m.) \\ (p.) \\ (a.s.) \end{array} \right\} \quad as \quad k \to \infty . \tag{20}$$

which q.m., p. and a.s. mean the convergence in quadratic mean, in probability and with probability one, respectively [5,6].

Provided that the Neumann series $A + A^2 + ...$ converges, and the path $i_0 \to i_1 \to i_2 \to ... \to i_k \to ...$ is infinitely long, we obtain an unbiased estimator of $< h, x >$.

The sample mean (Monte Carlo estimation) is then of the form $\hat{\Theta}_\infty = \frac{1}{N}\sum_{s=1}^{N} T_\infty^{(s)}(h)$, where

$$T_\infty^{(s)}(h) = \frac{h_{i_0}}{p_{i_0}^{(s)}} \sum_{i_m} W_m^{(s)} f_{i_m}, \qquad s = 1,2,..., N \tag{21}$$

and

$$W_m^{(s)} = \frac{a_{i_0^{(s)} i_1^{(s)}} a_{i_1^{(s)} i_2^{(s)}} \ldots a_{i_{m-1}^{(s)} i_m^{(s)}}}{p_{i_0^{(s)} i_1^{(s)}} p_{i_1^{(s)} i_2^{(s)}} \ldots p_{1_{m-1}^{(s)} i_m^{(s)}}}. \qquad (22)$$

Page simulate N such independent random paths, then we have:

$$\tilde{\Theta}_k(h) = \frac{1}{N} \sum \hat{T}_k^{(s)}(e_j), \qquad \text{j=1,2,…,n, } s = 1,2,..., N \qquad (23)$$

which estimates x_j, $j=1,2,\ldots,n$.

Therefore all random variables $T_k(e_j)$, $j=1,2,\ldots,n$, are defined on the same path and are calculated according to the same formula.

Markov Chains according to whether or not it is possible to go from a given state to another given state can be considered. If we consider

$$p_{ij}^{(n)} = p(X_n = j \mid X_0 = i) > 0 \qquad \textit{for} \quad \textit{some} \quad n \in \{1,2,...\}, \qquad (24)$$

it means the probability of that Markov Chain reaching from i to j in n steps ($n \geq 1$) [4]. Here, we use only the transition probability of the Markov chain with n=1. The number of Markov chains is given by $N \geq (\frac{0.6745}{\varepsilon} . \frac{\| f \|}{(1 - \| A \|)})^2$ and the length of Markov chain is also bounded by $T = k < \frac{\log(\delta / \| f \|)}{\log \| A \|}$ [1,6].

Different Transition Probabilities for Markov Chains

When we use Monte Carlo method for solving the linear system (1), we should specify our transition probabilities for walking on the rows and columns of the given matrix $\boldsymbol{A}$ in (1) for randomly selecting of its nonzero elements. It is expected that with considering different probabilistic nature in Monte Carlo algorithms we conclude difference accuracy and computational times from our employed algorithms.

Two Monte Carlo Algorithms for Solving SLAE

In solving System Linear Algebraic Equations (SLAE) if we use uniform transition probabilities for our Markov chain, we call it ***UM*** Monte Carlo method. We

use the discrete uniform density function for transition probability

$P=[p_{ij}]_{i,j=1,2,...,n}$ i.e. $p_{ij}=\frac{1}{n},\ i,j\in\{i_0,i_1,...,i_k\}$ and $i_0,i_1,...,i_k\in S=\{1,2,...,n\}$ where

n is the size of matrix $\boldsymbol{A}$

in the linear system (1). Then the corresponding transition probability for the coefficient matrix,

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2j} & \cdots & a_{2n} \\ & \vdots & & \vdots & & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ij} & \cdots & a_{in} \\ & \vdots & & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nj} & \cdots & a_{nn} \end{bmatrix}, \text{is given by, } P_{UM}=\begin{bmatrix} \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} & \cdots & \frac{1}{n} \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} & \cdots & \frac{1}{n} \\ & \vdots & & \vdots & & \vdots \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} & \cdots & \frac{1}{n} \\ & \vdots & & \vdots & & \vdots \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} & \cdots & \frac{1}{n} \end{bmatrix}.$$

If we consider

$$p_{ij}=\frac{|a_{ij}|}{\sum_{j=1}^{n}|a_{ij}|}, \qquad i,j=1,2,\ldots,n.$$

Then its transition probability matrix is given by:

$$P_{MAO}=\begin{bmatrix} \frac{|a_{11}|}{\sum_{j=1}^{n}|a_{ij}|} & \frac{|a_{12}|}{\sum_{j=1}^{n}|a_{ij}|} & \cdots & \frac{|a_{1j}|}{\sum_{j=1}^{n}|a_{ij}|} & \cdots & \frac{|a_{1n}|}{\sum_{j=1}^{n}|a_{ij}|} \\ \frac{|a_{21}|}{\sum_{j=1}^{n}|a_{ij}|} & \frac{|a_{22}|}{\sum_{j=1}^{n}|a_{ij}|} & \cdots & \frac{|a_{2j}|}{\sum_{j=1}^{n}|a_{ij}|} & \cdots & \frac{|a_{2n}|}{\sum_{j=1}^{n}|a_{ij}|} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{|a_{i1}|}{\sum_{j=1}^{n}|a_{ij}|} & \frac{|a_{i2}|}{\sum_{j=1}^{n}|a_{ij}|} & \cdots & \frac{|a_{ij}|}{\sum_{j=1}^{n}|a_{ij}|} & \cdots & \frac{|a_{in}|}{\sum_{j=1}^{n}|a_{ij}|} \\ \vdots & \vdots & & \vdots & & \vdots \\ \frac{|a_{n1}|}{\sum_{j=1}^{n}|a_{ij}|} & \frac{|a_{n2}|}{\sum_{j=1}^{n}|a_{ij}|} & \cdots & \frac{|a_{nj}|}{\sum_{j=1}^{n}|a_{ij}|} & \cdots & \frac{|a_{nn}|}{\sum_{j=1}^{n}|a_{ij}|} \end{bmatrix}.$$

We call Monte Carlo method using this transition probability, Monte Carlo algorithm using Almost Optimal (MAO) transition probabilities[1].

To compute the solution of the SLAE (1) by Monte Carlo methods, we select the non-zero elements of the coefficient matrix with regards to the specified transition probability. We note that if the random walk encounters a zero element, we ignore it in our computation and generate another random number to select the next non-zero element of *A*. We make two different algorithms by considering these two transition probabilities: UM Monte Carlo algorithm and MAO Monte Carlo algorithm.

Experimental Results

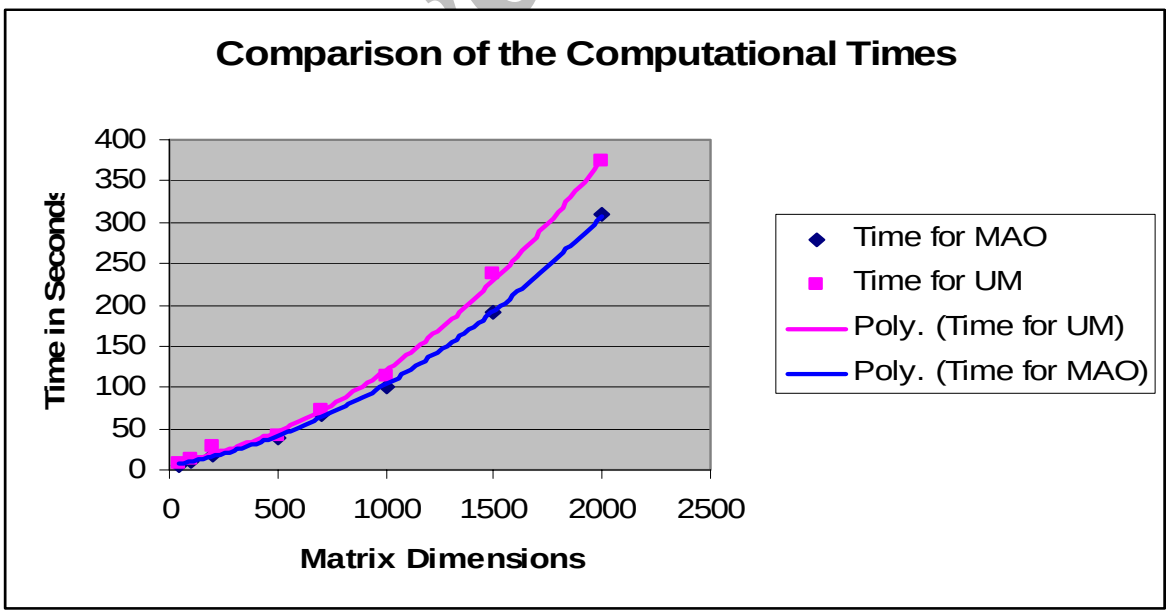
Now, we present the numerical results given by the above MAO and UM Monte Carlo Algorithms. The algorithm tested for solving linear systems with different dimensions of the coefficient matrices:

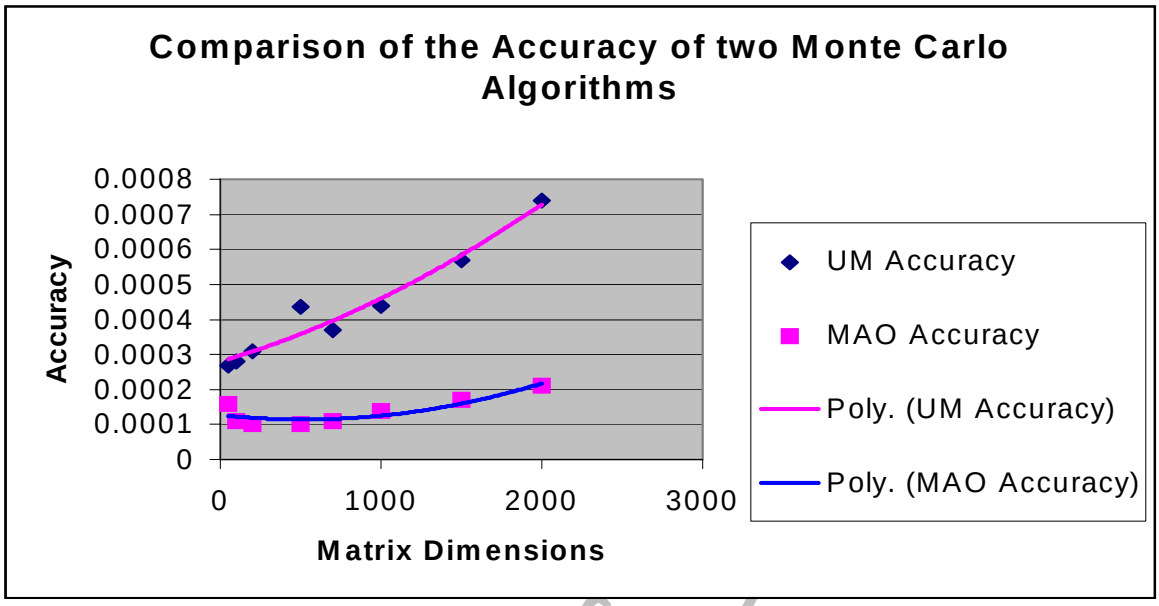
n	***UM***	***MAO***
50	0.585	0.80
100	1.025	1.035
200	1.715	1.775
500	3.895	4.21
700	5.770	6.31
1000	8.050	8.450
1500	14.19	14.78
2000	20.85	23.31

Table 1: Computational time for larger matrices.

n	*UM*	*MAO*
50	0.00027	0.00016
100	0.00028	0.00011
200	0.00031	0.00010
500	0.000436	0.00010
700	0.00037	0.00011
1000	0.00044	0.00014
1500	0.00057	0.00017
2000	0.00074	0.00021

Table2: Computational error (accuracy) of methods.





References:

[1] Alexandrov V.N., Efficient parallel Monte Carlo methods for matrix computation, Mathematics and computers in simulation Elsevier 47 (1998) 113-122, Netherland.

[2] Benzi M, and Szyld D., Existance and unique of splitting for stationary iterative methods with applications to alternating methods, Numerische Mathematik, 1997 76: 309-321.

[3] Bromley B. C., Quasi random number generators for parallel Monte Carlo algorithm, Journal of parallel and distributed computing 38, 101-104 (1996).

[4] Cinlar E., Introduction to Stochastic Processes, Prentice-hall, Englewood Cliffs, New Jersey, 1975.

[5] Dimov I.T., Dimov T.T. and Grov T.V., A new iterative Monte Carlo approach for inverse matrix problem, Journal of Computational and Applied Mathematics 92 (1998) 15-35.

[6] Dimov I.T. and Karaivanova, I.N., Iterative Monte Carlo method for linear algebra problems, First Workshop on Numerical Analysis and applications, Rousse, Bulgaria, June 24-27, 1996, Numerical anlysis and its applications, Springer Lecture Notes in Computer Science, Ser. 1196, pp. 150-160.

[7] Rubinstein R.Y., Simulation and the Monte Carlo method, John Wiley &Sons, New York , (1981) .