ELSEVIER

Mathematics and Computers in Simulation 47 (1998) 113–122

MATHEMATICS AND COMPUTERS IN SIMULATION

Efficient parallel Monte Carlo methods for matrix computations

V.N. Alexandrov[1]

Department of Computer Science, University of Liverpool, Liverpool, L69 7ZF, UK

Abstract

Three Monte Carlo methods for matrix inversion (MI) and finding a solution vector of a system of linear algebraic equations (SLAE) are considered: with absorption, without absorption with uniform transition frequency function, and without absorption with almost optimal transition frequency function. Recently Alexandrov, Megson, and Dimov have shown that an $n \times n$ matrix can be inverted in $3n/2 + N + T$ steps on a regular array with $O(n^2 NT)$ cells. Alexandrov and Megson have also shown that a solution vector of SLAE can be found in $n + N + T$ steps on a regular array with the same number of cells. A number of bounds on N and T have been established (N is the number of chains and T is the length of the chain in the stochastic process; these are independent of n), which show that these designs are faster than existing designs for large values of n. In this paper we take another implementation approach; we consider parallel Monte Carlo algorithms for MI and solving SLAE in MIMD environment, e.g. running on a cluster of workstations under PVM. The Monte Carlo method with almost optimal frequency function performs best of the three methods; it needs about six to ten times fewer chains for the same precision. © 1998 IMACS/Elsevier Science B.V.

Keywords: Monte Carlo methods; Parallel algorithms; Matrix inversion; System of linear algebraic equations

1. Introduction

The problem of inverting a real $n\times n$ matrix (MI) and/or solving system of linear algebraic equations (SLAE) is of unquestionable importance in many scientific and engineering applications: e.g. real-time speech coding, digital signal processing, communications, stochastic modelling, and many physical problems involving partial differential equations. The direct methods of solution require $O(n^3)$ sequential steps when the usual elimination or annihilation schemes (e.g. non-pivoting Gaussian elimination, Gauss–Jordan methods) are employed [3]. Consequently the computation time for very large problems or for real-time problems can be prohibitive and prevents the use of many established algorithms.

It is known that Monte Carlo methods give statistical estimates for elements of the inverse matrix, or for components of the solution vector of SLAE, by performing random sampling of a certain random variable, whose mathematical expectation is the desired solution [22,24]. We concentrate on Monte Carlo methods for MI and/or solving SLAE, since, *first*, only $O(NT)$ steps are required to find an

[1]vassil@csc.liv.ac.uk

PII S0378-4754(98)00097-4

Table 1
Area-time trade-off for various arrays for matrix inversion

Array	A	T	AT
Robert–Trystram [20]	n^2	$5n$	$5n^3$
Kung–Lo–Lewis [14]	n^2	$5n$	$5n^3$
Megson [15]	$(3/2)n^2 - (n/2)$	$4n$	$6n^3 - 2n^2$
Rajopadhye [21]	n^2	$4n$	$4n^3$
Delsome [9]	$5n^2/4$	$4n$	$5n^3$
Megson–Alexandrov–Dimov [18]	$n^2\ NT$	$<4n$	$<4n^3\ NT$

Table 2
Area-time trade-off for various arrays for solving SLAE

Array	A	T	AT
Ahmed–Delosme–Morf [19] and Kung–Leiserson [19]	$n(n+1)/2$	$5n$	$5/2(n^3+n^2)$
Gentleman–Kung [19] and Kung–Leiserson [19]	$n(n+3)/2$	$4n-1$	$2n^3+5n^2-3n/2$
Alexandrov–Megson [2]	$n^2\ NT$	$<3n$	$<3n^3\ NT$

element of the inverse matrix or component of the solution vector of SLAE (N is the number of chains and T is an estimate of the chain length in the stochastic process, which are independent of n) and *secondly*, the sampling process for stochastic methods is inherently parallel.

Recently Megson et al. [16,18] have presented regular arrays for MI by Monte Carlo method, exploiting the inherent parallelism of the method. The full matrix is inverted in $3n/2+N+T$ steps on O(n^2NT) cells. Similarly, for SLAE, Alexandrov and Megson [2] have presented regular arrays where a full solution vector is computed in $n+N+T$ steps on O(n^2NT) cells. A number of bounds on N and T were established, which show that these designs are faster than the existing ones for reasonably large values of n (see Tables 1 and 2). Thus for sufficiently large n the Monte Carlo approach is theoretically more efficient than the usual direct and iterative methods [5].

The purpose of this paper is to consider the implementation of Monte Carlo algorithms in MIMD environment. We use a cluster of workstations running under PVM.

The rest of this paper is organized as follows. Section 2 briefly outlines the essential details of MI by Monte Carlo methods. Section 3 introduces the concept of optimal and almost optimal frequency function. Section 4 discusses the parallel implementation. Section 5 outlines important bounds on T and N. Section 5 summarizes the results.

2. Stochastic methods and matrix inversion

Assume that the SLAE is presented in the form

$$x = Ax + \varphi \tag{1}$$

where A is a real square $n\times n$ matrix, $x=(x_1,x_2,\dots,x_n)^{\mathrm{T}}$ is an $n\times 1$ solution vector and

$\varphi = (\varphi_1, \varphi_2, \dots, \varphi_n)^{\mathrm{T}}$ is a given vector. (If we consider the system $Lx = b$, then it is possible to choose a non-singular matrix M such that $ML = I - A$ and $Mb = \varphi$, and so $Lx = b$ can be presented as $x = Ax + \varphi$.) Assume that A satisfies the condition $\max_{1\le i\le n} \sum_{j=1}^{n} |a_{ij}| < 1$, which implies that all the eigenvalues of A lie in the unit circle. The matrix and vector norms are determined as follows: $\|A\| = \max_{1\le i\le n} \sum_{j=1}^{n} |a_{ij}|$, $\|\varphi\| = \max_{1\le i\le n} |\varphi_i|$.

Suppose that we have Markov chains with n states. The random trajectory (chain) T_i of length i starting in state k_0 is defined as $k_0 \to k_1 \to \cdots \to k_j \to \cdots \to k_i$ where k_j is the number of the state chosen, for $j = 1, 2, \dots, i$. The following probability definitions are also important: $P(k_0 = \alpha) = p_\alpha, P(k_j = \beta | k_{j-1} = \alpha) = p_{\alpha\beta}$ where p_α is the probability that the chain starts in state α and $p_{\alpha\beta}$ is the transition probability to state β from state α. Probabilities $p_{\alpha\beta}$ define a transition matrix P. We say that the distribution $(p_1, \dots, p_n)^{\mathrm{t}}$ is acceptable [22] to vector g and similarly that the distribution $p_{\alpha\beta}$ is acceptable to A if

$$p_\alpha > 0 \quad \text{when } g_\alpha \neq 0, \qquad p_\alpha \geq 0 \quad \text{when } g_\alpha = 0$$

and

$$p_{\alpha\beta} > 0 \quad \text{when } a_{\alpha\beta} \neq 0, \qquad p_{\alpha\beta} \geq 0 \quad \text{when } a_{\alpha\beta} = 0,$$

and we require that $\sum_{\alpha=1}^{n} p_\alpha = 1$, $\sum_{\beta=1}^{n} p_{\alpha\beta} = 1$ for any $\alpha = 1, 2, \dots, n$.

Consider the problem of evaluating the inner product of a given vector g with the vector solution of (1):

$$(g, x) = \sum_{\alpha=1}^{n} g_\alpha x_\alpha. \tag{2}$$

It is known [22] that the mathematical expectation $E\Theta^*[g]$ of random variable $\Theta^*[g]$ is

$$E\Theta^*[g] = (g, x), \qquad \text{where} \quad \Theta^*[g] = \frac{gk_0}{pk_0} \sum_{j=0}^{\infty} W_j \varphi k_j \quad \text{and} \quad W_0 = 1,\ W_j = W_{j-1} \frac{a_{k_{j-1}k_j}}{p_{k_{j-1}k_j}} \tag{3}$$

We use the following notation for a partial sum (3): $\theta_i[g] = (g_{k_0}/p_{k_0}) \sum_{j=0}^{i} W_j \varphi_{k_j}$. According to the above conditions on the matrix A, the series $\sum_{j=0}^{\infty} W_j \varphi_{k_j}$ converges for any given vector φ and $E\theta_i[g]$ tends to (g, x) as $i \to \infty$. Thus $\theta_i[g]$ can be considered an estimate of (g, x) for i sufficiently large.

Now we define the Monte Carlo method. To find one component of the solution, e.g. the rth component of x, we must choose $g = e(r) = (0, \dots, 0, 1, 0, \dots, 0)$ such that $(e(r))_\alpha = \delta_{r\alpha}$. It follows that

$$(g, x) = \sum_{\alpha=1}^{n} (e(r))_\alpha x_\alpha = x_r$$

and the corresponding Monte Carlo method is given by

$$x_r \approx \frac{1}{N} \sum_{s=1}^{N} \theta_i[e(r)]_s \tag{4}$$

where N is the number of chains and $\theta_i[e(r)]_s$ is the value of $\theta_i[e(r)]$ in the sth chain.

To find the inverse C of some matrix B, we first compute the elements of matrix $A = I - B$ where I is the identity matrix; clearly $C = \sum_{i=0}^{\infty} A^i$ which converges if $\|A\| < 1$. To find the element $c_{rr'}$ of the inverse matrix C, we set $\varphi = e(r')$. The Monte Carlo method becomes

$$c_{rr'} \approx \frac{1}{N}\sum_{s=1}^{N}\left[\sum_{(j|k_j=r')} W_j\right]_s \tag{5}$$

where $W_j = W_{j-1}(a_{k_{j-1}k_j}/p_{k_{j-1}k_j})$ and $(j|k_j = r')$ means that only W_j for which $k_j = r'$ are included in the sum. Since W_j is included only into the corresponding sum for $r' = 1, 2, \ldots, n$, *the same set of N chains can be used to compute a single row of the inverse* [22] with a consequent saving in computation.

The *probable error* of the method is defined as

$$r_N = 0.6745\sqrt{D\theta/N}, \quad \text{where} \quad P\{|\bar{\theta} - E(\theta)| < r_N\} \approx 1/2 \approx P\{|\bar{\theta} - E(\theta)| > r_N\}$$

if we have N independent realizations of random variable (r.v.) θ with mathematical expectation $E\theta$ and average $\bar{\theta}$ [22].

It is clear from the formula for r_N that the number of chains N can be reduced by a suitable choice of the transition probabilities that reduce the variance for a given probable error. This idea leads to Monte Carlo methods with minimal probable error.

3. Monte Carlo methods with minimal probable error

The key results concerning minimization of probable error and the definition of *almost* optimal transition frequency for Monte Carlo methods applied to the calculation of inner product via iterated functions are presented in [6,18]. The main results and theorems are outlined here.

Define vectors Ψ and $\hat{\Psi}$ where

$$\psi_\alpha = \left(\sum_\beta \frac{a^2_{\alpha\beta}\varphi^2_\beta}{p_{\alpha\beta}}\right)^{1/2} \quad \text{and} \quad \hat{\psi}_\alpha = \sum_\beta |a_{\alpha\beta}\varphi_\beta| \qquad \forall\alpha = 1, 2, \ldots, n. \tag{6}$$

Lemma 1. The transition probability $p_{\alpha\beta} = |a_{\alpha\beta}\varphi_\beta| / \sum_\beta |a_{\alpha\beta}\varphi_\beta|$ of the Markov chain minimizes ψ_α, and $\min_p \psi_\alpha = \hat{\psi}_\alpha$ $\forall\alpha = 1, 2, \ldots, n$.

Lemma 2. The probability $p_\alpha = |g_\alpha\hat{\psi}_\alpha| / \sum_\beta |g_\beta\hat{\psi}_\beta|$ in the Markov chain minimizes the sum $\sum_\alpha (g^2_\alpha\hat{\psi}^2_\alpha/p_\alpha)$ and $\min_p \sum_\alpha (g^2_\alpha\hat{\psi}^2_\alpha/p_\alpha) = (\sum_\beta |g_\beta\hat{\psi}_\beta|)^2$.

Theorem 3. The frequency function $\hat{p}_i = c_0 g_{\alpha_0} | \prod_{j=1}^{i} |a_{\alpha_{j-1}\alpha_j}|\varphi_{\alpha_i}$, where $c_0 = (\sum_\beta |g_\beta\hat{\psi}_\beta|)^{-1}$ minimizes the second moment $E\theta^2_i[g]$ of the r.v. $\theta_i[g]$.

Theorem 4. If $\varphi_{\alpha_i} A^{(k-i)}\varphi_{k_i} \geq 0$ for any $k > i$, then the frequency function

$$\hat{p}_{k,i} = c_0|g_{\alpha_0}| \prod_{j=1}^{i} |a_{\alpha_{j-1}\alpha_j}| [\varphi_{\alpha_i} A^{(k-i)}\varphi_{\alpha_i}]^{1/2},$$

where $c_0 = (\sum_\beta |g_\beta\hat{\psi}_\beta|)^{-1}$ minimizes $E(\theta_j[g]\theta_k[g])$.

According to Theorems 3 and 4 the minimizing frequency functions of the second moment $E\Theta^{*2}[g]$ of r.v. $\Theta^{*}[g]$ are

$$\hat{p}_i = c_0|g_{\alpha_0}|\prod_{j=1}^{i}|a_{\alpha_{j-1}\alpha_j}|\varphi_{\alpha_i} \quad \text{for } i = 1,2,\ldots$$

and

$$\hat{p}_{k,i} = c_0|g_{\alpha_0}|\prod_{j=1}^{i}|a_{\alpha_{j-1}\alpha_j}|[\varphi_{\alpha_i}A^{(k-i)}\varphi_{\alpha_i}]^{1/2} \quad \text{for } i = 1,2,\ldots$$

where $A^{k-1}=\sum_{\alpha_{i+1}}\cdots\sum_{\alpha_k}a_{\alpha_i\alpha_{i+1}}\cdots a_{\alpha_{k-1}\alpha_k}\varphi_{\alpha_k}$. The *almost optimal frequency* is $\bar{p}_i=c_0|g_{\alpha_0}|\prod_{j=1}^{i}|a_{\alpha_{j-1}\alpha_j}|$.

In the Monte Carlo method we can employ either the *optimal* or *almost optimal* frequency function. According to the previous discussion and the principle of collinearity of norms [18] we can choose $p_{\alpha\beta}$ proportional to the $|a_{\alpha\beta}|$.

Similar transition probabilities are outlined by Halton [12,13] and Ermakov and Mikhailov [10].

The algorithmic efficiency of Monte Carlo method for MI is discussed in Section 5.

4. Parallel implementation

We implement parallel Monte Carlo algorithms on a cluster of workstations under PVM. We assume a virtual star topology and we apply the master/slave approach.

Notice that one row of the inverse matrix can be computed in parallel on the same set of Markov chains. Thus the natural partitioning of the inverse matrix C, among the processors is to partition C into p blocks of $\lceil n/p\rceil$ consecutive rows and to allocate each block of rows to a separate processor. Similarly we can partition the solution vector into p blocks of $\lceil n/p\rceil$ components.

Inherently, the Monte Carlo method allows us to have minimal communication, i.e. to pass the matrix A to every processor, to run the algorithm and to collect the results from slaves at the end without any communication between sending A and receiving C or x. The only communication is at the beginning and at the end of the algorithm execution which allows us to obtain very high efficiency of parallel implementation. Therefore, by locating the master in the central node of the star and the slaves in the remaining nodes, the communication is minimized.

5. Parameters – estimation and discussion

Let us outline the method of estimation of N and T in case of *Monte Carlo method without absorbing states*. We will consider Monte Carlo methods with uniform (UM) and with almost optimal (MAO) transition frequency function. We assumed that the following conditions $\sum_{\beta=1}^{n}p_{\alpha\beta} = 1$ for any $\alpha = 1,2,\ldots,n$ must be satisfied and that the transition matrix $\bar{P}$ might have entries $\bar{p}_{\alpha\beta} = |a_{\alpha\beta}|/\sum_{\beta}|a_{\alpha\beta}|$ for $\alpha,\beta = 1,2,\ldots,n$.

The estimator Θ^* for the matrix inverse was defined as follows:

$$E\Theta^*[g] = (g,x), \qquad \text{where} \quad \Theta^*[g] = \frac{gk_0}{pk_0}\sum_{j=0}^{\infty} W_j\varphi k_j \quad \text{and} \quad W_0 = 1,\ W_j = W_{j-1}\frac{a_{k_{j-1}k_j}}{p_{k_{j-1}k_j}} \tag{7}$$

where $\varphi_{k_j} = \delta_{k_j\beta}$ if $\alpha\beta$th entry of inverse matrix is computed. The sum for Θ^* must be dropped when $|W_i| < \delta$ [11,22], where δ is any given small number. Note that

$$|W_i| = \left|\frac{a_{\alpha_0\alpha_1}\ldots a_{\alpha_{i-1}\alpha_i}}{|a_{\alpha_0\alpha_1}|/\|A\|\ldots|a_{\alpha_{i-1}\alpha_i}|/\|A\|}\right| = \|A\|^i < \delta \tag{8}$$

Then to reach $\|A\|^i < \delta$ it follows that

$$T = i \le \frac{\log\delta}{\log\|A\|} \tag{9}$$

It is easily verified [22] that $\Theta^* \le \|\varphi\|/(1-\|A\|)$, which means that the variance of the r.v. Θ^* is bounded by its second moment:

$$D\Theta^* \le E\Theta^{*2} = \frac{\|\varphi\|^2}{(1-\|A\|)^2} \le \frac{1}{(1-\|A\|)^2} \tag{10}$$

In case of solving SLAE the stopping rule is $|W_i\varphi_{k_i}| < \delta$ [22]. Note that

$$|W_i\varphi_{k_i}| = \left|\frac{a_{\alpha_0\alpha_1}\ldots a_{\alpha_{i-1}\alpha_i}}{|a_{\alpha_0\alpha_1}|/\|A\|\ldots|a_{\alpha_{i-1}\alpha_i}|/\|A\|}\right||\varphi_{k_i}| = \|A\|^i\|\varphi\| < \delta$$

Then it follows that $T = i \le \log(\delta/\|\varphi\|)/\log\|A\|$.

It is easily verified [22] that $|\Theta^*| \le \|\varphi\|/(1-\|A\|)$, which means that the variance of the r.v. Θ^* is bounded by its second moment: $D\Theta^* \le E\Theta^{*2} = \|\varphi\|^2/(1-\|A\|)^2 \le f^2/(1-\|A\|)^2$.

Recall MI. Now consider the *Monte Carlo methods with absorption* (*MA*). There are several possibilities to build Markov chains using one [11] or n absorbing states [4,5]. The Monte Carlo method in this case is following [11]:

$$E\eta_T[g] = (g,x) \qquad \text{where} \quad \eta_T[g] = \frac{g_{k_0}}{p_{k_0}}W_T\frac{\varphi_{k_T}}{p_{k_T}} \quad \text{and} \quad W_0 = 1,\ W_T = W_i = W_{i-1}\frac{a_{k_{i-1}k_i}}{p_{k_{i-1}k_i}}, \tag{11}$$

where $\eta_T[g]$ is the r.v. taken over the chain of random length T (T is m.e. of the length of the chain when absorption takes place). From any transition probabilities $p_{\alpha\beta}$ such that $p_{\alpha\beta} \ge 0$, $\sum_\beta p_{\alpha\beta} < 1\ \forall\alpha = 1,2,\ldots,n$, and $p_{\alpha,n+1} = p_\alpha = 1 - \sum_{\beta=1}^n p_{\alpha\beta}$ is the probability that the trajectory ends in state α, we choose $p_{\alpha\beta} = |a_{\alpha\beta}|$, for $1 \le \alpha \le n$, $1 \le \beta \le n$. (For matrix inversion when $\alpha\beta$ entry is computed $\varphi_{k_T} = \delta_{k_T\beta}$.) Later we use the notation $\eta^*[g]$ for the r.v. $\eta_T[g]$ taken over infinitely long Markov chain.

The conditional mathematical expectation of chain length ET [4,5,23], if the chain starts in state $r=\alpha$ when $p_{\alpha\beta} = (|a_{\alpha\beta}|)$, is given by

$$E(T|r=\alpha) \le \max_\alpha \sum_{\beta=1}^{n}(I + P + P^2 + \cdots)_{\alpha\beta} \le \frac{1}{(1-\|A\|)} \tag{12}$$

The variance of $\eta^*[g]$ can be measured by the second moment of the r.v. $\eta^*[g]$ [4] and $D\eta^*[g] \leq (I-K)^{-1}\ Q\varphi^2$ where matrix K has entries $a^2_{\alpha\beta}/p_{\alpha\beta}$ and Q is a principal diagonal matrix with entries $1/p_\beta$, $\beta = 1, 2, \ldots, n$ on the principal diagonal. If we estimate the $\alpha\beta$th entry of the inverse matrix, its variance is bounded by

$$D\eta^*[g]_{\alpha\beta} < (I-K)^{-1}_{\alpha\beta}/p_\beta < \frac{1}{(1-\|A\|)^2} \tag{13}$$

According to the central limit theorem for the given error ϵ,

$$N \geq \frac{0.6745^2 D\eta^*[g]}{\epsilon^2} \quad \text{and thus} \quad N \geq \frac{0.6745^2}{\epsilon^2}\frac{1}{(1-\|A\|)^2} \tag{14}$$

is a lower bound on N which is independent of n. (In case of SLAE $N \geq (0.6745^2/\epsilon^2)(f^2/(1-\|A\|)^2)$.)

It is clear that T and N depend only on the matrix norm and precision. Furthermore, the size of N can be controlled by an appropriate choice of ϵ once P and A are known.

Consider N and T as functions of $1/(1-\|A\|)$. It is obvious from Eq. (12) and Eq. (14) that $T = O(\sqrt{N})$. It is easy also to show that $T = O(\sqrt{N})$ for MAO. In addition there are computational experiments in [7,18] showing this fact that for sufficiently large N we can take $T \approx \sqrt{N}$.

Next we consider the results from computational experiments involving different matrix $\|A\|$ norms and precision ϵ for MI [1]. The MAO and UM with stopping rules $|W_i| < \delta$ and MA have been implemented.

To illustrate the algorithmic efficiency of the MAO, consider the inversion of the 3×3 matrix B with B^{-1} found using MATLAB:

$$B = \begin{bmatrix} 0.7 & -0.2 & -0.0 \\ 0.0 & 0.67 & -0.1 \\ -0.1 & 0.0 & 0.8 \end{bmatrix}, \qquad B^{-1} = \begin{bmatrix} 1.4362 & 0.4287 & 0.0536 \\ 0.0268 & 1.5005 & 0.1876 \\ 0.1795 & 0.0536 & 1.2567 \end{bmatrix} \tag{15}$$

The B^{-1} matrices calculated by applying the three methods – MA, MAO and UM are as follows:

$$\begin{bmatrix} 1.3897 & 0.3933 & 0.1159 \\ 0.1265 & 1.2350 & 0.3321 \\ 0.4869 & 0.1568 & 0.9895 \end{bmatrix}, \quad \begin{bmatrix} 1.4721 & 0.4336 & 0.0751 \\ 0.0357 & 1.5468 & 0.2324 \\ 0.2013 & 0.0372 & 1.2599 \end{bmatrix}, \quad \begin{bmatrix} 2.2203 & 0.9757 & 0.0855 \\ 0.0161 & 2.9538 & 0.3837 \\ 0.3615 & 0.0871 & 1.4148 \end{bmatrix}.$$

The results are obtained with $\epsilon = 0.05$ and $N = 727$ for all the methods and with $\delta = 0.1$ for the MAO and UM. The MAO attains the required precision for $N = 727$ chains. The other two methods need more chains: e.g. the experiments show that UM needs about 6–10 times as many chains to reach the same precision as MAO. The minimal and maximal values for the length of Markov chains $T_{\min}$ and $T_{\max}$ and bounds on N for this example are presented in Table 3.

Further experiments have been carried out for matrices with different norms and different values of ϵ. Experimental results for $T_{\min}$, $T_{\max}$, and N for MAO are summarized in Table 4. From Tables 3 and 4, it is seen that the MAO needs fewer chains to reach a given precision.

To consider the efficiency of parallel implementation, we apply the three Monte Carlo methods to a randomly generated 100×100 dense matrix which has an inverse. A comparison between MA and MAO with $\epsilon = 0.02$ for such a matrix is given in Table 5. The figures show that both methods have