Comparison of Three Monte Carlo Methods for Matrix Inversion

V.N.Alexandrov, S. Lakka

Department of Statistics and Computational Mathematics, University of Liverpool, Liverpool, U.K.

Abstract. Three Monte Carlo methods for Matrix Inversion (MI) 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 has shown that an $n \times n$ matrix can be inverted in $3n/2 + N + T$ steps on regular arrays with $O(n^2 NT)$ 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, which are independent of matrix size n), which show that these designs are faster than the existing designs for large values of n.
In this paper we take another implementation approach, we consider parallel Monte Carlo algorithms for MI on MIMD environment, i.e. running on a cluster of workstations under PVM. The Monte Carlo method with almost optimal frequency function performs best of the three methods as it needs about six-ten times less chains for the same precision.

1 Introduction

The problem of inverting a real $n \times n$ matrix is of unquestionable importance in many scientific and engineering applications: for example 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 [1]. 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 a SLAE, by performing random sampling of a certain chance variable, whose mathematical expectation is the desired solution [16, 18]. We concentrate on Monte Carlo methods for MI since: **firstly,** only $O(NT)$ steps are required to find an element of the inverse matrix (N is a number of chains and T is a measure on the chains length in the stochastic process, which are independent of n) and **secondly,** the sampling process for stochastic methods is inherently parallel.

Recently Megson, Alexandrov, and Dimov have presented regular arrays for Matrix Inversion by Monte Carlo method [13, 11], exploiting the inherent par-

allelism of the method. The full matrix is inverted in $3n/2 + N + T$ 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 Table 1). Thus for sufficiently large n the Monte Carlo approach is theoretically more efficient than the usual direct and iterative methods[3].

Array	A	T	AT
Robert-Trystram [14]	n^2	$5n$	$5n^3$
Kung-Lo-Lewis [9]	n^2	$5n$	$5n^3$
Megson [10]	$(3/2)n^2 - (n/2)$	$4n$	$6n^3 - 2n^2$
Rajopadhye [15]	n^2	$4n$	$4n^3$
Delosme [5]	$5n^2/4$	$4n$	$5n^3$
Megson-Alexandrov-Dimov [13]	n^2NT	$< 4n$	$< 4n^3NT$

Table 1. Area-Time trade-off for various arrays for matrix inversion

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 organised as follows. Section 2 briefly outlines the essential details of matrix inversion 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 system of linear algebraic equations (SLAE) is presented in the form:

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

where A is a real square $n \times n$ matrix, $x = (x_1, x_2, ..., x_n)^t$ is a $1 \times n$ solution vector and $\varphi = (\varphi_1, \varphi_2, ..., \varphi_n)^t$ is a given vector. (If we consider the system $Lx = b$, then it is possible to choose 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 $j = 1, 2, \cdots, i$. 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 require that $\sum_{\alpha=1}^{n} p_\alpha = 1$, $\sum_{\beta=1}^{n} p_{\alpha\beta} = 1$ for any $\alpha = 1, 2, ..., n$, the distribution $(p_1, ..., p_n)^t$ is acceptable to vector g and similarly the distribution $p_{\alpha\beta}$ is acceptable to A [16].

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 [16] that the mathematical expectation $E\Theta^*[g]$ of random variable $\Theta^*[g]$ is:

$$\begin{aligned} E\Theta^*[g] &= (g, x) \\ \text{where } \Theta^*[g] &= \frac{g_{k_0}}{p_{k_0}} \sum_{j=0}^{\infty} W_j \varphi_{k_j} \\ \text{and} \quad W_0 &= 1 \,, \quad W_j = W_{j-1} \frac{a_{k_{j-1}k_j}}{p_{k_{j-1}k_j}} \end{aligned} \tag{3}$$

We use the following notation for a partial sum (3) $\theta_i[g] = \frac{g_{k_0}}{p_{k_0}} \sum_{j=0}^{i} W_j \varphi_{k_j}$. 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 $i \longrightarrow \infty$. 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, for example the r-th component of x, we choose $g = e(r) = (0, ..., 0, 1, 0, ..., 0)$ where the one is in the r-th place. It follows that $(g, x) = \sum_{\alpha=1}^{n} e_\alpha(r) x_\alpha = x_r$ 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 s-th 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

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

where $W_j = W_{j-1} \frac{a_{k_{j-1}k_j}}{p_{k_{j-1}k_j}}$ 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 $r' = 1, 2, ..., n$, **the same set of N chains can be used to compute a single row of the inverse [16]** with a consequent saving in computation.
The **probable error** of the method, is defined as $r_N = 0.6745\sqrt{D\theta/N}$, where $P\{|\bar{\theta} - E(\theta)| < r_N\} \approx 1/2 \approx P\{|\bar{\theta} - E(\theta)| > r_N\}$, if we have N independent

realizations of random variable (r.v.) θ with mathematical expectation $E\theta$ and average $\bar{\theta}$ [16].

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 reduces 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 [13]. The main results and Theorems are outlined here.

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

$$\psi_\alpha = (\sum_\beta \frac{a_{\alpha\beta}^2 \varphi_\beta^2}{p_{\alpha\beta}})^{1/2} \quad \text{and } \widehat{\psi}_\alpha = \sum_\beta |a_{\alpha\beta}\varphi_\beta| \ \forall \alpha = 1, 2, ..., n \tag{6}$$

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* $\min_p \psi_\alpha = \widehat{\psi}_\alpha \ \forall \alpha = 1, 2, ..., n$

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

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

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

where $c_0 = (\sum_\beta |g_\beta \widehat{\psi}_\beta|)^{-1}$ *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 $\widehat{p}_i = c_0 |g_{\alpha_0}| \prod_{j=1}^i |a_{\alpha_{j-1}\alpha_j}| \varphi_{\alpha_i}$, *for* $i = 1, 2, ...$ and $\widehat{p}_{k,i} = c_0 |g_{\alpha_0}| \prod_{j=1}^i |a_{\alpha_{j-1}\alpha_j}| [\varphi_{\alpha_i} A^{(k-i)} \varphi_{\alpha_i}]^{1/2}$, *for* $i = 1, 2, ...$ where $A^{k-i} = \sum_{\alpha_{i+1}} \cdots \sum_{\alpha_k} a_{\alpha_i \alpha_{i+1}} \cdots a_{\alpha_{k-1}\alpha_k} \varphi_{\alpha_k}$. 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 principal of collinearity of norms [13] we can choose $p_{\alpha\beta}$ proportional to the $|a_{\alpha\beta}|$.

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 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 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 on a separate processor.

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. 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 allocating 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 $\alpha = 1, 2, ..., n$ must be satisfied and transition matrix $\bar{P}$ might have entries $\bar{p}_{\alpha\beta} = \frac{|a_{\alpha\beta}|}{\sum_{\beta}|a_{\alpha\beta}|}$ for $\alpha, \beta = 1, 2, ..., n$.

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

$$\begin{aligned} E\Theta^*[g] &= (g, x), \\ \text{where } \Theta^*[g] &= \frac{g_{k_0}}{p_{k_0}} \sum_{j=0}^{\infty} W_j \varphi_{k_j} \\ \text{and} \quad W_0 &= 1\,, \quad W_j = W_{j-1} \frac{a_{k_{j-1}k_j}}{p_{k_{j-1}k_j}}. \end{aligned} \tag{7}$$

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 $|W_i| < \delta$ [16, 6], where δ is any given small number. Note that

$$|W_i| = |\frac{a_{\alpha_0\alpha_1} \cdots a_{\alpha_{i-1}\alpha_i}}{\frac{|a_{\alpha_0\alpha_1}|}{\|A\|} \cdots \frac{|a_{\alpha_{i-1}\alpha_i}|}{\|A\|}}| = \|A\|^i < \delta. \tag{8}$$

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

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

It is easy to find [16] that $|\Theta^*| \leq \frac{\|\varphi\|}{(1-\|A\|)}$, which means that the variance of r.v. Θ^* is bounded by its second moment:

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

Now consider the **Monte Carlo methods with absorption (MA)**. There are several possibilities to build Markov chains using one [6] or n absorbing states [2, 3]. The Monte Carlo method in this case is following [6]:

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

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 take place). From any transition probabilities $p_{\alpha\beta}$ such that $p_{\alpha\beta} \geq 0$, $\sum_\beta p_{\alpha\beta} < 1 \forall \alpha = 1, 2, ..., n$ 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 $1 \leq \alpha \leq n$, $1 \leq \beta \leq n$. (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 [17, 2, 3], if the chain starts in state $r = \alpha$ when $p_{\alpha\beta} = (|a_{\alpha\beta}|)$, is given by

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

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

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

is a lower bound on N which is independent of n. 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 $\frac{1}{(1-\|A\|)}$. It is obvious from (14) and (12) 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 [4, 13] 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. 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 3×3 matrix B with B^{-1} found using MATLAB:

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

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: for example, the experiments show that UM needs about 6-10 times more chains to reach the same precision in comparison with MAO. The minimal and maximal values for the length of Markov chains Tmin and Tmax and bounds on N for this example are presented in Table 2.

$\|A\|$	=	0.5	MA		MAO		UM	
N	ϵ	δ	Tmin	Tmax	Tmin	Tmax	Tmin	Tmax
45	0.2	0.2	1	7	3	5	3	7
727	0.05	0.1	1	14	3	6	3	12
4549	0.02	0.1	1	17	3	7	3	14

Table 2. A comparison between parameters of MA, MAO and UM methods

Further experiments has been carried out for matrices with different norms and different values of ϵ. Experimental results for Tmin, Tmax and N for MAO are summarized in Table 3. From Tables 2 and 3, 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 randomly generated 100×100 dense matrix which has an inverse. A comparison between MA and MAO with $\epsilon = 0.02$ for such matrix is given in Table 4. The figures show that both methods have efficient parallel implementation (efficiency above 0.9) but MA is about two times slower (the time is in seconds) compared with MAO. The UM method has been also

ϵ	δ	$\|\|A\|\|$	T_{min}	T_{max}	N	$\|\|A\|\|$	T_{min}	T_{max}	N
0.5	0.1	0.9	8	16	181 - 1600	0.5	0	3	7 - 64
0.2	0.1	0.9	6	17	1137 - 10000	0.5	0	4	45 - 399
0.1	0.1	0.9	7	17	4549 - 40000	0.5	0	4	181 - 1599
0.05	0.05	0.9	10	21	18198 - 160000	0.5	0	5	727 - 6398
0.01	0.01	0.9	15	28	454950 - 4000003	0.5	0	6	18198 - 160000

Table 3. Method without absorption

method	nproc	1	2	3	4	5	10
MA	TIME	1308.9	664.417	447.029	338.392	272.4	142.52
MA	Efficiency	1	0.985	0.976	0.967	0.961	0.918
MAO	TIME	757.829	384.685	258.556	193.917	156.67	79.31
MAO	Efficiency	1	0.987	0.979	0.977	0. 967	0.955

Table 4. A comparison between parameters of MA and MAO

implemented, but needs 6-10 times more chains to reach the same precision as MAO.

Generally, The results from the experiments show that MAO has fewer and shorter chains compared with MA. The comparison between MAO and UM shows that MAO methods require about six-ten times less chains to reach the same given precision [13, 12]. In addition MAO is roughly twice as fast as MA and faster than UM. In addition, the experiments show that convergence is very slow when the matrix norm is close to 0.9 or is tending to 1, so the question of fast Monte Carlo preconditioning and reducing the norm of the matrix is still open. Further experiments are needed to investigate parallel Monte Carlo methods for dense, sparse and structured matrices and to identify the break-even points compared with other best know parallel direct and iterative methods.

6 Conclusion

In our parallel implementation we have to compute n rows in parallel. To compute one row we need N independent chains with length T, and for n rows in parallel we need nN such independent chains of length T, where N and T are the mathematical expectations of the number of chains and chain length, respectively. So the execution time on p processors for MI by Monte Carlo is bounded by $O(nNT/p)$ (excluding communication time). According to the discussion and results above N and T depend only on the matrix norm and precision and do not depend on the matrix size. Therefore the Monte Carlo methods can be efficiently implemented on MIMD environment and in particular on a cluster of workstations under PVM.

In particular it should be noted that the Monte Carlo methods are well suited to large problems where other solution methods are impractical or impossible for computational reasons, for calculating quick rough estimate of inverse matrix, and when only an element or one row of the inverse matrix are desired. Consequently, if massive parallelism is available and if low precision is acceptable, Monte Carlo algorithms could become favourable for $n >> N$.

References

1. Bertsekas D.P. and Tsitsiklis , *Parallel and Distributed Computation* , Prentice Hall, 1989
2. Curtiss J.H., Monte Carlo methods for the iteration of linear operators. *J. Math Phys.*, vol 32, No 4 (1954), pp.209–232.
3. Curtiss J.H. *A Theoretical Comparison of the Efficiencies of two classical methods and a Monte Carlo method for Computing one component of the solution of a set of Linear Algebraic Equations.* ,Proc. Symposium on Monte Carlo Methods , John Wiley and Sons, 1956, pp.191-233.
4. Dimov I. and O.Tonev, Random Walk on Distant Mesh Points Monte Carlo Method. *Journal of Statistical Physics,* Vol. 70, Nos.5/6, pp. 1333–1342,1993.
5. Delosme J.M., *A parallel algorithm for the algebraic path problem* Parallel and Distributed Algorithms, eds. Cosnard et all, Bonas, France, 1988, pp67-78.
6. Ermakov S.M. *Method Monte Carlo and related topics* , Moscow, Nauka, 1975.
7. El-Amawy A., A Systolic Architecture for Fast Dense Matrix Inversion, *IEEE Transactions on Computers*, Vol.C-38, No3, pp.449-455, 1989.
8. Holton J.H., A Retrospective and Prospective Survey of the Monte Carlo Method, *SIAM Rev.* , Vol.12, No1, 1970 , pp.1-63.
9. Kung S.Y., S.C.Lo and P.S.Lewis, Optimal Systolic Design for the Transitive Closure and shortest path problems, *IEEE Transactions on Computers* , Vol.C-36, N05, 1987, pp.603-614.
10. Megson G.M., A Fast Faddeev Array, *IEEE Transactions on Computers*, Vol.41, N012, December 1992, pp.1594-1600.
11. G.M.Megson, V.Aleksandrov, I.T. Dimov *A Fixed Sized Regular Array for Matrix Inversion by Monte Carlo Method,* In Advances in Numerical Methods and Applications , World Scientific, pp.255-264, 1994.
12. S. Lakka *Parallel Matrix Inversion Using a Monte Carlo Method* MSc. Dissertation, University of Liverpool, September 1995.
13. G.M.Megson, V.Aleksandrov, I. Dimov *Systolic Matrix Inversion Using Monte Carlo Method,* J. Parallel Algorithms and Applications , Vol.3, pp.311-330, 1994.
14. Robert Y. and D. Trystram, *Systolic Solution of the Algebraic Path Problem* , in Systolic Arrays, W.Moore et. al, Adam Hilger, 1987 pp.171-180
15. Rajopadhye S., *An Improved Systolic Algorithm for the Algebraic Path Problem* , Integration the VLSI journal, 14, pp279-290, 1993.
16. Sobol' I.M. *Monte Carlo numerical methods.* Moscow, Nauka, 1973 (Russian)(English version Univ. of Chicago Press 1984).
17. Snell J.L. *Introduction to Probability*, Random House, New York, 1988
18. Westlake J.R., *A Handbook of Numerical Matrix Inversion and Solution of Linear Equations*, John Wiley and Sons, New York, 1968.