Modified conjugate gradient method for diagonalizing large matrices

Quanlin Jie* and Dunhuan Liu
Department of Physics, Wuhan University, Wuhan 430072, People's Republic of China

(Received 16 May 2003; published 24 November 2003)

We present an iterative method to diagonalize large matrices. The basic idea is the same as the conjugate gradient (CG) method, i.e, minimizing the Rayleigh quotient via its gradient and avoiding reintroducing errors to the directions of previous gradients. Each iteration step is to find lowest eigenvector of the matrix in a subspace spanned by the current trial vector and the corresponding gradient of the Rayleigh quotient, as well as some previous trial vectors. The gradient, together with the previous trial vectors, play a similar role as the conjugate gradient of the original CG algorithm. Our numeric tests indicate that this method converges significantly faster than the original CG method. And the computational cost of one iteration step is about the same as the original CG method. It is suitable for first principle calculations.

DOI: 10.1103/PhysRevE.68.056706 PACS number(s): 02.70.−c, 31.15.Ar, 31.15.−p, 95.75.Pq

I. INTRODUCTION

In first principle calculations, such as band structure calculations, atomic and molecular structure calculations, one of the basic tasks is to find several lowest eigenvalues and the corresponding eigenvectors iteratively (and very often self-consistently) of the effective Hamiltonian [1]. The matrix dimension of the Hamiltonian may range from tens of thousands to several millions, and one may need up to several thousands of lowest eigenvectors of the effective Hamiltonian. Diagonalizing matrices on such a scale needs considerable CPU time and memory. It is one of the major numerical costs in the first principle calculations, and the efficiency of the algorithm is crucial for the performances of the whole program. There are many efforts to improve the algorithm [2–6].

Among widely used algorithms, such as Lanczos [5,7], Dividson [8], relaxation method [4,9], DIIS (Direct Inversion in the Iterative Subspace, which minimizes all matrix elements between trial vectors) [10], and its later version RMM-DIIS [2,11] (RMM stands for residual minimization, i.e., minimizing the norm of the residual vector in iterative subspace), and the conjugate gradient (CG) method [1,2] are valuable tools to find a set of lowest eigenvectors of a large matrix. Briefly speaking, to obtain the lowest eigenvector of a matrix H for the general form eigenvalue problem

$$H|\psi\rangle = ES|\psi\rangle, \tag{1}$$

the CG method iteratively minimizes the Rayleigh quotient

$$E_n = \frac{\langle\phi_n|H|\phi_n\rangle}{\langle\phi_n|S|\phi_n\rangle}, \tag{2}$$

where S is the overlap matrix, and $|\phi_n\rangle$ is a refined trial vector at step n. Each iteration step has to search the minimization point in the direction of the conjugate gradient which is a combination of the current gradient and previous conjugate gradient. One can obtain higher eigenvectors in the same way, as provided to keep the trial vector orthogonal to lower eigenvectors. In practical calculations, the CG method is stable and reasonably efficient in many cases, and it is easy to implement. The iteration procedure needs only to store the trial vector and its gradient, as well as one previous conjugate gradient.

The conjugate gradient method is originally designed to minimize positive definite quadratic functions iteratively. In nth step of iteration, the CG method is equivalent to finding a minimum in an n-dimensional subspace spanned by the initial trial vector and the subsequent $n-1$ gradients of the quadratic function. Due to special properties of a quadratic function, one needs only to do the minimization in a two-dimensional space spanned by current state and the conjugate gradient, which is a combination of current gradient and last step's conjugate gradient. In principle, one needs atmost N steps to obtain final solution in an n-dimensional space. Practical calculations usually need more steps due to round off errors. The conjugate gradient method is virtually the most effective method to minimize a quadratic function iteratively. And it is a formally established algorithm to solve the linear algebraic equation.

For general functions, such as Rayleigh quotient, there are several ways to define the conjugate gradient, and the behaviors of conjugate gradient algorithm are unclear. However, near an exact minimum point, any function behaves like a quadratic function. If one starts with a good guess, one may find the solution very quickly. This partially explains the successes of the conjugate gradient method in diagonalizing a large matrix.

II. THE MODIFIED CONJUGATE ALGORITHM

Our method is based on the following two observations.

First, each iteration step of minimizing the Rayleigh quotient by CG algorithm is equivalent to finding lowest eigenvector in a two-dimensional subspace. The subspace at nth step is spanned by the current state $|\phi_n\rangle$ and the conjugate gradient $|F_n\rangle$. Note that the conjugate gradient $|F_n\rangle$ is a combination of the gradient of nth step's Rayleigh quotient, and the $(n-1)$th step's conjugate gradient $|F_{n-1}\rangle$. One may expect a better result at nth iterative step by finding the low-

*Email address: qljie@whu.edu.cn

est eigenvector in a three-dimensional subspace spanned by $|\phi_n\rangle$, $|G_n\rangle$, and $|F_{n-1}\rangle$, where $|G_n\rangle$ is the gradient of the Rayleigh quotient at nth step.

Second, we note that, within the CG algorithm, $|\phi_n\rangle$ is a combination of $|\phi_{n-1}\rangle$ and $|F_{n-1}\rangle$. Thus the three-dimensional subspace spanned by $|\phi_n\rangle$, $|G_n\rangle$, and $|F_{n-1}\rangle$ is the same as the subspace spanned by $|\phi_n\rangle$, $|G_n\rangle$, and $|\phi_{n-1}\rangle$. This means that one may obtain a better result $|\phi_{n+1}\rangle$ at nth step by replacing the nth step iteration of CG algorithm with finding the lowest eigenvector at the three-dimensional subspace spanned by $|\phi_n\rangle$, $|G_n\rangle$, and $|\phi_{n-1}\rangle$. Of course, the result will further improve if one finds the lowest eigenvector in a larger subspace spanned by $|\phi_n\rangle$,$|G_n\rangle$, $|\phi_{n-1}\rangle, \ldots, |\phi_{n-m+2}\rangle$.

The above observations indicate that one may improve the efficiency of the CG algorithm by replacing each iteration step of the CG algorithm with finding the lowest eigenvector in a small subspace spanned by the current vector $|\phi_n\rangle$ and the corresponding gradient $|G_n\rangle$, as well as some previous vectors $|\phi_{n-1}\rangle$, $|\phi_{n-2}\rangle, \ldots$. In our numeric tests, the effect is significant in many cases. Since diagonalizing a small matrix of several dimensions is numerically very cheap, each step's numeric cost of the modified version is about the same as that of the original CG algorithm.

Practical implementation of the modified conjugate gradient method is similar to that of the original CG method. For finding a single lowest eigenvalue and its corresponding eigenvector, it goes through the following steps.

(1) Choose the dimension M of the iteration subspace, and the maximum iteration step N_{max}. In our numerical test, it is enough to set the dimension $M \leqslant 10$. In many cases, $M=3$ works quite well. In this case, the three-dimensional subspace is spanned by current trial vector $|\phi_n\rangle$, the corresponding gradient $|G_n\rangle$ and one previous trial vector $|\phi_{n-1}\rangle$ obtained in the last step.

(2) Choose an initial normalized trial vector $|\phi_0\rangle$, $\langle\phi_0|S|\phi_0\rangle=1$; and calculate the expectation value (Rayleigh quotient) $E_0=\langle\phi_0|H|\phi_0\rangle$.

(3) For $n=0,1,2,\ldots,N_{max}$, do the following iteration loop to refine the trial vector from $|\phi_0\rangle$ to $|\phi_1\rangle,|\phi_2\rangle,\ldots$.

(a) Calculate the gradient of the Rayleigh quotient

$$|G_n\rangle=H|\phi_n\rangle-E_n|\phi_n\rangle. \tag{3}$$

Here the refined trial function $|\phi_n\rangle$ is normalized at the end of each iteration.

(b) In the m-dimensional subspace spanned by $|\psi_1\rangle=|G_n\rangle$, $|\psi_2\rangle=|\phi_n\rangle$, $|\psi_3\rangle=|\phi_{n-1}\rangle,\ldots,|\psi_m\rangle=|\phi_{n-m+2}\rangle$, calculate the matrix elements of the matrix H, $\mathcal{H}_{ij}=\langle\psi_i|H|\psi_j\rangle$, and the overlap matrix of the basis vector $\mathcal{S}_{ij}=\langle\psi_i|S|\psi_j\rangle$. Here, the dimension is $m=n+1$ if $n+1\leqslant M$, otherwise $m=M$, i.e., in the first $M-1$ loops, the subspace has only $n+1$ basis vector.

(c) Find the lowest eigenvalue ϵ and eigenvector φ for the general form eigenvalue problem

$$\mathcal{H}\varphi=\epsilon\mathcal{S}\varphi. \tag{4}$$

(d) From the above eigenvector φ, construct the refined trial vector $|\phi_{n+1}\rangle$,

$$|\phi_{n+1}\rangle=\sum_{i=1}^{m}\varphi_i|\psi_i\rangle, \tag{5}$$

and calculate the expectation value $E_{n+1}=\langle\phi_{n+1}|H|\phi_{n+1}\rangle$.

(e) If $|E_{n+1}-E_n|$ is less than the required value or $n>N_{max}$, stop the iteration loop, otherwise continue the iteration loop.

Impose a maximum iteration step if necessary in many cases. For example, in self-consistent calculations, one needs to update the Hamiltonian after some steps of iterations. The trial vector can be chosen, in principle, arbitrarily, provided it is not orthogonal with the lowest eigenvector. However, even if the initial trial vector accidently becomes orthogonal to the lowest eigenvector, due to the numeric roundoff errors in the iterations, one can always arrive at the lowest eigenvector.

Check that the convergence is usually testing the difference between the trial vector and its refined version after an iteration. In our numeric tests, check that the difference between two consecutive trial vectors' Rayleigh quotients also works well, and it is also numerically faster.

For large matrices, calculation of the gradient is a main numeric task in each loop of iteration. It involves a multiplication of matrix and vector. Other numeric costs are mainly the calculation of the matrix elements $\mathcal{H}_{ij}$ and $\mathcal{S}_{ij}$ in the small subspace, as well as the combination of the gradient and previous trial vectors to form a refined trial vector. The numeric cost of diagonalizing the small matrix $\mathcal{H}$ is almost nothing as compared to other operations. In each loop of iteration, the subspace changes two basis vectors, i.e., the current gradient $|G_n\rangle$ replaces the previous one $|G_{n-1}\rangle$, and the refined trial vector $|\phi_n\rangle$ replace the old one $|\phi_{n-m+2}\rangle$. One needs only to calculate the matrices elements $\mathcal{H}_{ij}$ and $\mathcal{S}_{ij}$ related to the two vectors in each iteration loop. If the subspace is three dimensional, the numerical cost of one iteration loop is about the same as that of the original CG method.

After finding the lowest eigenvector, one can find the second lowest eigenvector in a similar way. One starts with a trial vector orthogonal to the lowest eigenvector, and in the following iterations, gradients of the Rayleigh quotient, as well as the updated trial vectors, must be kept orthogonal to the lowest eigenvector. Similarly, after working out k lowest eigenvectors, the $k+1$ eigenvector can be worked out by maintaining the orthogonality with k lower eigenvectors.

In this strict sequential procedure, the accuracy of lower eigenvectors affect the higher ones. A remedy to this problem, according to Ref. [2] is rediagonalizing the matrix in the subspace spanned by the refined trial vectors, which is referred as subspace rotation in Ref. [2]. After this subspace rotation, one can use these resultant vectors as trial vectors for further iteration to improve the accuracy. In practical implementations, we only iterate every trial vector for some steps, then perform a subspace rotation. The convergence check is to test the eigenvalues differences between two consecutive subspace rotations. This procedure improves the

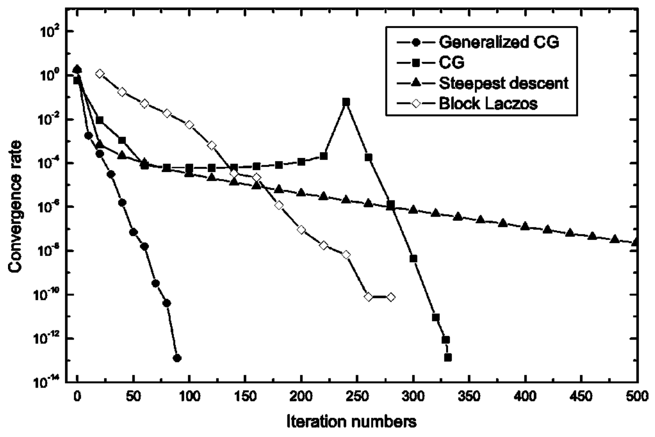

FIG. 1. Convergence rate of the modified conjugate gradient method in comparing with other algorithms.

overall efficiency. In Ref. [2], there is a detailed discussion on the role of the subspace rotation.

III. NUMERICAL RESULTS

We test the efficiency of the above outlined algorithm by comparing its performance with other algorithms for various matrices. In all cases, the modified CG algorithm outperforms the original CG algorithm. We observe significant improvement to the convergence rate in many cases.

As an illustration, we show in Fig. 1, a typical result for a banded matrix with bandwidth $2L$. The matrix's diagonal element is $a_{ii}=2\sqrt{i}-a$, and its off-diagonal elements within the bandwidth is a constant, $a_{ij}=a$. Due to its simple form and its relation with Hamiltonian describing the pairing effects, this matrix has been investigated by some other authors, see, e.g., Ref. [4]. Here we choose the matrix's dimension $N=200\,000$ with half bandwidth $L=300$, the parameter a is set to be 20. For finding first eight lowest eigenvectors, the modified CG algorithm converges within 100 steps with an accuracy of machine's precision limit. It is more than three times faster than the original CG algorithm. As a comparison, we also show the result for the block Lanczos method [7], as well as the steepest decent method. In Fig. 1, the convergence rate of one iteration step is defined as the relative error of the two consecutive Rayleigh quotients, $(E_n-E_{n-1})/[(E_n+E_{n-1})/2]$, where E_{n-1} and E_n are two consecutive Rayleigh quotients. When every eigenvalue reaches the required accuracy, we perform a subspace rotation and repeat the iteration. Convergence is to test the corresponding relative error for every eigenvalue between two consecutive rotations. In our implementation, the maximum iteration number $N_{max}=500$, i.e., we go at most 500 steps of iteration for each trial vector before performing a subspace rotation.

In the above calculations, we use a three-dimensional iteration subspace for the modified CG algorithm, i.e., the subspace is constituted of the current trial vector, its corresponding gradient, as well as one previous trial vector. In such case, each iteration step needs to calculate one gradient, and some combinations of the three vectors, as well as solving a three-dimensional eigenvalue problem. From the above argument, when the iteration subspace is three-dimensional, the numeric cost of each iteration step is almost the same as that of the original CG method. For the block Lanczos algorithm, however, to ensure a reasonable convergence rate, the iteration subspace is 50 dimensional, i.e., one needs to calculate 50 gradients for each iteration step. As per our experience, one step of Lanczos iteration needs longer CPU time than 50 steps of the modified CG method. Thus, one Lanczos step is counted as 50 steps in Fig. 1.

In the three-dimensional iteration subspace spanned by $\{|G_n\rangle, |\phi_n\rangle$, and $|\phi_{n-1}\rangle\}$, the gradient vector $|G_n\rangle$, together with the previous trial vector $|\phi_{n-1}\rangle$, plays the same role as that of the conjugate gradient in the minimization of a quadratic function. This is especially the case when the Rayleigh quotient closes to the minimum point, i.e., it is approximately a quadratic function of the iteration trial vector. In fact, without the previous trial vector $|\phi_{n-1}\rangle$, the lowest eigenvector obtained in the two-dimensional subspace spanned by $\{|G_n\rangle,|\phi_n\rangle\}$ is just a result of steepest descent method. By including one previous trial vector which contains information about previous gradients, one is able to prevent reintroduction of errors to the refined trial vector in the direction of previous gradients. This is the reason we call this method as modified CG algorithm.

On the other hand, in the context of relaxation algorithm for finding lowest eigenvector [4,9], the refined trial vector $|\phi_n\rangle$ at step *n*, is an approximation to the lowest eigenvector of the matrix in the subspace spanned by $\{|\phi_0\rangle,|G_1\rangle,|G_2\rangle, \dots ,|G_n\rangle\}$, which is equivalent to the subspace spanned by $\{|\phi_0\rangle,|\phi_1\rangle,|\phi_2\rangle, \dots ,|\phi_{n-1}\rangle,|\phi_n\rangle\}$. According to the relaxation algorithm, to find the lowest eigenvector in the subspace spanned by the above basis vectors, one starts from an initial trial vector $|\psi_0\rangle$, and minimizes the Rayleigh quotient iteratively. Each iterative step is to minimize the Rayleigh quotient in a two-dimensional subspace spanned by the (updated) trial vector, and one basis vector. The basis vector can be chosen consecutively from the first one to the last one. After going through all basis vectors, one continues the next round of iteration by choosing the first basis vector as next basis vector. This iteration will converge after going through all basis vectors in several rounds. Note that, if one starts with the first basis vector $|\phi_0\rangle$ as initial trial vector, in the two-dimensional subspace spanned by two consecutive basis vector $|\phi_i\rangle$ and $|\phi_{i+1}\rangle$, the second basis vector $|\phi_{i+1}\rangle$ minimizes the Rayleigh quotient. After going through all basis vector for one round, the refined trial vector is $|\phi_n\rangle$, which represents an approximate lowest eigenvector in the above subspace.

The above two factors explain the rapid convergence of the modified CG algorithm. One consequence of the above arguments is that, if we increase the dimension of the iteration subspace by including more previous trial vectors, the convergence rate will not increase too much. In other words, one needs only to do the modified CG algorithm in a small iteration space. From our experience, one needs at most five-dimensional iteration subspace. In most cases, it is enough to do the iteration in the three-dimensional iteration subspace. Figure 2 shows our numeric result to confirm this property of

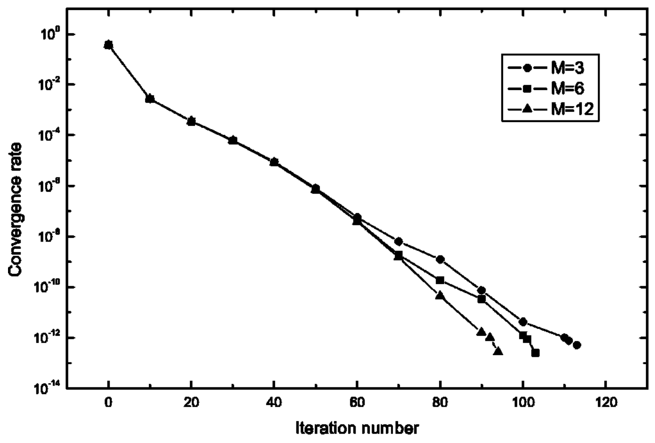

FIG. 2. Convergence rates of the modified CG algorithm for different dimensions of the iteration subspace.

the modified CG algorithm. Here we do the same calculation using different iteration subspaces. The filled circle connected line is the same as Fig. 1 with three-dimensional iteration subspace, and the filled square and triangle are results for 6 dimensional and 12 dimensional iteration subspaces, respectively. There is almost no difference within 50 steps where the convergence rate is about 10^{-8}. One needs almost the same iteration steps to arrive at the final precision. However, the three-dimensional iteration runs faster for each iteration step since it involves less combination and production of the basis vectors that span the iteration subspace.

For some matrices or some properly chosen initial trial vectors, the Rayleigh quotients are approximately quadratic functions of the trial vectors. In such cases, the modified CG algorithm converges by almost the same rate as the original CG algorithm. And a trial vector ϕ_n at step n, is an almost exact minimum in the subspace spanned by $\{|\phi_0\rangle,|\phi_1\rangle,\ldots,|\phi_n\rangle,|G_n\rangle\}$. We have encountered such cases in our numeric tests. In fact, near a minimum, any function behaves like a quadratic function. Some matrices with special structures also make the Rayleigh quotient like a quadratic function in a quite large region of the vector space. For such matrices, the CG method is indeed a very efficient method. Of course, in any case the modified CG method always outperforms the original CG method.

The refined trial vector $|\phi_n\rangle$ becomes closer and closer to the previous step's trial vector $|\phi_{n-1}\rangle$ when iteration closes to final solution. In higher dimensional iteration, one may encounter (numerical) degeneracy of basis vectors that span the iteration subspace. This problem is easy to solve. One simple solution is to replace this step by the steepest descent's step. Other more sophisticated way is to choose some independent vectors from the basis vectors and do this step in a small subspace. Both methods are easy to implement. In fact, one can detect the degeneracy when solving the general form eigenvalue problem (4), which can be conveniently solved by the conventional Choleski-Householder procedure [12]. If there is a degeneracy, the Choleski decomposition of the overlap matrix $\mathcal{S}$ returns an error code. When this happens, one can simply redo this step with the steepest descent's step. Alternatively, one can use a more sophisticated Choleski decomposition program that automatically chooses independent basis vector. In doing so, one must adjust the matrix element of $\mathcal{H}$ simultaneously. These two methods need almost the same numerical cost. Of course, the first method is easy to implement. In our numerical tests, there is almost no degeneracy in the three-dimensional iteration subspace.

It is straightforward to implement preconditioning treatment for the modified CG algorithm. Preconditioning treatment can significantly improve the convergence rate for matrices with a large difference between lowest and highest eigenvalues. Due to the fact that there is no need to construct explicitly the conjugate gradient in the modified CG algorithm, it is easier to implement the preconditioning treatment by directly modifying each step's gradient. Since preconditioning treatment depends on specific system, we don't go into more details about such topic.

The modified CG algorithm shares a common feature with many other iterative methods of diagonalizing matrices, such as Lanczos, Dividson, RMM-DIIS, and relaxation method. In all these algorithms, one refines the trial vector in iterative subspaces. What makes the modified CG algorithm different from other algorithms is that the iteration subspaces are spanned by the trial vectors of previous iteration steps, as well as the latest trial vector and its gradient. The trial vectors of previous steps are already prepared, one needs to calculate one gradient vector (and possibly does some preconditioning treatment) to construct the basis vectors of the iterative subspace. Only two basis vectors of the iterative subspace are different from previous one, it needs only update two columns of the matrix elements in the iteration subspace. By including previous trial vectors into the iterative subspace, one avoids reintroducing errors to the trial vectors in the previous directions of gradients. These properties of the iterative subspace make the modified CG algorithm numeric efficient. And the common feature of the algorithm makes it easy to implement.

It is easy to formulate block algorithm for the modified CG algorithm to find several lowest eigenvectors simultaneously. For this end, one refines several trial vectors at each iteration step. Here the iteration subspace includes all current trial vectors, their gradients, and all trial vectors of some previous steps. In this implementation, one needs to find several eigenvectors by solving the general form eigenvalue problem (4). Trial vectors obtained in this way are automatically orthogonal with each other, and one needs no additional subspace rotation.

However, one step of block algorithm usually needs more floating point operations than sequentially processing each trial vector and maintaining orthogonality between trial vectors by Schmidt orthogonalization method. This is mainly because the block algorithm needs more flips to form the matrix elements of $\mathcal{H}$ and the corresponding overlap matrix $\mathcal{S}$. If one needs n_0 lowest eigensolutions for n-dimensional matrix, the block algorithm's iterative subspace is $M=mn_0$ dimensional with $m=3,4,\ldots$. Each step of block algorithm needs the following floating point operations: (a) n_0NL flips for n_0 matrix multiplying vector operations to obtain n_0 gradients, where $L\leqslant N$ is the band width of the matrix; (b)

$2(mn_0)^2N$ flips for the formation of the matrix elements of $\mathcal{H}$ in the iterative subspace and the corresponding overlap matrix $\mathcal{S}$; (c) an $O[(mn_0)^3]$ floating point operation for solving the general form eigenvalue problem (4); (d) $2mn_0^2N$ flips for combination of the mn_0 basis vectors to form n_0 refined trial vectors. Here, the flips in step (c) is negligible when $n_0 \ll N$. The total flips of one step block algorithm is $\sigma(m,n_0,N) = n_0NL + 2(mn_0)^2N + 2mn_0^2N$. If $n_0 = 1$, the above floating point operations $\sigma(m,1,N) = NL + 2m^2N + 2mN$ is the flips for processing one trial vector in sequential algorithm. On the other hand, sequentially processing each trial vector one round needs $n_0\sigma(m,1,N) + 4n_0^2N$ flips. Here the second term is the flips to maintain the orthogonality of trial vectors, including making gradients orthogonal to previous trial vectors. Even including subspace rotation which is performed after some rounds of sequential steps, the sequential implementation needs less floating point operations than the block algorithm.

If n_0 is small, e.g., $n_0 < 10$, the difference of flips between block and sequential algorithm is small. The block algorithm may be one choice in such cases. Like the block Lanczoz [7], and block Dividson [8], there are some other ways to form the iterative subspace to implement the block version of modified CG algorithm. For example, the iterative subspace may contain only one gradient, plus all the current trial vectors and some previous trial vectors. The choice of iterative subspace affects the convergence properties which needs further investigations. For large n_0, e.g., $n_0 > 100$, to our experiences, block algorithm need more numeric cost and is less efficient as compared with the above sequential implementation. The dimension of the iteration subspace grows quickly with the number of needed eigenvectors, and one needs more memory to store the basis vectors and much more CPU time to solve the general from eigenvalue problem (4) which increases drastically with the dimension of the iterative subspace. Since the lowest eigenvector usually converges faster than the higher ones, the number of iteration steps in a block algorithm is determined by the vector with the slowest convergence rate.

IV. CONCLUSIONS

In summary, in the sense of conjugate gradient algorithm, we formulate an iterative method to find a set of lowest eigenvalues and eigenvectors of a matrix. This method minimizes the Rayleigh quotient of a trial vector via the gradient of the Rayleigh quotient, and at the same time, prevents reintroducing errors in the direction of previous gradients. We realize such idea by refining the trial vectors in a special kind of iteration subspace. Each iteration subspace is spanned by the latest trial vector and the gradient of its Rayleigh quotient, as well as some trial vectors of previous steps. Each iteration step is to find lowest eigenvector in the iteration subspace. The gradient, together with the previous trial vector, plays the role of the conventional conjugate gradient. In our numerical test, it is usually enough to include only one previous trial vector, i.e., one needs only refining the trial vector in a three-dimensional subspace. As compared to the conventional conjugate gradient algorithm, which is designed to minimize a general function, the current method exploits special properties of eigenvalue problems, and thus converges much faster in many cases. During iterations, the trial vector at the step *n*, is an approximate lowest eigenvector in the subspace spanned by the initial trial vector and *n* subsequent gradient vectors. This is the reason for the rapid convergence rate. The easy implementation of this algorithm makes it suitable for first principle calculations.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation, the Research Fund of the State Education Ministry of China, and the Research Fund of Wuhan University. We thank Professor W. Wang for helpful discussions.

[1] M.C. Payne, M.P. Teter, D.C. Allan, T.A. Arias, and J.D. Joannopoulos, Rev. Mod. Phys. **64**, 1045 (1992); M.P. Teter, M.C. Payne, and D.C. Allan, Phys. Rev. B **40**, 12 255 (1989).

[2] G. Kresse and J. Furthmüller, Phys. Rev. B **54**, 11 169 (1996).

[3] D.M. Bylander, L. Kleinman, and S. Lee, Phys. Rev. B **42**, 1394 (1990).

[4] F. Andreozzi, A. Porrino, and N.L. Iudice, J. Phys. A **35**, L61 (2002).

[5] H.Q. Lin and J.E. Gubernatis, Comput. Phys. **7**, 400 (1993).

[6] N. Wijesekera, G. Feng, and T.L. Beck, e-print cond-mat/0304374.

[7] J.K. Cullum and R.A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations* (Birkhäuser, Boston, 1985).

[8] E.R. Davidson, J. Comput. Phys. **17**, 87 (1975); B. Liu, in *Proceedings of the Workshop Numerical Algorithms in Chemistry: Algebraic Methods*, edited by C. Moler and I. Shavitt (University of California, Berkeley, 1978), p. 49.

[9] I. Shavitt, C.F. Bender, A. Pipano, and R.P. Hosteny, J. Comput. Phys. **11**, 90 (1973).

[10] P. Pulay, Chem. Phys. Lett. **73**, 393 (1980).

[11] D.M. Wood and Z. Zunger, J. Phys. A **18**, 1343 (1985).

[12] G.H. Golub and C.F. Van Loan, *Matrix Computation* (The Johns Hopkins University Press, Baltimore, 1996).