Matrix spectroscopy: Computation of interior eigenstates of large matrices using layered iteration

Robert E. Wyatt

Department of Chemistry and Biochemistry, University of Texas at Austin, Austin, Texas 78712

The computation of a small group of interior eigenstates of a large matrix H is an important problem in chemical physics. This general problem is approached in this study through the application of a two-layer iteration scheme. The design of the outer loop is based upon the strategy developed by Ericsson and Ruhe [Math. Comput. **35**, 1251 (1980)]. Starting with a variable input energy E, this loop uses the Lanczos algorithm (with no reorthogonalization) to develop a reduced tridiagonal representation of the total Green function $G(E)$. Diagonalization of this small matrix (denoted T) yields, through a simple mapping, excellent approximations to the eigenvalues of H that lie near energy E. The inner iteration loop of the algorithm computes the matrix-vector product $G(E)Q_j$, where Q_j is one of the Lanczos vectors. Both direct and indirect methods may be used for this operation. The direct method uses a perturbative expansion of the Green function, while the indirect method is based upon the iterative solution of a linear algebraic system. The indirect method yields a parametrized polynomial representation of the Green function, with the coefficients chosen variationally. The method is adaptive in the sense that these coefficients change during the course of the Lanczos iteration. Convergence of these iterative methods is improved significantly by developing the Green function in the partial adiabatic representation. Conversion to this representation involves diagonalization of the diagonal block of H involving those states whose zeroth-order energies surround the energy of interest. The two-layer scheme is applied to a model Hamiltonian matrix that has a variable coupling strength within and between a number of dense bands of states. With this scheme it is possible to accurately compute a group of eigenstates near energy E in just a few Lanczos iteration steps (< 10).

PACS number(s): 02.70.−c, 02.60.Dc

I. INTRODUCTION

The study of molecular excited states is a central theme in chemical physics. Computational approaches are normally based upon first choosing a representation of dimension N (such as the finite basis representation, the discrete variable representation, the pseudospectral method, etc.) and then constructing the matrix representation of the system Hamiltonian H. In most cases, this matrix representation of the Hamiltonian, denoted H (assumed to be real valued and symmetric), is very large, typically of dimension $> 10^4$. Since direct diagonalization is out of the question, there are then a variety of methods that can possibly be used for extracting eigenstate information from H; some of these will be reviewed later in this section. However, in most cases, we are not interested in obtaining all N eigenstates, there are just too many of them and we do not need all of this information. Rather we want a small set of eigenstates near a test input energy E. Just as the experimental spectroscopist would do in the laboratory, we want to move around to different regions of the spectrum and examine closely the eigenstates near E_1, then move to another region near E_2, etc. This process of computing eigenstates in selected regions of the spectrum of a (large) Hamiltonian matrix we term *matrix spectroscopy*.

Over the past 10–15 years, a number of methods have been proposed for computing excited states. Some of the most important of these that have been used by the quantum dynamics community to calculate molecular excited states are categorized in Table I. Later, in Sec. IV, we will comment on several other methods that are not listed in this table. In order to set the stage for further developments, we will comment briefly on each of these methods.

The *relaxation method* of Kosloff and Tal-Ezer [1] is based upon the propagation of a wave packet in imaginary time ($t \rightarrow \tau = -it$). This causes excited state components to relax at rates proportional to their eigenvalues, thus leaving a multiple of the ground state wave function. If the ground state is then removed by using the projected Hamiltonian $H_1 = (1-P_0)H(1-P_0)$, then another propagation sequence will result in a wave packet that collapses down to the first excited state wave function. The whole process can then be repeated to give the lowest few eigenstates. This method is not designed to give eigenstates in the dense interior region of the spectrum.

The *spectral method* involves the real time propagation of a wave packet [2]. This information is used to compute a time correlation function, such as $\langle \psi(0)|\psi(t)\rangle$. The Fourier transform of this quantity (after weighting it with a window function) from the time to the energy domain gives an energy-dependent amplitude, the absolute square of which peaks at the eigenvalues. The spectral method may also be referred to as employing a *direct Fourier filter*. Skodje *et al.* have recently used this method to compute resonance energies in the $H + H_2$ reaction [3]. A severe disadvantage of this method is that very long propagation times are necessary to resolve small energy differences. (Although quite different in im-

TABLE I. Comparison of eigenstate algorithms.

Method	Propagation of wave packet	Fourier transform ($t \rightarrow E$)	Develop Krylov subspace	Matrix diagonalization in subspace
imaginary time propagation (relaxation)[a]	yes (imaginary time)	no	no	no
spectral[b]	yes	yes	no	no
filter diagonalization[c]	yes	yes	yes	yes
Davidson and generalizations[d]	no	no	no	yes
spectral filter Lanczos[e]	no	no	yes	yes

[a]Kosloff and Tal-Ezer, Ref. [1].
[b]Feit, Fleck, and Steiger, Ref. [2].
[c]Neuhauser, Ref. [5].
[d]Davidson, Ref. [6]; Morgan and Scott, Refs. [7–9].
[e]Ericsson and Ruhe, Ref. [11]; Kono, Ref. [13]; Webster, Rossky, and Friesner Ref. [12]; present work.

plementation, a discrete fast Fourier transform algorithm for computing eigenstates was also developed by Yau and Lu [4].)

An improvement on the spectral method is provided by the *filter-diagonalization method* developed by Neuhauser [5]. This method uses short-time wave packet propagation with Fourier energy filtering at a number of energies, say N, of the product of the evolving wave packet times a window function. The N functions so generated (after Gram-Schmidt orthogonalization) are then used to form the $N \times N$ matrix representation of the system Hamiltonian. Diagonalization then yields approximations (which can be very good) to some of the eigenvalues. This method has recently been used [5] to compute the energies of narrow resonances in H_3 and to compute vibration-rotation energies in LiCN($J = 0$).

These first three methods all involve some sort of time propagation, either real time or complex time. However, one might naively wonder about the answer to the following question: If the goal is to obtain eigenstate information, which is inherently time independent, then why launch wave packets and watch them move around? In fact, the answer is that one does not have to do this. This brings us to the next two methods, which are time independent, and then to the central theme of this study.

Continuing with Table I, the *Davidson method* [6] and the *generalized Davidson (GD) method* [7–9] have been used to compute interior eigenvalues of large matrices. At iteration step k, the GD method constructs the vector $v_{k+1} = (M - \vartheta I)^{-1}(H - \vartheta I)z_k$, where $\{\vartheta, z_k\}$ is an approximate eigenpair. In the original Davidson algorithm $M = D$, the diagonal of H, but in the GD algorithm, M is a larger but easily invertible portion of H. After orthogonalization of v_{k+1} to the previous vectors, the $(k+1)$-dimensional Hamiltonian matrix is constructed and diagonalized. The iterative process then continues.

Closer to the theme of this study are methods that employ the Lanczos algorithm [10] (LA). Along with other

matrix iteration methods, the LA is used to repeatedly improve an approximate solution until it reaches sufficient accuracy. Some well known advantages of matrix iterative methods are that H is used only as an operator in computing matrix-vector products, the matrix-vector product is straightforward to parallelize, and sparse storage schemes can be used in appropriate cases. The conventional LA begins with the Hamiltonian H (although the full matrix might not be constructed in advance) and a starting vector v_0 and builds, step by step, a relatively small $M \times M$ tridiagonal matrix T. Diagonalization of T then yields approximations to some of the eigenvalues of H. These approximations are excellent for eigenvalues on the extreme edges of the spectrum and are frequently very good for interior eigenvalues in the sparse region where there are relatively large gaps between the adjacent eigenvalues. However, in the dense interior region where the gaps may be very small, the conventional LA converges so slowly that it ceases to be a viable approach.

This somewhat pessimistic view of the LA as a procedure for tackling the interior eigenproblem changes completely when the iteration is driven not by H itself but with a *spectral filter*, a specially designed function of H, denoted $f(H)$. This function generates a transformed spectrum, which we will try to compute, but what do we desire of this new spectrum? We want the desired eigenvalues to be well separated from the others (we want to improve the gap distribution) and it is advantageous if these eigenvalues are at the extreme edges of the spectrum. Also, we should be able to invert the computed spectrum of the filter to obtain uniquely and exactly the desired eigenvalues of H. Before proceeding, we note that in some respects, $f(H)$ is related to time-independent iterative methods in the way that the Fourier transform is to the time-dependent wave packet methods described earlier.

Several filters have been used with the LA in the past,

the earliest of which was suggested by Ericsson and Ruhe [11]. Their method, appropriately termed *shift and invert*, uses the filter $f(\mathbf{H}) = (E\mathbf{1}_N - \mathbf{H})^{-1}$. (The $N \times N$ unit matrix is denoted $\mathbf{1}_N$.) This has the highly desirable effect of throwing the eigenvalues of $\mathbf{H}$ that lie near E to the extreme edges of the spectrum. The LA shows its appreciation of this shift and invert strategy by converging these eigenvalues very quickly. We will return to this filter after mentioning two others. An *exponential filter* $f(\mathbf{H}) = \exp[-\beta \mathbf{H}]$ was used by Webster, Rossky, and Friesner to extract lower eigenvalues of $\mathbf{H}$ in studies of the spectroscopy of the $e^- + H_2O$ system [12]. Recently, Kono suggested use of the *Gaussian derivative filter* $f(\mathbf{H}) = (\mathbf{H} - E\mathbf{1})\exp[-\beta(\mathbf{H} - E\mathbf{1})^2]$, where E is chosen in the middle of the interesting part of the spectrum and where β controls the width of the filter [13]. The Kono filter has recently been applied in studies of the overtone spectroscopy [14] of CD_3H and CF_3H. It is also possible to design special purpose filters to emphasize any portion of the spectrum, for example, by taking an appropriate combination of Chebyshev polynomials. In fact, Lanczos described how a single Chebyshev polynomial can be used to magnify the separation between two close eigenvalues at one edge of the spectrum [15(a)]. About the same time, Flanders and Shortley also used a Chebyshev filter to accelerate eigenvalue calculations [15(b)]. Several recent texts have excellent sections on the use of these polynomial accelerators [16,17].

In many respects, an excellent filter to use in the LA is the Green function $f(\mathbf{H}) = (E\mathbf{1} - \mathbf{H})^{-1}$. The gap separations near energy E that result from the use of this filter are more extreme and therefore better than those produced by the Kono filter. The result of using the Green function filter is that the eigenvalues of $\mathbf{H}$ that are far from the input value E are mapped to a cluster near the value 0, while the desired eigenvalues near the test energy E are mapped to very large positive or negative values. One excellent feature of the conventional LA is that convergence is improved for well separated eigenvalues and for eigenvalues at the extreme edges of the spectrum. The hyperbolic map achieves these very desirable features and, as demonstrated later in this study, works extremely well for eigenstates in the dense interior region of the spectrum.

In the present study, we will employ the Green function filter in the LA, with one major change from the work of Ericcson and Ruhe [11]. Instead of *factoring* the shifted operator, as suggested by Ericcson and Ruhe and subsequently implemented this way in almost all applications [18–24] (more on this will come later in Sec. II), we will emphasize *iterative methods* for applying $f(\mathbf{H})$ to a vector. Factorization usually requires that the matrix and the factor, denoted $\mathbf{L}$, fit in the central memory, but we want an algorithm that will work effectively on platforms that have relatively modest central memory (i.e., typical workstations) as well as supercomputers. For this reason, we will explore methods that do not utilize matrix factorization.

The overall algorithm that is described in Sec. II involves two iteration loops. A schematic overview of these two loops is shown in Fig. 1. The *outer Lanczos*

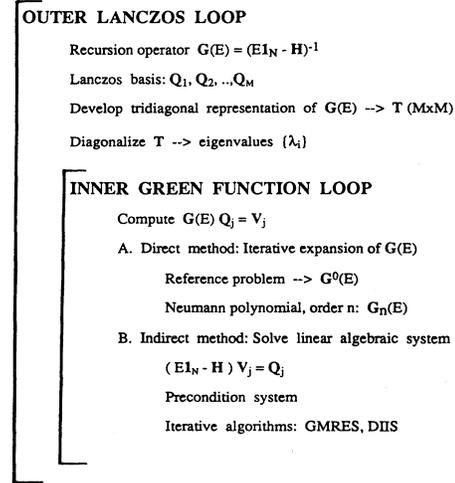## TWO-LAYER LANCZOS-GREEN FUNCTION ITERATION ALGORITHM

FIG. 1. Schematic diagram of the two loops involved in the iteration scheme for generating internal eigenstates.

loop develops a small subspace in terms of which the filter $f(\mathbf{H})$ is tridiagonal. Direct diagonalization of the small tridiagonal matrix yields excellent approximations for some of the eigenvalues of $f(\mathbf{H})$. Inversion of these eigenvalues then gives the desired eigenvalues of $\mathbf{H}$. The *inner Green function loop* applies the Green function to a Lanczos vector. There are two distinct ways to do this. In the *direct method*, we expand the $f(\mathbf{H})$ operator and then apply the terms in the series expansion one by one to the "old" vector. In the *indirect method*, we iteratively solve the linear algebra problem $f(\mathbf{H})^{-1}\mathbf{V}_{new} = \mathbf{V}_{old}$. There are several effective ways to do this. As expected, the direct and indirect methods are closely related, but they do *not* have the same convergence characteristics. In this study, only nondegenerate eigenvalues will be considered; the block Lanczos algorithm works very well when there are degeneracies [24].

II. COMPUTATIONAL METHOD

A. Outer Lanczos loop

The outer recursion loop is driven with the Green function (or resolvent matrix) $\mathbf{G}(E)$ replacing the traditional Hamiltonian matrix in the conventional Lanczos algorithm. Beginning with the normalized $N \times 1$ starting vector $\mathbf{Q}_1$, the vectors $\mathbf{Q}_2, \mathbf{Q}_3, \ldots$ are generated through application of the Lanczos three-term recurrence equations (the superscript t denotes the matrix transpose)

$$\mathbf{R}_j = \mathbf{G}(E)\mathbf{Q}_j - a_j\mathbf{Q}_j - b_{j-1}\mathbf{Q}_{j-1}, \quad \mathbf{Q}_0 = 0,$$

$$b_j = (\mathbf{R}_j^t\mathbf{R}_j)^{1/2}, \quad \mathbf{Q}_{j+1} = \mathbf{R}_j/b_j, \quad a_j = \mathbf{Q}_j^t\mathbf{G}(E)\mathbf{Q}_j. \tag{1}$$

Each recursion step develops a new Lanczos vector $\mathbf{Q}_{j+1}$ along with the quantities a_j and b_j. These equations develop a tridiagonal representation for the Green function;

after M recursion steps, we have

$$Q^t G(E) Q = T , \qquad (2)$$

where the $N \times M$ matrix Q is built by stacking the M Lanczos vectors side by side. In addition, the $M \times M$ matrix T has diagonal elements $\{a_1, a_2, \ldots, a_M\}$ and off-diagonal elements $\{b_1, b_2, \ldots, b_{M-1}\}$. [Equation (2) has formal significance only; the matrix Q is not actually constructed. What we are after is T and this is constructed from the quantities a_i and b_i that are formed during each step of the recursion. However, if the eigenvectors are desired, then the columns of Q must be stored, possibly on disk.] The small tridiagonal matrix T is then diagonalized (the QR algorithm is efficient at doing this) $S^t T S = \lambda$ to produce M eigenvalues of the Green function, which are denoted λ_i. The eigenvalues of H, the ones that we are trying to find, are then given by $E_i = E - 1/\lambda_i$. In Sec. III, we will show that even with just a few recursion steps, the eigenvalues of H closest to the test energy E are very accurate.

The Lanczos vectors Q_i span an M-dimensional subspace denoted L_M. The tridiagonal Green function T is the restriction of $G(E)$ to L_M. Another way to view the Lanczos vectors is to imagine forming the power vectors $Q_1, GQ_1, G^2 Q_1, \ldots, G^{M-1} Q_1$, which form the basis for the Krylov subspace $K_M(G, Q_1)$. Starting with the vector Q_1 and orthonormalizing these vectors with the Gram-Schmidt algorithm, we then obtain the Lanczos vectors Q_i. The power vectors themselves become increasingly skewed in the direction of the eigenvectors having eigenvalues near E. This feature is evident if we expand the starting vector in the (so far unknown) eigenvectors $Q_1 = c_1 Z_1 + c_2 Z_2 + \ldots$, so that

$$G(E)^m Q_1 = c_1 / (E - E_1)^m Z_1 + c_2 / (E - E_2)^m Z_2 + \cdots . \qquad (3)$$

From this equation we note that when E is near eigenvalue E_j, for example, then as m increases, the power vectors becomes increasingly dominated by the eigenvectors whose eigenvalues are near E_j.

In order to generate the next Lanczos vector Q_{j+1}, Eq. (1) shows that we need to form the new vector $V_j = G(E) Q_j$. In one approach to this problem, following the shift-and-invert strategy of Erickson and Ruhe [11], the equation for V_j would first be rearranged and the matrix $(E 1_N - H)$ would be factored

$$LL^t V_j = Q_j , \qquad (4)$$

where L is a dense lower triangular $N \times N$ matrix. Unfortunately, L does not take advantage of any sparse structure in H. After defining the scratch vector $S_j = L^t V_j$, the equation $LS_j = Q_j$ is then easily solved for the vector S_j and then the equation $S_j = L^t V_j$ is solved for the desired vector V_j.

The shift-invert-factor strategy works exceedingly well provided N is small enough that L can be computed and then stored in central memory. In practice, this frequently means that $N < 5000$. However, many eigensystems

that we would like to solve have $N \gg 10\,000$. For this reason, we will abandon any attempt to factor $G(E)^{-1}$ and we will now focus upon iterative methods for computing $G(E) Q_j$.

B. Inner Green function loop

1. Overview

As mentioned earlier, the inner Green function loop applies the Green function to a Lanczos vector. There are two distinct ways to do this. In the direct method, we perturbatively expand the Green matrix and then apply the terms in the series expansion one by one to the old Lanczos vector. In the indirect method, we iteratively solve the linear algebra problem $G(E)^{-1} Q_{new} = Q_{old}$. There are several effective iterative algorithms for doing this. As expected, the direct and indirect methods are closely related, but they do not have the same convergence characteristics.

2. Dyson expansion of $G(E)$

The familiar Dyson expansion of the Green function [25] provides possibly the simplest way to act with $G(E)$ on a vector. In the N-dimensional zeroth-order basis set, we have

$$\begin{aligned} G(E) &= (E 1_N - H)^{-1} \\ &= G^0(E) + G^0(E) V G^0(E) + \cdots \\ &= [1_N + G^0(E) V + G^0(E) V G^0(E) V + \cdots] G^0(E) , \end{aligned} \qquad (5)$$

where $G^0(E) = (E 1_N - H^0)^{-1}$ is the diagonal "free-particle" (in the terminology of scattering theory) Green function and V is the off-diagonal interaction matrix. All of these matrices are of dimension N. When this expansion is truncated at the nth degree, we obtain one of the Neumann polynomials

$$G_n(E) = \{ 1_N + G^0(E) V + \cdots + [G^0(E) V]^n \} G^0(E) . \qquad (6)$$

The degree of the polynomial is chosen such that convergence is obtained in the matrix-vector product $G_n(E) Q_i$. These polynomials are sometimes used to approximate the inverse of a matrix [26]. However, this brings up an unfortunate aspect: the Dyson expansion may not converge. One factor leading to divergence is that V may be "too strong." However, the Neumann polynomial expansion is efficient in the weak coupling limit.

In order to set the stage for further developments, we note that the matrix $G^0(E) V$ has the elements

$$(G^0 V)_{ij} = V_{ij} / (E - E_i^0) \qquad (7)$$

so that if the shift E is close to the zero-order energy E_i^0, then this matrix element will be very large, provided V_{ij} is nonzero. This is the second factor that can destroy the convergence of the Dyson expansion. In addition, we note that the matrix $G^0 V$ is not symmetric, but this feature is not very harmful.

3. The partial adiabatic representation

Convergence of the iterative methods for applying $G(E)$ to a vector is enhanced by converting to the partial adiabatic representation (PAR). This desirable feature arises because the largest off-diagonal terms in Eq. (7) are eliminated. In order to convert to the PAR, we select M states whose energies (E_i^0) are near E and that are coupled relatively strongly to each other. The indices in the matrix H for these states range from n_1 to n_2. The number of these states should be as large as possible (but not too large) and may be 1000 or more in a large problem. These states span the P space (the "important" space); the relatively strong couplings within the small P space will be treated by matrix diagonalization. Coupling of these states to the remaining basis vectors in the Q space, as well as the Q-to-Q coupling, will then be treated iteratively. The practical results of this is that the algorithm used for the inner loop has much less difficulty converging.

The nondiagonal "small" $M \times M$ Hamiltonian matrix that is extracted from the full $N \times N$ matrix H is denoted H_a. We begin by diagonalizing this block,

$$H_a Z = Z E_a , \tag{8}$$

where Z denotes the orthogonal eigenvector matrix and E_a denotes the diagonal eigenvalue matrix. For later use, we note from Eq. (8) the spectral decomposition $H_a = Z E_a Z^t$. We are now ready to partition the full Hamiltonian matrix H into the block diagonal matrix H_1 and the interaction matrix H_2. With D_1 and D_2 denoting diagonal blocks, the matrix H_1 has the structure

$$H_1 = \begin{bmatrix} D_1 & 0 & 0 \\ 0 & Z E_a Z^t & 0 \\ 0 & 0 & D_2 \end{bmatrix} = D_1 \oplus Z E_a Z^t \oplus D_2 , \tag{9}$$

where $\oplus$ denotes the direct sum. The matrix H_2, by design, has a "hole" where H_a used to reside,

$$H_2 = \begin{bmatrix} V_{11} & V_{12} & V_{13} \\ V_{21} & 0 & V_{23} \\ V_{31} & V_{32} & V_{33} \end{bmatrix} . \tag{10}$$

We will now define the transformation matrix X from the PAR to the zeroth-order representation such that H_1 can be written

$$H_1 = X \varepsilon X^t , \tag{11}$$

where ε is a diagonal matrix containing the zeroth-order energies in positions $1, \ldots, (n_1 - 1)$ and $(n_2 + 1), \ldots, N$ and the adiabatic energies in positions $n_1, \ldots, n_2$. The matrix X is block diagonal,

$$X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & Z & 0 \\ 0 & 0 & 1 \end{bmatrix} . \tag{12}$$

The transformation matrix X is introduced for formal purposes only; it does not need to be stored as such during computations. In addition, the matrix H_2 can be written

$$H_2 = X v X^t . \tag{13}$$

In the following subsection, the coupling matrix in the PAR will play a major role. From Eqs. (11) and (12), this matrix is

$$v = \begin{bmatrix} V_{11} & V_{12} Z & V_{13} \\ Z^t V_{21} & 0 & Z^t V_{23} \\ V_{31} & V_{32} Z & V_{33} \end{bmatrix} . \tag{14}$$

At this point, the total Hamiltonian matrix is $H = X h X^t$, where $h = \varepsilon + v$ is the Hamiltonian matrix in the PAR. Everything is now set up so that we may continue with the iterative computation of $G(E) Q_i$.

4. Green function in the partial adiabatic representation

Using the above matrices, the partial Green function $(E 1_N - H_1)^{-1}$, which is defined in the zeroth-order basis, can be written in terms of the Green function in the PAR,

$$G_1(E) = (E X X^t - X \varepsilon X^t)^{-1} = X(E 1_N - \varepsilon)^{-1} X^t = X g X^t , \tag{15}$$

where g is the diagonal zeroth-order Green function in the PAR. The full Green function, using Eq. (13), is then

$$G(E) = [E 1_N - (H_1 + H_2)]^{-1} = G_1(E) + G_1(E) V G_1(E) + \cdots$$

$$= X g X^t + X g X^t X v X^t X g X^t + \cdots$$

$$= X[g + g v + g v g v + \cdots] X^t = X G_a(E) X^t , \tag{16}$$

where $G_a(E)$ is the full Green function evaluated in the PAR. The Dyson expansion of this Green function $g + g v + g v g v + \cdots$ is then used to evaluate the result of acting with $G_a(E)$ on a vector. In effect, the mth degree Neumann polynomial in the PAR is being used. In contrast to $G^0 V$ in Eq. (7), the iteration term $g v$ is well behaved, in part because of the hole in v.

5. Iterative linear system solvers

As mentioned above, each pass through the outer Lanczos loop requires evaluation of the matrix-vector product

$$V_j = G(E) Q_j , \tag{17}$$

where Q_j is a known Lanczos vector from the jth step. In addition to direct expansions for the Green function, we can proceed along a different route, referred to as the indirect method, for computing V_j. Equation (17) can be written as a linear algebraic system for the new vector V_j

$$[E1_N - (H_1 + H_2)]V_j = Q_j . \qquad (18)$$

Since our goal is to solve this equation iteratively for V_j, we will first convert to the PAR and then precondition the system (it could be argued that conversion to the PAR is the first step in a sequence of two preconditioning operations). There are many ways to precondition and then iteratively solve linear systems. For example, Oppe, Joubert, and Kincaid, in a review [27], list 20 preconditioners and 21 accelerators (iterative "solvers"). In this study, we will use only point Jacobi preconditioning followed by the application of two accelerators.

Conversion to the PAR is accomplished by substituting, in Eqs. (11) and (13),

$$[E1_N - (\varepsilon + v)]X^t V_j = X^t Q_j . \qquad (19)$$

In general, preconditioning of a linear system is accomplished by multiplying both sides of the equation by an approximate inverse. In the case of Eq. (19), we will left multiply by the diagonal matrix $g = (E1_N - \varepsilon)^{-1}$. We then obtain

$$(1_N - gv)S_j = gT_j , \qquad (20)$$

where, for example, $S_j = X^t V_j$. After solving this equation for S_j, we then obtain the desired solution $V_j = XS_j$. One important aspect of Eq. (20) is that even though g and v are symmetric matrices, the product gv is not; as a result, the left-side matrix $(1_N - gv)$ is not symmetric. The algorithm chosen to solve Eq. (20) must be appropriate for nonsymmetric systems.

There are several accelerators that may be used to solve Eq. (20). Two of these, GMRES (generalized minimum residual method) and DIIS (direct inversion in the iterative subspace), were investigated in this study. Each will be briefly described so that the differences between them can be appreciated. An important feature of both GMRES and DIIS is that, when applied to the solution of Eq. (20), they develop an iteration-variation polynomial representation of the Green function $(1_N - gv)^{-1}$. This aspect is considered further in the Appendix. In addition, the linear variational parameters in the Green function change with each Lanczos iteration step, so that the method is also adaptive. Now we will briefly describe these accelerators.

(i) GMRES. In the GMRES algorithm that was developed by Saad and Schultz [28(a)], the solution to Eq. (20) would be written

$$S_j = v_0 + c_1 q_1 + c_2 q_2 + \cdots + c_m q_m , \qquad (21)$$

where v_0 is an initial guess (such as the normalized version of the vector gT_j), the q_i are orthonormal Arnoldi iteration vectors [29], and m is the number of iteration steps. The vector q_1, which primes the Arnoldi recursion, is the normalized residual associated with the start-

ing vector v_0. The m coefficients $\{c_i\}$ are then determined by minimizing the square of the length of the residual vector, where the residual associated with the approximate solution in Eq. (20) is defined by

$$R = gT_j - (1_N - gv)(v_0 + c_1 q_1 + \cdots + c_m q_m) . \qquad (22)$$

Minimization of $R^t R$ with respect to the $\{c_i\}$ leads to a small $m \times m$ linear algebraic equation of the least-squares type. (Independent of the work of Saad and Schulz, Schneider and Collins [28(b)] employed essentially the same scheme in their linear algebraic approach to electron-molecule collisions.) GMRES is designed to generate the smallest residual over the current iteration space. A disadvantage of the GMRES algorithm is that storage of the m iteration vectors is required and m may be relatively large. However, when the central memory is limited, it may be possible to store these vectors on a secondary device. In some applications, a restarted version has been used. In this case, after m steps, the approximate solution vector is used to prime a new iteration sequence. However, for the applications described later in Sec. III, the restarted GMRES tended to stall; the residual decreased very little after restarting with the approximate solution. In contrast, the following algorithm applied to the same preconditioned linear system did not stall and only a small number of iteration vectors needed to be retained before restarting.

(ii) DIIS. In the DIIS algorithm developed by Pulay [30], the solution is written in the m-dimensional iterative subspace

$$S_j = c_0 v_0 + c_1 v_1 + c_2 v_2 + \cdots + c_{m-1} v_{m-1} . \qquad (23)$$

The coefficients c_i are again determined by minimizing the square of the length of the residual vector, but they are now subject to the normalization constraint

$$c_0 + c_1 + c_2 + \cdots + c_{m=1} = 1 . \qquad (24)$$

This leads to a small $(m+1)$-dimensional linear algebraic system for the expansion coefficients; this system is then solved using a standard direct method. (The dimension is $m+1$, rather than m, because of the added linear constraint.) All of this sounds very similar to GMRES; however, the expansion vectors v_i are not required to be orthonormal (they are not determined by the Arnoldi algorithm). The recent work by Sidi and co-workers [31] on accelerating the convergence of vector sequences establishes interesting mathematical properties of vector sums, such as those in Eqs. (23) and (24).

There are several iterative methods for generating these vectors, including the Jacobi method $\mathcal{M}_J$, the weighted Jacobi (WJ) method $\mathcal{M}_{WJ}$, the Gauss-Seidel (GS) method $\mathcal{M}_{GS}$, and the successive overrelaxation (SOR) method $\mathcal{M}_{SOR}$. (These iterative methods have been described in several excellent monographs [32,33].) If v_n denotes the current iteration vector, let $v_n(\mathcal{M}_J)$ and $v_n(\mathcal{M}_{GS})$ denote the Jacobi and Gauss-Seidel updates, respectively. The new weighted Jacobi and the SOR iteration vectors are then

$$\mathbf{v}_{n+1}(\mathcal{M}_{WJ}) = (1-\omega)\mathbf{v}_n(\mathcal{M}_{WJ}) + \omega\mathbf{v}_n(\mathcal{M}_J) ,$$

$$\mathbf{v}_{n+1}(\mathcal{M}_{SOR}) = (1-\omega)\mathbf{v}_n(\mathcal{M}_{SOR}) + \omega\mathbf{v}_n(\mathcal{M}_{GS}) .$$

(25)

For a few well structured problems, the value of the best relaxation parameter can be predicted (it is the one that minimizes the spectral radius of the iteration matrix). However, for the problems dealt with in Sec. III, a few test cases run with different values of ω quickly yielded a good value for this parameter.

To cite one example, test calculations were run on a 300×300 matrix with random small off-diagonal elements (magnitude < 0.003) and 1.0 on the diagonal. In order to bring the length of the residual vector down to 10^{-4}, 27, 8, 9, and 8 iterations were required using the Jacobi, WJ, GS, and SOR algorithms, respectively (these four free-standing algorithms were used independent of DIIS). At least with respect to this application, it is clear that the latter three algorithms provide a considerable advantage over the simple Jacobi iteration scheme.

A significant advantage of DIIS is the capability for restarting; after m iteration steps, the approximate solution is formed and relabeled $\mathbf{v}_0$. This $\mathbf{v}_0$ then primes the next sequence of at most m additional recursion steps. In practice, m can be set very small, for example, in the range 5–10. With DIIS, there is no need to store a large number of iteration vectors. For the applications described in Sec. III, this restart feature worked very well.

In anticipation of another implementation of the DIIS algorithm, the above version will be termed DIIS1. The second version, suggested by Stanton [34] and termed DIIS2 in this study, begins by building N^0 iteration vectors, which can be the DIIS iterates. Then, the size of the iteration space is fixed at N^0 ($=m$) and this dimension can be quite small (5–10). The oldest iterate is destroyed as the N^0-1 higher iterates are moved down by one index $\mathbf{v}_j \rightarrow \mathbf{v}_{j-1}$. The newest iterate then replaces the previous highest iteration vector and is labeled $\mathbf{v}_{N^0}$. In this way, we always work with the N^0 most recent DIIS iteration vectors. As shown later in Sec. III, this "fixed-size" version of DIIS works very well.

We close this section by mentioning that there are methods different from those studied here for using DIIS to solve eigenproblems [35–38].

(iii) *Other methods.* The well known conjugate gradient algorithm is not applicable to nonsymmetric matrices, but several other methods are [33]; including BiCG (biconjugate gradient), QMR (quasiminimal residual), CGS (conjugate-gradient squared), and BiCGSTAB (biconjugate gradient stabilized). The investigation of these methods would be useful future research projects.

C. Computation of the eigenvectors

Two methods were used in this study to compute eigenvectors. These are described below.

(a) Backtransformation of Lanczos vectors. If the Lanczos vectors are saved as they are computed, then the $N \times M$ matrix $\mathbf{Q}$ in Eq. (2) can be backtransformed to yield the eigenvectors of $\mathbf{G}(E)$, which are also eigenvectors of $\mathbf{H}$. At the end of M Lanczos steps, if we diagonalize the small $M \times M$ matrix $\mathbf{T}$, $\mathbf{S}^t\mathbf{T}\mathbf{S} = \mathbf{E}$ (diagonal), then

$\mathbf{T} = \mathbf{S}\mathbf{E}\mathbf{S}^t$. From Eq. (2),

$$\mathbf{Q}^t\mathbf{G}(E)\mathbf{Q} = \mathbf{S}\mathbf{E}\mathbf{S}^t$$

or

$$(\mathbf{QS})^t\mathbf{G}(\mathbf{E})(\mathbf{QS}) = \mathbf{E} ,$$

(26)

which shows that the $N \times M$ eigenvector matrix of $\mathbf{G}(E)$ is given by $\mathbf{Z} = \mathbf{QS}$. The $M \times M$ matrix $\mathbf{S}$ is thus used to form linear combinations of the Lanczos vectors in order to construct M approximate eigenvectors of the Green function.

(b) Inverse iteration. Once we have calculated an eigenvalue E_j close to the shift energy E, inverse iteration [39] can be used to generate the eigenvector. This method is just the inverse power method applied to the matrix $(E_j\mathbf{1}_N - \mathbf{H})$. If $\mathbf{U}$ is a starting or "source" vector (such as a vector with random components), the eigenvector $\mathbf{V}_j$ is given by the solution to the linear equations,

$$(\mathbf{E}_j\mathbf{1}_N - \mathbf{H})\mathbf{V}_j = \mathbf{U} .$$

(27)

This equation is formally identical to Eq. (17), except that the test energy has been replaced by the approximate eigenvalue. [Of course, Eq. (27) is singular if E_j if an exact eigenvalue, so the input value should not be "too good."] Equation (27) may also be written $\mathbf{V}_j = \mathbf{G}(E_j)\mathbf{U}$, which emphasizes that the Green function projects the appropriate eigenvector from the source vector. Depending upon the nature of the potential, Eq. (27) may be solved by any of the methods mentioned previously in Sec. II B 5.

III. COMPUTATIONAL RESULTS

A. The model Hamiltonian

In order to test the iterative methods described in Sec. II, a flexible model Hamiltonian was developed. The system has N_b bands of states, with N_s states in each band. For most of these studies, $N_b = 10$ and N_s was varied from 2 to 260. The states within each band are relatively strongly coupled, with weaker coupling between states in different bands. The zeroth-order diagonal energies were chosen to lie in the interval [0,1], so that the average spacing between successive states is $1/(N_bN_s)$. The Hamiltonian matrix elements are specified as follows (where i denotes the band index, $i = 1, 2, \ldots, N_b$, and j denotes the index for states in this band, $j = 1, 2, \ldots, N_s$):

for diagonal energies

$$H_{ij,ij} = (i-1)\Delta + (j-1)\delta \quad \text{where } \delta \ll \Delta ,$$

for intraband coupling

$$H_{ij,ij'} = C\exp(-|j-j'|) ,$$

and for interband coupling

$$H_{ij,i'j'} = \{C/(n_{od}|i-i'+1|)\}\exp(-|j-j'|) .$$

This model Hamiltonian has six parameters, but usually only N_s and the coupling strength C were varied. As im-

plied above, N_s was used to vary the density of states. The parameter n_{od} was used to adjust the interband coupling relative to the intraband coupling. For fixed zeroth-order state densities, the rate of convergence of the iterative methods was studied with respect to rather large variations in the off-diagonal coupling strength C. For the studies described later, we chose $\Delta = 0.1$, $\delta = 0.0001$, and (unless specified otherwise) $n_{od} = 5$. As a result, the bands begin at the zeroth-order energies $0.0, 0.1, \ldots, 0.9$.

Eigenvalues of this model Hamiltonian are shown in Fig. 2 for two choices of parameters. The two plots in the upper part of the figure show the eigenvalues for a 500 state system ($N_b = 10$, $N_s = 50$, $C = 0.05$). The blowup in the upper right shows the band structure near $E = 0.5$. The second blowup in the lower part of the figure shows the eigenvalues between 0.48 and 0.52 for a 2000 state system ($N_b = 10$, $N_s = 200$, $C = 0.1$). The goal in the following computational studies will be to accu-

rately compute a few eigenstates near $E = 0.5$ in these dense spectra.

B. Computational studies

1. Importance of the partial adiabatic representation

Before considering the convergence of the eigenvalues with respect to the number of Lanczos iteration steps, we will first comment on the importance of using the partial adiabatic representation. For this purpose, the Neumann expansion of $G(E)$ was used in the original representation for a set of increasing coupling strengths within the 2000 state model. It was found that the Neumann expansion converged only in the weak coupling limit with $C < 0.0015$. A way to calibrate this coupling strength is through the ratio of the coupling strength to the energy gap between successive states near the test energy. In this example, the coupling strength ratio is $\beta = 0.0015 / 0.000\,17 = 8.8$ (the denominator is the separation between adjacent eigenvalues near the test energy). It will be shown later in this section that convergence can be obtained for much larger values of β, provided the PAR is utilized. All of the computational results to follow were thus obtained by utilizing the PAR.

2. Outer loop convergence characteristics

We will now examine the convergence of the eigenvalues near the test energy $E = 0.5$, again for the 2000 state system in which the coupling strength parameter is $C = 0.04$ ($\beta = 235$). In addition, the DIIS1 algorithm with the Gauss-Seidel accelerator was used to solve the linear algebraic equations in the inner loop. For all of the iterative calculations reported here, the starting Lanczos vector Q_1 had constant coefficients $c_i = N^{-1/2}$. The eigenvalues and residues obtained with the iterative algorithm will be compared to the corresponding exact values, which were obtained by directly diagonalizing the Hamiltonian matrix. For later reference, a group of 18 of the exact residues and eigenvalues that are closest to the test energy are listed in Table II. The term residue denotes the square of the projection of an eigenvector upon an input test vector. For these comparisons, this normalized test vector was chosen to have constant coefficients.

Table III shows the convergence of the eigenvalues and residues for 4, 6, 8, 10, and 20 Lanczos iteration steps. Column 3 in this table gives the eigenvalues of the Green function; the extreme eigenvalues of $G(E)$ have very large magnitudes, greater than 10^4, and these converge in about six Lanczos steps. Column 4 then gives the eigenvalues of H that were computed from the eigenvalues of $G(E)$. Finally, column 5 lists the residues. These residues were computed with the recursive residue generation method [40], which does not require computation of eigenvectors. In the last two columns, the digits in the eigenvalues and residues that agree with the exact values in Table II are underscored. There are now a number of observations that can be made with respect to these results. (i) Even for four Lanczos steps, the two eigenvalues closest to the test energy are accurate to five or six digits. (ii) For six Lanczos steps, two eigenvalues, numbers 1 and
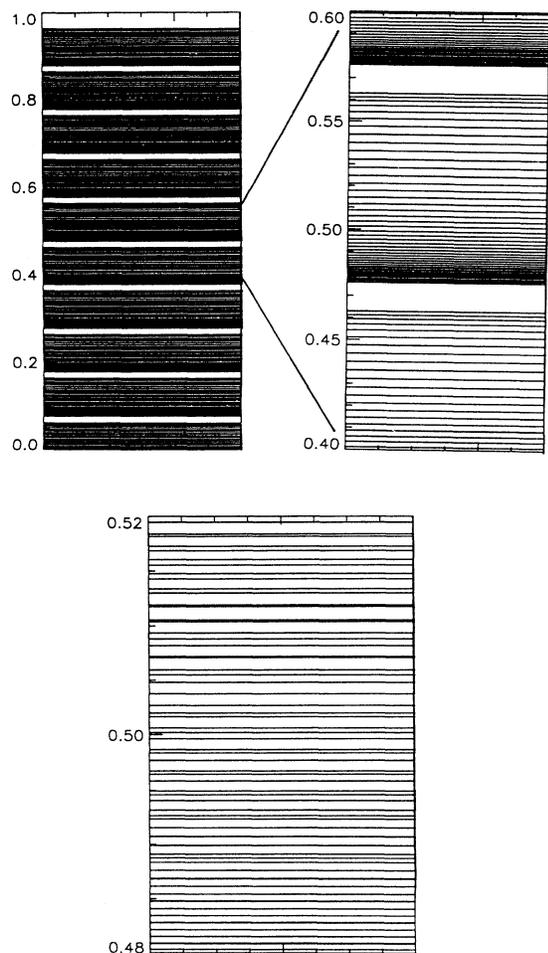

FIG. 2. Eigenvalue plots for the model Hamiltonian. The top two figures show the eigenvalues for a 500 state system (10 bands, 50 states per band, coupling strength $C = 0.05$). The lower figure shows eigenvalues near $E = 0.5$ for a 2000 state system (10 bands, 200 states per band, coupling strength $C = 0.1$).

TABLE II. Eigenvalues and residues near $E = 0.5$ for the $N = 2000$ state system. Direct diagonalization was used. The dashed line ($E = 0.5$) indicates the position near which the eigenvalues will be computed by iterative Green function methods. Residue is the square of the projection of a test vector upon each eigenvector. The test vector is a normalized vector with constant components.

State No.	Eigenvalue	Residue (unit of 10^{-5})
1076	0.498 570 25	0.035 717 51
1077	0.498 741 18	0.035 937 27
1078	0.498 911 54	0.036 213 71
1079	0.499 081 33	0.036 322 92
1080	0.499 250 59	0.036 909 59
1081	0.499 419 31	0.036 262 40
1082	0.499 587 55	0.038 628 72
1083	0.499 755 51	0.034 546 49
1084	0.499 923 78	0.043 835 15
1085	0.500 093 69	0.030 322 76
1086	0.500 267 39	0.055 320 02
1087	0.500 447 05	0.026 175 90
1088	0.500 634 13	0.070 673 26
1089	0.500 829 20	0.023 901 73
1090	0.501 032 27	0.085 953 63
1091	0.501 243 15	0.022 838 44
1092	0.501 461 62	0.100 602 67
1094	0.501 920 65	0.115 156 83

6, are accurate to eight digits (which is about 10^{-4} of the average level spacing) and when $N_L = 10$, four eigenvalues are accurate to at least eight digits. (iii) As N_L increases, the accuracy of the eigenvalues and residues improves in the direction from the outer edges toward the inter region of the spectrum of $G(E)$. (iv) When $N_L = 20$, multiple copies of eigenvalues start to appear on both edges of the spectrum of $G(E)$. This characteristic byproduct of the Lanczos algorithm is actually a welcome feature; it indicates convergence of these eigenvalues to machine precision. However, when the Green function is used to drive the Lanczos algorithm, it is only necessary to use a small number of Lanczos steps, so that multiple copies of eigenvalues will not usually appear.

Another way to summarize the data in Table III is shown in Table IV. The number of decimal digits in the eigenvalues listed in Table III that agree with the exact results (Table II) are tabulated for N_L ranging from 4 through 20. From this table, it is clear that the number of eigenvalues accurate to six or more digits increases rapidly with increasing N_L. In addition, with only eight to ten Lanczos steps, two to four very accurate eigenvalues are obtained; these are the nearest neighbors to the test energy. Finally, this table shows that the number of Lanczos steps per accurate eigenvalue ($a > 5$) is remarkably low; for four or more Lanczos steps, each additional 2.5 Lanczos steps produces an additional accurate eigenvalue.

3. Comparison of inner loop iteration methods

We will now consider the convergence characteristics of four inner loop iteration methods as a function of the

coupling strength parameter. Table V lists the total number of inner loop iteration steps that are required for ten outer loop iteration steps. The four methods listed here are: the direct use of the Neumann expansion for $G(E)$, DIIS1 with the Jacobi and Gauss-Seidel algorithms, and the GMRES algorithm. For the lowest coupling strength, the direct expansion for $G(E)$ worked very well. However, when C was larger than 0.06, the number of iteration steps increased rapidly such that when $C = 0.08$, convergence was not obtained in 8000 iteration steps. The DIIS1 algorithm worked very well for coupling strengths < 0.08, with the Gauss-Seidel version gaining the advantage over the Jacobi version as the coupling strength parameter C was increased. For the strongest coupling strength, GMRES required the smallest number of iteration steps.

The DIIS2 algorithm was also tested over the same range of coupling strengths that are listed in Table V. Recall that this is the fixed-size version of the DIIS algorithm. When $C = 0.04$, 108 DIIS iteration steps were required during ten Lanczos steps; when $C = 0.08$, 549 steps were required. These numbers are similar to the GMRES results listed in Table V. However, DIIS2 does have the significant advantage that fewer iteration vectors need to be stored (about 50 per Lanczos step for GMRES vs 10 for DIIS2, but fewer vectors could have been used for the latter method).

The recommendation from this set of results would thus be to use the Neumann expansion for $G(E)$ when the coupling strength is low and then switch to DIIS1, DIIS2, or GMRES for stronger couplings. GMRES is useful when there is enough space to store the iteration vectors, while DIIS2 is highly recommended when the storage space is limited.

For any of the inner loop iteration methods utilized in this section, the CPU time scales quadratically with respect to N, the dimension of the Hamiltonian matrix. This is to be expected since matrix-vector multiplies dominate the inner loop. To give one example, for ten Lanczos steps using GMRES for the inner loop, the CPU time on one processor of the Cray Y-MP is given by $CPU(sec) = 5.58 \times 10^{-5} N^2$ ($N < 2500$). (If H had the sparse structure that was utilized in the algorithm, then the CPU time would vary as $N^{(1+\alpha)}$, where $\alpha < 1$.)

The final aspect of the inner loop iterative methods that we will consider concerns the rate of convergence with respect to the number of inner loop iteration steps. For this purpose, Fig. 3 shows the convergence history of the DIIS1 algorithm for Lanczos step number 4. The abscissa shows the log of the residual plotted vs the iteration step. This nearly linear plot indicates exponential convergence with respect to the iteration step. This very desirable feature was also observed for the other inner loop iteration methods, except, of course, when the coupling strength C was made so large that the method failed to converge.

4. Additional convergence characteristics

In this section, convergence of the inner loop iterative methods will be examined with respect to two parame-

TABLE III. Eigenvalues and residues near $E = 0.5$ for the $N = 2000$ state system. The Lanczos-Green function algorithm was used. The DIIS1 algorithm was used to solve the algebraic equations, with a Gauss-Seidel accelerator. The numbers in square brackets denote multiplicative powers of 10. The underscore indicates all digits that agree with the direct diagonalization results in Table II. N_L denotes the number of Lanczos iteration steps.

N_L	State No.	Green function eigenvalue λ_i	Eigenvalue E_i	Residue (units of 10^{-5})
4	1	−0.105[5]	0.500 095 20	0.033 423 65
	2	−0.251[1]	0.898 253 72	92 926.360 730 67
	3	0.417[2]	0.476 005 65	7073.526 071 28
	4	0.131[5]	0.499 923 52	0.044 774 40
6	1	−0.107[5]	0.500 093 70	0.030 330 06
	2	−0.352[4]	0.500 284 09	0.079 230 36
	3	−0.413[1]	0.742 318 56	83 555.147 791 18
	4	0.247[2]	0.459 530 20	16 444.646 840 52
	5	0.385[4]	0.499 740 21	0.051 971 58
	6	0.131[5]	0.499 923 78	0.043 836 31
8	1	−0.107[5]	0.500 093 69	0.030 322 73
	2	−0.373[4]	0.500 267 81	0.056 294 95
	3	−0.167[4]	0.500 597 09	0.149 761 43
	4	−0.546[1]	0.683 106 20	74 849.853 849 94
	5	0.187[2]	0.446 530 57	25 149.741 613 48
	6	0.211[4]	0.499 525 51	0.089 208 46
	7	0.408[4]	0.499 755 18	0.035 113 88
	8	0.131[5]	0.499 923 78	0.043 835 23
10	1	−0.107[5]	0.500 093 69	0.030 322 73
	2	−0.374[4]	0.500 267 39	0.055 339 40
	3	−0.215[4]	0.500 464 41	0.039 231 98
	4	−0.117[4]	0.500 857 33	0.232 977 50
	5	−0.669[1]	0.649 576 78	66 999.608 565 34
	6	0.154[2]	0.435 231 40	32 999.765 896 74
	7	0.132[4]	0.499 240 13	0.145 306 37
	8	0.240[4]	0.499 582 72	0.043 972 09
	9	0.409[4]	0.499 755 51	0.034 552 63
	10	0.131[5]	0.499 923 78	0.043 835 23
20	1	−0.107[5]	0.500 093 69	0.000 686 59[a]
	2	−0.107[5]	0.500 093 69	0.029 636 14
	3	−0.374[4]	0.500 267 39	0.055 320 07
	4	−0.224[4]	0.500 447 05	0.026 175 86
	5	−0.158[4]	0.500 634 13	0.070 679 54
	6	−0.120[4]	0.500 832 11	0.025 539 94
	7	−0.949[3]	0.501 054 10	0.102 573 32
	8	−0.610[3]	0.501 638 19	0.260 679 08
	9	−0.213[3]	0.504 588 29	4.702 596 45
	10	−0.611[1]	0.663 630 90	69 760.150 075 41
	11	0.162[2]	0.438 139 19	30 233.668 404 37
	12	0.388[3]	0.497 424 40	0.541 918 67
	13	0.755[3]	0.498 674 89	0.122 045 18
	14	0.106[4]	0.499 052 34	0.052 671 00
	15	0.133[4]	0.499 249 81	0.037 723 17
	16	0.172[4]	0.499 419 30	0.036 264 84
	17	0.242[4]	0.499 587 55	0.038 628 67
	18	0.409[4]	0.499 755 51	0.034 546 48
	19	0.131[5]	0.499 923 78	0.042 737 58[b]
	20	0.131[5]	0.499 923 78	0.001 097 64

[a]Total residue 0.030 322 73 for this eigenvalue.
[b]Total residue 0.043 835 22 for this eigenvalue.

TABLE IV. Number of eigenvalues accurate of α decimal digits for the $N = 2000$ state system. The Lanczos-Green function algorithm was used. The DIIS1 algorithm was used to solve the algebraic equations, with a Gauss-Seidel accelerator. N_L denotes the number of Lanczos iteration steps.

			N_L		
α	4	6	8	10	20
4	0	2	1	0	3
5	1	0	0	2	0
6	1	0	2	0	0
7	0	1	0	0	1
8	0	1	2	4	7

ters: the dimension of the block H_a in the PAR [recall Eq. (8)] and n_{od} in the Hamiltonian, which determines the strength of the interband coupling. First, Table VI shows the number of iteration steps (GMRES was used in the inner loop) as a function of the dimension of H_a. Recall that this block of the Hamiltonian matrix is centered around the test energy. We note that when this dimension is < 200, a relatively large number of iteration steps (> 500) is required. As the dimension is increased, the number of iteration steps levels off near 130–140. The conclusion is that the dimension of H_a should be as large as possible in order to decrease the total number of iteration steps.

Now we will consider the influence of the interband coupling strength. Table VII lists the number of iteration steps (when GMRES or DIIS1 Gauss-Seidel are used in the inner loop) with respect to n_{od}. Recall that the interband coupling strength is determined by the ratio C/n_{od}, so that the interband coupling decreases as n_{od} increases. It is clear from this table that the number of inner loop iteration steps increases significantly when $n_{od} < 4$ and that in this circumstance, GMRES requires fewer iteration steps than DIIS1.

5. Iterative computation of eigenvectors

In Sec. II C we indicated how the eigenvectors may be generated by backtransforming the Lanczos vectors. At least in principle, this is a straightforward procedure; however, the following issue should be addressed: How

TABLE V. Total number of iteration steps (Green function vector multiplies) for various methods. Results for four coupling strengths are shown for the $N = 2000$ state system with $N_L = 10$ Lanczos steps.

		Coupling strength C		
Method[a]	0.01	0.04	0.06	0.08
$G(E)$ direct	58	97	189	diverge[b]
DIIS1 Jacobi	87	196	216	5140
DIIS1 Gauss-Seidel	65	129	143	864
GMRES	70	141	283	483

[a]In all cases, the partial adiabatic representation was used.
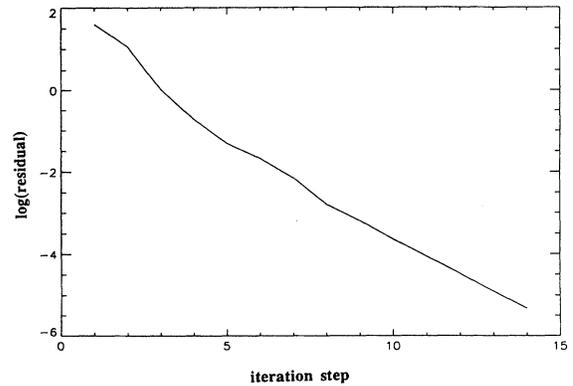[b]No convergence in 8000 steps.

FIG. 3. Natural log of the residual vs the DIIS1 iteration step index for Lanczos step 4. This result is for the $N = 2000$ state system with $C = 0.04$.

accurate are these eigenvectors when we perform only a few Lanczos steps? Before answering this question, we will examine plots of several Lanczos vectors and then one of the eigenvectors formed by taking a linear combination of the Lanczos vectors. In Fig. 4 the coefficients in the original basis set of the three Lanczos vectors Q_1, Q_5, and Q_9 are plotted. These vectors are for a 500 state model system in which the coupling strength parameter is $C = 0.04$. The bulge of relatively large coefficients near state index 280 in vector Q_1 is due to the preconditioning factor $g = (E\mathbf{1}_N - \varepsilon)^{-1}$, which appears on the right-hand side of Eq. (20). The higher Lanczos vectors have a broadened bulge near state index 280, but they gradually show the buildup of a band structure as more iterations are performed. The eigenvector formed by taking the appropriate linear combination of these Lanczos vectors is shown in Fig. 5. This eigenvector corresponds to the highest eigenvalue less than the test energy $E = 0.5$. This semilog plot displays $\ln|c_i|$ vs the state index i for $i = 1, 2, \ldots, 500$. The structure associated with ten bands of states is very evident.

We will now consider the accuracy of the eigenvectors generated by backtransforming the Lanczos vectors. In

TABLE VI. Variation of the number of iteration steps with the dimension (M) of the block H_a for the $N = 2000$ state model problem. The number of Lanczos iteration steps is $N_l = 10$ and the coupling strength parameter is $C = 0.04$.

M	No. of iterations[a]
100	> 1000
200	591
300	161
400	141
500	137
600	132
1000	79

[a]The GMRES algorithm was used to solve the linear algebraic equations. The number of GMRES iteration steps was then summed over the Lanczos iteration steps to obtain the total number of iterations.

order to define a measure of accuracy, we compared each coefficient for the approximate eigenvector with the corresponding exact coefficient (these were obtained through direct diagonalization of the Hamiltonian matrix). From the error in each coefficient $\delta_i = c_i^{exact} - c_i^{iteration}$, the root-

mean-square error per coefficient was computed,

$$\mathscr{E}_{rms} = \frac{1}{N} \left\{ \sum_{i=1}^{N} \delta_i^2 \right\}^{1/2} . \tag{28}$$
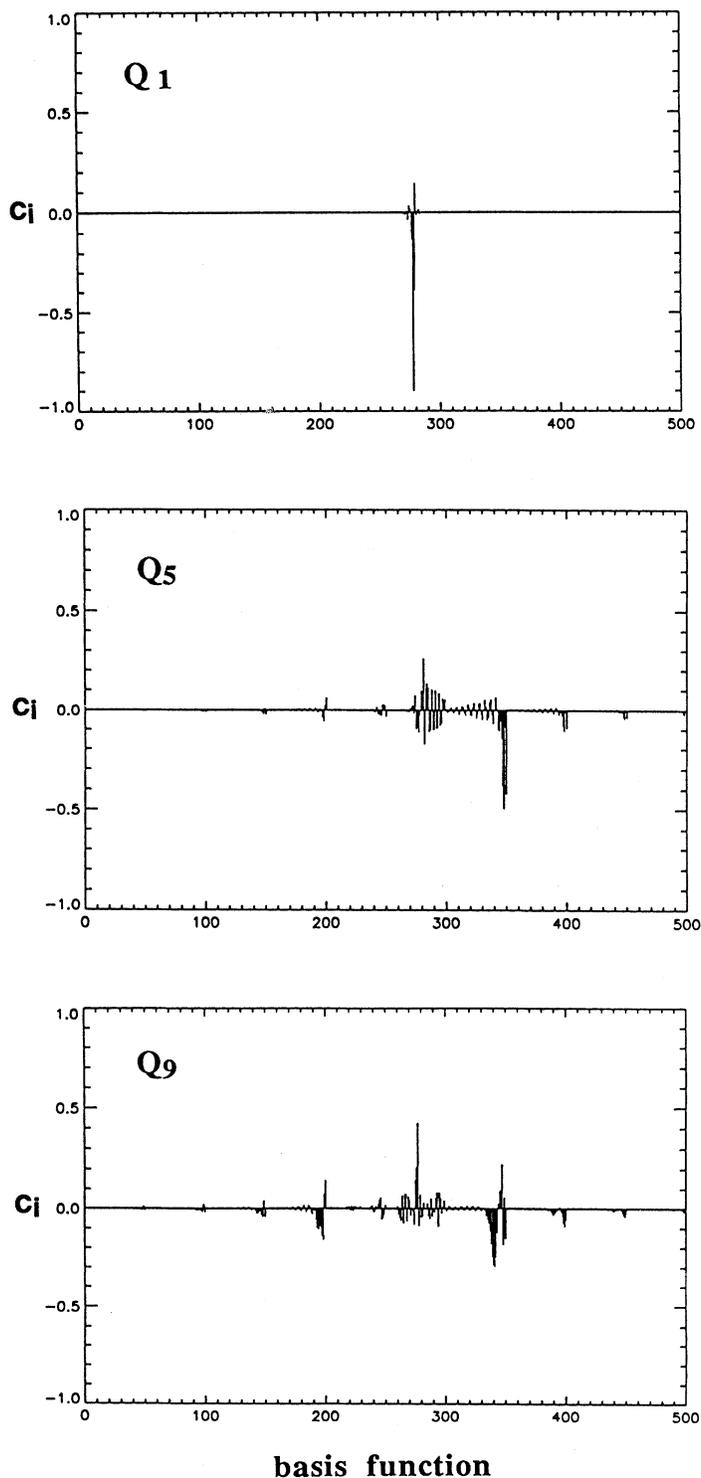
Lanczos vectors

FIG. 4. Lanczos vectors for a 500 state system (10 bands, 50 states per band, coupling strength $C = 0.04$). The expansion coefficients c_i of each vector in the zeroth-order basis set are shown for three vectors Q_1, Q_5, and Q_9.

basis function

TABLE VII. Dependence of the total number of iteration steps (obtained by summing the number of GMRES or DIIS1 steps over $N_L = 10$ Lanczos steps) upon the strength of the interband coupling, C/n_{0d}, where $C = 0.04$ and n_{0d} is varied. Note that the coupling becomes weaker as n_{0d} is increased.

Method	n_{0d}				
	3	4	5	6	7
GMRES	348	157	139	126	120
DIIS1-GS[a]	910	134	114	109	102

[a]The Gauss-Seidel accelerator was used in the DIIS1 algorithm.

TABLE VIII. Root-mean-square error per eigenvector coefficient [see Eq. (28)] for one eigenvector [associated with the largest eigenvalue *below* the test energy $(E = 0.5)$] for the $N = 2000$ model system. The number of Lanczos steps N_L and the coupling strength in the potential are varied.

N_L	Coupling strength C	
	0.01	0.08
4	2×10^{-6}	8×10^{-6}
6	1×10^{-8}	1×10^{-8}
10	$< 1 \times 10^{-8}$	$< 1 \times 10^{-8}$

The rms error is listed in Table VIII for the 2000 state model system for four, six, and ten Lanczos steps, for both weak and strong coupling strengths. Even for the higher coupling strength, the rms error decreases rapidly as the number of Lanczos steps was increased from four to six. For six Lanczos steps, the eigenvector coefficients have an extremely low rms error, about 1×10^{-8}.

IV. SUMMARY AND DISCUSSION

A. Summary of the two-layer iteration algorithm

The computation of a small group of interior eigenstates of a large matrix **H** was approached in this study through the application of a two-layer iteration scheme. Starting with a test input energy E, the outer iteration loop uses the Lanczos algorithm to develop a small $M \times M$ tridiagonal representation of the Green function $\mathbf{G}(E)$. Diagonalization of this matrix (denoted **T**) yields, through a simple mapping, excellent approximations to the eigenvalues of **H** that lie near energy E. The inner iteration loop computes the matrix-vector product $\mathbf{G}(E)\mathbf{Q}_j$, where $\mathbf{Q}_j$ is one of the Lanczos vectors. Both direct and indirect methods may be used for this operation. The direct method uses a perturbative expansion of the Green function, while the indirect method is based upon the iterative solution of a linear algebraic system. In the latter method, a polynomial representation of the Green function is developed with adaptive, variationally
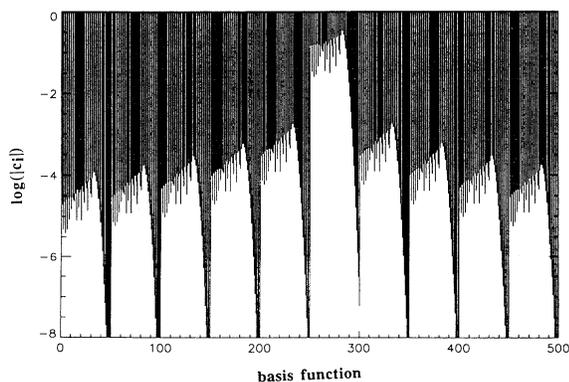

FIG. 5. Eigenvector coefficients for one eigenvector (with energy just below 0.5) in the $N = 500$ state system with $C = 0.04$. The natural log of $|c_i|$ is plotted vs the basis function index.

determined linear expansion coefficients. The GMRES, DIIS1, and DIIS2 algorithms were found to be effective methods for solving the linear system. If storage of the iteration vectors is a problem, then the DIIS2 algorithm (with a small value for the dimension of the iteration subspace N^0) is highly recommended. Convergence of these inner loop iterative methods was improved significantly by developing the Green function in the partial adiabatic representation. The two-layer iteration method was applied to a dense 2000 state model Hamiltonian matrix. With this scheme, it was possible to accurately compute a small group of eigenvalues and eigenvectors in the dense interior region of the spectrum in just a few Lanczos iteration steps (< 10).

B. Additional approaches to the matrix eigenproblem

The two-layer iteration scheme described in this study is related to other recent studies that share the same goal: compute a few eigenpairs in the interior of a dense spectrum. Recently, Dul and Arczewski [41] described a closely related two-layer scheme that they applied to large three-dimensional elliptic and two-dimensional biharmonic problems. Their outer loop involves K-dimensional subspace iteration (rather than Lanczos iteration), while the inner loop also uses a preconditioned linear system solver. Eigenvectors are then found by Rayleigh quotient iteration. This method is also fully iterative and avoids matrix factorizations. In addition, Morgan [8] has discussed several methods for implementing inverse operators in the Lanczos algorithm.

An algorithm related to those mentioned above has been developed by Sorensen [42]. The implicitly restarted Lanczos method (IRLM) develops the solution in a K-dimensional Krylov subspace by using a low degree polynomial filter in the outer loop of the Arnoldi or the Lanczos algorithm. The filter, which can be constructed from Chebyshev polynomials, is designed to accelerate convergence in a spectral interval, say $[E_{min}, E_{max}]$. This method has been used to compute surface basis functions for atom-diatom reactive scattering [43] and for the vibrational analysis of mechanical structures [44]. In addition, the subroutines *sym__lanczos* and *gen__arnoldi* on the Thinking Machines CM-5 utilize the IRLM [45].

There is considerable interest in developing parallel algorithms for dense matrix eigensystems. From time to time, the Jacobi plane rotation algorithm has been used

in parallel mode to compute all eigenvalues of real symmetric matrices [46,47]. As an example, the subroutine *sym__jacobi__ eigensystem* on the CM-5 uses this technique [45]. In a very different approach, the PRISM [48] (parallel research on invariant subspace methods) group has emphasized the invariant subspace decomposition approach of Auslander and Tsao [49]. This approach involves a treelike breakdown of the matrix into a set of smaller matrices, which can then be directly diagonalized. Another approach, based upon the IRLM, is incorporated in the ARPACK software package where parallelism is achieved, at least in part, through the matrix-vector products [50].

C. Polynomial expansion of $G(E)$

An efficient way to compute the time propagator associated with the Hamiltonian H is through the expansion in Chebyshev polynomials [51]

$$\exp(-iHt) = \sum_{n=0}^{\infty} a_n(t) T_n(H) , \qquad (29)$$

where $T_n(H)$ is the nth-order Chebyshev polynomial in which the argument is the scaled Hamiltonian matrix (the spectral range is shifted to the interval $[-1,1]$). In addition, the coefficient $a_n(t)$ is a multiple of the nth-order Bessel function, $J_n(\Delta E t)$, where the energy half-width is $\Delta E = (E_{max} - E_{min})/2$. The half Fourier transform of Eq. (29) over the interval $[t=0, \infty]$ then gives the Chebyshev expansion of the Green function [a convergence factor $\exp(-\varepsilon t)$ is first introduced in the integrand, where $\varepsilon > 0$]

$$(E^+ 1_N - H)^{-1} = \sum_{n=0}^{\infty} a_n(E^+) T_n(H) \qquad (30)$$

in which $E^+ = E + i\varepsilon$ and where the $a_n(E)$ are analytic expansion coefficients [52,53]. In some cases, such as scanning through a set of energies, this separation of the action of H and the energy E is advantageous [53,54]. Closely related to this is the polynomial expansion of the spectral density operator (SDO), defined by

$$\delta(E-H) = \frac{i}{2\pi} \lim_{\varepsilon \to 0+} [(E^+ - H)^{-1} - (E^- - H)^{-1}] . \qquad (31)$$

The real valued polynomial expansion of the SDO has been used to evaluate spectral intensities [55]. An improved version of the Chebyshev expansion of the SDO has recently been presented [55].

The number of terms required in order to converge the right-hand side of Eq. (29) [and Eq. (30)] can be determined by noting that when $n > \Delta E t$, the Bessel function decays exponentially. If the eventual goal is to compute eigenvalues, then the propagation time must be sufficiently long to resolve the energy spacings in the spectrum. If the spacing is δE and we want to resolve energies to the fraction f of this value, the propagation time must be the order of $t = 1/(f \delta E)$. As a result, the number of Chebyshev terms must be $n > \Delta E/(f \delta E)$. Now, for the model system in Sec. III, $\Delta E = 0.5$, $\delta E = 0.0002$,

and let us assume that we want energy resolution to $f = \frac{1}{10}$ of the spacing. The required number of Chebyshev terms is then about 25 000. This number is about 10^2 times the total number of matrix-vector multiplies used to compute very accurate ($f = \frac{1}{1000}$) eigenvalues for the 2000 state model system. For dense interior spectra, this spectral method does not seem like the best way to proceed. However, the filter-diagonalization method of Neuhauser [5] would likely be more efficient.

The Chebyshev expansion in Eq. (30) was tested on the model problem in Sec. III. In order to resolve the dense interior spectrum, it was necessary to choose the imaginary component of the energy ε to be smaller than the local energy gap $\delta E = 0.0002$. This small value of ε, in turn, led to a large number of terms in the Chebyshev expansion (about 32 000 terms were required for convergence).

There are alternatives to the Chebyshev expansion of the Green function. Another approach was used by Auerbach and Miller in their applications to reactive scattering [56,57]. They developed a power series [57] for $G(E)$ by using N evenly spaced quadrature points along the time axis to evaluate the half Fourier transform over the finite interval $[t=0, T]$. We conclude by noting that an advantage of the iteration-variation expansion (the Appendix) in the partial adiabatic representation (Sec. II) is that the pole structure of $G(E)$ may be accurately built in by including additional terms in the expansion.

ACKNOWLEDGMENTS

APPENDIX: ITERATION-VARIATION EXPANSION OF THE GREEN FUNCTION

An improvement over the familiar Dyson expansion for the Green function [see Eq. (5)] is provided by the following approach. The problem in simplest terms is to find the inverse of a matrix. After preconditioning, we will write the linear system as

$$(1 - A)x = b , \qquad (A1)$$

so that the usual Dyson expansion for the solution is

$$x = (1 - A)^{-1} b = (1 + A + A^2 + A^3 + \cdots) b . \qquad (A2)$$

The iteration-variation approach within the DIIS formalism using the Jacobi accelerator may be initiated with the vector $x^{(0)} = y^{(0)} = b$, for which the residual vector is $R^{(0)} = Ab$. The Jacobi update would then provide the next iteration vector $y^{(1)} = y^{(0)} + R^{(0)} = b + Ab$ and the next residual vector $R_{(1)} = A^2 b$. At this stage, the approximate solution to Eq. (A1) would be

$\mathbf{x}^{(1)}=c_0\mathbf{y}^{(0)}+c_1\mathbf{y}^{(1)}$, where the constraint $c_0+c_1=1$ is imposed. The corresponding residual vector is then $\mathbf{R}=c_0\mathbf{R}^{(0)}+c_1\mathbf{R}^{(1)}=(1-c_1)\mathbf{Ab}+c_1\mathbf{A}^2\mathbf{b}$. The variational aspect then enters with the minimization of $\mathbf{R}\cdot\mathbf{R}$ with respect to c_1. This gives the "best" value for c_1, denoted c_1', and then $c_0'=1-c_1'$. The iteration-variation solution would then be

$$\mathbf{x}^{(1)}=[c_0'\mathbf{b}+c_1'(\mathbf{b}+\mathbf{Ab})]=(1+c_1'\mathbf{A})\mathbf{b} . \qquad (A3)$$

Continuing beyond the first iteration step, the iteration-variation solution is

$$\mathbf{x}=(1+c_1'\mathbf{A}+c_2'\mathbf{A}^2+c_3'\mathbf{A}^3+\cdots)\mathbf{b} . \qquad (A4)$$

In order to relate more closely to the Green function analysis, we will now make a change of notation; let $\mathbf{x}=\mathbf{G}$, $\mathbf{A}=\mathbf{G}^0\mathbf{V}$, and $\mathbf{b}=\mathbf{G}^0$. Equation (A4) then becomes

$$\mathbf{G}=c_0'\mathbf{G}^0+c_1'\mathbf{G}^0\mathbf{V}\mathbf{G}^0+c_2'\mathbf{G}^0\mathbf{V}\mathbf{G}^0\mathbf{V}\mathbf{G}^0+\cdots . \qquad (A5)$$

When truncated at the nth degree in $\mathbf{G}^0\mathbf{V}$, the resulting polynomial represents the parametrized incomplete inverse [56] of the matrix $(E\mathbf{1}-\mathbf{H})^{-1}$. Finally, we emphasize that the weight of each term in this polynomial is chosen through a constrained variational procedure.

[1] R. Kosloff and H. Tal-Ezer, Chem. Phys. Lett. **127**, 223 (1986).

[2] M. D. Feit, J. A. Fleck, and A. Steiger, J. Comput. Phys. **47**, 412 (1982).

[3] R. T. Skodje, R. Sadeghi, H. Koppel, and J. L. Krause, J. Chem. Phys. **101**, 1725 (1994).

[4] S.-T. Yau and Y. Y. Lu, SIAM J. Sci. Comput. **14**, 121 (1993).

[5] D. Neuhauser, J. Chem. Phys. **93**, 2611 (1990); **95**, 4927 (1991); **100**, 5076 (1994).

[6] E. R. Davidson, Comput. Phys. Commun. **53**, 49 (1989).

[7] R. B. Morgan and D. S. Scott, SIAM J. Sci. Stat. Comput. **7**, 817 (1986).

[8] R. B. Morgan, Linear Algebra Appl. **154**, 289 (1991).

[9] R. B. Morgan and D. S. Scott, SIAM J. Sci. Comput. **14**, 585 (1993).

[10] C. Lanczos, J. Res. Natl. Bur. Stand. **45**, 255 (1950).

[11] T. Ericsson and A. Ruhe, Math. Comput. **35**, 1251 (1980).

[12] F. Webster, P. J. Rossky, and R. A. Friesner, Comput. Phys. Commun. **63**, 494 (1991).

[13] H. Kono, Chem. Phys. Lett. **214**, 137 (1993).

[14] C. Iung and C. Leforestier, J. Chem. Phys. (to be published); A. Maynard, C. Iung, and R. E. Wyatt (unpublished).

[15] (a) J. B. Rosser, C. Lanczos, M. R. Hestenes, and W. Karush, J. Res. Natl. Bur. Stand. **47**, 291 (1951); (b) D. A. Flanders and G. Shortley, J. Appl. Phys. **21**, 1326 (1950).

[16] Y. Saad, *Numerical Methods for Large Eigenvalue Problems* (Halsted, New York, 1992).

[17] F. Chatelin, *Eigenvalues of Matrices* (Wiley, New York, 1993).

[18] J. Cullum, W. Kerner, and R. Willoughby, Comput. Phys. Commun. **53**, 19 (1989).

[19] Y. Saad, Comput. Phys. Commun. **53**, 71 (1989).

[20] For applications in chemical physics, see J. D. Kress, G. A. Parker, R. T. Pack, B. J. Archer, and W. A. Cook, Comput. Phys. Commun. **53**, 91 (1989); S. Dallwig, N. Fahrer, and C. Schlier, Chem. Phys. Lett. **191**, 69 (1992).

[21] T. J. Gil, C. L. Winsted, and P. W. Langhoff, Comput. Phys. Commun. **53**, 123 (1989).

[22] B. N. Parlett and B. Nour-Omid, Comput. Phys. Commun. **53**, 169 (1989).

[23] D. S. Scott, Comput. Phys. Commun. **53**, 271 (1989).

[24] R. G. Grimes, J. G. Lewis, and H. D. Simon, SIAM J. Matrix Anal. Appl. **15**, 228 (1984).

[25] P. Roman, *Advanced Quantum Theory* (Addison-Wesley, Reading, MA, 1965).

[26] P. F. Dubois, A. Greenbaum, and G. H. Rodrigue, Computing **22**, 257 (1979).

[27] T. C. Oppe, W. D. Joubert, and D. C. Kincaid, Comput. Phys. Commun. **53**, 283 (1989).

[28] (a) Y. Saad and M. Schultz, SIAM J. Sci. Stat. Comput. **7**, 856 (1986); (b) B. I. Schneider and L. A. Collins, Phys. Rev. A **33**, 2970 (1986).

[29] W. E. Arnoldi, Q. Appl. Math. **9**, 17 (1951).

[30] P. Pulay, Chem. Phys. Lett. **73**, 393 (1980); T. P. Hamilton and P. Pulay, J. Chem. Phys. **84**, 5728 (1986).

[31] A. Sidi, W. F. Ford, and D. A. Smith, SIAM J. Numer. Anal. **23**, 178 (1986); A. Sidi, J. Comput. Appl. Math. **36**, 305 (1991).

[32] W. L. Briggs, *A Multigrid Tutorial* (SIAM, Philadelphia, 1987).

[33] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romaine, and H. van der Vorst, *Templates* (SIAM, Philadelphia, 1994).

[34] J. F. Stanton (unpublished).

[35] P. Bendt and A. Zunger, Phys. Rev. B **26**, 3114 (1982).

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

[37] J. L. Martins and M. L. Cohen, Phys. Rev. B **37**, 6134 (1988).

[38] H. Kim, B. D. Yu, and J. Ihm, J. Phys. A **27**, 1199 (1994).

[39] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in Fortran* (Cambridge University Press, New York, 1992).

[40] A. Nauts and R. E. Wyatt, Phys. Rev. Lett. **51**, 2238 (1983).

[41] F. A. Dul and K. Arczewski, J. Comput. Phys. **111**, 89 (1994).

[42] D. C. Sorensen, in *Large-Scale Numerical Optimization*, edited by T. F. Coleman and Y. Li (SIAM, Philadelphia, 1990), p. 228.

[43] P. Pendergast, Z. Darakjian, E. Hayes, and D. C. Sorensen, J. Comput. Phys. **113**, 201 (1994).

[44] H. A. Smith, D. C. Sorensen, and R. K. Singh, Int. J. Numer. Methods Eng. **36**, 1987 (1993).

[45] CMSSL Manual, Ver. 3.1 (Thinking Machines Corp., Cambridge, MA 1993), Chap. 8.

[46] R. Carbo, L. Molino, and B. Calabuig, J. Comput. Chem. **13**, 155 (1992).

[47] R. P. Brent and F. T. Luk, SIAM J. Sci. Stat. Comput. **6**, 69 (1985).

[48] PRISM working notes are available via anonymous ftp to ftp.super.org in the directory pub/prism.

[49] L. Auslander and A. Tsao, Adv. Appl. Math. **13**, 253

(1992).

[50] J. Choi, J. J. Dongarra, R. Pozo, D. C. Sorrensen, and D. W. Walker, Int. J. Supercomputer Appl. **8**, 99 (1994).

[51] H. Tal-Ezer and R. Kosloff, J. Chem. Phys. **81**, 3967 (1984).

[52] R. Kosloff, J. Phys. Chem. **92**, 2087 (1988); R. Kosloff, Annu. Rev. Phys. Chem. **45**, 145 (1994).

[53] Y. Huang, W. Zhu, D. J. Kouri, and D. K. Hoffman, Chem. Phys. Lett. **214**, 451 (1993).

[54] Y. Huang, D. J. Kouri, and D. K. Hoffman, Chem. Phys.

Lett. **225**, 37 (1994).

[55] W. Zhu, Y. Huang, D. J. Kouri, C. Chandler, and D. K. Hoffman, Chem. Phys. Lett. **217**, 73 (1994); D. J. Kouri, W. Zhu, G. A. Parker, and D. K. Hoffman, Chem. Phys. Lett. (to be published).

[56] S. M. Auerbach and W. H. Miller, J. Chem. Phys. **100**, 1103 (1994).

[57] S. M. Auerbach and W. H. Miller, J. Chem. Phys. **98**, 6917 (1993).