Available online at www.sciencedirect.com

ScienceDirect

Journal of Computational Physics 219 (2006) 172–184

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

# Self-consistent-field calculations using Chebyshev-filtered subspace iteration ☆

Yunkai Zhou [a,*], Yousef Saad [a], Murilo L. Tiago [b], James R. Chelikowsky [b]

[a] *Department of Computer Science and Engineering, University of Minnesota, 200 Union Street SE, Minneapolis, MN 55455, USA*
[b] *Institute for Computational Engineering and Sciences, University Station, University of Texas at Austin, Austin, TX 78712, USA*

Received 28 September 2005; received in revised form 1 March 2006; accepted 18 March 2006
Available online 22 May 2006

**Abstract**

The power of *density functional theory* is often limited by the high computational demand in solving an eigenvalue problem at each *self-consistent-field* (SCF) iteration. The method presented in this paper replaces the explicit eigenvalue calculations by an approximation of the wanted invariant subspace, obtained with the help of well-selected Chebyshev polynomial filters. In this approach, only the initial SCF iteration requires solving an eigenvalue problem, in order to provide a good initial subspace. In the remaining SCF iterations, no iterative eigensolvers are involved. Instead, Chebyshev polynomials are used to refine the subspace. The subspace iteration at each step is easily five to ten times faster than solving a corresponding eigenproblem by the most efficient eigen-algorithms. Moreover, the subspace iteration reaches self-consistency within roughly the same number of steps as an eigensolver-based approach. This results in a significantly faster SCF iteration.
© 2006 Elsevier Inc. All rights reserved.

*Keywords:* Density functional theory; Self-consistent-field; Chebyshev polynomial filter; Subspace iteration; Eigenproblem; Real-space pseudopotential

## 1. Introduction

Since its formulation, *density functional theory* (DFT) [18,21] has been recognized as a major achievement in the development of quantum many-body theories. Its basis lies in describing the ground state of a many-electron system solely in terms of the charge density function. As a result, the task of solving the *Schrödinger equation* for a many-electron system is replaced by the immensely simpler one of solving a nonlinear, self-consistent eigenvalue problem: the *Kohn–Sham equations*. During the last several decades, DFT has been successfully applied to a range of problems in condensed matter physics, material sciences, chemistry and biology [29,13]. In typical numerical implementations of DFT, the most time-consuming part is spent in computing the self-consistent

☆ Work supported by DOE under Grants DE-FG02-03ER25585 and DE-FG02-03ER15491, by NSF Grants ITR-0551195 and ITR-0428774, and by the Minnesota Supercomputing Institute.
* Corresponding author.
*E-mail addresses:* zhou@msi.umn.edu (Y. Zhou), saad@cs.umn.edu (Y. Saad), mtiago@ices.utexas.edu (M.L. Tiago), jrc@ices.utexas.edu (J.R. Chelikowsky).

0021-9991/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2006.03.017

solution of the Kohn–Sham equations. Because of the high computational demand of matrix diagonalizations when the number of wanted eigenvalues as well as the matrix dimension are large, applying DFT to very large system (e.g., molecules containing thousands of atoms) still remains a highly challenging problem.

It has long been realized that full diagonalization is too expensive for large problems. Many researchers have considered avoiding matrix diagonalization or reducing its cost. Typical approaches that have been explored include the use of the conjugate gradient (CG) method which directly minimizes the total energy (or Rayleigh-quotient) [30], the Car-Parrinello molecular dynamics method [7] and the DIIS variants [31,32,42,22] which minimize the residual vectors instead of Rayleigh-quotients.

Iterative eigensolvers which only compute the wanted eigenpairs have also been utilized to reduce the cost of diagonalization. In the early days of the development of DFT codes, the linear *subspace iteration* algorithm was used. For example, the 1991 paper [27], see also [26], uses a linear *subspace iteration* algorithm similar to *Ritzit* [33], but the Chebyshev acceleration in *Ritzit* was replaced by a modified Jacobi preconditioner. In contrast with [27], the classical Ritzit code (with Chebyshev acceleration) was found to be an excellent partial diagonalization tool in the solution of self-consistent Schrödinger equations which arise in the simulation of electrons in quantum wires [20]. The main difference with our approach is that we do not use subspace iteration for computing eigenvalues and eigenvectors. Instead, our approach can be viewed as a nonlinear subspace iteration in that after each subspace filtering the Hamiltonian is updated.

More recent examples of the use of iterative diagonalization include the multigrid approaches [5,6,11,14,17,24] and preconditioned Davidson method [35,39]. We note that the complexity of the minimization-based methods is similar to that of the iterative eigensolver-based methods. The computational cost of an eigensolver approach is usually dominated by the cost of the matrix–vector products and that of maintaining orthogonality of the basis vectors. In [4], partial reorthogonalized Lanczos without restart is used in order to reduce the orthogonalization cost, at the expense of a higher memory requirement.

In this paper we present a method which avoids solving large eigenvalue problems explicitly. The method utilizes Chebyshev polynomial filtered subspace iteration. In this approach only the initial SCF iteration requires solving an eigenvalue problem, by means of any available efficient eigensolver. This step is used to provide a good initial subspace. Because the subspace dimension is slightly larger than the number of wanted eigenvalues (denoted by $k_{\mathrm{want}}$), the method does not require as much memory as standard restarted eigensolvers such as ARPACK (see [38,25]) and TRLan (Thick-Restart Lanczos) [43,44]. Moreover, the cost of orthogonalization is much reduced. This is because the new approach only requires a subspace of dimension around $k_{\mathrm{want}}$, and the orthogonalization is done only once per SCF iteration. In contrast, standard eigensolvers using restart usually require a subspace of dimension $2k_{\mathrm{want}}$ in order to compute $k_{\mathrm{want}}$ eigenpairs efficiently, the orthogonalization cost approximately amounts to orthogonalizing $2\ k_{\mathrm{want}}k_{\mathrm{restart}}$ number of vectors, where $k_{\mathrm{restart}}$ is the number of restarts needed in order to converge the eigenvectors.

Chebyshev polynomial filtering has been utilized in electronic structure calculations (see e.g. [36,40,16,2,19]), where the focus is on approximating the Fermi-Dirac operator, i.e., Chebyshev polynomials only over interval $[-1, 1]$ is considered. The approach we take here is different from the existing Chebyshev methods in electronic structure calculations. The fundamental difference is that we exploit the exponential growth property of Chebyshev polynomial outside the $[-1, 1]$ interval, hence the polynomial degree required is much lower. In our approach, we never map the full spectrum of the Hamiltonian into $[-1, 1]$; instead, we adaptively decide on the unwanted part of the spectrum and map only this part into $[-1, 1]$ for damping it. Our Chebyshev filtering is based on [45]. Note that the focus in [45] is on eigenvalue computations while here we only use Chebyshev polynomial to filter subspaces. More details are presented in Section 4.

The current approach has been implemented in PARSEC, our real-space DFT code based on [10,9], and it has been observed to be significantly faster than iterative eigensolver-based approaches. As an example, it took less than 2 hours for the sequential implementation of the new method to reach self-consistency for the Silicon cluster $Si_{525}H_{276}$ *on a single* SGI 1.3 GHz Madison processor. In contrast, a parallel calculation done with the preceding version of PARSEC around the year 1997, on the CRAY T3E, took a total of 20 hours [39] *using 48 processors*. Back in 1997 the problem could not be solved in fewer than 48 processors due to memory requirements. The remarkable gains do not come only from a reduced cost of diagonalization but also from exploiting symmetry [41] and from gains made in other parts of the code. Section 5 presents comparisons with methods based on two of the most efficient eigensolvers currently available (ARPACK and TRLan). All three methods utilize the same symmetry operations, therefore the reported speedup is from the Chebyshev filtering approach.

## 2. Basics of self-consistent-field calculation

In this section we briefly review the self-consistent-field calculation. Here we focus discussion on SCF in DFT calculations, but we note that SCF is also used in Hartree-Fock and other approximations. As mentioned in the introduction, the most time-consuming part of an SCF calculation is in matrix diagonalization, which consists of computing the self-consistent solutions of the following Kohn–Sham equation (in atomic units):

$$\left[-\frac{\nabla^2}{2} + V_{\text{total}}(\rho(r), r)\right]\Psi_i(r) = E_i \Psi_i(r), \tag{2.1}$$

where $\Psi_i(r)$ is a wave function, $E_i$ is a Kohn–Sham eigenvalue. The total potential

$$V_{\text{total}}(\rho(r), r) = V_{\text{ion}}(r) + V_{\text{H}}(\rho(r), r) + V_{\text{XC}}(\rho(r), r) \tag{2.2}$$

includes the ionic potential $V_{\text{ion}}$, the Hartree potential $V_{\text{H}}$ and the exchange–correlation potential $V_{\text{XC}}$. In DFT the total potential depends only on $\rho(r)$—the *charge density*. The *charge density* is given by

$$\rho(r) = 2\sum_{i=1}^{n_{\text{occ}}} |\Psi_i(r)|^2, \tag{2.3}$$

where $n_{\text{occ}}$ is the number of occupied states (half the number of valence electrons in the system) and the factor of two comes from spin multiplicity. Eq. (2.3) can be easily extended to situations where the highest occupied states have fractional occupancy or when there is an imbalance in the number of electrons for each spin component.

Because the potential (2.2) depends on the charge density (2.3), which in turn depends on eigenfunctions of the Hamiltonian in Eq. (2.1), the eigenvalue problem (2.1) is in fact nonlinear. Self-consistent iterations for solving this problem consist of starting with an initial guess of the charge density $\rho_0(r)$, then obtaining a guess for $V_{\text{total}}$ and solving (2.1) for $\Psi_i(r)$'s to update $\rho(r)$ and $V_{\text{total}}$. Then (2.1) is solved again for the new $\Psi_i(r)$'s and the process is carried on until the difference between two consecutive $V_{\text{total}}$'s is below a certain tolerance (equivalently, the wave functions are close to stationary). Algorithm 2.1 contains a pseudo code for this SCF loop.

We note that, since the charge density does not depend on eigenstates beyond $n_{\text{occ}}$, the number of eigenvectors needed in Step 2 of Algorithm 2.1 is limited. Nevertheless, the eigenvalue problem is still very challenging in complex systems (i.e., systems with a very large number of electrons), when the Hamiltonian has large dimension and $n_{\text{occ}}$ is also large.

Our computational code uses a real-space implementation of the above SCF method [10,35,39]. In this implementation, wave functions are expressed directly as functions of position, and they are required to vanish outside a specified boundary that encloses the physical system (alternatively, periodic boundary conditions can be imposed [1]). The region inside this boundary is discretized by using a regular grid with adjustable spacing between neighbouring points. We use pseudopotentials to describe the interaction between valence electrons and ionic cores (core electrons + nuclei) and solve the SCF problem for valence electrons only. In addition, we make use of symmetry operations in the arrangement of atoms and reduce the sampled region to a smaller one: the "irreducible wedge". Appropriate boundary conditions and the existing symmetry operations are used to expand wave functions from the irreducible wedge to the full volume. For highly symmetric systems, such as the atom clusters analyzed in Section 5, reducing the volume of interest to an irreducible wedge can easily lead to a 10-fold reduction in the computational load [41] than without exploiting symmetry.

**Algorithm 2.1** (*Self-Consistent Iteration*)

```
1. Initial guess for $\rho(r)$, get $V_{\text{total}}(\rho(r), r)$.
2. Solve $\left[-\frac{\nabla^2}{2} + V_{\text{total}}(\rho(r), r)\right]\Psi_i(r) = E_i\Psi_i(r)$ for $\Psi_i(r)$, $i = 1, 2, \ldots$
3. Compute new charge density $\rho(r) = 2\sum_{i=1}^{n_{\text{occ}}}|\Psi_i(r)|^2$.
4. Solve for new Hartree potential $V_{\text{H}}$ from $\nabla^2 V_{\text{H}}(r) = -4\pi\rho(r)$.
```

5. Update $V_{\mathrm{XC}}$ and $V_{\mathrm{ion}}$; get new $\tilde{V}_{\mathrm{total}}(\rho, r) = V_{\mathrm{ion}}(r) + V_{\mathrm{H}}(\rho, r) + V_{\mathrm{XC}}(\rho, r)$ (often followed by a potential-mixing step).
6. If $\|\tilde{V}_{\mathrm{total}} - V_{\mathrm{total}}\| < \mathrm{tol}$, stop; Else, $V_{\mathrm{total}} \leftarrow \tilde{V}_{\mathrm{total}}$, goto 2.

## 3. Self-consistent-field calculation without explicit eigenvectors

While the charge density $\rho(r)$ depends explicitly on the wave functions $\Psi_i(r)$, see (2.3), it has been observed that any basis for the subspace spanned by these eigenvectors can be used. After discretization each $\Psi_i(r)$ becomes an eigenvector (denoted as $\psi_i$) of the discretized Hamiltonian. The charge density is then simply the diagonal of the density matrix

$$P = \Phi\Phi^{\mathrm{T}}, \tag{3.1}$$

in which $\Phi$ is the matrix with column vectors the $\{\psi_i\}$'s. Entry $(j,j)$ of this projector $P$ is equal to the charge density at the mesh-point $r_j$. Notice that for any orthonormal matrix $Q$ of a suitable dimension, $P = (\Phi Q)(\Phi Q)^{\mathrm{T}}$. Explicit eigenvectors are therefore not needed to calculate the charge density in Algorithm 2.1. Any orthonormal basis of the eigensubspace corresponding to occupied states will give the correct charge density.

Techniques based on this observation have appeared in, e.g. [40,37,19,4,3]. In particular, the recent article [4] stresses the importance of de-emphasizing eigenvectors in favor of the underlying eigenspace. This observation has also been exploited in the linear scaling approaches to DFT, see e.g. [15] for a survey.

In our approach we progressively refine a subspace by rather low degree Chebyshev polynomials. After self-consistency is reached, this subspace includes the eigensubspace corresponding to occupied states. Explicit eigenvectors can be easily obtained by a *Rayleigh–Ritz refinement* step [28] (called a *subspace rotation* in materials science terminology).

## 4. Chebyshev-filtered subspace iteration for SCF calculations

The main idea of the proposed approach is to start with a good initial eigen-basis $V$ corresponding to occupied states of the initial Hamiltonian $H_0$, and then to adaptively improve the subspace by polynomial filtering. That is, at a given self-consistent step, a degree-$m$ polynomial filter $p_m(t)$ is constructed for the current Hamiltonian $H$. Note that the polynomial will be different at each SCF step since $H$ will change. The goal of the filter is to make the subspace spanned by $p_m(H)V$ approximate the eigensubspace corresponding to the occupied states of $H$. There is no need to make $p_m(H)V$ approximate the wanted eigensubspace of $H$ to high accuracy at the intermediate steps. Instead, the filtering is designed so that the new subspace obtained at each self-consistent iteration step will progressively approximate the wanted eigensubspace of the final Hamiltonian when self-consistency is reached. This can be efficiently achieved by exploiting the Chebyshev polynomials, specifically the fast growth property outside the $[-1, 1]$ interval. All that is required to obtain a good filter at a given SCF step, is to provide a lower bound and an upper bound of an interval of the spectrum of the current Hamiltonian $H$ in which we want $p_m(t)$ to be small. Moreover, the lower bound can be readily obtained from the Ritz values computed from the previous step, and the upper bound can be inexpensively obtained by a very small number of (say, 4 or 5) Lanczos steps [23]. Hence the main cost of the filtering at each iteration is in computing $p_m(H)V$. This computation is accomplished by exploiting the convenient three-term recurrence property of Chebyshev polynomials.

### 4.1. The main algorithms

The Chebyshev-filtered subspace iteration algorithm for SCF calculations mainly depends on the following Chebyshev-filtered subspace method presented in Algorithm 4.1.

**Algorithm 4.1** (*Chebyshev-filtered subspace method*)

1. Get the lower bound $b_{\mathrm{low}}$ from previous Ritz values (use the largest one).
2. Compute the upper bound $b_{\mathrm{up}}$ of the spectrum of the current discretized Hamiltonian $H$ (call Algorithm 4.4).

3. Perform Chebyshev filtering (call Algorithm 4.3) on the previous basis $\Phi$, where $\Phi$ contains the discretized wavefunctions of $\Psi_i(r)$, $i = 1, \ldots, s$: $\Phi = \texttt{Chebyshev\_filter}(\Phi, m, b_{\text{low}}, b_{\text{up}})$.
4. Ortho-normalize the basis $\Phi$.
5. Perform the Rayleigh–Ritz step:
   (a) Compute $\hat{H} = \Phi^{\mathrm{T}} H \Phi$.
   (b) Compute the eigen-decomposition of $\hat{H} : \hat{H}Q = QD$, where $Q$ contains eigenvectors of the $\hat{H}$, $D$ contains non-increasingly ordered Ritz values of $H$.
   (c) 'Rotate' the basis: $\Phi := \Phi Q$.

The purpose of Algorithm 4.1 is to replace, except at the first SCF iteration, the eigenvalue problem at Step 2 in Algorithm 2.1 by a single Chebyshev filtering step. At the first SCF loop, Step 2 in Algorithm 2.1 is carried out by an iterative eigensolver like ARPACK or TRLan to provide an initial orthonormal basis $\Phi = [\psi_1, \ldots, \psi_s]$. In order not to miss occupied eigenstates, a standard practice is to compute $s$ eigenstates, where $s$ is an integer with $s > n_{\text{occ}}$. Here we follow this practice by fixing an integer $n_{\text{state}}$ which is slightly larger than $n_{\text{occ}}$, then set $s = n_{\text{state}} + n_{\text{add}}$ (typically $n_{\text{add}} \leqslant 10$).

Note that Algorithm 4.1 does not iterate over the subspace spanned by $\Phi$ apart from the single filtering step applied on $\Phi$, using a degree-$m$ Chebyshev polynomial. This is in contrast with the standard Chebyshev subspace iteration for computing eigenvalues, where another loop is applied to iterate on the subspace until accurate eigenvectors are extracted.

The estimated cost of Algorithm 4.1 with respect to the number of discretization points denoted by $N$, and the number of computed states, denoted by $s$ previously, is as follows:

- The Chebyshev filtering in Step 3 costs $\mathrm{O}(s * N)$ flops. The discretized Hamiltonian is sparse and each matrix–vector product costs $\mathrm{O}(N)$ flops. Step 3 requires $s * m$ matrix–vector products, at a total cost of $\mathrm{O}(s * m * N)$ where the degree-$m$ of the polynomial is small (typically between 8 and 20).
- The ortho-normalization in Step 4 costs $\mathrm{O}(N * s^2)$ flops. Note that $N$ is usually much larger than $s$ but that $N$ depends on $s$. In particular, note that $N$, the number of grid-points, must be increased proportionally to the number of atoms.
- The eigen-decomposition at Step 5 costs $\mathrm{O}(s^3)$ flops.
- The final basis refinement step $\Phi = \Phi Q$ costs $\mathrm{O}(N * s^2)$, which is identical with the cost of ortho-normalization.

Standard iterative diagonalization methods also require the ortho-normalization of a (typically larger) basis, the eigen-decomposition of the projected Rayleigh-quotient matrix, and the basis refinement. Therefore, Algorithm 4.1 scales in a similar way to standard eigenvalue-based methods. Specifically it is of the order of the cube of the number of atoms, since $N$ scales like $s$, which is proportional to the number of atoms. However, eigenvalue-based methods need to perform these same operations several times during the diagonalization process. In contrast, these operations need to be performed only once in Algorithm 4.2.

The complete Chebyshev-filtered subspace iteration (CheFSI) for SCF calculations is summarized in Algorithm 4.2. We note that the subspace iterations in Algorithm 4.2 correspond to the SCF iterations. Since the Hamiltonian changes with each iteration, we essentially perform a nonlinear subspace iteration over the basis of the initial subspace.

In other words, one can view Algorithm 4.2 as a simplification of standard SCF iterations, in which one loop has been omitted. A standard SCF method would have an outer SCF loop (the usual nonlinear SCF loop) and a diagonalization loop (iterate until eigenvectors are computed accurately). In a sense Algorithm 4.2 merges these 2 loops into a single one. When convergence takes place in the end, the basis $\Phi$ will indeed become an eigenvector basis. However, in the intermediate steps $\Phi$ will just be an evolving basis of some converging subspace. Algorithm 4.2 can thus be considered as *a nonlinear form of the subspace iteration algorithm*. In summary, although Algorithm 4.2 bears the same overall scaling behavior as standard methods, the scaling constant is likely to be much smaller.

**Algorithm 4.2** (*SCF loop with CheFSI*)

```
1. Initial guess for $\rho(r)$, get $V_{\rm total}(\rho(r), r)$.
2. Solve $\left[-\frac{\nabla^2}{2} + V_{\rm total}(\rho(r), r)\right]\Psi_i(r) = E_i\Psi_i(r)$ for $\Psi_i(r)$, $i = 1, 2, \ldots, s$.
3. Compute new charge density $\rho(r) = 2\sum_{i=1}^{n_{\rm occ}}|\Psi_i(r)|^2$.
4. Solve for new Hartree potential $V_{\rm H}$ from $\nabla^2 V_{\rm H}(r) = -4\pi\rho(r)$.
5. Update $V_{\rm XC}$ and $V_{\rm ion}$; get new $\tilde{V}_{\rm total}(\rho, r) = V_{\rm ion}(r) + V_{\rm H}(\rho, r) + V_{\rm XC}(\rho, r)$ (often followed by a potential-
   mixing step).
6. If $\|\tilde{V}_{\rm total} - V_{\rm total}\| <$ tol, stop; Else, $V_{\rm total} \leftarrow \tilde{V}_{\rm total}$ (update $H$ implicitly), call Algorithm 4.1 to get $s$ approx-
   imate wavefunctions; goto 3.
```

We now discuss some additional details of Algorithm 4.1. For the ortho-normalization step in Algorithm 4.1 (Step 4) we use the DGKS method [12] which uses iterated classical Gram-Schmidt.

Algorithm 4.1 does not compute an eigen-basis of the current Hamiltonian $H$. Instead, it computes an eigen-decomposition of the projected Rayleigh-quotient matrix $\hat{H}$ of size $s$. Even though $s$ is normally much smaller than the size of $H$, the Rayleigh–Ritz refinement (Step 5) may still be expensive when $s$ becomes very large (e.g., large complex systems without physical symmetry), since the eigen-decomposition step is of complexity $O(s^3)$.

An important feature of the Chebyshev-filtered subspace method is that Step 5 in Algorithm 4.1 can be omitted for huge systems where $s$ is very large. The reason for this is that $\Phi Q$ spans the same subspace as $\Phi$. In this case we can construct another suitable filter so that the components of $\Phi$ corresponding to unoccupied states will be filtered out. Rayleigh–Ritz refinement is performed when $s$ is moderate because it is not expensive, and this refinement makes columns in $\Phi Q$ approximate the eigenvectors of $H$ better than $\Phi$ does. That is, the first $n_{\rm occ}$ columns in $\Phi Q$ that correspond to the $n_{\rm occ}$ smallest Ritz values can be readily returned to the main program for the calculation of $\rho(r)$ to continue the SCF loop. Results in the literature (e.g. [22,11]) for different approaches also show that *subspace rotation* improves stability and convergence rate. In our approach, the Rayleigh–Ritz step is also helpful because it provides a convenient lower bound for the next Chebyshev filtering.

If the Rayleigh–Ritz step is not performed, the wanted lower bound can still be estimated from the largest Rayleigh-quotient among $\psi_j^{\rm T}H\psi_j$, where $\psi_j$ $(j = 1, \ldots, s)$ is the column vector of $\Phi$. Thanks to the Courant-Fisher min–max theorem [28, p.206] this will still be a good approximation for the filtering to work well.

The next section will show useful properties of Chebyshev polynomials and discuss how to adaptively change these filters.

### *4.2. Controlling the Chebyshev polynomial filters*

The well-known Chebyshev polynomials of the first kind are defined by ([28, p. 371], [34, p. 142])

$$C_k(t) = \begin{cases} \cos(k\cos^{-1}(t)), & -1 \leqslant t \leqslant 1, \\ \cosh(k\cosh^{-1}(t)), & |t| > 1. \end{cases}$$

Note that $C_0(t) = 1, C_1(t) = t$. The following important 3-term recurrence is easy to derive from properties of $\cos(t)$ and $\cosh(t)$,

$$C_{k+1}(t) = 2tC_k(t) - C_{k-1}(t), \quad t \in \mathbb{R}. \tag{4.1}$$

The rapid growth property of the Chebyshev polynomial outside $[-1, 1]$ is discussed in [28]. Fig. 4.1 illustrates this property. Here we only plot the $[-2, 1]$ interval, but note that the farther away from $-1$ or 1, the larger the magnitude of the real part of the Chebyshev polynomials.

Assume that the full spectrum of $H$ (denoted as $\sigma(H)$) is contained in $[a_0, b]$. As is well known [33,28,34,45], in order to approximate the eigensubspace associated with the lower end of the spectrum, say $[a_0, a]$ with $a_0 < a < b$, essentially one only needs to map $[a, b]$ into $[-1, 1]$. This can be easily realized by a linear mapping defined as

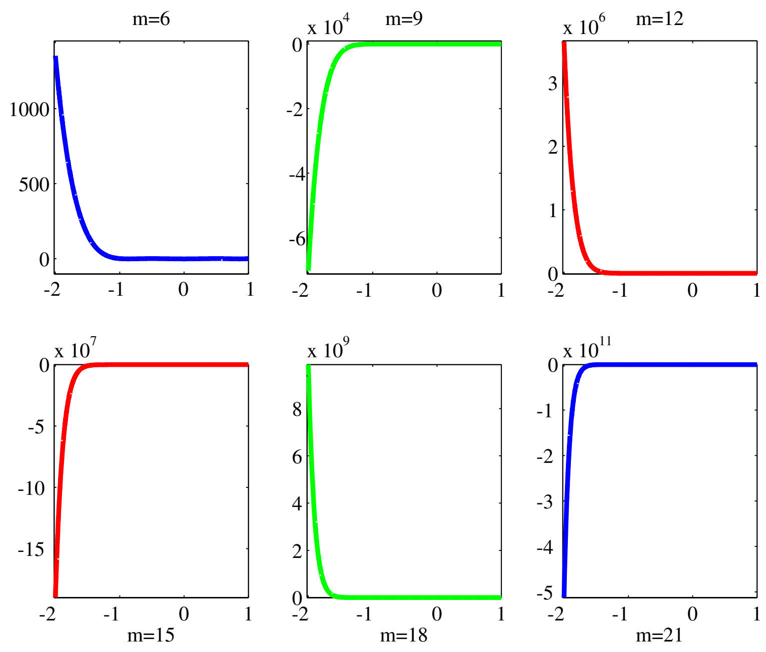

Fig. 4.1. Rapid increase outside $[-1,1]$ of the $m$th degree Chebyshev polynomial.

$$l(t) := \frac{t-c}{e}; \qquad c = \frac{a+b}{2}, \quad e = \frac{b-a}{2},$$

where $c$ denotes the center and $e$ the half-width.

The Chebyshev iteration, which utilizes the three-term recurrence (4.1) with the goal to dampen values on a given $[a,b]$ is presented in Algorithm 4.3. Here the formula derived in [34, p. 223] for the complex Chebyshev iteration is adapted to the real case. The iteration of the algorithm is equivalent to computing

$$Y = p_m(H)X, \quad \text{where } p_m(t) = C_m\left[\frac{t-c}{e}\right]. \tag{4.2}$$

Although Algorithm 4.3 only explicitly filters the $[a,b]$ interval, we note that by the property of the Chebyshev polynomial, the filter values on the interval to the left of $[a,b]$ will be magnified—which is what is needed to approximate the eigensubspace associated with the lower end of $\sigma(H)$.

**Algorithm 4.3** ($[Y] =$ `Chebyshev_filter`$(X, m, a, b)$)

*Purpose*: Filter vectors in $X$ by an $m$ degree Chebyshev polynomial that dampens on the interval $[a,b]$, and output the filtered vectors in $Y$.

```
1. $e = (b-a)/2$; $c = (b+a)/2$;
2. $\sigma = e/(a-c)$;
3. $\sigma_1 = \sigma$;
4. $Y = (HX - cX)\sigma_1/e$;
5. For $i = 2{:}m$
6.     $\sigma_2 = 1/(2/\sigma_1 - \sigma)$;
7.     $Y_{\text{new}} = 2(HY - cY)\sigma_2/e - \sigma\sigma_2 X$;
8.     $X = Y$;
9.     $Y = Y_{\text{new}}$;
10.    $\sigma = \sigma_2$;
11. End For
```

As seen from Algorithm 4.3, a desired filter can be easily controlled by providing two endpoints of the higher end of $\sigma(H)$. The higher endpoint can be estimated by a few steps of standard Lanczos (which is presented in Algorithm 4.4). A lower bound of the full $\sigma(H)$ is not needed. Instead, the wanted lower bound is any value which is larger than the Fermi-level but smaller than the higher endpoint. As presented in Algorithm 4.1, this lower bound is readily available as the largest Rayleigh-quotient of the previous iteration.

Hence there is negligible extra work associated with computing bounds for the Chebyshev filtering, the major cost being in the three-term recurrences in Algorithm 4.3 which involve matrix–vector products. The polynomial degree-$m$ is left as a free parameter. Our experience indicates that an $m$ between 8 and 20 is good enough to achieve overall fast convergence in the SCF loop.

### *4.3. Estimating an upper bound of the spectrum*

Now we present an inexpensive way to estimate an upper bound of $\sigma(H)$. As pointed out in [45], the upper bound has to bound the full spectrum of $H$. This is because the Chebyshev polynomial also grows fast on the right of $[-1,1]$. So if $[a,b]$ with $b < \sigma_{\max}(H)$ is mapped into $[-1,1]$, then the $[b,\sigma_{\max}(H)]$ portion of the spectrum will also be magnified, which will cause the whole procedure to fail. We need $b$ to be larger than $\sigma_{\max}(H)$ but it cannot be too large as this will affect convergence. There are several ways to estimate this type of upper bound, for example, by using the one-norm of $H$, or by applying Gerschgorin's Circle Theorem if possible. Bounds obtained this way can, however, overestimate $\sigma_{\max}(H)$.

We found that a few steps of the standard Lanczos procedure are sufficient to provide an effective upper bound. For example, a $k$-step Lanczos leads to a Lanczos-decomposition

$$HV_k = V_kT_k + f_ke_k^{\mathrm{T}},$$

where $V_k$ contains the $k$ Lanczos basis, $T_k$ is a size-$k$ tridiagonal matrix, $f_k$ is a residual vector and $e_k$ is a length $k$ unit vector with only the first element nonzero.

Notice that Lanczos iteration often quickly approximate the outermost eigenvalues, and that

$$\|HV_k\|_2 = \|V_kT_k + f_ke_k^{\mathrm{T}}\|_2 \leqslant \|T_k\|_2 + \|f_k\|_2.$$

We can start with a random unit vector, carry out $k$ steps of the Lanczos procedure, and use $\|T_k\|_2 + \|f_k\|_2$ as an upper bound for $\sigma(H)$. This is presented in Algorithm 4.4. For simplicity we skip the safeguards for $\beta = 0$, as this does not happen for small values of $k$ in general.

In practice, we found that $k = 4$ or $k = 5$ is sufficient to yield a proper upper bound of $\sigma(H)$. In fact we found that it is often counter-productive to take the bounds obtained from larger values of $k$ (say, $k > 10$).

**Algorithm 4.4** (*Estimating an upper bound of* $\sigma(H)$ by $k$-step Lanczos)

```
 1. Generate a random vector $v$, set $v \leftarrow v/\|v\|_2$.
 2. Compute $f = Hv$; $\alpha = f^T v$; $f \leftarrow f - \alpha v$; $T(1,1) = \alpha$.
 3. Do $j = 2$ to $\min(k,10)$
 4.    $\beta = \|f\|_2$;
 5.    $v_0 \leftarrow v$; $v \leftarrow f/\beta$;
 6.    $f = Hv$; $f \leftarrow f - \beta v_0$;
 7.    $\alpha = f^T v$; $f \leftarrow f - \alpha v$;
 8.    $T(j,j-1) = \beta$; $T(j-1,j) = \beta$; $T(j,j) = \alpha$;
 9. End Do
10. Return $\|T\|_2 + \|f\|_2$ as the upper bound.
```

## 5. Numerical results

The numerical experiments are performed using our own DFT package called PARSEC which is written in Fortran 95. PARSEC is based on the real-space pseudopotential method [9,10], where higher order finite differences are used for the discretization of the Kohn–Sham equation on a uniform grid, as discussed in Section 2. We use a 12th order centered finite difference scheme. At each point $(x, y, z)$ the local stencil involves 37-points. Fig. 5.1 illustrates this stencil.

We compare three different methods in PARSEC, two of which are based on solving eigenvalue problems (2.1) at each SCF iteration. The solvers used are ARPACK [25] and TRLan [43,44] which represent two of the most efficient publicly available eigenvalue packages. The third method is our Chebyshev filtered subspace iteration, with the initial SCF step solved using TRLan. Note that each method is applied in the same package to the same problems, that is, each method can exploit the same physical symmetry operations if they exist, hence the performance difference is due solely to the three different methods.

Note that we only use the solver for standard symmetric eigenproblems in ARPACK. We mention that ARPACK is currently one of the most efficient general purpose eigensolvers, especially for nonsymmetric eigenproblems. It is by far the best known package for general eigenvalue calculations.

TRLan is a Fortran 90 package for standard symmetric eigenproblems. It uses reduced full orthogonalization and several *thick restart* strategies [43,44], hence it can be more efficient than the symmetric eigensolver in ARPACK. We call TRLan for the first step SCF iteration for our Chebyshev-filtered subspace iteration *CheFSI* method.

The test problems are a Silicon clusters $Si_{525}H_{276}$, a Silicon Germanium cluster $Si_{65}Ge_{65}H_{98}$, a Gallium Arsenide cluster $Ga_{41}As_{41}H_{72}$ and two iron clusters $Fe_{27}$ and $Fe_{51}$. These problems constitute four typical materials in electronic structure calculations. We note that metallic systems are difficult for SCF calculation because of the *charge sloshing* effect [22].

Fig. 5.2 shows the atomic structure of $Si_{525}H_{276}$. Table 5.1 contains some numerical parameters related to these test problems. The number of symmetry operations used to construct the irreducible wedge is denoted $n_{\text{symm}}$, and "reduced $H$ size" is the number of grid-points in the wedge (equal to the dimension of the discretized reduced Hamiltonian). At each SCF iteration, ARPACK and TRLan compute approximately $n_{\text{state}}$ eigenpairs from $n_{\text{symm}}$ reduced Hamiltonians at Step 2 of Algorithm 2.1, while *CheFSI*, as presented in Algorithm 4.2, essentially replaces Step 2 of Algorithm 2.1 by Algorithm 4.1 and returns the same number of wave vectors as ARPACK or TRLan does from Step 2 to Step 3 in Algorithm 2.1. The number of states of $Fe_{27}$ and $Fe_{51}$ is doubled because these clusters are magnetized and spin degeneracy in eigenvalues is absent, i.e., for each spin channel at least 520 eigenpairs are computed.

In each table, the `total_eV/atom` counts the total energy per atom, given in electron-volts. This is a control parameter used to assess the accuracy of the final result. The `# SCF steps` is the iteration steps used to reach self-consistency; the `# MV products` counts the number of matrix–vector products. Clearly this is not the only factor that determines CPU time, the reduced orthogonalization can also have a crucial effect in CPU time.

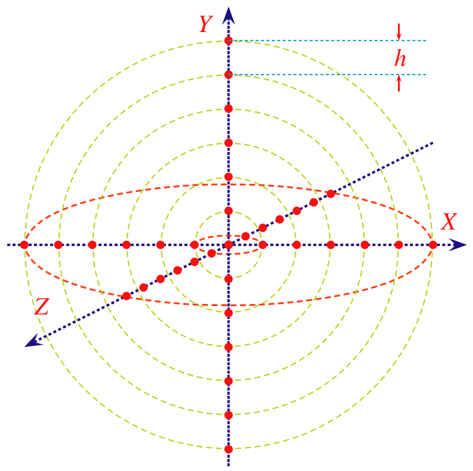

Fig. 5.1. 37-Stencil of a 12th order centered finite difference.

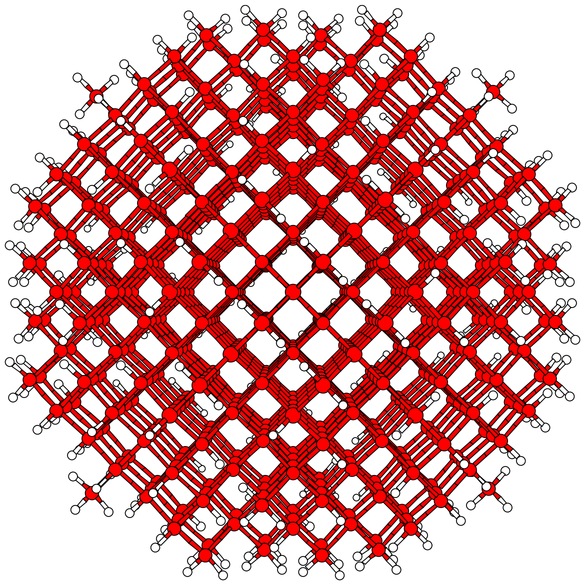

Fig. 5.2. Atomic structure of the quantum dot $Si_{525}H_{276}$. The red and white balls represent Si and H atoms respectively . The quantum dot contains 25 shells of Si atoms ad is 27.2 Å in diameter [8] (For interpretation of the references to the color in this figure legend, the reader is referred to the web version of this article).

Table 5.1
Relevant data of the test problems

| Model | Size of $H$ | $n_{\text{state}}$ | $n_{\text{symm}}$ | Reduced $H$ size |
|---|---|---|---|---|
| $Si_{525}H_{276}$ | 292,584 | 1194 | 4 | 73,146 |
| $Si_{65}Ge_{65}H_{98}$ | 185,368 | 313 | 2 | 92,684 |
| $Ga_{41}As_{41}H_{72}$ | 268,096 | 210 | 1 | 268,096 |
| $Fe_{27}$ | 697,504 | $520 \times 2$ | 8 | 87,188 |
| $Fe_{51}$ | 874,976 | $520 \times 2$ | 8 | 109,372 |

All the numerical runs are performed on the SGI Altix 3700 cluster of the Minnesota Supercomputing Institute. The CPU type is a 1.3 GHz Intel Madison processor. The Operating System is a 64bit Linux with kernel version 2.4.21. The compiler used is the Intel Fortran compiler `ifort`, with optimization flag `-O3` for all codes.

As seen from Tables 5.2–5.6, the CheFSI method is usually five to ten times faster than the eigenvector-based methods represented by two of the best available iterative sparse eigensolvers ARPACK and TRLan. Although eigenspaces are not explicitly and accurately computed at each SCF step, CheFSI only requires a few more SCF steps to reach self-consistency. However, each SCF step using CheFSI is much cheaper than an SCF step based on eigenvectors. About ten other tests on smaller systems (Hamiltonian sizes around 100–160 K) not reported here showed similar results. We also mention that there are cases where the Chebyshev filtering may become less effective. We observed these in Gallium Arsenide clusters, as seen from Table 5.4. The cases of reduced efficiency correspond to situations where the full spectrum of a given Hamiltonian spreads over a very large interval. In such situations, high degree Chebyshev polynomials must be used. However, we see that even in these unfavorable cases, CheFSI is still reasonably faster than eigenvector-based methods.

Finally, we should mention that CheFSI is as robust as eigenvector-based methods. In fact, it has been used successfully for unreported periodic systems, with performance gains similar to the ones obtained in clusters. On the other hand, an eigenvector-based method using a preconditioned Davidson method failed to converge for the same periodic systems.

Table 5.2
$Si_{525}H_{276}$, polynomial degree used is 8

| Method | # MV products | # SCF steps | Total_eV/atom | CPU (s) |
|---|---|---|---|---|
| CheFSI | 124,761 | 11 | −77.316873 | 5946.69 |
| ARPACK | 142,047 | 10 | −77.316873 | 62026.37 |
| TRLan | 145,909 | 10 | −77.316873 | 26852.84 |

Table 5.3
$Si_{65}Ge_{65}H_{98}$, polynomial degree used is 8

| Method | # MV products | # SCF steps | Total_eV/atom | CPU (s) |
|---|---|---|---|---|
| CheFSI | 42,919 | 13 | −140.076118 | 2344.06 |
| ARPACK | 51,752 | 9 | −140.076118 | 12770.81 |
| TRLan | 53,892 | 9 | −140.076118 | 6056.11 |

Table 5.4
$Ga_{41}As_{41}H_{72}$, polynomial degree used is 16

| Method | # MV products | # SCF steps | Total_eV/atom | CPU (s) |
|---|---|---|---|---|
| CheFSI | 138,672 | 37 | −89.634940 | 12923.27 |
| ARPACK | 58,506 | 10 | −89.634940 | 44305.97 |
| TRLan | 58,794 | 10 | −89.634940 | 16733.68 |

The spectrum of each reduced Hamiltonian spans a large interval, making the Chebyshev filtering not as effective as other examples.

Table 5.5
$Fe_{27}$, polynomial degree used is 9

| Method | # MV products | # SCF steps | Total_eV/atom | CPU (s) |
|---|---|---|---|---|
| CheFSI | 363,728 | 30 | −776.575290 | 15408.16 |
| ARPACK | 750,883 | 21 | −776.586420 | 118693.64 |
| TRLan | 807,652 | 21 | −776.586422 | 83726.20 |

Table 5.6
$Fe_{51}$, polynomial degree used is 9

| Method | # MV products | # SCF steps | Total_eV/atom | CPU (s) |
|---|---|---|---|---|
| CheFSI | 474,773 | 37 | −777.019294 | 37701.54 |
| ARPACK | 1,272,441 | 34 | −777.038081 | 235662.96 |
| TRLan | 1,241,744 | 32 | −777.038086 | 184580.33 |

## 6. Conclusions

An algorithm based on Chebyshev-filtered subspace iteration has been developed for performing SCF calculations in *density functional theory*, which has the advantage of not explicitly relying on eigenvectors, except at the first SCF step. This leads to significant gains in computational time relative to traditional eigenvector-based methods which are inevitably constrained by the high computational cost of diagonalization at each SCF step. Numerical results show that the new method is five to ten times faster than eigenvector-based methods using two of the best iterative eigensolvers. Even though implemented sequentially at present, the CheFSI method can solve realistic systems of moderate size within a reasonable time frame. Parallel implementation of the CheFSI method will be reported in a forthcoming paper, where we study larger and more complex material systems.

## Acknowledgments

We thank Dr. Manuel Alemany for his help in using PARSEC and Dr. Shen Li for providing us with the physical data of the iron clusters. Valuable comments from the referees are also acknowledged. Calculations were performed at the Minnesota Supercomputing Institute.

## References

[1] M.M.G. Alemany, M. Jain, L. Kronik, J.R. Chelikowsky, Real-space pseudopotential method for computing the electronic properties of periodic systems, Phys. Rev. B 69 (2004) 075101-1–075101-6.

[2] R. Baer, M. Head-Gordon, Chebyshev expansion methods for electronic structure calculations on large molecular systems, J. Chem. Phys. 107 (1997) 10003–10013.

[3] S. Baroni, P. Giannozzi, Towards very large scale electronic structure calculations, Europhys. Lett. 17 (1992) 547–552.

[4] C. Bekas, Y. Saad, M.L. Tiago, J.R. Chelikowsky, Computing charge densities with partially reorthogonalized Lanczos, Comp. Phys. Comm. 171 (2005) 175–186.

[5] A. Brandt, Multiscale scientific computation: review 2001, in: T. Barth, T. Chan, R. Haimes (Eds.), Multiscale and Multiresolution Methods, Lecture Notes in Computer Science and Engineering, vol. 20, Springer-Verlag, Berlin, 2002, pp. 3–95.

[6] E.L. Briggs, D.J. Sullivan, J. Bernholc, Large-scale electronic-structure calculations with multigrid acceleration, Phys. Rev. B 52 (1995) R5471–R5474.

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

[8] J. Chelikowsky, The pseudopotential-density functional method applied to nanostructures, J. Phys. D: Appl. Phys. 33 (2000) R33–R50.

[9] J.R. Chelikowsky, N. Troullier, Y. Saad, Finite-difference-pseudopotential method: electronic structure calculations without a basis, Phys. Rev. Lett. 72 (1994) 1240–1243.

[10] J.R. Chelikowsky, N. Troullier, K. Wu, Y. Saad, Higher-order finite-difference pseudopotential method: an application to diatomic molecules, Phys. Rev. B 50 (1994) 11355–11364.

[11] S. Costiner, S. Táasan, Adaptive multigrid techniques for large-scale eigenvalue problems: solutions of the Schrödinger problem in two and three dimensions, Phys. Rev. E 51 (1995) 3704–3717.

[12] J. Daniel, W.B. Gragg, L. Kaufman, G.W. Stewart, Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization, Math. Comp. 30 (1976) 772–795.

[13] R.M. Dreizler, E.K.U. Gross, Density Functional Theory: An Approach to the Quantum Many-Body Problem, Springer-Verlag, Berlin, 1990.

[14] J.-L. Fattebert, J. Bernholc, Towards grid-based O($N$) density-functional theory methods: optimized nonorthogonal orbitals and multigrid acceleration, Phys. Rev. B 62 (2000) 1713–1722.

[15] S. Goedecker, Linear scaling electronic structure methods, Rev. Mod. Phys. 71 (1999) 1085–1123.

[16] S. Goedecker, L. Colombo, Efficient linear scaling algorithm for tight-binding molecular dynamics, Phys. Rev. Lett. 73 (1994) 122–125.

[17] M. Heiskanen, T. Torsti, M.J. Puska, R.M. Nieminen, Multigrid method for electronic structure calculations, Phys. Rev. B 63 (2001) 245106.

[18] P. Hohenberg, W. Kohn, Inhomogeneous electron gas, Phys. Rev. 136 (1964) B864–B871.

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

[20] T. Kerkhoven, A. Galick, U. Ravaioli, J. Arends, Y. Saad, Efficient numerical simulation of electron states in quantum wires, J. Appl. Phys. 68 (1990) 3461–3469.

[21] W. Kohn, L.J. Sham, Self-consistent equations including exchange and correlation effects, Phys. Rev. 140 (1965) A1133–A1138.

[22] G. Kresse, J. Furthmüller, Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set, Phys. Rev. B 54 (1996) 11169–11186.

[23] C. Lanczos, An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, J. Res. Nat. Bur. Stand. 45 (1950) 255–282.

[24] I. Lee, Y. Kim, R.M. Martin, One-way multigrid method in electronic-structure calculations, Phys. Rev. B 61 (2000) 4397–4400.

[25] R.B. Lehoucq, D.C. Sorensen, C. Yang, ARPACK Users Guide: Solution of Large Scale Eigenvalue Problems by Implicitly Restarted Arnoldi Methods, SIAM, Philadelphia, 1998. Available from: <http//www.caam.rice.edu/software/ARPACK/>.

[26] J.L. Martins, M. Cohen, Diagonalization of large matrices in pseudopotential band-structure calculations: dual-space formalism, Phys. Rev. B 37 (1988) 6134–6138.

[27] J.L. Martins, N. Troullier, S.-H. Wei, Pseudopotential plane-wave calculations for ZnS, Phys. Rev. B 43 (1991) 2213–2217.

[28] B.N. Parlett, The Symmetric Eigenvalue Problem, Classics in Applied Mathematics, vol. 20, SIAM, Philadelphia, PA, 1998.

[29] R. Parr, W. Yang, Density Functional Theory of Atoms and Molecules, Oxford University Press, 1989.

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

[31] P. Pulay, Convergence acceleration of iterative sequences: the case of SCF iteration, Chem. Phys. Lett. 73 (1980) 393–398.

[32] P. Pulay, Improved SCF convergence acceleration, J. Comput. Chem. 3 (1982) 556–560.

[33] H. Rutishauser, Simultaneous iteration method for symmetric matrices, in: J.H. Wilkinson, C. Reinsh (Eds.), Handbook for Automatic Computation (Linear Algebra), vol. II, Springer-Verlag, Berlin, 1971, pp. 284–302.

[34] Y. Saad, Numerical Methods for Large Eigenvalue Problems, John Wiley, New York, 1992. Available from: <http://www.cs.umn.edu/saad/books.html>.

[35] Y. Saad, A. Stathopoulos, J. Chelikowsky, K. Wu, S. Öğüt, Solution of large eigenvalue problems in electronic structure calculations, BIT 36 (1996) 563–578.

[36] O.F. Sankey, D.A. Drabold, A. Gibson, Projected random vectors and the recursion method in the electronic-structure problem, Phys. Rev. B 50 (1994) 1376–1381.
[37] A.P. Seitsonen, M.J. Puska, R.M. Nieminen, Real-space electronic-structure calculations: combination of the finite-difference and conjugate-gradient methods, Phys. Rev. B 51 (1995) 14057–14061.
[38] D.C. Sorensen, Implicit application of polynomial filters in a *k*-step Arnoldi method, SIAM J. Matrix Anal. Appl. 13 (1992) 357–385.
[39] A. Stathopoulos, S. Öğüt, Y. Saad, J. Chelikowsky, H. Kim, Parallel methods and tools for predicting materials properties, IEEE Comput. Sci. Eng. 2 (2000) 9–32.
[40] U. Stephan, D.A. Drabold, R.M. Martin, Improved accuracy and acceleration of variational order-*N* electronic structure computations by projection techniques, Phys. Rev. B 58 (1998) 13472–13481.
[41] M.L. Tiago, J.R. Chelikowsky, Technical Report, University of Texas, Austin, in preparation.
[42] D. Wood, A. Zunger, A new method for diagonalising large matrices, J. Phys. A: Math. Gen. 18 (1985) 1343–1359.
[43] K. Wu, A. Canning, H.D. Simon, L.-W. Wang, Thick-restart Lanczos method for electronic structure calculations, J. Comput. Phys. 154 (1999) 156–173.
[44] K. Wu, H. Simon, Thick-restart Lanczos method for large symmetric eigenvalue problems, SIAM J. Matrix Anal. Appl. 22 (2000) 602–616.
[45] Y. Zhou, Y. Saad, A Chebyshev–Davidson algorithm for large symmetric eigenvalue problems, Technical Report, Minnesota Supercomputing Institute, University of Minnesota, under review.