Outline

• Motivation

• Markov Chain Monte Carlo
  – Bilinear Forms of Matrix Powers
  – The Algorithm

• Applicability and Acceleration Analysis

• Numerical Experiments

• Concluding Remarks
Motivation

• The direct methods for solving SLAE require $O(n^3)$ sequential steps (e.g. Gaussian elimination, Gauss-Jordan methods, LU-factorisation) is used;

• "Classical" Iterative Methods (Jacobi, Gauss-Seidel, SOR, SUR) require $O(kn^2)$ operations;

• Conjugate Gradient methods require $O(n^2)$ operations;

• Monte Carlo - $O(Nkd)$ or $O(Nk \log d)$; $O(n^2)$ or $O(n \log n)$ operations for dense matrix format and $O(\log n)$ operations for sparse matrix format.¹

Formulation of the Monte Carlo Algorithm

Bilinear Forms of Matrix Powers

We are interested in bilinear forms of matrix powers:

\[(v, A^k h).\] (1)

For \(x\), the solution of a SLAE \(Bx = b\) then

\[(v, x) = \left( v, \sum_{i=0}^{k} A^i h \right),\] (2)

where the Jacobi Over-relaxation Iterative Method has been used to transform the SLAE into the problem \(x = Ax + h\). In cases where the Neumann series does not converge a resolvent method can be used.
Markov Chain Monte Carlo

The random trajectory (chain) $T_k$ of length $k$ starting in the state $\alpha_0$ is defined as follows:

$$T_k = \alpha_0 \rightarrow \alpha_1 \rightarrow \ldots \rightarrow \alpha_j \rightarrow \ldots \rightarrow \alpha_k,$$

where $\alpha_j$ means the number of the state chosen, for $j = 1, \ldots, k$.

Assume that

$$P(\alpha_0 = \alpha) = p_\alpha, \quad P(\alpha_j = \beta|\alpha_{j-1} = \alpha) = p_{\alpha\beta},$$

where $p_\alpha$ is the probability that the chain starts in state $\alpha$ and $p_{\alpha\beta}$ is the transition probability to state $\beta$ after being in state $\alpha$. Probabilities $p_{\alpha\beta}$ define a transition matrix $P$. We require that

$$\sum_{\alpha=1}^{n} p_\alpha = 1 \quad \text{and} \quad \sum_{\beta=1}^{n} p_{\alpha\beta} = 1, \quad \text{for any} \quad \alpha = 1, 2, \ldots, n.$$
We will consider a special choice of density distributions $p_i$ and $p_{ij}$ defined as follows:

$$p_i = \frac{|v_i|}{\|v\|}, \quad \|v\| = \sum_{i=1}^{n} |v_i| \quad \text{and} \quad p_{ij} = \frac{|a_{ij}|}{\|a_i\|}, \quad \|a_i\| = \sum_{j=1}^{n} |a_{ij}|.$$

(5)
Monte Carlo Algorithm for Computing Bilinear Forms of Matrix Powers \((v, A^k h)\)

The pair of density distributions (5) defines a finite chain of vector and matrix entrances:

\[
v_{\alpha_0} \rightarrow a_{\alpha_0\alpha_1} \rightarrow \ldots \rightarrow a_{\alpha_{k-1}\alpha_k}.
\] (6)

The latter chain induces the following product of matrix/vector entrances and norms:

\[
A^k_v = v_{\alpha_0} \prod_{s=1}^{k} a_{\alpha_{s-1}\alpha_s}; \quad \| A^k_v \| = \| v \| \times \prod_{s=1}^{k} \| a_{\alpha_{s-1}} \|.
\]

The rule for creating the value of \(\| A^k_v \|\) is the following: the norm of the initial vector \(v\), as well as norms of all row-vectors of matrix \(A\) visited by the chain (6), defined by densities (5), are included. For such a choice of densities \(p_i\) and \(p_{ij}\) we can prove the following theorem:
\[ E\{h_{\alpha_k}\} = \frac{\text{sign}\{A^k_v\}}{\|A^k_v\|} (v, A^k h) \]  

(7)

The standard deviation \(\sigma\{h_{\alpha_k}\}\) is finite. Since random variable

\[ \theta^{(k)} = \text{sign}\{A^k_v\} \times \|A^k_v\| h_{\alpha_k} \]

is an unbiased estimate of the form \((v, A^k h)\), (7) can be used to construct a MC algorithm.

Let us consider \(N\) realizations of the Markov chain \(T_k\) (3) defined by the pair of density distributions (5). Denote by \(\theta^{(k)}_i\) the \(i^{th}\) realization of the random variable \(\theta^{(k)}\). Then the value

\[ \bar{\theta}^{(k)} = \sum_{i=1}^{N} \theta^{(k)}_i = \text{sign}\{A^k_v\} \|A^k_v\| \sum_{i=1}^{N} \{h_{\alpha_k}\}_i \]

(8)
can be considered as a MC approximation of the form \((v, A^k h)\). The probability error of this approximation can be presented in the following form:

\[
R_N^{(k)} = \left| (v, A^k h) - \bar{\theta}^{(k)} \right| = c_p \sigma \{ \theta^{(k)} \} N^{-\frac{1}{2}}. \tag{9}
\]

In fact, (8) together with the sampling rules using probabilities (5) defines a MC algorithm. The expression (8) gives a MC approximation of the form \((v, A^k h)\) with a probability error \(R_N^{(k)}\). Obviously, the quality of the MC algorithm depends on the behavior of the standard deviation \(\sigma \{ \theta^{(k)} \}\). So, there is a reason to consider a special class of robust MC algorithms.

We compute the first 10 iterations by Monte Carlo and by simple matrix-vector multiplication with double precision assuming that the obtained results are ”exact” (they still contain some roundoff errors that are relatively small). The results of computations are presented on Figure 1. The first impression is that the results are good.
Figure 1: Monte Carlo and "Exact" values $\frac{v^T A^k v}{\|v\|}$ for a random non-balanced matrix $A_3$ of size $128 \times 128$. In all experiments the number $N$ of Markov chains is $10^6$. 
Figure 2: Relative MC error for values $\frac{v^T A^k v}{\|v\|}$ for a random non-balanced matrix of size $128 \times 128$. In all experiments the number $N$ of Markov chains is $10^6$. 

IMACS MCM 2007 The University of Reading 18-21 June
Applicability and Acceleration Analysis

Robust MC Algorithms

Definition 1. *MC algorithm for which the standard deviation does not increase with increasing of matrix power* \( k \) *is called robust MC algorithm.*

Lemma 1. *If MC algorithm is robust, then there exist a constant* \( M \) *such that*

\[
\lim_{k \to \infty} \sigma\{\theta^{(k)}\} \leq M.
\]

It is interesting to answer the question:

- How small could be the probability error? and
- Is it possible to have MC algorithms with zero probability error?
Interpolation MC Algorithms

Definition 2. *MC algorithm for which the probability error is zero is called interpolation MC algorithm.*

Theorem 1. *Let*

\[
\hat{h} = \{h_i^2\}^n_{i=1},
\]

\[
\bar{v} = \{|v_i|\}^n_{i=1},
\]

\[
\bar{A} = \{|a_{ij}|\}^n_{i,j=1}.
\]

*Then*

\[
D\{\theta^{(k)}\} = \|A_v^k\| \left(\bar{v}, \bar{A}^k\hat{h}\right) - (v, A^k\hat{h})^2.
\]

Corollary 1. *For a perfectly balanced singular stochastic matrix the MC algorithm defined by density distributions (5) is an interpolation MC algorithm.*
Experimental Results

- IBM p690+ Regatta system cluster of IBM SMP nodes, containing a total of 1536 IBM POWER5 processors;

- Sun Fire 15K server with 52x0.9GHz UltraSPARC III processors;

- SGI Prism equipped with 8 x 1.5 GHz Itanium II processors and 16 GByte of main memory.

- Dense and unstructured sparse random matrices of sizes

  - $n = 1000$, $n = 5000$, $n = 10000$, $n = 15000$, $n = 20000$, $n = 40000$. are used.
Figure 3: Comparison of Robust Monte Carlo algorithm results with the *exact* solution for the bilinear form of a dense matrix of size $n = 15000$. 
Figure 4: Comparison of the Relative MC error for the robust and non-robust algorithms. Matrices of size \( n = 15000 \) are used.

\[
R_N^{(k)} = \left| \frac{1}{N} \sum_{i=1}^{N} \theta_i^{(k)} - \frac{(v, A^k h)}{(v, h)} \right|, \quad \text{Rel}_N^{(k)} = \frac{(v, h)}{(v, A^k h)} R_N^{(k)}.
\]
Figure 5: The relative MC error for the robust and non-robust algorithms. the matrix size is $n = 15000$. 
Figure 6: Comparison of the MC results for bilinear form of matrix powers for a sparse matrix of size $n = 10000$ with the exact solution. 5 or 6 Monte Carlo iterations are enough for solving the system of linear algebraic equations or for computing the dominant eigenvalue for the robust Monte Carlo.
Figure 7: Comparison of the MC results for bilinear form of matrix powers for a sparse matrix of size $n = 40000$ with the exact solution.
Concluding Remarks

- We are focused on the computing bilinear form of matrix powers \((v, A^k h)\) as a basic subtask of MC algorithms for solving a class of Linear Algebra problems.

- We study the applicability and robustness of Markov chain Monte Carlo. The robustness of the Monte Carlo algorithm with large dense and unstructured sparse matrices has been demonstrated.

We can conclude that the **balancing** of the input matrix is very important for MC computations. A balancing procedure should be performed as an initial (preprocessing) step in order to improve the quality of Monte Carlo algorithms. For matrices that are ”close” in some sense to the stochastic matrices the accuracy of the MC algorithm is very high.
References


Questions?

http://www.personal.reading.ac.uk/ sis04itd/