A Monte Carlo Approach for Finding More than One Eigenpair*

Michael Mascagni[1] and Aneta Karaivanova[1,2]

[1] Department of Computer Science and School of Computational Science and Information Technology,
Florida State University, Tallahassee, FL 32306, USA,
mascagni@cs.fsu.edu, http://www.cs.fsu.edu/∼mascagni
[2] Bulgarian Academy of Sciences, Central Laboratory for Parallel Processing,
Acad G. Bonchev St., bl. 25A, 1113 Sofia, Bulgaria,
aneta@csit.fsu.edu, http://parallel.bas.bg/∼anet/

Abstract. The Monte Carlo method has been successfully used for computing the extreme (largest and smallest in magnitude) eigenvalues of matrices. In this paper we study computing eigenvectors as well with the Monte Carlo approach. We propose and study a Monte Carlo method based on applying the ergodic theorem and compare the results with those produced by a Monte Carlo version of the power method. We also study the problem of computing more than one eigenpair combining our Monte Carlo method and deflation techniques.

1 Introduction

Many important problems in computational physics and chemistry can be reduced to the computation of dominant eigenvalues of matrices of high or infinite order (for example, quantum mechanical Hamiltonians, Markov matrices and transfer matrices, [11]). The analogy of the time-evolution operator in quantum mechanics, on the one hand, and the transfer matrix and the Markov matrix in statistical mechanics, on the other, allows these two fields to share numerical techniques. Specifically, a transfer matrix, G, of a statistical-mechanical system in d dimensions often can be interpreted as the evolution operator in discrete, imaginary time, t, of a quantum-mechanical analog in $d-1$ dimensions. That is, $G \approx exp(-tH)$, where H is the Hamiltonian of a system in $d-1$ dimensions, the quantum mechanical analog of the statistical-mechanical system. From this point of view, the computation of the partition function in statistical mechanics, and of the ground-state energy in quantum mechanics are essentially the same problems: finding the largest eigenvalue of G and $exp(-tH)$, respectively. Another issue is the computation of the relaxation time of a system with stochastic dynamics. This problem is equivalent to the computation of the second largest eigenvalue of the Markov matrix.

* Supported, in part, by the U.S. Army Research Office under Contract # DAAD19-01-1-0675

I. Dimov et al. (Eds.): NMA 2002, LNCS 2542, pp. 123–131, 2003.

Another important problem is estimating the Fiedler vector, [13], which is the eigenvector corresponding to the second smallest eigenvalue of a matrix. This task is the most computationally intensive component of several applications, such as graph partitioning, graph coloring, envelope reduction, and seriation. Some difficulties in applying Monte Carlo approach to this problem have been reported in [13]

These numerous examples which require the calculation of more than just the dominant eigenpair have motivated us to study the problem of finding one or more eigenvectors of a matrix via Monte Carlo. The problem of using Monte Carlo and quasi-Monte Carlo methods for finding an extremal eigenvalue has been extensively studied, for example, [7,3,4,8,9]. In this paper, we study the problem how to accurately find both the dominant eigenvector and the second largest eigenpair using deflation techniques coupled with Monte Carlo or quasi-Monte Carlo power iterations. Due to space considerations we do not describe here the quasirandom sequences used as this can be found in many articles and books, for example, [2,10].

2 The Problem and the Method of Solution

Consider the eigenvalue problem for determining complex number-vector pairs, (λ, x), for which the following matrix equation

$$Ax = \lambda x,$$

has a non-trivial solution. One (extremal) solution depends upon the convergence, for almost all choices of initial values, x_0, on the sequence

$$x^{(k)} = Ax^{(k-1)}/\lambda_k,$$

where λ_k is chosen for normalization, i. e., so that $||x^{(k)}|| = 1$ in some vector norm, [1]. Then the λ_k converge to the dominant (largest) eigenvalue, λ_1, of A, and $x^{(k)}$ converges to the corresponding eigenvector. We suppose that A is $n \times n$ and that its n eigenvalues are ordered as follows $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_{n-1}| \geq |\lambda_n|$. Choosing the initial vector, f, and a vector h (both of dimension n), the

construction of the desired Monte Carlo Method (MCM) begins with defining a Markov chain $k_0 \to k_1 \to \ldots \to k_i$, of the natural numbers, $k_j \in \{1, 2, \ldots, n\}$ for $j = 1, \ldots, i$. The k_j's defining the Markov chain can be thought of as a random walk on the n dimensions in the vector space we find ourselves. We then define an initial density vector, $p = \{p_\alpha\}_{\alpha=1}^n$, to be permissible to the vector h and a transition density matrix, $P = \{p_{\alpha\beta}\}_{\alpha\beta=1}^n$, to be permissible to A, [3]. We then define the following random variable on the given Markov chain:

$$W_0 = \frac{h_{k_0}}{p_{k_0}}, \;\; W_j = W_{j-1}\frac{a_{k_{j-1}k_j}}{p_{k_{j-1}k_j}}, \;\; j = 1, \ldots, i. \tag{1}$$

The Monte Carlo method for estimating the eigenvector that corresponds to the dominant eigenvalue is based on the following expected value identity[12,3]:

$$(h, A^i f) = E[W_i f_{k_i}], \;\; i = 1, 2, \ldots.$$

By setting $h = e(r) = (0, \ldots, 0, \underbrace{1}_{r}, 0, \ldots, 0)^T$, the rth canonical unit vector, for $r = 1, \ldots, n$ the above random variable has expected value equal to the r-th component of the dominant eigenvector. The Monte Carlo estimate for the dominant eigenvalue is based on the fact that [14,3]:

$$\lambda_{max} \approx \frac{E[W_i f_{k_i}]}{E[W_{i-1} f_{k_{i-1}}]}. \tag{2}$$

2.1 Convergence

The fact that the system of the eigenvectors $\{v^{(1)}, v^{(2)}, \ldots, v^{(n)}\}$ is linearly independent and spans $\mathbb{R}^n$ implies that any vector, $x^{(0)}$, can be presented as a linear combination of them as follows

$$x^{(0)} = \sum_{j=1}^{n} \alpha_j v^{(j)}.$$

If we repeatedly apply A to $x^{(0)}$ we get

$$A^k x^{(0)} = \lambda_1^k \sum_{j=1}^{n} \alpha_j (\frac{\lambda_j}{\lambda_1})^k v^{(j)}.$$

Since $|\lambda_1| > |\lambda_j|$ for all $j = 2, 3, \ldots, n$, we have $\lim_{k\to\infty}(\lambda_j/\lambda_1)^k = 0$, and so

$$\lim A^k x^{(0)} = \lim_{k\to\infty} \lambda_1^k \alpha_1 v^{(1)}. \tag{3}$$

Obviously, this sequence converges to zero if $|\lambda_1| < 1$, and diverges if $|\lambda_1| \geq 1$, provided, of course, that $\alpha_1 \neq 1$. Advantage can be made of the relationship (3) by scaling the powers of $A^k x^{(0)}$ appropriately to ensure that the limit in (3) is finite and nonzero. The convergence of this process, and therefore of the Monte Carlo process as well, is most rapid if $x^{(0)}$ is close to $v^{(1)}$. We may be able to enhance the probability of this occurring by replacing x^k by $x^{(k)}/\sum_1^n x_i^{(k)}$ after performing our sampling. This, in effect, gives new weights to the same set of Markov-chain paths, which we hope will give better estimates for the eigenpair.

The power method has rate of convergence of $O((\frac{\lambda_2}{\lambda_1})^k)$. When we use the power method with Monte Carlo iterations we have additional error from the approximate, probabilistic, computation of matrix powers. This error is stochastic, and thus the uncertainty in this average taken from N samples is $O(N^{-1/2})$, by virtue of the central limit theorem. One generic approach to improving the convergence

of MCMs has been the use of highly uniform, **q**uasi**r**andom, **n**umbers (QRNs) in place of the usual **p**seudo**r**andom **n**umbers (PRNs). While PRNs are constructed to mimic the behavior of truly random numbers, QRNs are constructed to be distributed as evenly as mathematically possible. Quasi-MCMs use quasirandom sequences, which are deterministic, and may have correlations between points, and they were designed primarily for integration. For example, with QRNs, the convergence of numerical integration can sometimes be improved to as much as $O(N^{-1})$!

Since we wish to discuss both MCMs and and quasi-MCMs for these problems, it is timely to recall some pertinent estimates. Using MCMs, [3,4], we have

$$|h^T A^i f - \frac{1}{N}\sum_{s=1}^{N}(\theta)_s| \approx Var(\theta)^{1/2} N^{-1/2},$$

where $Var(\theta) = \{(E[\theta])^2 - E[\theta^2]\}$. In addition, we have

$$E[\theta] = E[\frac{h_{k_0}}{p_{k_0}} W_i f_{k_i}] = \sum_{k_0=1}^{n} \frac{h_{k_0}}{p_{k_0}} p_{k_0} \sum_{k_1=1}^{n} \cdots \sum_{k_i=1}^{n} \frac{a_{k_0 k_1} \ldots a_{k_{i-1} k_i}}{p_{k_0 k_1} \cdots p_{k_{i-1} k_i}} p_{k_0 k_1} \cdots p_{k_{i-1} k_i}.$$

Using quasi-MCMs, [8,9], we obtain:

$$|h_N^T A_N^i f_N - \frac{1}{N}\sum_{s=1}^{N} h(x_1^{(s)}) a(x_1^{(s)}, x_2^{(s)}) \ldots a(x_i^{(s)}, x_{i+1}^{(s)}) f(x_{i+1}^{(s)})| \le |h|^T |A|^i |f| D_N^*, \tag{4}$$

where the $i+1$-dimensional quasirandom sequence $\{(x_1^{(s)}, x_2^{(s)}, \ldots, x_{i+1}^{(s)})\}$, has a star discrepancy of D_N^*.

Let us compare MCM and QMCM errors for computing $(h, A^m f)$: Both are products of two factors (first depends on A, second - on the sequence). The order is $N^{-1/2}$ for MCM and $(log^{m+1} N)N^{-1}$ for QMCM. Moreover, the MCM error is a probabilistic error bound while the QMCM error is the worst-case bound (inequality). In the same time, computational complexity for MCM and QMCM is the same: $O((m+1)N)$, where N is the number of chains, $m+1$ is the length of a single Markov chain.

Then, the convergence rates for power method with MCM or QMCM iterations are:

$$O(\|\frac{\lambda_2}{\lambda_3}\|^m + \sigma N^{-1/2})$$

$$O(\|\frac{\lambda_2}{\lambda_3}\|^m + (log^m N)N^{-1})$$

2.2 The Complex Conjugate Eigenvalue Pair Case

The above approach can be as is used when the dominant eigenvalue is real, a singleton, and separated from the second eigenvalue. However, the method can

also be extended to estimate also a complex conjugate pair of eigenvalues, $\lambda_1 = a + bi$ and $\lambda_2 = a - bi$, and their corresponding eigenvectors. These eigenvectors are approximately equal to

$$bx^{k-1} \pm i(ax^{k-1} - x^k),$$

provided the power iterations, $x^k = Ax^{k-1}$, are real. Unfortunately, the eigenvector estimate is inaccurate when the conjugate pair is close to the real axis, i. e., when b is small.[1]

What is the Monte Carlo procedure in this case? It can be shown ([6]) that λ_1 and λ_2 are the roots of the quadratic equation $\lambda^2 + c_1\lambda + c_2 = 0$. In addition, it can be shown that the coefficients, c_2, c_1, and 1 have the property that for any three consecutive iterations x_k, x_{k+1} and x_{k+2} in the power method, they are the coefficients of an approximate linear relation, i. e. , $x_{k+2} + c_1 x_{k+1} + c_2 x_k \approx 0$.

Let $h \in \mathbb{R}^{1\times n}$ be an n-dimensional vector. One can prove that the above system is equivalent to the following one, [4]:

$$d_1(h, Ax_k) + d_2(h, x_k) = -(h, A^2 x_k).$$

Suppose, we have the values of four MC iterations:

$$(h, x_{k-3});\ (h, Ax_{k-3}) = (h, x_{k-2});$$

$$(h, A^2 x_{k-3}) = (h, x_{k-1});\ (h, A^3 x_{k-3}) = (h, x_k).$$

One can prove that the dominant complex conjugate eigenvalues, λ_1 and λ_2, of the matrix A are the solution of the following quadratic equation: $\lambda^2 + d_1\lambda + d_2 = 0$, where

$$d_1 = \frac{(h, x_{k-1})(h, x_{k-2}) - (h, x_k)(h, x_{k-3})}{(h, x_{k-1})(h, x_{k-3}) - (h, x_{k-2})^2} \tag{5}$$

and

$$d_2 = \frac{(h, x_k)(h, x_{k-2}) - (h, x_{k-1})^2}{(h, x_{k-1})(h, x_{k-3}) - (h, x_{k-2})^2}. \tag{6}$$

These expressions give us a Monte Carlo procedure to estimate the complex conjugate eigenvalues and their corresponding eigenvectors.

2.3 A Fast and Rough Estimate of the Eigenvector

By using the ergodic theorem, [5], we can estimate the first eigenvector very fast. In this case we use a single, very long, Markov chain:

$$EigV = lim_{k\to\infty} A^k x^{(0)}.$$

[1] It is obvious, that as b goes to zero, $\lambda_1 = a + bi$ and $\lambda_2 = a - bi$ both approach a and the two eigenvalues coalesce into a single eigenvalue of multiplicity 2.

This approach can be used only when the matrix is a stochastic matrix with principal eigenvalue equal to 1. Such a matrix describes a Markov chain where all states communicate.[2] Our test matrices are not stochastic, but still a rough estimate for the first eigenvector can be computing using the matrix $\frac{1}{\lambda_{max}}A$. The numerical results are given in the Table 2.

2.4 Numerical Tests

We performed many numerical tests using PRNs, and Sobol', Halton and Faure quasirandom sequences. Some of the results for the L_2-norm of the error: $err = \sqrt{\sum_{i=1}^{n}(x^k(i) - EigV(i))^2}$, where $EigV$ is the exact eigenvector and x^k is the our approximation after k iterations, are presented in the Tables 1 and 2.

Table 1. Accuracy in computing the first eigenvector of sparse matrices of order n using N_w random or quasirandom walks of length k

n	N_w	k	URAND	SOBOĹ	FAURE	HALTON
128	1280	5	0.282e-01	0.642e-02	0.632e-02	0.496e-02
1024	10240	6	0.375e-02	0.958e-03	0.837e-03	0.778e-03
2000	20000	6	0.158e-02	0.153e-02	0.158e-02	0.156e-02

Table 2. Accuracy in computing the first eigenvector of sparse matrices of order n using one long random or quasirandom walk

n	URAND	SOBOĹ	FAURE	HALTON
128	0.52	0.22	1.32	0.41
1024	0.67	0.14	1.22	0.17
2000	0.86	0.003	1.32	0.003

For our numerical tests we use randomly generated sparse matrices of order $n = 128$, 1024 and 2000. With $N_w = 10n$ walks we achieve sufficiently good accuracy in estimating the eigenvector using the power method with MCM or quasi-MCM iterations. For large n, this makes the power MCM and the power quasi-MCMs computationally very efficient. We should recall that the computational complexity of both methods is kN_w, where k is the length of the Markov chains (the power in the power method) and N_w is the number of the walks. It

[2] For example, if the matrix has a block structure, it does not satisfy these communication requirements.

should also be noted that the results using QRNs are better than with PRNs, as we hoped; in order to compare the Monte Carlo and the quasi-Monte Carlo approach, we implemented them using the same number and length of walks for both of the methods, and the quasi-Monte Carlo gives better accuracy.

Let us also note, that when we compute the dominant eigenvalue we need to perform many more walks to achieve an accuracy similar to that in case we compute the dominant eigenvector (see the results given in the Table 3).

The Table 2 shows the results using the ergodic theorem - they confirm the theoretical assumption for giving us a rough approximation.

Table 3. Accuracy in computing the dominant eigenvalue of a sparse matrix of order n using N_w random or quasirandom walks of length k

n	N_w	k	URAND	SOBOĹ	FAURE	HALTON
128	12800	5	0.42e-01	0.36e-02	0.14e-01	0.18e-01
1024	20480	6	0.37e-02	0.23e-02	0.57e-02	0.43e-02
2000	50000	6	0.17	0.15	0.14	0.17

3 Finding More Eigenpairs

Once having the eigenpair, (λ_1, v_1), we deflate the computed eigenvalue from A by constructing a smaller matrix, B. The deflated matrix, B, has one less row and column than A, and the eigenvalues of B are the same as those of A, except that the previously computed λ_1 is missing from B's spectrum.

The deflated matrix is constructed using a Householder matrix, $H = I - 2ww^T$, where w is an n-dimensional vector with components, [6],

$$w(1) = \frac{1}{\sqrt{2s(s + |v_1(1)|)}}(v_1(1) + sign(v_1(1))s),$$

and

$$w(i) = \frac{1}{\sqrt{2s(s + |v_1(1)|)}}v_1(i), \quad i = 2, \ldots, n,$$

where $s = (\sum_{i=1}^{n} v_1(i)^2)^{1/2}$. Then ensures that the matrix HAH has the form

$$HAH = \begin{pmatrix} \lambda_1 & b^T \\ 0 & B \end{pmatrix},$$

with eigenpairs (λ, Hv), where λ is an eigenvalue of A and v is the corresponding eigenvector of A. The characteristic polynomial for HAH is: $\det(HAH - \lambda I) = (\lambda_1 - \lambda)\det(B - \lambda I)$. One zero of the characteristic polynomial for HAH is

$\lambda = \lambda_1$, while the remaining zeros are the roots of the equation $det(B - \lambda I) = 0$, i. e., the eigenvalues of B. Hence, the eigenvalues of B are the same as the eigenvalues of A except that λ_1 is missing. Now we can apply power iterations to estimate λ_2 using B.

Our goal is to study the applicability of MCMs and quasi-MCMs for finding the second eigenpair. We need to understand how the use of the approximate eigenvector (estimated by MC or by QMC iterations) will "spoil" the deflated matrix, and how spectrum of B will be changed. Generally speaking, the answer depends on what type of matrix A is, and on how accurate its first eigenvector is computed. If we perform exactly the above procedure, we will obtain the deflated matrix with the accuracy of the approximate eigenvector. Then we again apply the power method with MC or quasi-MC iterations. It is important to note that this second iteration has a rate of convergence depending additionally on $\frac{\lambda_2}{\lambda_3}$.

3.1 Numerical Tests

In our numerical tests, we use a matrix with size $n = 128$ which produced estimates for the first eigenpair with the worst accuracy among the test matrices. We obtained the deflated matrix using the eigenvector computed using PRNs and the Sobol' sequence. The "approximate" deflated matrix has the same spectral radius as the exactly deflated matrix, and the eigenvalues very close to the exact ones as well. The expected rate of convergence for the power method applied to the deflated matrix for estimating λ_2 is $\frac{\lambda_3}{\lambda_2} = \frac{13}{16}$. The results using PRNs and Sobol' sequence are presented in the Table 4. We have tabulated the relative error for computing λ_2 and the L_2-error in the eigenvector. Again, the quasi-MCM shows slightly better results than the MCM, but the best results are obtained when we combine results using both types (pseudorandom and quasirandom) of sequences.

4 Conclusions and Future Plans

We have shown that Monte Carlo approach can be used for finding more than one eigenpair of a matrix. We tested the method by finding the first- and second-largest eigenvalues and their eigenvectors using PRNs and quasirandom sequences. In all of our numerical experiments, the quasi-MCM gives better results than the MCM. Given these results, we now have to study the use of Monte

Table 4. The error in computing the second eigenpair of a matrix of order 128 using PRNs, Sobol', and both sequences

	Error(λ_2)	Error($v^{(2)}$)
PRNs	0.22	0.3571
Sobol'	0.022	0.3555
PRNs+Sob	0.007	0.3482

Carlo on the analogous problem at the "other end of the spectrum," i. e., for estimating the second smallest eigenpair based on prior knowledge of the smallest eigenpair. This problem can be solved in a similar way; however, using the resolvent Monte Carlo method on a deflated matrix. However, it is well known that resolvent-type algorithms have more restrictions on applicability and have worse rate of convergence than the power-type algorithms studies here, so a careful study should be undertaken.

References

1. Burden, R. and Faires, J. D.: Numerical Anaysis. PWS Publ. Company, Boston (1997)
2. Caflisch, R. E.: Monte Carlo and quasi-Monte Carlo methods. Acta Numerica, 7 (1998) 1–49.
3. Dimov, I. and Karaivanova, A.: Parallel computations of eigenvalues based on a Monte Carlo approach. Journal of Monte Carlo Methods and Applications, Vol. 4,.1 (1998) 33–52.
4. Dimov, I. and Karaivanova, A.: A Power Method with Monte Carlo iterations. In: Iliev, O., Kaschiev, M., Margenov, S., Sendov, Bl., Vassilevski, P. (eds.): Recent Advances in Numerical Methods and Applications II. World Scientific, Singapore (1999) 239–247.
5. Feller, W.: An introduction to probability theory and its applications. Vol. 1, Wiley, New York (1968)
6. Hager, William W.: Applied Numerical Linear Algebra. Prentice Hall, London (1989)
7. Hammersley, J. M. and Handscomb, D. C.: Monte Carlo Methods. Methuen, London (1964)
8. Mascagni, M. and Karaivanova, A.: Matrix Computations Using Quasirandom Sequences. In: L.Vulkov, Waśniewski, J., Yalamov, P. (eds.): Numerical Analysis and its Applications. Lecture Notes in Computer Science, Vol. 1988, Springer-Verlag (2001) 552–559.
9. Mascagni, M. and Karaivanova, A.: A Parallel Quasi-Monte Carlo Method for Computing Extremal Eigenvalues. Monte Carlo and Quasi-Monte Carlo Methods 2000, Lecture Notes in Statistics, Springer-Verlag (2002) 369–380.
10. Niederreiter, H.: Random number generation and quasi-Monte Carlo methods. SIAM, Philadelphia (1992)
11. Nightingale, M. P., Umrigar, C. J.: Monte Carlo Eigenvalue Methods in Quantum Mechanics and Statistical Mechanics. In: Ferguson, D. M., Siepmann, J. Ilja, and Truhlar, D. G. (eds.): Monte Carlo Methods in Chemistry. Advances in Chemical Physics, Vol. 105, Wiley, NY, in press.
12. Soboĺ, I. M.: Monte Carlo numerical methods. Nauka, Moscow (1973)
13. Srinivasan, A., Mascagni, M.: Monte Carlo Techniques for Estimating the Fiedler Vector in Graph Applications. Lecture Notes in Computer Sccience, Vol. 2330, 2, Springer-Verlag (2002) 635–645.
14. Vladimirov, V. S.: On the application of the Monte Carlo method to the finding of the least eigenvalue, and the corresponding eigenfunction of a linear integral equation. in Russian: Teoriya Veroyatnostej i Yeye Primenenie, 1, No 1 (1956) 113–130.