Applicability and Robustness of Monte Carlo Algorithms for Very Large Linear Algebra Problems

Ivan Dimov

ACET, The University of Reading
and
IPP - BAS, Sofia

IMACS MCM 2007 The University of Reading 18-21 June

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]

[1] J. Curtiss: 1956; J. Hammersly and D. Handscomb: 1964; D. Knuth: 1964; J.R. Westlake: 1968; I. Sobol: 1973, 1979; S. Ermakov and G. Mikhailov: 1982; M. Kalos and P. Whitlock: 1986; I.D.: 1989, 1991, 2000; I.D., T. Dimov and T. Gurov: 1997; R. Rubinstein: 1992; I.D. and A. Karaivanova: 1994, 1999; M. Mascagni and A. Karaivanova: 2000; I.D. and V. Alexandrov: 2001; V. Alexandrov, E. Atanassov and I. D.: 2004; I.D., V. Alexandrov, S. Branford, and C. Weihrauch: 2006; I.D., V. Alexandrov, R. Papancheva, and C. Weihrauch: 2007

Formulation of the Monte Carlo Algorithm

Bilinear Forms of Matrix Powers

We are interested in bilinear forms of matrix powers:

$$(v, A^k h). \tag{1}$$

For x, the solution of a SLAE $Bx = b$ then

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

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 α_0 is defined as follows:

$$T_k = \alpha_0 \to \alpha_1 \to \ldots \to \alpha_j \to \ldots \to \alpha_k, \tag{3}$$

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

$$\sum_{\alpha=1}^{n} p_\alpha = 1 \quad \text{and} \quad \sum_{\beta=1}^{n} p_{\alpha\beta} = 1, \quad \text{for any} \quad \alpha = 1, 2, ..., n. \tag{4}$$

We will consider a special choice of density distributions p_i and p_{ij} defined as follows:

$$p_i = \frac{|v_i|}{\parallel v \parallel}, \quad \parallel v \parallel = \sum_{i=1}^{n} |v_i| \quad \text{and} \quad p_{ij} = \frac{|a_{ij}|}{\parallel a_i \parallel}, \quad \parallel a_i \parallel = \sum_{j=1}^{n} |a_{ij}|. \qquad (5)$$

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:

$$v_{\alpha_0} \rightarrow a_{\alpha_0\alpha_1} \rightarrow \ldots \rightarrow a_{\alpha_{k-1}\alpha_k}. \tag{6}$$

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

$$A_v^k = v_{\alpha_0} \prod_{s=1}^{k} a_{\alpha_{s-1}\alpha_s}; \quad \| A_v^k \| = \| v \| \times \prod_{s=1}^{k} \| a_{\alpha_{s-1}} \| \,.$$

The rule for creating the value of $\| A_v^k \|$ 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:

$$E\{h_{\alpha_k}\} = \frac{sign\{A_v^k\}}{\parallel A_v^k \parallel} (v, A^k h) \,. \tag{7}$$

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

$$\theta^{(k)} = sign\{A_v^k\} \times \parallel A_v^k \parallel h_{\alpha_k}$$

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 $\theta_i^{(k)}$ the i^{th} realization of the random variable $\theta^{(k)}$. Then the value

$$\bar{\theta}^{(k)} = \sum_{i=1}^{N} \theta_i^{(k)} = sign\{A_v^k\} \parallel A_v^k \parallel \sum_{i=1}^{N} \{h_{\alpha_k}\}_i \tag{8}$$

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.

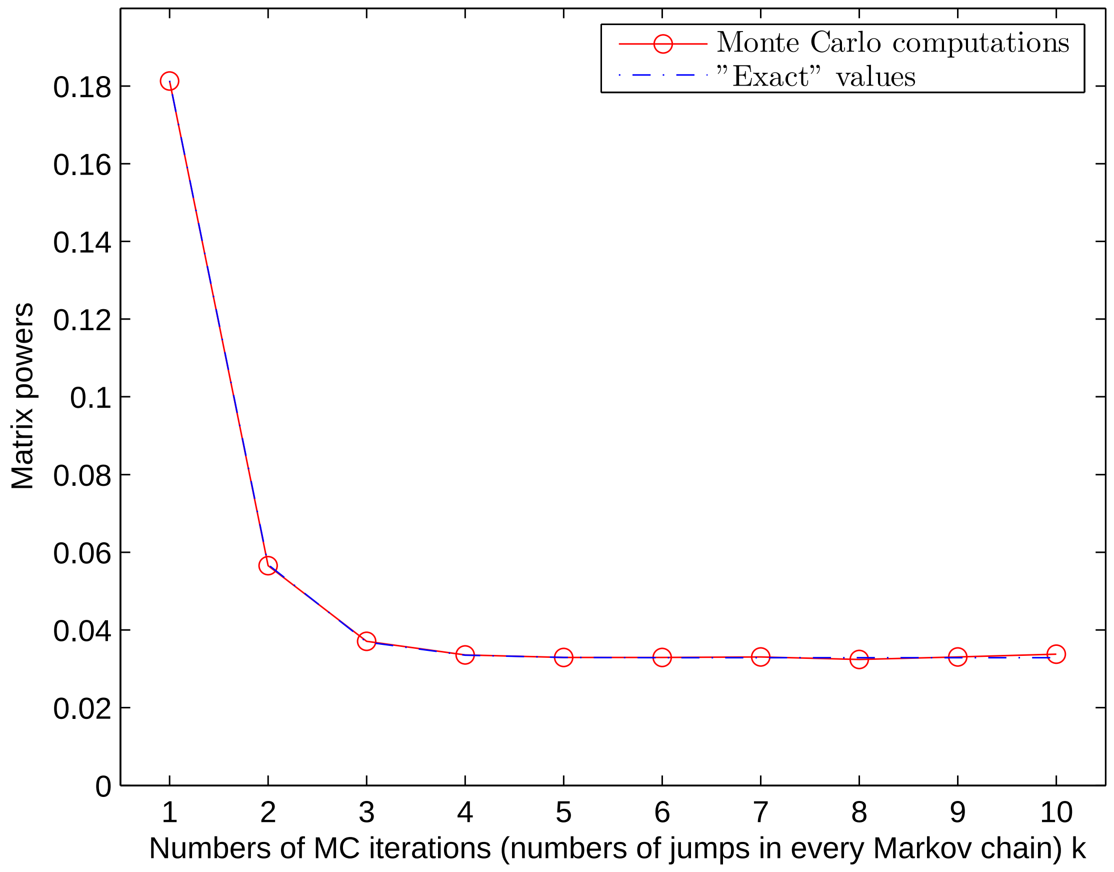

Figure 1: Monte Carlo and "Exact" values $\dfrac{v^T A^k v}{\|v\|}$ for a random non-balanced matrix A_3 of size 128×128. In all experiments the number N of Markov chains is 10^6.

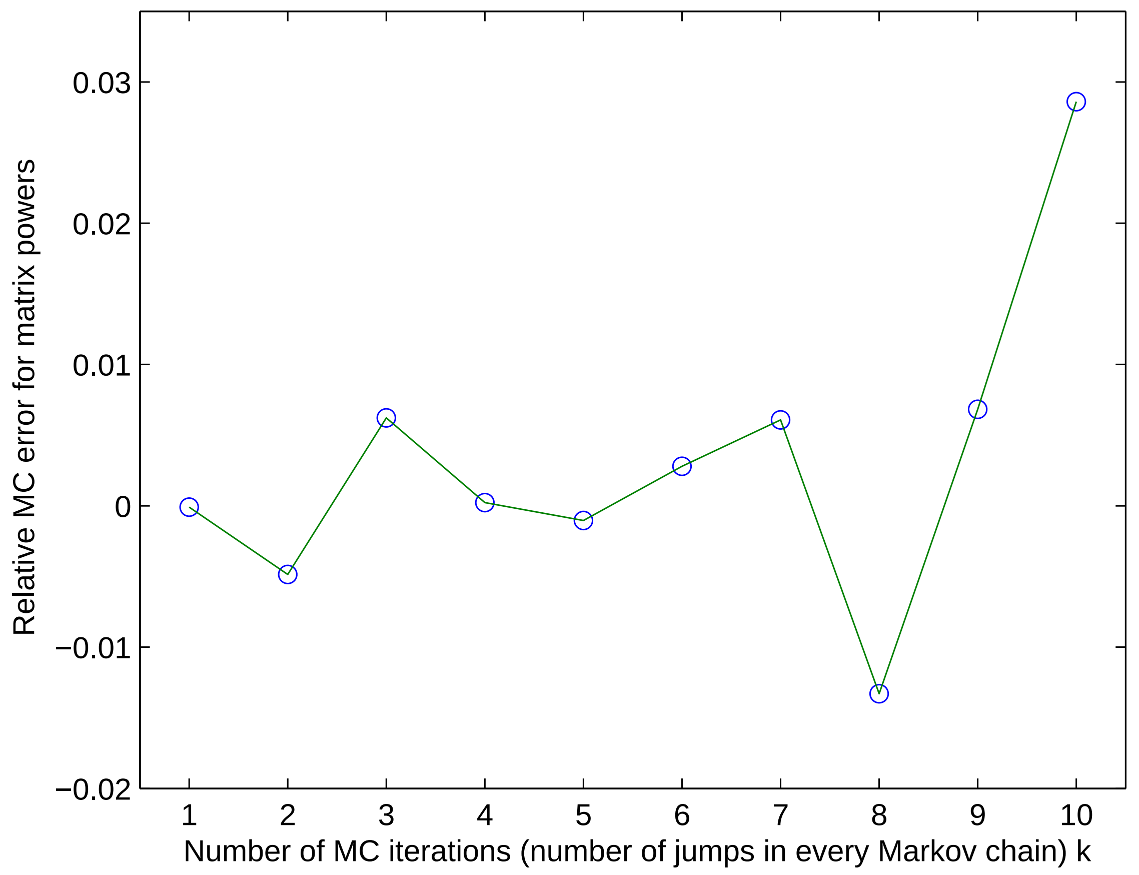

Figure 2: Relative MC error for values $\dfrac{v^T A^k v}{\|v\|}$ for a random non-balanced matrix of size 128×128. In all experiments the number N of Markov chains is 10^6.

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*

$$\begin{aligned} \hat{h} &= \{h_i^2\}_{i=1}^n, \\ \bar{v} &= \{|v_i|\}_{i=1}^n, \\ \bar{A} &= \{|a_{ij}|\}_{i,j=1}^n. \end{aligned}$$

Then

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

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.

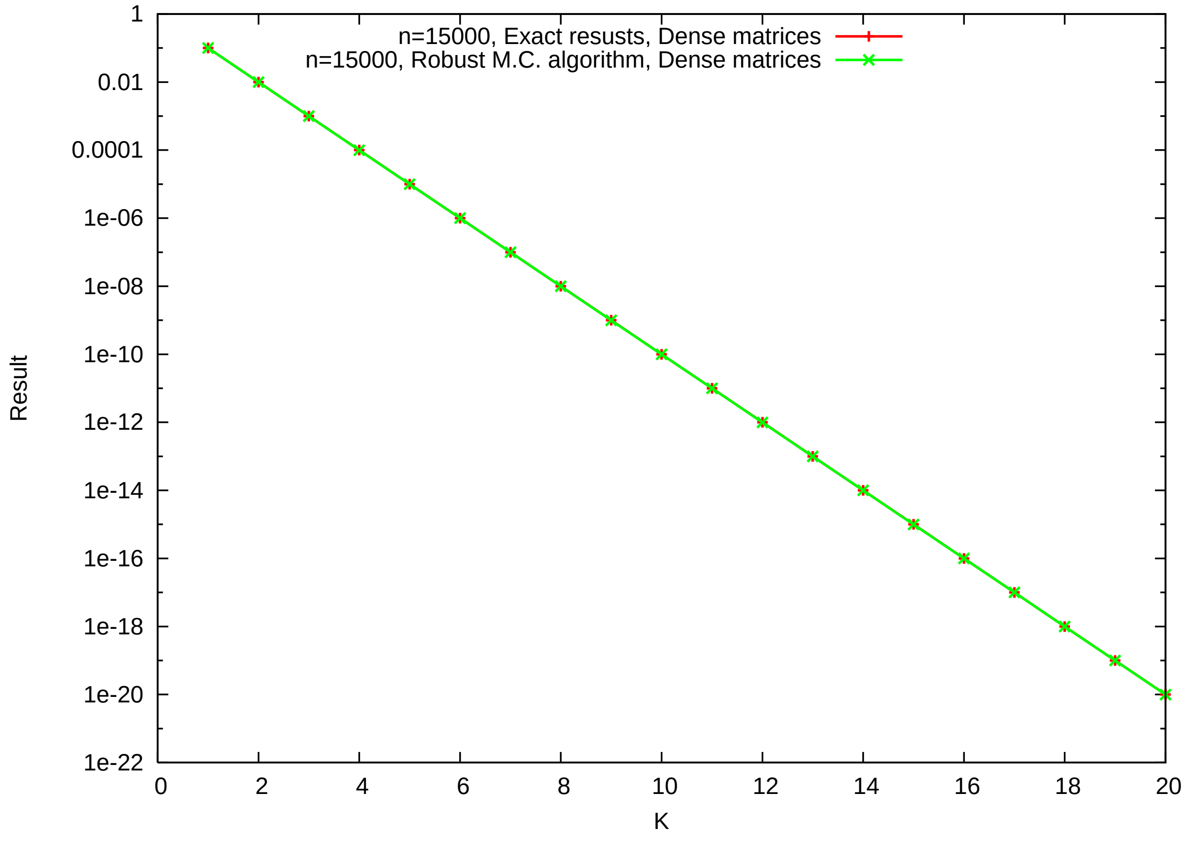

Figure 3: Comparison of Robust Monte Carlo algorithm results with the *exact* solution for the bilinear form of a dense matrix of size $n = 15000$.

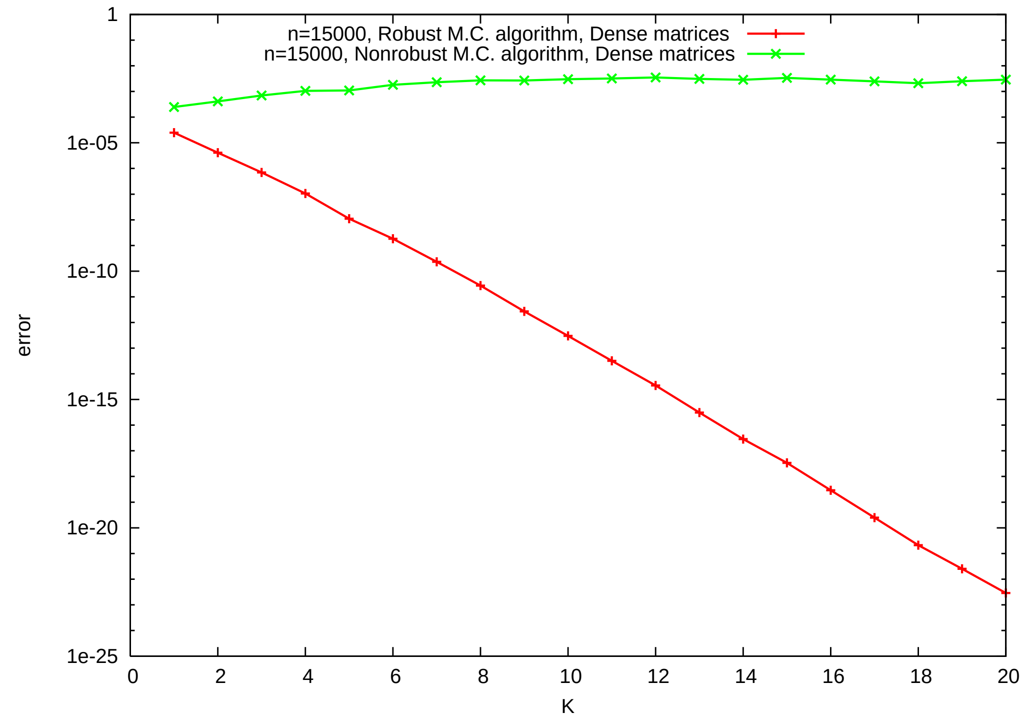

Figure 4: Comparison of the Relative MC error for the robust and non-robust algorithms. Matrices of size $n = 15000$ are used.

$$R_N^{(k)} = \left| \frac{1}{N} \sum_{i=1}^{N} \theta_i^{(k)} - \frac{(v,\, A^k h)}{(v,\, h)} \right|, \quad Rel_N^{(k)} = \frac{(v,\, h)}{(v,\, A^k h)} R_N^{(k)}.$$

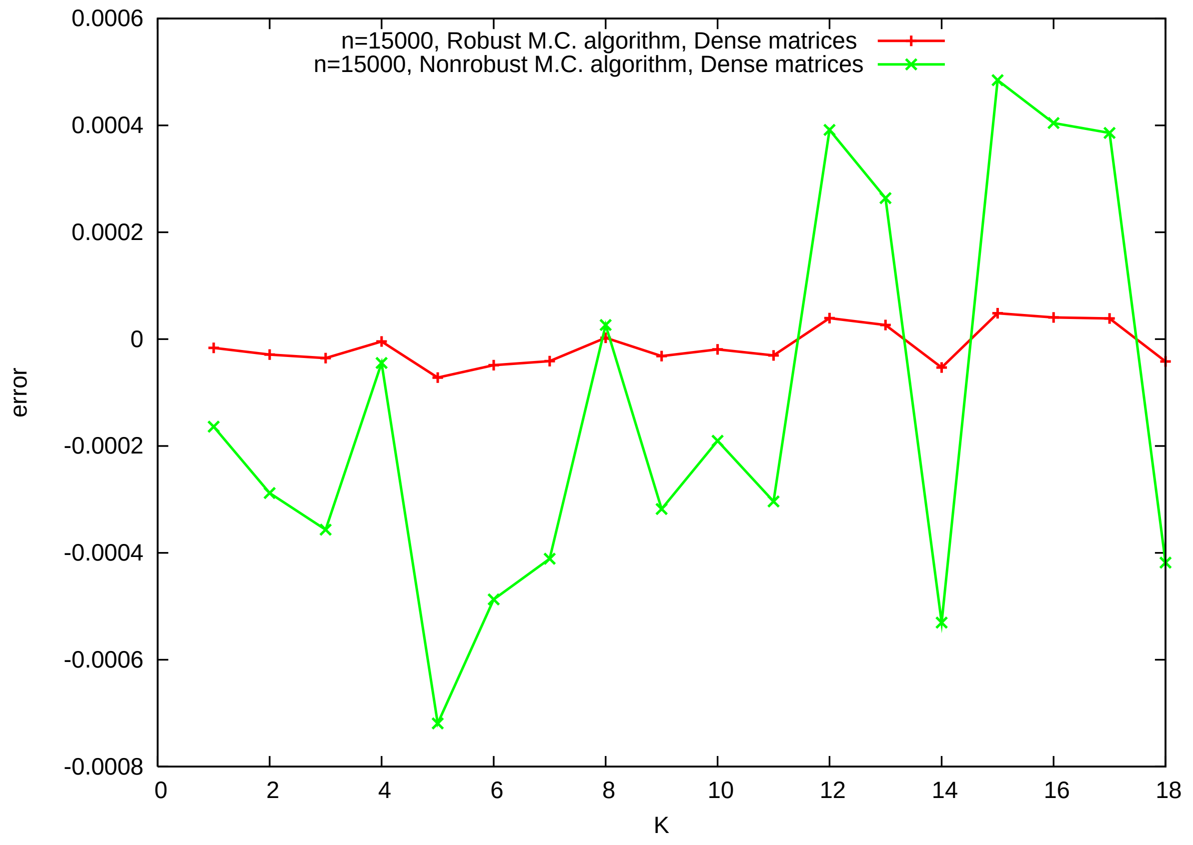

Figure 5: The relative MC error for the robust and non-robust algorithms. the matrix size is $n = 15000$.

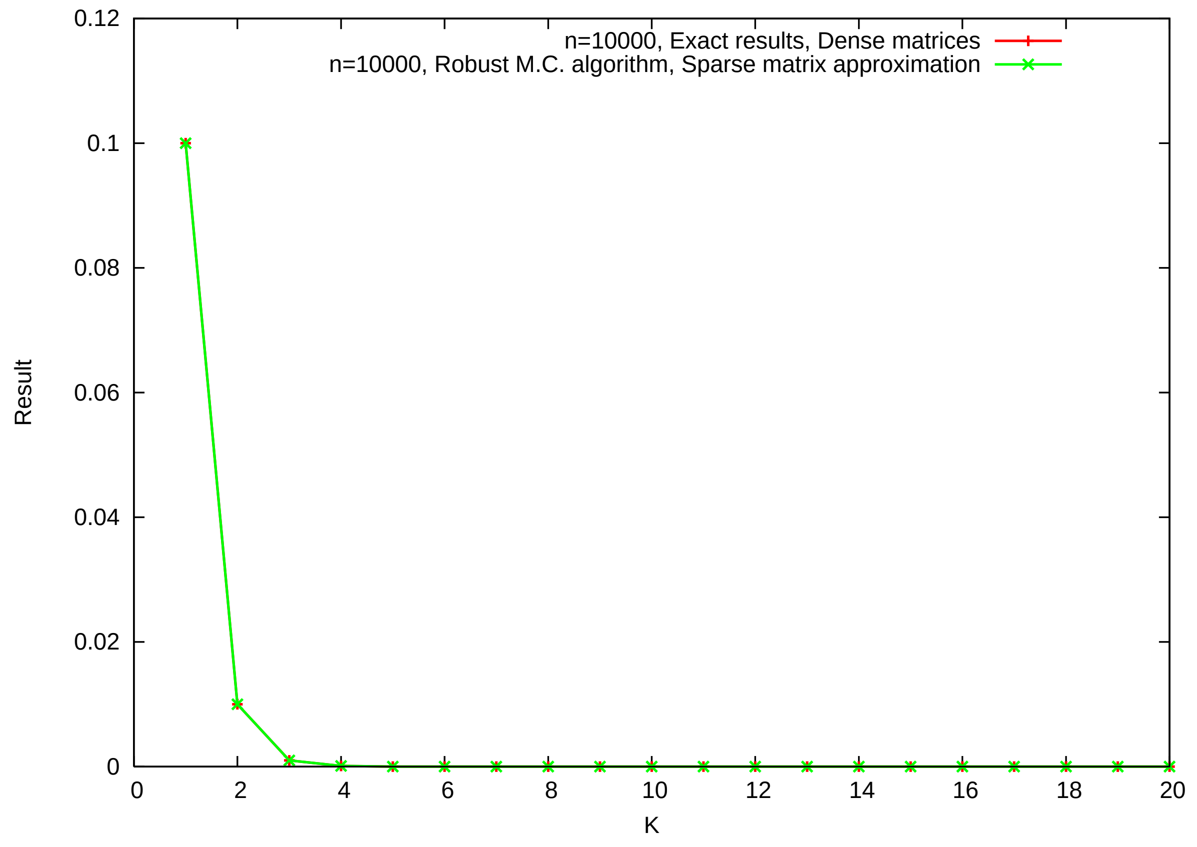

Figure 6: Comparison of the MC results for bilinear form of matrix powers for a sparse matrix of size $n = 10000$ with the exact solution. 5 or 6 Monte Carlo iterations are enough for solving the system of linear algebraic equations or for computing the dominant eigenvalue for the robust Monte Carlo.

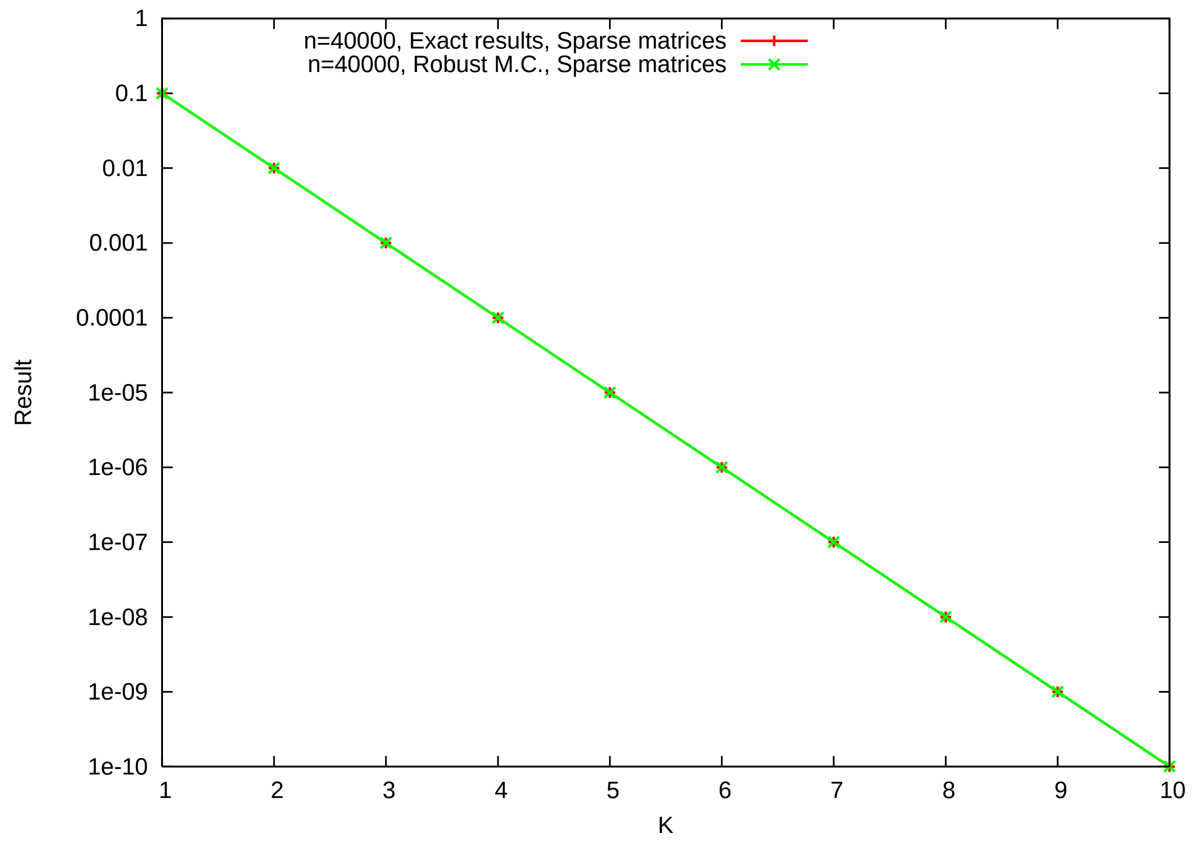

Figure 7: Comparison of the MC results for bilinear form of matrix powers for a sparse matrix of size $n = 40000$ with the exact solution.

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

[1] V. Alexandrov, E. Atanassov, I. Dimov: *Parallel Quasi-Monte Carlo Methods for Linear Algebra Problems*, Monte Carlo Methods and Applications, Vol. 10, No. 3-4 (2004), pp. 213-219.

[2] R. E. Bank and C. C. Douglas: *Sparse matrix multiplication package (SMMP)*, Advances in Computational Mathematics, Vol. 1, Number 1 / February (1993), pp. 127-137.

[3] I. Dimov: *Minimization of the Probable Error for Some Monte Carlo methods*. Proc. Int. Conf. on Mathematical Modeling and Scientific Computation, Albena, Bulgaria, Sofia, Publ. House of the Bulgarian Academy of Sciences, 1991, pp. 159-170.

[4] I. Dimov: *Monte Carlo Algorithms for Linear Problems*, Pliska (Studia Mathematica Bulgarica), Vol. 13 (2000), pp. 57-77.

[5] I. Dimov, V. Alexandrov, S. Branford, and C. Weihrauch: *Error Analysis of a Monte Carlo Algorithm for Computing Bilinear Forms of Matrix Powers*, Computational Science (V.N. Alexandrov et al. Eds.), Lecture Notes in Computing Sciences, Springer-Verlag Berlin Heidelberg, Vol. 3993, (2006), 632-639.

[6] C. Weihrauch, I. Dimov, S. Branford, and V. Alexandrov: *Comparison of the Computational Cost of a Monte Carlo and Deterministic Algorithm for Computing Bilinear Forms of Matrix Powers*, Computational Science (V.N. Alexandrov et al. Eds.), Lecture Notes in Computing Sciences, Springer-Verlag Berlin Heidelberg, Vol. 3993, (2006), 640-647.

[7] I. Dimov, A. Karaivanova: *Parallel computations of eigenvalues based on a Monte Carlo approach*, Journal of Monte Carlo Method and Applications, Vol. 4, Nu. 1, (1998), pp. 33-52.

[8] J.R. Westlake: *A Handbook of Numerical matrix Inversion and Solution of Linear Equations*, John Wiley & Sons, inc., New York, London, Sydney, 1968.

[9] M. Mascagni, A. Karaivanova: *A Parallel Quasi-Monte Carlo Method for Computing Extremal Eigenvalues*, Monte Carlo and Quasi-Monte Carlo Methods (2000), Springer, pp. 369-380.

Questions?

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