Computing Charge Densities with Partially Reorthogonalized Lanczos *

Constantine Bekas † Yousef Saad † Murilo L. Tiago ‡ James R. Chelikowsky §

March 30, 2005

Abstract

This paper considers the problem of computing charge densities in a density functional theory (DFT) framework. In contrast to traditional, diagonalization-based, methods, we utilize a technique which exploits a Lanczos basis, without explicit reference to individual eigenvectors. The key ingredient of this new approach is a partial reorthogonalization strategy whose goal is to ensure a good level of orthogonality of the basis vectors. The experiments reveal that the method can be a few times faster than ARPACK, the implicit restart Lanczos method. This is achievable by exploiting more memory and BLAS3 (dense) computations while avoiding the frequent updates of eigenvectors inherent to all restarted Lanczos methods.

1 Introduction

Matrix diagonalization constitutes the most time-consuming part of typical electronic structures calculation codes. Many efforts have been put into avoiding diagonalization, or at least, reducing its cost. Thus, a number of techniques bypass the need of diagonalization by attacking the original optimization problems directly, see, e.g., [1] for an overview. Typical examples of this approach are the Car-Parrinello method [2] and methods based on direct minimization of the Kohn-Sham energy functional [1]. While these methods do not compute eigenvectors directly, they do solve a (nonlinear) optimization problem whose complexity is comparable to that of solving a sequence of eigenproblems. In this paper we take a middle ground approach. Individual eigenvectors are not explicitly obtained but a good basis of the invariant subspace associated with the occupied states is computed, in which the charge density is expressed. When desired, eigenvectors can be computed easily, making this method also useful for cases when not only occupied states are needed but also many virtual (unoccupied) ones are needed. This happens frequently when Kohn-Sham eigenstates are used to compute optical absorption spectra within the framework of time-dependent DFT [3, 4].

In the proposed approach, approximate eigenvectors are only used implicitly, in the sense that there is no attempt to compute them individually. Rather, we focus on the subspace which they generate. Specifically, the charge density is viewed as the diagonal of the density matrix P [5, 6, 7]. Given *any* basis $v_1, v_2, \ldots, v_{n_o}$ of the space spanned by the occupied states, the charge density can be obtained as the diagonal of $P \equiv VV^T$, where $V = [v_1, v_2, \ldots, v_{n_o}]$ denotes the matrix with column vectors $v_1, \ldots, v_{n_o}$. Thus, the basic principle of the method is to focus on subspaces rather than on individual eigenvectors. We first compute an orthonormal basis $q_1, \ldots, q_m$ whose span contains the span of V. This is done with the help of a Lanczos procedure. Since this procedure is prone to a loss of orthogonality of the basis vectors produced, an inexpensive form of re-orthogonalization is employed. Just as important an ingredient for the overall procedure, is the convergence test which in this case measures how accurate a subspace $span(V)$

*Work supported by NSF grants ITR-0082094, DMR-0325218, by DOE under Grants DE-FG02-03ER25585, DE-FG02-03ER15491, and by the Minnesota Supercomputing Institute

†Computer Science & Engineering, University of Minnesota, Twin Cities. {bekas,saad}@cs.umn.edu

‡Department of Chemical Engineering and Materials Science, Institute for the Theory of Advanced Materials in Information Technology, Digital Technology Center, University of Minnesota, Minneapolis, MN 55455, mtiago@msi.umn.edu

§ICES, 1 University Station, University of Texas at Austin, Austin, Texas 78712. jrc@ices.utexas.edu

we would get if we were to stop the Lanczos procedure at a given step. This test uses eigenvalues (only) of the tridiagonal matrix obtained from the Lanczos algorithm and its cost is negligible relative to the rest of the calculation.

It is important to compare this procedure, at the outset, with a competitive approach based on diagonalization by a restarted Lanczos procedure. For example, a procedure such as the one in the ARPACK code can be used for this task [8]. In an implicitly restarted Lanczos procedure, eigenvectors are computed one (or a few) at a time, i.e., the basis V is computed one vector at a time. The main drawback of this method from a computational point of view is that eigenvectors are (implicitly) updated at every step, or at frequent intervals, to check convergence. In addition, restarting causes the number of matrix-vector multiplications to be quite high, because of the induced loss of optimality. In contrast, if we were to perform one single run (no restarts) of the Lanczos procedure, the eigenvectors would converge much faster in principle because of the optimality of the underlying procedure. However, there are two issues to address. First, there is the problem of loss of orthogonality. It is known that the vectors obtained from the Lanczos procedure loose their orthogonality fairly rapidly [9] after the first few eigenvectors start converging. Reorthogonalizing these vectors can be quite expensive if full reorthogonalization is performed. Fortunately, an inexpensive form of reorthogonalization exists which monitors loss of orthogonality with an inexpensive recurrence relation and performs reorthogonalization only when needed. The second issue with the non-restarted Lanczos procedure is that it would be difficult to know when to stop. Normally, we would require that desired eigenvectors, namely those of the occupied states, be accurate enough. This would mean that we would compute these eigenvectors at frequent intervals, or obtain by some means estimates of their accuracies as the algorithm proceeds. We adopted a much less expensive alternative which consists of checking the convergence of the sum of the n_o smallest eigenvalues, those corresponding to the n_o lowest states. An advantage of this technique is that this number is fairly inexpensive to compute: it is simply the sum of the smallest eigenvalues (without eigenvectors) of the tridiagonal matrix obtained from the Lanczos procedure. At the m-th step of Lanczos, this cost is of the order of m^2. We found that its cost is negligible relative to the cost of the overall procedure.

Throughout the paper, the descriptions of the algorithms rely on `MATLAB` notation. For example, $A(:, 1:k)$ refers to the first k columns of matrix A and $A(1:k, :)$ to its first k rows. The symbol $\|x\|_2$ means the Euclidian norm of the vector x and, for a matrix A, $\|A\|_2$ is the induced norm of A.

2 Charge densities without eigenvectors

The charge density $\rho(r)$ at a point r in space is commonly computed from the eigenvectors Ψ_i of the Hamiltonian matrix H via the formula

$$\rho(r) = \sum_{i=1}^{n_o} |\Psi_i(r)|^2, \tag{1}$$

where the summation is taken over all occupied states of the system under study. For simplicity, we assume no explicit spin dependence. In order to compute the charge density $\rho(r)$ via (1), eigenvectors are normally required. However, it is also possible to compute $\rho(r)$ without explicitly resorting to using eigenvectors.

Let the vectors ψ_i be the discretizations of $\Psi_i(r)$ with respect to r. Then, the charge densities are the diagonal entries of the "functional density matrix"

$$P = V_{n_o} V_{n_o}^\top \quad \text{with} \quad V_{n_o} = [\psi_1, \ldots, \psi_{n_o}]. \tag{2}$$

Specifically, the charge density at the j-th point r_j is the j diagonal entry of P. Observe that any orthogonal basis $\mathcal{V}$ that spans the same subspace as the eigenvectors $\psi_i,\ i = 1, \ldots, n_o$ can be used.

In particular, consider the Heaviside function $\mathcal{F}$, which is illustrated in Fig. 1. With $\lambda_{\max}$ we denote the largest eigenvalue of the matrix at hand, while λ_{n_o} is the eigenvalue which corresponds to the last occupied state of the system. Then, it is not difficult to see that

$$P = \mathcal{F}(H).$$

Indeed, if we consider the spectral decomposition of the Hamiltonian

$$H = V \Lambda V^\top, \quad \text{where} \quad V = [\ V_{n_o} \quad \tilde{V}\ ], \quad V_{n_o}^\top V_{n_o} = I,$$

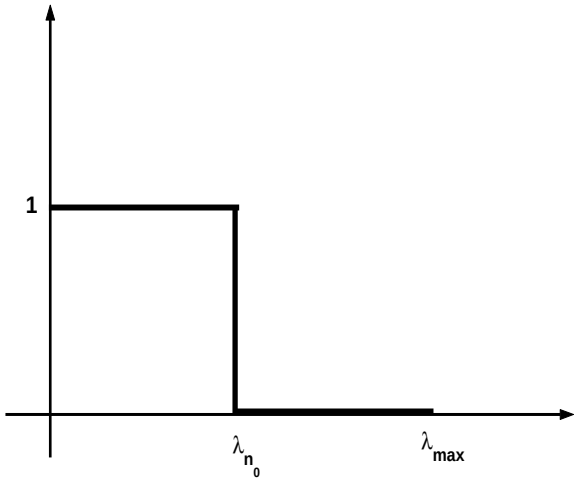

Figure 1: The Heaviside function.

then,

$$\begin{aligned}\mathcal{F}(H) &= V\mathcal{F}(\Lambda)V^\top \\ &= [\ V_{n_o} \quad \tilde{V}\ ] \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_{n_o}^\top \\ \tilde{V}^\top \end{bmatrix} \\ &= V_{n_o} V_{n_o}^\top.\end{aligned}$$

One can approximate P using this expression and many linear-scaling methods have exploited this, see e.g., [5, 6, 7, 10].

Observe that instead of the eigenvectors in V, all that is needed is a basis that spans the same subspace. In this paper we follow this approach in using Krylov subspaces to approximate the Heaviside function of the Hamiltonian. For a given unit vector q_1, the Krylov subspace $\mathcal{K}_m(A, q_1)$ of a square matrix A, with respect to q_1 is defined as (see [11])

$$\mathcal{K}_m(A, q_1) = \mathtt{span}\{q_1, Aq_1, A^2q_1, \ldots, A^{(m-1)}q_1\}. \tag{3}$$

When A is a real symmetric matrix ($A = A^\top$), as is the case of the Hamiltonians considered in this paper, the best known method for computing an orthonormal basis Q_m for $\mathcal{K}_m(A, q_1)$ is the Lanczos algorithm, proposed by C. Lanczos in 1950 [12] (see also [11, 13, 14] for a detailed description of the algorithm). Figure 2 provides an algorithmic description of the method. Note that, in exact arithmetic, the algorithm can be recast as a simple three-term recurrence, namely,

$$\beta_{i+1}q_{i+1} = Aq_i - \alpha_i q_i - \beta_i q_{i-1} \tag{4}$$

What is remarkable about this algorithm is that, in exact arithmetic, it is capable of computing an orthonormal basis of the above Krylov subspace, with the simple three term recurrence of (4). That is, only three vectors are required in memory at any step. However, as is well known, roundoff errors cause severe loss of orthogonality among the basis vectors (columns of matrix Q_m), and some form of reorthogonalization must be applied. In standard electronic structures calculations where Hamiltonian matrices can be very large and many eigenvectors must be computed, reorthogonalization costs become prohibitive and dominate the overall calculation. However, as we will see in the following, clever reorthogonalization techniques have been designed that can considerably reduce these costs.

An additional appealing characteristic of the Lanczos algorithm is that the matrix A is only needed in "functional" form, meaning that all that is needed is a routine to compute the product Aq_i for any given vector q_i (line 3, Figure 2)). Thus, the matrix can either be accessible in stencil form or it can be stored in

Lanczos

*(*Input*)* Matrix $A \in \mathbb{R}^{n \times n}$, starting vector q_1, $\|q_1\|_2 = 1$, scalar m
*(*Output*)* Orthogonal basis $Q_m \in \mathbb{R}^{n \times m}$ of $\mathcal{K}_m(A, q_1)$,
unit norm vector q_{m+1} such that $Q_m^\top q_{m+1} = 0$

1. Set $\beta_1 = 0$, $q_0 = 0$
2. **for** $i = 1, \ldots, m$
3. $\quad w_i = Aq_i - \beta_i q_{i-1}$
4. $\quad \alpha_i =< w_i, q_i >$
5. $\quad w_i = w_i - \alpha_i q_i$
6. $\quad \beta_{i+1} = \|w_i\|_2$
7. $\quad$ **if** $(\beta_{i+1} == 0)$ **then** `stop`
8. $\quad q_{i+1} = w_i / \beta_{i+1}$
9. **end**

Figure 2: The Lanczos algorithm. The inner product for vectors is denoted by $< ., . >$.

sparse data structures [15]. We stress that this property of the Lanczos algorithm is particularly convenient in electronic structure codes that are based on finite difference discretizations, where Hamiltonians are available in stencil form, and so do not need to be explicitly stored.

We now show how one can use the Lanczos basis Q_m to approximate the functional density matrix P (see (2)). After m steps of the Lanczos algorithm on the Hamiltonian H and a random unit norm starting vector q_1 the following factorization holds

$$HQ_m = Q_m T_m + \beta_{m+1} q_{m+1} e_m^\top, \tag{5}$$

where T_m is the tridiagonal symmetric matrix

$$T = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{m-1} & \alpha_{m-1} & \beta_m \\ & & & \beta_m & \alpha_m \end{bmatrix}, \tag{6}$$

q_{m+1} is the last vector computed by Lanczos and e_m is the $m-th$ column of the canonical basis (thus e_m has 1 at the $m-$th entry and zeros elsewhere). If we were to run all n steps of Lanczos, then $H = Q_n T_n Q_n^\top$ and the functional density matrix P could be obtained as $P = \mathcal{F}(H) = Q_n \mathcal{F}(T_n) Q_n^\top$. However, the cost for computing a full Lanczos basis is $O(n^3)$, which is comparable to the cost of computing all eigenvectors of the Hamiltonian H.

Instead, we can approximate $\mathcal{F}(H)$ by stopping Lanczos early for some $m \ll n$. In particular, let $T_m = U \Lambda_m U^\top$ be the spectral decomposition of T_m. Furthermore, let $U = \begin{bmatrix} U_{n_o} & \tilde{U} \end{bmatrix}$, where $U_{n_o} \in \mathbb{R}^{m \times n_o}$ are eigenvectors of T_m that correspond to its smallest n_o eigenvalues. Then, we write

$$\begin{aligned} \mathcal{F}(H) &\approx Q_m \mathcal{F}(T_m) Q_m^\top \\ &= Q_m U \mathcal{F}(\Lambda_m) U^\top Q_m^\top \\ &= Q_m \begin{bmatrix} U_{n_o} & \tilde{U} \end{bmatrix} \begin{bmatrix} I_{n_o} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_{n_o}^\top \\ \tilde{U} \end{bmatrix} Q_m^\top \\ &= Q_m U_{n_o} (Q_m U_{n_o})^\top. \end{aligned} \tag{7}$$

Thus, we approximate the charge densities $\rho(r_i), i = 1, \ldots, n$ by the diagonal of matrix (7). These values are equal to the squared norms of the rows of the matrix $Q_m U_{n_o}$.

Partially Reorthogonalized Lanczos

*(*Input*)* Matrix $A \in \mathbb{R}^{n\times n}$, starting vector q_1, $\|q_1\|_2 = 1$, scalar m, tolerance `tol`
*(*Output*)* Partially Orthogonal basis $Q_m \in \mathbb{R}^{n\times m}$ of $\mathcal{K}_m(A, q_1)$

1. Set $\beta_1 = 0$, $q_0 = 0$
2. **for** $i = 1, \ldots, m$
3. $w_i = Aq_i - \beta_i q_{i-1}$
4. $\alpha_i =< w_i, q_i >$
5. $w_i = w_i - \alpha_i q_i$
6. $\beta_{i+1} = \|w_i\|_2$
7. **if** $(\beta_{i+1} == 0)$ **then** `stop`
8. $q_{i+1} = w_i/\beta_{i+1}$
9. Update level of orthogonality η_i for q_{i+1}
10. **if** $\eta_i >$ `tol` **then** reorthogonalize q_{i+1} again
    all previous vectors $q_1, \ldots, q_{i+1}$ using Gram-Schmidt
11. **end**

Figure 3: The Partially Reorthogonalized Lanczos algorithm. The inner product for vectors is denoted by $< ., . >$.

2.1 Partially reorthogonalized Lanczos

In exact arithmetic the three term recurrence of the Lanczos algorithm should build an orthonormal basis Q_m of the Krylov subspace $\mathcal{K}_m(H, q_1)$ for arbitrarily large m. However, in floating point computations, errors will cause the basis vectors to rapidly loose orthogonality [16]. As an example, consider the Hamiltonian ($n = 17,077$) corresponding to $\mathtt{Si_{10}H_{16}}$, produced by a real space pseudopotential discretization code [17]. We test the orthogonality of the bases $Q_i, i = 1, \ldots, m$, with $m = 200$ by computing the norm $\|Q_i^\top Q_i - I_i\|_2$, where I_i is the identity matrix of size i. The left plot in Figure 4 illustrates the rapid deterioration of orthogonality among basis vectors.

A number of existing reorthogonalization schemes are often employed to remedy the problem. The simplest of these consists of a full reorthogonalization approach, whereby the orthogonality of the basis vector q_i is enforced against all previous vectors at each step i. This means that the vector q_i, which in theory is already orthogonal against $q_1, \ldots, q_{i-1}$, is orthogonalized (a second time) against these vectors. In principle we no longer have a 3-term recurrence but this is not an issue as the corrections are small and usually ignored (see however Stewart [18]). The additional cost at the m-th step will be $O(nm)$. So, if reorthogonalization is required at each step, then we require an additional cost of $O(nm^2)$ which are consumed the by Gram-Schmidt process. In general all basis vectors need to be available in main memory, therefore rendering the method impractical for bases of large dimension,

An inexpensive alternative to full reorthogonalization is *partial reorthogonalization*. In contrast to full reorthogonalization, this scheme performs a reorthogonalization step only when it is deemed necessary. The goal is not so much to guarantee that the vectors are exactly orthogonal, but to ensure that they are at least nearly orthogonal. Typically, the loss of orthogonality is allowed to grow up to roughly the square root of the machine precision, before a reorthogonalization is performed. A result by Simon ([19]) ensures that we can get fully accurate approximations to the Ritz values (eigenvalues of the tridiagonal matrix T_i) in spite of a reduced level of orthogonality among the Lanczos basis vectors. Furthermore, a key to the successful utilization of this result is the existence of clever recurrences which allow us to estimate the level of orthogonality among the basis vectors [20, 21]. It must be stressed that the cost of updating the recurrence is a very modest one. Thus, we can cheaply and efficiently probe the level of orthogonality of the current vector (say q_i) and determine whether a reorthogonalization step against previous basis vectors is required.

Although not utilized in the present paper, an important side benefit of this procedure, is that it becomes unnecessary to store all basis vectors in main memory. We can instead use secondary storage and

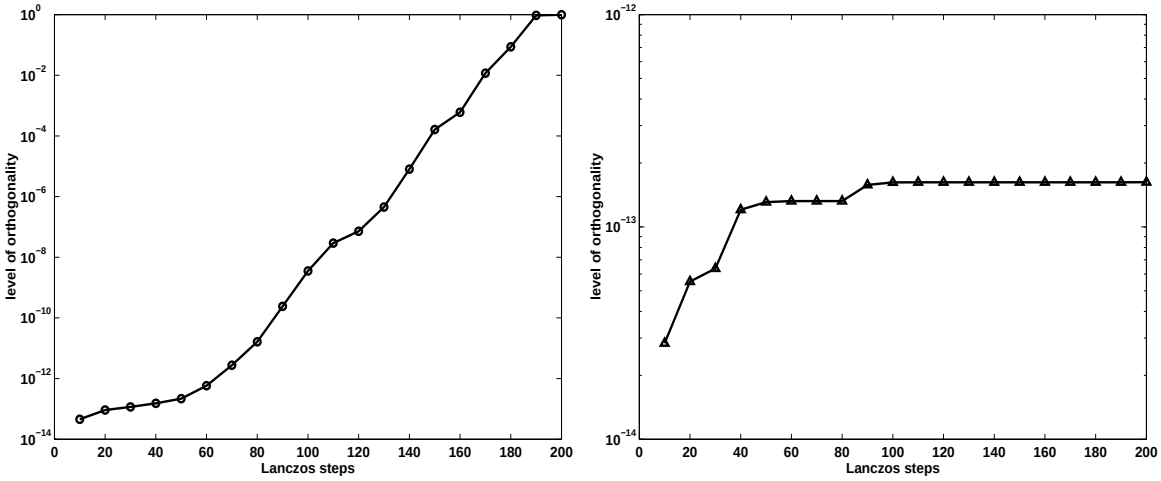

Figure 4: Levels of orthogonality of the Lanczos basis for the Hamiltonian ($n = 17077$) corresponding to $\mathtt{Si_{10}H_{16}}$. Left: Lanczos without reorthogonalization. Right: Lanczos with partial reorthogonalization. The number of reorthogonalizations was 34 with an additional 3400 inner vector products.

bring these vectors back to main memory, say a few at a time, when they are needed for reorthogonalization. The rationale is that previous vectors will only be needed infrequently, so the cost of accessing secondary storage will not hamper overall performance significantly. In addition, due to the simple access pattern, there are many ways to dampen the cost of accessing secondary storage by overlapping computations with read/writes from disk.

Implementations of Lanczos with partial reorthogonalization are available in `MATLAB` [22] as well as in `FORTRAN` [22, 23]. In our experiments we have used the `PLAN` software package by K. Wu and H. Simon [23]. The right plot in Figure 4 illustrates the level of orthogonality for the same Hamiltonian as in the previous example ($\mathtt{Si_{10}H_{16}}$). We used the default orthogonality threshold, $\mathtt{tol} = \mathtt{SQRT(EPS)}$, where `EPS` is the machine precision ($\mathtt{tol} \approx \mathtt{10^{-16}}$ for double precision arithmetic). The number of required reorthogonalizations was 34, compared to 200 for full reorthogonalization, which induced 3400 additional inner vector products.

3 The Algorithm

A method which utilizes the Lanczos algorithm in order to approximate the charge densities must include a test of convergence. It is desirable that this test be inexpensive to perform, as well as accurate, in the sense that it should not lead to unnecessarily long runs or premature termination.

A natural test would be to monitor the progress of the approximate diagonal of $\mathcal{F}(H)$. However, equation (7) indicates that this entails computing the matrix product $Q_i U_{n_o}$ at each step $i, i > n_0$. Since Q_i is $n \times i$ and U_{n_o} is $i \times n_0$ the cost at each step is in the order $O(i^2 n n_0)$. Therefore, the overall cost is $O(m^3 n n_0)$. If a large Lanczos basis Q_m is required for convergence, this cost will grow rapidly and dominate the overall cost of the method.

An alternative method is to monitor the eigenvalues of the tridiagonal matrices $T_i,\ i = 1, \ldots, m$. The cost for computing only the eigenvalues of T_i is $O(i^2)$. If we were to apply the test at every single step of the procedure, the total cost for all m Lanczos steps would be $O(m^3)$, which can be quite high. This cost can be reduced drastically, to the point of becoming negligible relative to the overall cost, by employing a number of simple strategies. Foremost among these is the fact that the test need only be applied very infrequently.

A very simple technique is to monitor the eigenvalues of the tridiagonal matrix T_i at fixed intervals, i.e. when $MOD(i, s) = 0$, where s is a stride. Of course, large values of s will induce infrequent convergence tests, thus reducing the cost from $O(m^3)$ to $O(\frac{m^3}{3s})$. On the other hand, a large stride may inflict

Charge Densities with Partial Lanczos

*(*Input*)* Matrix $A \in \mathbb{R}^{n \times n}$, starting vector q_1, $\|q_1\|_2 = 1$,
scalars m, n_0, $s \geq n_0$, tolerance `tol`
*(*Output*)* Charge densities $\rho(r_i)$, $i = 1, \ldots, n$

1. Set $\beta_1 = 0$
2. Set $l = 1$, $\Sigma_{old} = 0$
3. **repeat**
4. Perform Partial Lanczos steps $l : l + s$ (lines 3-10, Figure 3)
5. Compute **only** the n_0 smallest eigenvalues of μ_j of T_{l+s}
6. Sort eigenvalues μ_j in ascending order and compute $\Sigma_{new} = \sum_{j=1}^{n_0} \mu_i$
7. $l = l + s + 1$
8. if $|(\Sigma_{new} - \Sigma_{old})/\Sigma_{old}| \leq$ `tol` **then** `goto` 11
9. $\Sigma_{old} = \Sigma_{new}$
10. **until** $(l > m)$
11. Compute the eigenvectors V_{n_o} corresponding to the n_0 algebraically smallest eigenvalues μ_j of T_l using inverse iteration
12. Approximate charge densities $\rho(r_i)$ by the squares of the norms of the rows of matrix $Q_l V_{n_o}$

Figure 5: Algorithm for approximating charge densities by means of Partial Lanczos.

unnecessary $O(s)$ additional Lanczos steps before convergence is detected.

Though not implemented in this paper, a better strategy is to use the bisection algorithm (see [11] Sec. 8.5) to track the latest eigenvalue that has converged. Here the important property that is exploited is the following: the Lanczos procedure is a variational technique in the sense that when an eigenvalue converges, later steps can only improve it. In addition, convergence tends to occur from left to right in the spectrum, meaning that typically the smallest eigenvalue converges first followed by the second smallest, etc. This suggests a simple procedure based on the bisection algorithm. We will first track the smallest eigenvalue until it converges, then the second until it converges, etc., until the n_o-th eigenvalue converges. Computing the i-th eigenvalue by a bisection procedure incurs only an $O(m)$ cost. Again, the checking can be performed infrequently. In addition, as the procedure progresses, the initial interval to bisect becomes quite small as previous information can be exploited to narrow it.

When convergence has been detected (say at step $l \leq m$) then the charge densities are approximated as the squares of the norms of matrix $Q_l U_{n_o}$. Figure 5 provides an algorithmic description of the proposed technique. Line 4 is essentially the Lanczos algorithm with partial reorthogonalization (see Figure 3).

3.1 Implementation issues

Although the final length $l \leq m$ of the Lanczos basis Q_l can become much larger than the number n_o of desired eigenvalues, we only require the eigenvectors that correspond to these eigenvalues. This can be accomplished with a cost $O(n_o l)$ using inverse iteration on matrix $(T_l - \mu_j I)$, $j = 1, \ldots, n_0$ (see [24], routine `xSTEIN`).

Clearly, virtually all of the memory required by the proposed algorithm is consumed in storing the Lanczos basis Q_l. Thus, when computing the norms of the rows of matrix $Q_l V_{n_o}$ we would like to avoid computing the latter matrix explicitly. Observe that we could compute each row at a time, thus essentially minimizing the amount of the required additional memory. In particular, the $i-$th row of Q_l is multiplied from the right by matrix V_{n_o}. This is a `BLAS 2` type operation. However, with a modest increase in the required memory, a more efficient `BLAS 3` implementation is also possible. We partition the basis Q_l row-wise in $\lfloor \frac{n}{l} \rfloor$ chunks of dimension $l \times l$ (see Figure 6). Then, at each step l rows of matrix $Q_l V_{n_o}$ are computed.

We mentioned earlier the possibility to develop an out-of-core code, whereby the Lanczos vectors are

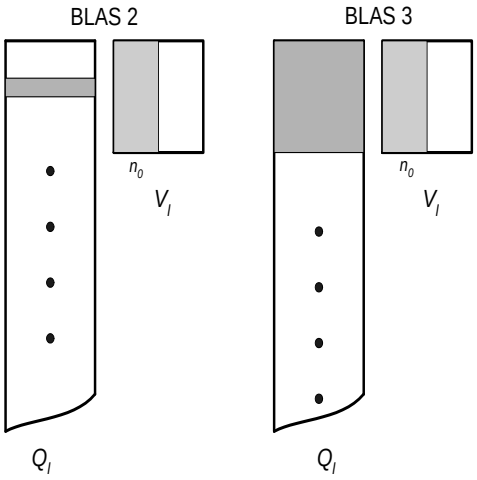

Figure 6: `BLAS 2` (left) and `BLAS 3` (right) implementations for computing the rows of matrix $Q_l V_{n_o}$.

saved periodically to secondary storage and recalled only when needed in a reorthogonalization step. Our current implementation is not out-of-core. Rather, all Lanczos basis vectors are always kept in main memory. As it will be shown in the experiments section that follows, in the test cases that we considered (involving Hamiltonians of dimension around $n = 10^5$ and $n_0 \leq 496$ eigenvalues) memory requirements never exceeded 5 `GBytes`. These memory requirements can be considered to be well within the range of standard high performance workstations and certainly rather small for high end systems, similar to the ones housed at MSI[1]. However, for much larger Hamiltonians (say $n = O(10^6)$) and thousands of eigenvalues, we can anticipate a ten-fold increase in memory requirements. For such cases, an out-of-core implementation of the proposed method will be mandatory.

4 Experiments

We are interested in testing the proposed technique for a wide range of Hamiltonian sizes as well as for a large number of computed eigenvalues. Note that for a given Hamiltonian H with size n, if the number of occupied states n_o of the molecular system under study is large, the dimension m of the Krylov basis has to become also large. This will inevitably cause an increase in the number of required reorthogonalizations, as well as increased memory requirements for storing the Krylov basis. We designed the following experiments in order to explore the limits of the algorithm and compare it with a state-of-the-art eigenvalue method such as `ARPACK` [8].

The test systems are silicon and germanium clusters with 10 to 100 atoms. They are constructed as undistorted fragments of the bulk structures, and therefore retain tetrahedral coordination. The cluster is passivated by attaching hydrogen atoms. Optical properties of these compounds are well known from theoretical analysis [4, 25], and they are ideal for the study of small quantum dots. We solve the Kohn-Sham equations on a regular grid in real space and write the effective Hamiltonian as a $n \times n$ matrix $H_{ij} = H(r_i, r_j)$ [17]. The grid has n points. Finite difference expansion of the Laplace operator and the limited range of non-local terms in the pseudopotentials ensure that H_{ij} is a highly sparse matrix.

All experiments were run on an `SGI Origin 2000` system, equipped with 12 `MIPS R12000` processors. Each processor had `8 MB` cache and the total system memory was `23 GB`. Our experiments run on one cpu and in single user mode. The operating system was `IRIX 6.5`. `PLAN`[2] is written in `FORTRAN 77` as is `ARPACK`[3]. The compilation flags were `-TARG:platform=IP27 -r12000 -64 -O3`.

[1] Minnesota Supercomputing Institute (`http://www.msi.umn.edu`).

[2] `http://crd.lbl.gov/~kewu/planso.html`

[3] `http://www.caam.rice.edu/software/ARPACK/`

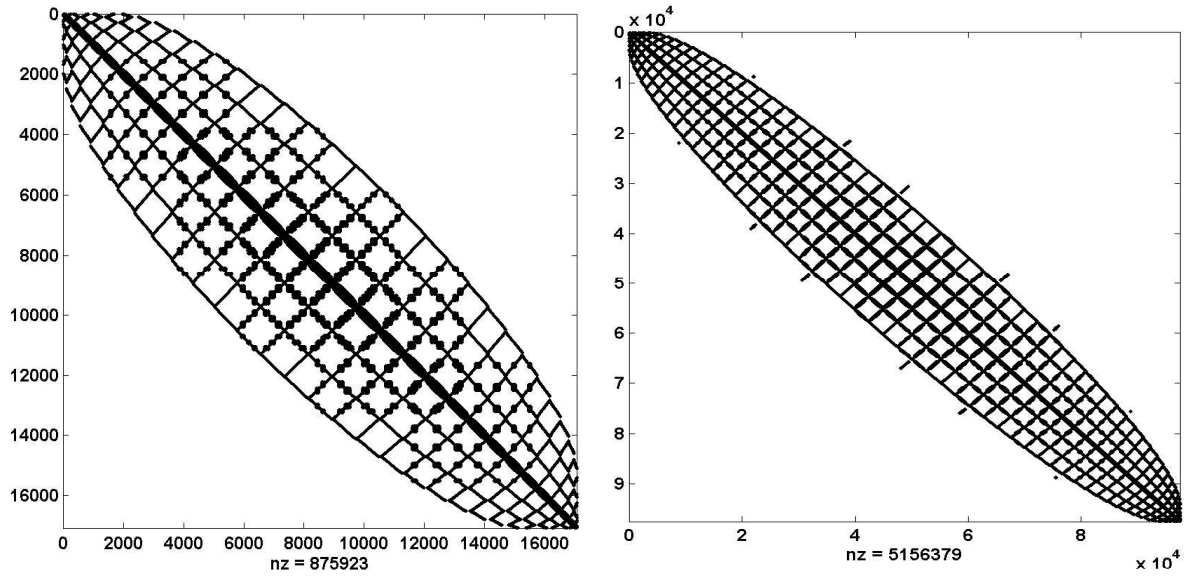

Figure 7: Left: Structure of Hamiltonian (n = 17077, nnz=875923) for $\mathtt{Si}_{10}\mathtt{H}_{16}$. Right: Structure of Hamiltonian (n = 97569, nnz=5156379) for $\mathtt{Si}_{34}\mathtt{H}_{36}$.

	Partial Lanczos				ARPACK			
n_o	MATVECS	REORTHS.	MEMORY MB	secs	MATVECS	RESTARTS	MEMORY MB	secs
28	539	16	72	24	592	27	7.5	58
50	930	35	124	61	1039	31	13.3	187
150	1940	97	259	273	2129	21	40	1111
200	2190	114	292	360	2676	20	53	1847

Table 1: Costs for Partial Lanczos and ARPACK for $\mathtt{Si}_{10}\mathtt{H}_{16}$.

In order to select the dimension of the Lanczos basis in ARPACK we conducted a parametric study, experimenting with different sizes. In line with the choice widely used by users of ARPACK we found that this dimension is best set to twice the number of sought eigenvalues. We point out that this choice is also adopted in the popular MATLAB routine eigs, which is an mex interface to the ARPACK library. On the other hand, for PLAN we let the Lanczos basis to increase until convergence was achieved.

$\mathtt{Si}_{10}\mathtt{H}_{16}$ For this cluster ($\mathtt{Si}_{10}\mathtt{H}_{16}$) the number of required (algebraically smallest) eigenvalues is neig=28. Furthermore we conducted experiments for $n_0 = 50, 150$ and 200.

$\mathtt{Si}_{30}\mathtt{H}_{36}$ For this cluster ($\mathtt{Si}_{30}\mathtt{H}_{36}$) the number of required (algebraically smallest) eigenvalues is neig=86. Furthermore we conducted experiments for $n_0 = 100, 150$ and 200.

$\mathtt{Ge}_{87}\mathtt{H}_{76}$ The number of occupied states for ($\mathtt{Ge}_{87}\mathtt{H}_{76}$) is neig=212. Furthermore we conducted experiments for $n_0 = 300$ and 424.

$\mathtt{Ge}_{99}\mathtt{H}_{100}$ The number of occupied states for ($\mathtt{Ge}_{99}\mathtt{H}_{100}$) is neig=248. Furthermore we conducted experiments for $n_0 = 350$ and 496.

Figures 7 and 8 illustrate the sparsity pattern of the Hamiltonians for the above clusters.

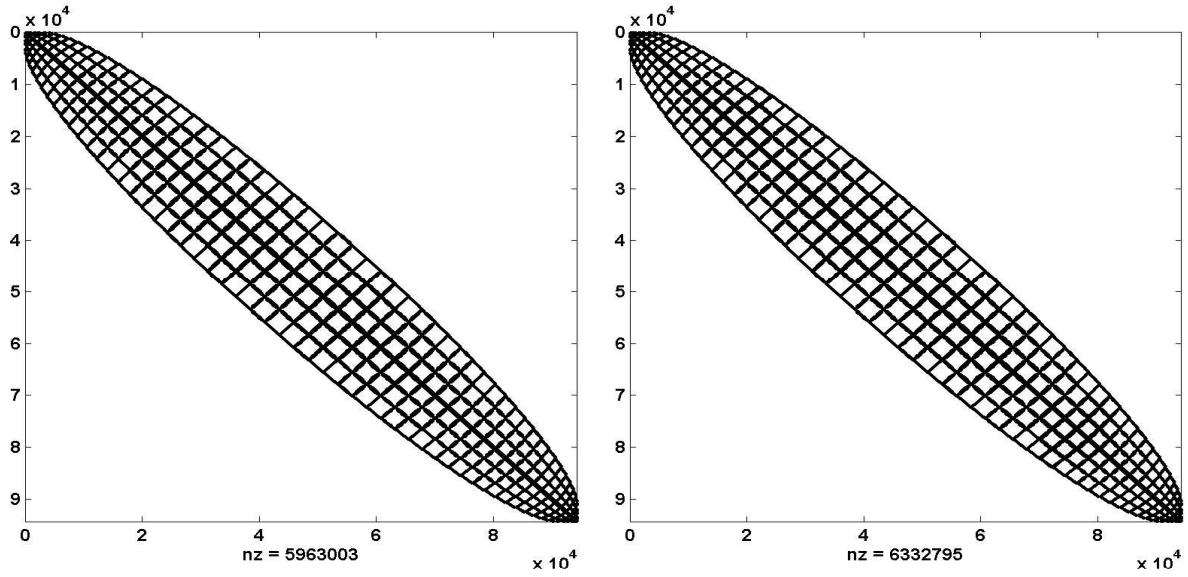

Figure 8: Left: Structure of Hamiltonian (n = 94341, nnz=5963003) for the $Ge_{87}H_{76}$ cluster. Right: Structure of Hamiltonian (n = 94341, nnz=6332795) for the $Ge_{99}H_{100}$ cluster.

	Partial Lanczos				ARPACK			
n_o	MATVECS	REORTHS.	MEMORY MB	secs	MATVECS	RESTARTS	MEMORY MB	secs
86	1440	36	1098	605	1537	24	131	2877
100	1810	50	1380	907	2164	32	152	4800
150	2880	96	2195	2191	3085	32	229	9993
200	3580	129	2729	3431	3803	30	305	16099

Table 2: Costs for Partial Lanczos and ARPACK for $Si_{34}H_{36}$.

4.1 Discussion

Tables 1-4 illustrate the results. For PLAN we tabulate the number of Matrix-Vector products (MATVECS), the number of reorthogonalizations (REORTHS), the amount of main memory (in MBytes) and the run times (in seconds). It it reminded that the dimension of the Lanczos basis for PLAN is equal to the number of MATVECS. For ARPACK we tabulate the number of Matrix-Vector products (MATVECS), the amount of main memory (in MBytes), the number of restarts of the method and the run times (in seconds).

Concerning run times PLAN is roughly 4.5-5.5 times faster than ARPACK. The difference in performance is particularly evident when large number of eigenvalues are requested. It is important to observe that although there exists a difference in the number of MATVECS between PLAN and ARPACK, in favor of the former, this is small and cannot account for the overall difference in performance.

The key is the small number of reorthogonalizations required by PLAN in all test cases (at least twenty times less than the dimension of the Lanczos basis). Observe that although the dimension of the Lanczos basis for PLAN is significantly larger than the one used by ARPACK (roughly 6-9 times larger across all test cases) the Lanczos basis vectors are rarely required by PLAN, due to the small number of reorthogonalizations. On the other hand, in ARPACK all basis vectors are needed at each restart. Therefore, in our test cases PLAN incurs far less traffic to memory than ARPACK, which accounts for the difference in performance.

On the other hand, we must point out that should a large number of reorthogonalizations be required in PLAN, we can expect the difference in performance between ARPACK and PLAN to be diminished or even reversed. However, our experience with a variety of Hamiltonians, arising from real space discretizations of different clusters, indicates that PLAN can achieve a five-fold increase of performance compared to ARPACK

	Partial Lanczos				ARPACK			
n_o	MATVECS	REORTHS.	MEMORY MB	secs	MATVECS	RESTARTS	MEMORY MB	secs
212	2710	88	1951	1993	2867	20	306	12145
300	4010	153	2887	4448	4673	25	432	28359
424	5740	252	4132	9804	6059	23	611	51118

Table 3: Costs for Partial Lanczos and ARPACK for $Ge_{87}H_{76}$.

	Partial Lanczos				ARPACK			
n_o	MATVECS	REORTHS.	MEMORY MB	secs	MATVECS	RESTARTS	MEMORY MB	secs
248	3150	109	2268	2746	3342	20	357	16454
350	4570	184	3289	5982	5283	24	504	37371
496	6550	302	4715	13714	6836	22	714	67020

Table 4: Costs for Partial Lanczos and ARPACK for $Ge_{99}H_{100}$.

across these cases.

5 Conclusion

By focussing on obtaining a good orthonormal basis of the subspace associated with the n_o lowest eigenvalues of the Hamiltonian rather than individual eigenvectors, the method discussed in this paper can lead to a significant reduction in arithmetic cost. In a few sample cases, the execution time was close to 6 times lower than that obtained by a standard implicitly restarted Lanczos procedure, as the one implemented in ARPACK. This can be achieved by using a Lanczos code with partial reorthogonalization.

One question that remains is whether or not it is possible to utilize the formalism exploited in this paper in a different approach that will not depend on a large Lanczos basis. It is clear that at minimum a basis of size n_o vectors is required. It is perfectly possible to perform the computation with exactly n_o vectors in memory. The algorithm for this is the well-known subspace iteration, [9, 14], see also [26] for related methods, based on minimizing traces. The reason this algorithm is no longer used in practice is that it is exceedingly slow. As can be seen the main issue is one of trading arithmetic cost for memory cost. In this paper we showed that if a sufficiently large memory is available, then the overall computational cost can be drastically reduced.

However, we should point out that not everything has been exploited to reduce memory costs further. Instead of using Lanczos directly on the Hamiltonian H we could, for example, apply it on $\pi(H)$ instead where π is a low degree polynomial whose goal is to speed-up convergence. Preliminary results with this approach are encouraging.

Acknowledgments This work would not have been possible without the availability of excellent source codes for diagonalization. Specifically, our experiments made use of the PLAN code developed by Wu and Simon [23] and the ARPACK code of Lehoucq, Sorensen, and Yang [8].

References

[1] M. C. Payne, M. P. Teter, D. C. Allan, T. A. Arias, and J. D. Joannopoulos. Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients. *Rev. Mod. Phys.*, 64:1045–1097, 1992.

[2] R. Car and M. Parrinello. Unified approach for molecular dynamics and density-functional theory. *Phys. Rev. Lett.*, 55:2471–2474, 1985.

[3] E. L. de la Grandmaison, S. B. Gowda, Y. Saad, M. L. Tiago, and J. R. Chelikowsky. Efficient computation of the coupling matrix in time-dependent density-functional theory. To appear in *J. Comput. Phys.*, 2005.

[4] I. Vasiliev, S. Öğüt, and J. R. Chelikowsky. First-principles density-functional calculations for optical spectra of clusters and nanocrystals. *Phys. Rev. B*, 65:115416, 2002.

[5] X.-P. Li, R. W. Nunes, and D. Vanderbilt. Density-matrix electronic-structure method with linear system-size scaling. *Phys. Rev. B*, 47:10891, 1993.

[6] S. Goedecker. Linear scaling electronic structure methods. *Reviews of Modern Physics*, 71:1085–1123, 1999.

[7] S. Goedecker and M. Teter. Tight-binding electronic-structure calculations and tight-binding molecular dynamics with localized orbitals *Phys. Rev. B*, 51:9455, 1995.

[8] R. Lehoucq, D. C. Sorensen, and C. Yang. ARPACK User's Guide: Solution of Large-Scale Eigenvalue Problems With Implicitly Restarted Arnoldi Methods. SIAM, Philadelphia, 1998.
URL `http://www.caam.rice.edu/software/ARPACK`.

[9] B. N. Parlett. The Symmetric Eigenvalue Problem. Prentice Hall, Englewood Cliffs, 1980.

[10] L. O. Jay, H. Kim, Y. Saad, and J. R. Chelikowsky. Electronic structure calculations using plane wave codes without diagonlization. *Comput. Phys. Comm.*, 118:21–30, 1999.

[11] G. H. Golub and C. F. Van Loan. Matrix Computations. The Johns Hopkins University Press, Baltimore, 3d edition, 1996.

[12] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. Nat. Bur. Standards*, 45:255–282, 1950.

[13] J. Cullum and R. A. Willoughby. Lanczos algorithms for large symmetric eigenvalue computations. Volumes 1 and 2. Birkhäuser, Boston, 1985.

[14] Y. Saad. Numerical Methods for Large Eigenvalue Problems. Halstead Press, New York, 1992.

[15] Y. Saad. SPASRKIT: A Basic Tool Kit for Sparse Matrix Computation, version2. 1994. Software available at `http://www.cs.umn.edu/~saad`.

[16] C. C. Paige. The Computation of Eigenvalues and Eigenvectors of Very Large Sparse Matrices. PhD Thesis, London University, London, England, 1971.

[17] J. R. Chelikowsky, L. Kronik, I. Vasiliev, M. Jain, and Y. Saad. Using Real Space Pseudopotentials for Electronic Structure Calculations. Volume 10 of Handbook of Numerical Analysis, Special Volume on Computational Chemistry, (C. Le Bris, Guest editor), pages 613–637. Elsevier Science B. V., 2003.

[18] G. W. Stewart Adjusting the Rayleigh Quotient in Semiorthogonal Lanczos Methods. *SIAM J. Scient. Comput.* 24(1):201-207, 2002.

[19] H. D. Simon. Analysis of the symmetric Lanczos algorithm with reorthogonalization methods. *Linear Algebra Appl.*, 61:101–132, 1984.

[20] R. M. Larsen. Efficient Algorithms for Helioseismic Inversion. PhD Thesis, Dept. Computer Science, University of Aarhus, DK-8000 Aarhus C, Denmark, October 1998.

[21] H. D. Simon. The Lanczos algorithm with partial reorthogonalization. *Math. Comp.*, 42(165):115–142, 1984.

[22] R. M. Larsen. PROPACK: A software package for the symmetric eigenvalue problem and singular value problems on Lanczos and Lanczos bidiagonalization with partial reorthogonalization, SCCM, Stanford University
URL: `http://sun.stanford.edu/~rmunk/PROPACK/`.

[23] K. Wu and H. Simon. A parallel Lanczos method for symmetric generalized eigenvalue problems. Technical Report 41284, Lawrence Berkeley National Laboratory, 1997.

[24] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Crop, A. Greenbaum, S. Hammerling, A. McKenney, and D. Sorensen. LAPACK Users' Guide. SIAM, Philadelphia, 3rd edition, 1999.

[25] L. X. Benedict, A. Puzder, A. J. Williamson, J. C. Grossman, G. Galli, J. E. Kelpies, J.-Y. Ratty, and O. Pandurate. Calculation of optical absorption spectra of hydrogenated Si clusters: Bethe-Salpeter equation versus time-dependent local-density approximation. *Phys. Rev. B*, 68:085310, 2003.

[26] A. Sameh and Z. Tong. The trace minimization method for the symmetric generalized eigenvalue problem. *J. Comput. Appl. Math.*, 123(1-2):155–175, 2000.