JOURNAL OF COMPUTATIONAL PHYSICS **124,** 146–161 (1996)
ARTICLE NO. 0050

Projective Block Lanczos Algorithm for Dense, Hermitian Eigensystems

FRANK WEBSTER* AND GEN-CHING LO†

*Chemistry Department, State University of New York, Stony Brook, New York 11794-3400 and †Department of Applied Mathematics and Statistics, State University of New York, Stony Brook, New York 11794-3600

Received June 20, 1994; revised July 25, 1995

Projection operators are used to effect ''deflation by restriction'' and it is argued that this is an optimal Lanczos algorithm for memory minimization. Algorithmic optimization is constrained to dense, Hermitian eigensystems where a significant number of the extreme eigenvectors must be obtained reliably and completely. The defining constraints are operator algebra without a matrix representation and semi-orthogonalization without storage of Krylov vectors. Other semi-orthogonalization strategies for Lanczos algorithms and conjugate gradient techniques are evaluated within these constraints. Large scale, sparse, complex numerical experiments are performed on clusters of magnetic dipoles, a quantum many-body system that is not block-diagonalizable. Plane-wave, density functional theory of beryllium clusters provides examples of dense complex eigensystems. Use of preconditioners and spectral transformations is evaluated in a preprocessor prior to a high accuracy self-consistent field calculation.

1. INTRODUCTION

The Lanczos algorithm for computing eigenvalues and eigenvectors of a Hermitian matrix was introduced by Cornelius Lanczos [1] in the early 1950s. Numerical analysis of the method may be considered mature, and sophisticated variants may be reliably applied to certain classes of problems. We present a double iteration scheme for the block Lanczos procedure, BL, with semi-orthogonalization that is suitable for applications where memory minimization is the dominant consideration. We identify the imposed constraints and focus upon the development of an appropriate variant of the block Lanczos algorithm. An assessment of facts related to experiments are provided for sparse and dense eigensystems where a significant fraction of the extreme eigenpairs is sought.

Optimization of the block Lanczos algorithm was explored within several constraints. The constraints arise from: intention to implement the optimized algorithm on a distributed memory multiprocessor and the need to compute a significant fraction of the extreme eigenpairs. Large eigenvector computation on parallel architectures with distributed memory will be described in a subsequent article.

First, we consider clusters of magnetic dipoles, quantum many-body problems where all sites are directly coupled by scalar operators. (If a matrix representation were constructed, it would be sparse.) In contrast to most electron-spin models, the solutions are not pure spin states, and the Hamiltonian is not block-diagonalizable. These examples are discussed in Section 4 and are used to illustrate the algorithm in considerable detail.

In Section 5 we consider plane-wave density functional theory of beryllium clusters. These examples are dense. The focus is upon the use of preconditioners and spectral transformations. For an applied mathematician these transformations provide a wide variety of exercises with very different eigenvalue spectra. For a solid-state physicist, these evaluate methods for accelerating computation of electronic states when all electrons are explicitly represented (i.e., no pseudo-potentials).

Well-established techniques are inappropriate to these constraints. The Householder transform as used in LAPACK [2] obtains a tridiagonal representation of full dimensionality and cannot compete with iterative procedures. (Memory requirements, alone, prohibit such procedures.) Efficient methods for one extreme eigenvector (e.g., Monte Carlo and some conjugate gradient algorithms, CG) become very inefficient for hundreds of vectors.

We believe that use of projection operators (and their generalizations) in an iterative BL is an optimal expression of semi-orthogonalization within our constraints. Our major contribution is identification of the constraints and synthesis of an appropriate algorithm from published work. Given the extensive research and development of BL, our new contributions are largely a synthesis of previously developed techniques in relationship to the constraints we have imposed. Cullum and Willoughby's book [3] and B. N. Parlett's book [4] are our principle resources.

Arnoldi methods and their variants [5, 6] are closely related to Lanczos algorithms incorporating full reorthogonalization. Parlett and Scott [7] developed ''selective orthogonalization'' to correct the loss of orthogonality that plagues all Lanczos procedures. Simon [8] generalized this

0021-9991/96 $18.00

concept and developed ''partial reorthogonalization.'' These improvements make great use of formal numerical analysis to semi-orthogonalize the iteratively constructed Krylov space. We believe that these are not applicable to our constraints, because we assume that recall of Krylov vectors, however infrequently, is prohibited. Indeed, unavailability of Krylov vectors essentially determines our approach and distinguishes it from the more sophisticated algorithms referenced above.

The most important algorithmic developments are based upon preconditioning, a ''shift and invert'' strategy [9]. The work of Grimes, Lewis, and Simon [10] is the state-of-the-art in finding the interior eigenvalues of sparse, generalized eigenvector problems. The success of this work stems from the sparseness of the applications and the necessity of decomposing a matrix (whether or not a shift is applied). Progress has been made in diagonal preconditioning [11] for dense, *simple* eigensystems using a generalization of Davidson's method [12] expressed as a doubly iterative Lanczos algorithm. This and other approximate spectral transformations (e.g., our earlier work [13]) are studied in Section 5. The goal is a BL algorithm using an accurate shift operator (like Grimes, Lewis, and Simon [10]) for large, dense eigensystems implemented on a distributed memory multiprocessor. Our projective block Lanczos algorithm, PBL, and our evaluation of approximate spectral transforms are a step towards this goal for specific classes of physical problems.

Our algorithm uses one form of semi-orthogonalization that is known as ''deflation by restriction.'' (See Parlett's book [4].) The operator is deflated using projection operators. More generally, the composite operator makes degenerate and shifts the converged vectors. In this technique, the restarts are explicit and new Krylov spaces are constructed after each change to the operator, although more efficient, implicit restarts require saving Krylov vectors. Indeed, our memory constraint is severe enough, that our recursions must be run twice: first to approximate the eigenvalues and again to construct the eigenvectors.

2. LANCZOS AND CONJUGATE GRADIENT ALGORITHMS

Among solid state physicists and many-body theorists, preconditioned conjugate gradient methods have achieved great success. Notable examples are the work of Teter, Payne, and Allen [14] for plane-wave, density functional theory and the work of Nightingale, Viswanath, and Müller [15] for many-body quantum mechanics of spin systems. The formal equivalence of BL and CG algorithms is not acknowledged in such work. We give one perspective on the debate concerning these techniques. We do not provide numerical comparisons.

When the aim is minimization of the residual, simple forms of the CG and Lanczos methods are equivalent (for positive definite matrices and with finite precision arithmetic). See Sections 4.2 and 4.3 of Ref. [3] and the references therein. Minimization of the residual is equivalent to minimization of the functional for quadratic functionals [16]. If the goal is calculating the best approximation to an eigenvalue or minimizing the energy, the two elementary methods are equally efficient.

Given this equivalence, then why do some authors claim that the CG is as much as an order of magnitude more efficient than the Lanczos? Nobody uses the simple forms of the algorithms. Fair comparisons require sophisticated numerical analysis or numerical tests using the best available algorithms. We believe there are only two important differences; unlike BL, CG methods do not need Krylov vectors to construct the eigenvectors, but BL methods are better suited to implicit and explicit restarts.

Important examples of quadratic energy functionals include many-body electron spin problems (e.g., Ising, Heisenberg, Hubbard, and Potts models). In many such exercises, only one or two states are needed, and the Hamiltonian matrix (if constructed) would be extremely sparse. In Nightingale *et al.* [15] the comparison was made between CG and a ''modified'' Lanczos [17], where the energy is minimized within iteratively optimized, two-dimensional, Krylov subspaces. The CG and Lanczos methods would be equivalent if the full dimensionality of the Krylov space was used in the Lanczos algorithm. This simplification apparently trades accuracy for ''efficiency.'' But, obtaining one or two eigenvalues of a large tridiagonal matrix is never the computational bottleneck.

For the density functional and Hartree–Fock theories of electronic structure, assessments of the efficiency and accuracy of BL and CG methods are important and nontrivial. The work of Teter *et al.* [14] is an example of a preconditioned CG with cyclic subspace iteration (band-by-band minimization) applied to a nonquadratic functional. Any BL algorithm with a semi-orthogonalization strategy would more efficiently maintain band orthogonality and the energy functional would be quadratic. However, extra effort would be required to construct the states. Our PBL is applied to a generalization of band-by-band minimization in Section 4. Section 5 evaluates the preconditioner of Ref. [14].

Some energy functionals are not quadratic, and some functionals other than the energy may be directly optimized (e.g., recursive residue generation for Green's functions [18]). One may speculate that observables related to wave function properties may be better obtained by minimizing the residual through BL algorithms. Direct calculation of transition moments without constructing states might provide such a test.

Further assessment of sophisticated CG and BL procedures applied to electronic structure theory will require

numerical experiments on observables besides the energy. We provide a preliminary study in Section 5. Fair comparisons should use blocked and doubly iterative versions of the algorithms. Cullum and Willoughby provide a historical account of the relationship between BL and doubly iterative CG algorithms, and physicists advocating CG methods would be well served by perusing this.

3. PROJECTIVE BLOCK LANCZOS

The constraints and assumptions guiding our developments are summarized. Memory use must be minimized, and this implies that Krylov vectors cannot be stored. Operator algebra must be assumed in the statement of the problem and in implementation of the algorithm.

By the use of "operator algebra," we mean a linear operator acting on a trial vector without constructing a matrix representation. All linear operators on finite vector spaces *may* be represented by matrices, but they *need not* be. The statement of the eigenvector problem *should not* use a matrix representation. Throughout this paper we will write the Hamiltonian operator $H(\mathbf{v})$ acting on a vector $\mathbf{v}$ as matrix multiplication $\boldsymbol{Hv}$. (We use $\boldsymbol{H}$ instead of the usual $\boldsymbol{A}$.)

If N is the dimension of the space in which the operators act, then no matrix $\boldsymbol{H} \in C^{N\times N}$ can be constructed or used. Storage of $\boldsymbol{H}$ dwarfs storage of Krylov vectors. In other words, if one stores $\boldsymbol{H}$ then the more sophisticated semi-orthogonalization strategies for the BL or Arnoldi procedures are the method of choice. Note that this also prevents consideration of generalized eigenvector problems, unless the decomposition step can be represented by operator algebra.

All many-body spin Hamiltonians, and the examples of Section 4, can be represented by operators, since the matrix elements are simple analytic functions and the nonzero elements are known in advance. More generally, any sparse matrix multiplication using indirect addressing may be considered an operator (in our use of the phrase). Plane-wave, density functional theory of Section 5 provides a beautiful example of an operator, a three-dimensional convolution in reciprocal space, plus a diagonal (not constant) matrix. Indeed, the convolution is easier to think about than its matrix representation (which is completely unnecessary).

We assume that most states are directly coupled by the operator. In other words, the matrix $\boldsymbol{H} \in C^{N\times N}$ would be dense. Nothing in our algorithm depends upon this assumption. As mentioned above, sparse matrices, whose storage does not scale as N^2 could be implemented as an operator while adhering to our strict memory requirements.

We need to compute a significant fraction of the extreme eigenpairs. For M eigenpairs, we assume that $M = O(N/100)$ although this is not required. The desired fraction of the spectrum should be obtained completely and reliably (i.e., within a predetermined tolerance). We assume a small degree of degeneracy; the multiplicity of a desired state should not exceed the block size. For simplicity, we assume a complex, Hermitian eigensystem, and we believe that generalization to nonHermitian systems would not be difficult.

We consider only block algorithms. This requirement arises from our parallel implementation, because data movement on most high-performance architectures is slow compared to floating point performance. Block algorithms imply reduced amounts of memory traffic [19]. Block algorithms also allow extensive use of BLAS level 3 routines [20], their parallel counterparts when available, and assembly coded libraries if available.

We will occasionally use SVL to refer to a class of single-vector Lanczos procedures that obtain a tridiagonal representation in a Krylov space. The BL iteratively constructs a band matrix (i.e., a banded matrix with a single band). Please refer to Refs. [3, 4].

We assume that evaluating the operator, $\boldsymbol{H}$, dominates the CPU time. The BL algorithm requires $O(Nm^2 + mN^2)$ flops, where m is the block size. $O(mN^2)$ dominates and arises from matrix multiplication $\boldsymbol{Hv}$. For operators, we write $O(mT_H) \approx O(mN^2)$, where T_H is the execution time for the operator. Assuming T_H dominates allows us to diminish optimizations of the component, linear-algebraic procedures.

We present the algorithm abstractly. Afterwards, we comment on theoretical details of projection operators, block sizes, and convergence parameters. Practical considerations and parameter choices are given in Sections 4 and 5. Finally, several questions are asked and answered, and several improvements are described.

3.1. *Algorithm*

Our projective block Lanczos algorithm is a double iteration scheme where the outer loop updates the initial vectors and the semi-orthogonalization algorithm and the inner loop is the BL algorithm. Explicit restarts are used in the outer loop. This is "deflation by restriction," and the BL inner loop constructs a new Krylov space for each restart.

We seek M extreme eigenvalues and eigenvectors of a dense Hermitian matrix. The Hermitian matrix $\boldsymbol{H}$ is evaluated as an operator, or represented as $\boldsymbol{H} \in C^{N\times N}$. All memory requriments and all results are reported for 16 byte complex numbers. If represented explicitly, the BLAS routine *ZHPMV* is used and $\boldsymbol{H}$ dominates the storage requirements.

The user specifies the initial block size m_1 and provides a set of linearly independent vectors. A range of residuals

$\varepsilon_1 < \varepsilon_2$ is specified for the M computed eigenvectors. A practical means for determining ε_1 and ε_2 is given in Section 5.2.

The algorithm is structured as a double loop, with the outer loop indexed by k. Step 0 sets up the outer loop, and step 5 executes the loop. The inner loop is the BL algorithm indexed by j. There are two distinct inner loops, steps 1 and 3. Step 1 is described in detail and step 3 is a slight modification of step 1.

Step 1 begins with the action of $\boldsymbol{H}$ upon $\boldsymbol{Q}_j^k$. All $\boldsymbol{Q} \in C^{N\times mk}$ are left unitary, $\boldsymbol{Q}^*\boldsymbol{Q} = \mathbf{1}$, where the asterisk suffix denotes conjugate transpose. The block size m_k varies with k but is fixed as j varies. The projection operator $\boldsymbol{C}_k\lambda\boldsymbol{C}_k^*$ in step 1a is never formed explicitly, but is evaluated as a sequence of left-multiplications, with λ, a real, diagonal matrix. See Section 3.2.

Matrices $\boldsymbol{X}$, $\boldsymbol{R}$, $\boldsymbol{A}$, and $\boldsymbol{W}$ are elements of $C^{N\times mk}$ and need not be distinct. All $\boldsymbol{Q}$ are formed by QR factorization, which is step 1e. The $\boldsymbol{B}$ are upper triangular. The $\boldsymbol{Q}_1^1$ is obtained from the user-specified initial vectors by QR factorization. Similarly, one may factor initial vectors to obtain $\boldsymbol{Q}_1^k$ for $k > 1$. This is required only if over-completeness is suspected (cf. Section 3.5).

The BL recursions in step 1 build a Hermitian band matrix $\boldsymbol{T}^{\mathbf{k}} \in C^{n\times n}$ with $n = m_k J_k$. The band matrix has band width m_k and is stored as $C^{nb\times nt}$ with $nb = m_k/2 + 1$ and $nt \geq \max\{m_k J_k .: k\}$, as specified in Ref. [2, Section 5.3]. Step 1g, constructs $\boldsymbol{T}^{\mathbf{k}}$, in sequence, from the lower-triangular part of $\boldsymbol{A}_j^k$ and the upper-triangular matrix $\boldsymbol{B}_{j+1}^k$. The compact storage scheme is a sequence of upper-right and lower-right triangles packed $\boldsymbol{A}_1\boldsymbol{B}_2\boldsymbol{A}_2\boldsymbol{B}_3$ ... into a $C^{nb\times nt}$ matrix.

The σ in step 1i represents the last m_k elements of $\boldsymbol{S}_1$, one approximate extreme eigenvector of the band matrix. Updating $\boldsymbol{S}_1$, is described in Section 3.4. This is used to evaluate convergence and to terminate the BL. The total number of BL steps to convergence is J_k.

A caution concerns the use of the scalar zero in BLAS routines (e.g., *ZGEMM* and *ZGEMV*) in step 1 for $k = 1$. This does not clear memory, as expected, if the memory has previously contained any integer data (i.e., zero times *NANQ* is *NANQ*).

Step 2 obtains several eigenvectors of $\boldsymbol{T}^{\mathbf{k}}$ using *ZHBEVX* from LAPACK [2] that "computes selected eigenvalues and, optionally, eigenvectors of a complex Hermitian band matrix." The eigenvectors are $\boldsymbol{S} \in C^{n\times m}$ with $n = m_k J_k$ and $m = m_k + \delta m$. Any SVL will be faster and smaller and should replace *ZHBEVX* once the algorithm is implemented and tested. See Section 3.5.

In step 2, $m_k + \delta m$ eigenvectors are found, as discussed in Section 3.3. Step 2b uses $\boldsymbol{s}$ to indicate the last m_k elements of $\boldsymbol{S}$. p_k counts the number of newly converged vectors, and p accumulates the total number converged to this point. Selection of newly converged vectors uses e_2 and appears in the algorithm before the vectors are themselves constructed. See Section 3.4.

Step 3 repeats the BL recursion to compute the eigenvectors. Construction of the band matrix and convergence testing are unnecessary. The second BL must exactly reproduce the $\boldsymbol{Q}_j^k$ in sequence. Alternatively, one can simply store these matrices on the disk to avoid the additional computation. Step 3f shows the matrix indices (i.e., submatrix multiplication).

Step 4a forms a matrix as a union of a matrix and a set of vectors, with $\boldsymbol{C}_k \in C^{N\times p}$ and $\boldsymbol{C}_{k+1} \in C^{N\times p'}$ with $p' = p + p_k$. Step 4b assigns the unconverged vectors to the starting block for the next iteration of the outer loop.

0. *set* $\boldsymbol{C}_1 = 0$, $m_1 = m$, $p = 0$, *and* $k = 1$
1. *BL recursions: initialize* $\boldsymbol{T}^{\mathbf{k}}$ *and* $\boldsymbol{S}_1^k = \hat{\mathbf{1}}$
   set $\boldsymbol{Q}_0^k = 0$, $\boldsymbol{B}_1^k = 0$, *and for* $j = 1, 2, \ldots$ *do*
   a. $\boldsymbol{X}_j^k = (\boldsymbol{I} - \boldsymbol{C}_k\,\lambda\boldsymbol{C}_k^*)\,\boldsymbol{H}\,\boldsymbol{Q}_j^k$
   b. $\boldsymbol{R}_j^k = \boldsymbol{X}_j^k - \boldsymbol{Q}_{j-1}^k\,\boldsymbol{B}_j^{k*}$
   c. $\boldsymbol{A}_j^k = \boldsymbol{Q}_j^{k*}\,\boldsymbol{R}_j^k$
   d. $\boldsymbol{W}_j^k = \boldsymbol{R}_j^k - \boldsymbol{Q}_j^k\,\boldsymbol{A}_j^k$
   e. $\boldsymbol{Q}_{j+1}^k\,\boldsymbol{B}_{j+1}^k = \boldsymbol{W}_j^k$
   f. *increase memory allocation for* $\boldsymbol{T}^{\boldsymbol{k}}$
   g. *store* $\boldsymbol{A}_j^k$ *and* $\boldsymbol{B}_{j+1}^k$ *in* $\boldsymbol{T}^{\boldsymbol{k}}$
   h. *update* $\boldsymbol{S}_1^k$
   i. *if* $\|\boldsymbol{B}_{j+1}^k\,\sigma\| < \varepsilon_1$ *goto* 2, *else goto* 1a.
2. *ZHBEVX applied to* $\boldsymbol{T}^{\mathbf{k}}$
   a. *store eigenpairs* $\{\Lambda_n^k, \boldsymbol{S}_n^k$ *for* $n = 1, \ldots, m_k + \delta m\}$
   b. $\{p_k = 0;\ n = 1\}$, *while* $\|\boldsymbol{B}_{j+1}^k\,\boldsymbol{s}_n^k\| < \varepsilon_2$ *do* $\{p_k = p_k + 1; n = n + 1\}$
   c. $p = p + p_k$; $m_{k+1} = \min(m, M\text{-}p)$
3. *BL recursion : set* $J_k = j$, *and vary* $j = 1,2, \ldots$
   3a *through* 3e *exactly as in steps* 1a *through* 1e; *or read* $\boldsymbol{Q}_j^k$ *from disk*
   f. *for i from* $jm_k\text{-}m_k + 1$ *to* jm_k *and for n from* 1 *to* m_k *do*

$$\boldsymbol{V}_{in}^k = \sum_{i'=1}^{m_k} \boldsymbol{Q}_{ji'}^k\,\boldsymbol{S}_{i'n}^k$$

   g. *if* $j = J_k$, *goto* 4, *else goto* 3a.
4. a. $\boldsymbol{C}_{k+1} = \boldsymbol{C}_k \cup \{\boldsymbol{V}_n^k$ *for* $n = 1, \ldots, p_k\}$
   b. $\boldsymbol{Q}_1^{k+1} = \{\boldsymbol{V}_n^k$ *for* $n = p_k + 1, \ldots, p_k + m_{k+1}\}$
5. *if* $p = M$ *stop*, *else* $\{k = k + 1$; *goto* 1$\}$.

3.2. *Projection Operators*

When λ is a unit matrix, step 1a applies a projection operator, $\boldsymbol{P} = \boldsymbol{I}\text{-}\boldsymbol{C}\boldsymbol{C}^*$, following the Hamiltonian operator. For unitary $\boldsymbol{C}$, the projection $\boldsymbol{C}\boldsymbol{C}^*$ is onto the subspace E spanned by vectors in $\boldsymbol{C}$, and $\boldsymbol{P}$ is the complementary projection onto the orthogonal subspace E^*. Both operators are Hermitian and idempotent. All is true with minor

modifications for $\boldsymbol{C}$ being nonorthogonal. If a matrix representation $\boldsymbol{H}$ is used, the projection operator may be incorporated directly into $\boldsymbol{H}$ using Hermitian rank-one updates, BLAS routine *ZHPRU*.

To improve efficiency, we perform the Lanczos recurrences using $\boldsymbol{PH}$ rather than $\boldsymbol{PHP}$. The matrix representation in space E^* of $\boldsymbol{PHP}$ is equal to $\boldsymbol{PH}$. Given an iterative method that can be written by a recurrence relation involving only vectors in E^* and powers of the operator, the sequence of vectors arising from $\boldsymbol{PH}$ is equivalent to the sequence arising from $\boldsymbol{PHP}$.

With $\lambda = \boldsymbol{I}$, all vectors in E are eigenvectors with zero eigenvalue. Finite precision arithmetic allows the full Krylov space to accumulate a component in E. Thus, multiple zero eigenvalues of $\boldsymbol{T^k}$ will be found for large J_k. This seriously affects the convergence of unknown eigenpairs lying near zero.

When $\lambda \neq \boldsymbol{I}$, $\boldsymbol{P = I\text{-}C\lambda\, C^*}$ is not a projection operator. However, the statements regarding subspaces and $\boldsymbol{PH}$ hold true. Use of $\boldsymbol{PH}$ instead of $\boldsymbol{PHP}$ holds a theoretical advantage: inaccuracies in $\boldsymbol{C}$ are multiplied by λ instead of λ^2. For $\lambda \neq \boldsymbol{I}$, the operator shifts the effective eigenvalues of the converged vectors away from the region of interest. A nonconstant λ_n, $= 1\text{-}e/\Lambda_n$, implies $\boldsymbol{PH\ v} = e\boldsymbol{v}$, $\forall \boldsymbol{v} \in E$. This degeneracy further accelerates convergence. We have found [21] the latter to be necessary for $J_k > 10m_k$.

An additive ''project and shift'' operator, $\boldsymbol{(I\text{-}CC^*)(H\text{-}\varepsilon)} + \varepsilon$, may have better numerical stability than the multiplicative expression when the converged Λ_n range over many orders of magnitude. The multiplicative variant is more general, encompassing the additive form. Advantages to further generalization (e.g., λ being any Hermitian matrix or operator) are not clear.

Given E spanned by approximate eigenvectors of $\boldsymbol{H}$, what is the smallest achievable residual for a new vector obtained from E^*? Obviously, a true eigenvector projecting entirely onto E^* could be computed very accurately. A true eigenvector could project largely onto both subspaces due to ''noise'' in E arising from inaccurate solutions for $\boldsymbol{C}$. The original question is restated, ''Can we assume that none of the desired eigenvectors project significantly onto the noise in E?'' The difficulty arises only if one desires interior or quasi-degenerate eigenvalues. (Well separated, extreme eigenvectors are the earliest members of E.) Failure results in a missed eigenvector, but such are easily identified.

3.3. *Block Size*

The choice of block size m has a major effect upon the performance of the algorithm. Two factors favor the choice of small m. The storage required for computation of the eigenvectors of $\boldsymbol{T^k}$ increases as a quadratic function of the block size; and, the asymptotic convergence of the algorithm favors large numbers of BL steps for each outer iteration.

TABLE I

The Effect of Block Size on Performance

M	m	k	$\Sigma J_k m_k$
64	32	3	2942
	64	3	2396
	96	3	2838
	128	2	3380
96	32	3	3132
	64	3	2820
	96	3	3145
	128	3	3565

Note. Block size m is varied for a magnetic dipole example with $N = 4096$, $\varepsilon_1 = 1{*}10^{-7}$, and $\varepsilon_2 = 50\ \varepsilon_1$. Performance is measured by the total number of Krylov vectors used, fourth column. This assumes that $\mathbf{Hv}$ dominates the performance and other linear algebra is negligible. The cost is higher for small m because some Ritz vectors must be recomputed. It is pointless to set $m > M$.

The principal factor favoring a large m is efficiency. This is critical to parallel computing. For our examples, the operator subroutine will be more efficient for large m_k, when the inner loop is unrolled. In other words, the cost per vector of any non-sparse operator decreases with m_k, asymptotically approaching the cost of matrix multiplication. An example is given in Section 5.1. Similarly, for preconditioners and other spectral transformations, the procedures are more efficient for larger m_k. In general, it is best to choose $m_0 \leq m \leq M$, where M is the number of desired eigenvectors. The value of m_0 is a problem-dependent parameter approximating the number of converged eigenvectors during the first iteration or the multiplicity of the desired eigenvalues.

Regardless of the procedure used to obtain $\boldsymbol{S}$, δm does not affect the memory requirements. An excess of eigenvectors, δm, may be computed with no effect on memory and little effect on speed. Usually, set $\delta m = \min(m, M\text{-}m)$. (*Note.* δm should be negative when $m > M$.)

Suppose $m = M/2$ and $p_1 = m/4$. Since the storage is defined by m, excess unconverged Ritz vectors are thrown away when $\delta m > p_1$. Ideally, one should set $\delta m = \max(p_k)$, the number of converged vectors. Since this is unknown, it is likely that unconverged Ritz vectors will be unused when $m < M$.

This is demonstrated in Table I, rows 1 and 5. In our experience with iterative, single vector Lanczos, the information is largely preserved by using linear combinations as starting vectors for the next iteration. Preliminary tests indicate that a good restart strategy is nontrivial.

In row 4, only two iterations were performed, but the initial number of BL recursions, J_1, is unchanged. The

large block size reduces the effort in subsequent iterations, but the initial cost, $128J_1$, is excessive.

3.4. *Convergence Criteria*

We describe the parameters ε_1 and ε_2, the convergence criteria, and initial vectors, $\boldsymbol{Q}_1^1$. Section 5.2 suggests a strategy for quantitatively choosing ε_1 and ε_2. Furthermore, the effect of scaling the matrix $\alpha\boldsymbol{H}$ is critical to the comparison of different operators in Section 5.4. Section 5 is more important in practice. This section provides a better understanding of the algorithm.

The algorithm provides eigenvectors whose residuals lie between ε_1 and ε_2. The stopping criterion within the BL iteration is determined by ε_1, the *lower* bound. Once stopped, vectors are considered converged if their residual lies below the upper bound ε_2, the maximum acceptable residual. An estimate is required for $\boldsymbol{S}_1^k$, one eigenvector of the band matrix. The estimate from the previous iteration, $\boldsymbol{S}_1^{k-1}$, is updated using several iterations of CG followed by an SVL procedure. Without the CG, the SVL could generate many spurious eigenvalues before converging. We require the residual of $\boldsymbol{S}_1$ to be $\varepsilon_1/1000$, and the CG alone is too inaccurate [21]. A poor approximate vector can underestimate the true residual and lead to premature termination of the BL recursions.

The selection criterion for converged vectors is determined by ε_2, the maximum allowable residual in the final answers. A convenient formula is used in step 2 to evaluate the residuals *before* the eigenvectors are constructed in step 3. This is cheaper, and more importantly, it helps to identify over-completeness.

Finally, the initial block, $\boldsymbol{Q}_1^1$, must be specified. In many applications to quantum mechanics, a good estimate for the eigenvectors is available, being constructed from formal arguments or generated in a previous computation (e.g., see Section 5.1). When such is not the case, an orthogonal set of random vectors is used (e.g., Section 4). In exact arithmetic, the BL would not obtain an eigenvector unless the initial block contained a (possibly infinitesimal) component in that direction.

3.5. *Improvements*

First, one should shun *ZHBEVX* in step 2. Since the SVL is used in step 1h to estimate $\boldsymbol{S}_1$, replacing *ZHBEVX* would appear to be trivial. However, obtaining one extreme eigenvector $\boldsymbol{S}_1$ accurately is easy compared to obtaining the entire set $\{\boldsymbol{S}_n\}$ for large m_k. A sophisticated SVL is required. The BLAS routine *ZHBMV* permits this change without restructing the storage of $\boldsymbol{T}^{\boldsymbol{k}}$.

The LAPACK [2] routine *ZHBEVX* calls *ZHBTRD* to reduce the band matrix to tridiagonal form *of full order*, $n = m_kJ_k$. The unitary matrix for the corresponding similarity transformation is constructed. This unitary matrix cannot be compactly represented, requiring $(m_kJ_k)^2$ double complex words. Ignoring the compactly stored $\boldsymbol{T}^{\boldsymbol{k}}$ and single vectors, memory must exceed $(m_kJ_k)^2 + m_kJ_k$ $(m_k + \delta m)$. This waste of memory is the primary reason why one should shun *ZHBEVX*. Notice that Ref. [2, p. 194] grossly overstates the memory required by *ZHBEVX* for the eigenvectors. The above is correct.

As given, the algorithm can compute a set of eigenvectors M much larger than m. In such an application, $\boldsymbol{C} \in C^{N\times p}$ dominates the memory requirements. The difficulty is that $\boldsymbol{C}$ is required often and should not be stored on disk. The projection operator need not be applied at every step (i.e., semi-orthogonalization instead of full orthogonalization). References [7, 8] provide criteria and techniques for less frequent use of $\boldsymbol{C}$ while maintaining comparable accuracy.

The projection operator may contain vectors obtained from other sources. Although not shown [21] the example in Section 5 benefits from separately converging "core" and "valence" states by including the complementary set of vectors in the projection operator (even when those vectors are not converged). Thus, the PBL can implement a generalization of "band-by-band minimization" of Ref. [14].

The final set of eigenvectors may be overcomplete. Linear dependence and duplication of eigenvectors arise from global loss of orthogonality in the BL. Loss of global orthogonality may be checked before each restart, by performing a QR factorization of $\boldsymbol{A} = \boldsymbol{C}_k \cup \boldsymbol{Q}_1^k$, with $\boldsymbol{A} \in \boldsymbol{C}^{N\times(p+m)}$. This additional operation was mentioned in Section 3.1, paragraph 6. Whenever we have observed this problem [21], it is preceded by a breakdown of the monotonically decreasing behavior of the quantity appearing in step 2b. We routinely abort the BL recursion when this quantity increases over three recursions. In such an event, no new eigenvectors are obtained and only the initial vectors for the next iteration are updated. In such an event, the initial QR mentioned here must be preformed.

One may perform a Rayleigh–Ritz procedure as a postprocessor. See Ref. [4, Section 11-3]. The residual matrix $\boldsymbol{R} \in C^{M\times M}$ and the subspace approximation to $\boldsymbol{H}$, *together*, require M additional operator calls. The final residuals are computed using Section 11-8 of Ref. [4]. The total cost is M operator calls (not $2M$).

3.6. *Comments*

To complete the exposition of the algorithm, we answer several questions. Why does one change the operator with each BL iteration? How can one check the final set of eigenvectors for completeness? Finally, does storing $\boldsymbol{C}$ violate our own stringent memory restrictions? Our answers rely upon some experimental evidence.

Why are projection operators necessary? Suppose, we

set $\lambda = 0$ in step 1a. If the $\boldsymbol{Q}_1^k$ contains good Ritz vectors, columns of $\boldsymbol{W}_1^k$ will zero. In finite precision, roundoff error dominates these columns and either the QR (step 1e) will fail or meaningless quantities will be propagated. The second block $\boldsymbol{Q}_2^k$ will be nonorthogonal to the first block, and this may be measured using

$$O_{12}^k = \|\boldsymbol{Q}_1^{k*}\boldsymbol{Q}_2^k\|.$$

Loss of orthogonality limits the minimum achievable residual from an iterative BL without projection operators (or other semi-orthogonalization strategy).

We test the PBL without projection operators. For the $M = m = 96$ example in Table I, the PBL with $\lambda = 0$ is applied with a fixed inner loop, $J = 8$. The Euclidean matrix norm of the residual steadily decreases for six iterations of the outer loop, to 10^{-2}. Coincidentally, the O_{12}^k increases with k to $O_{12}^7 = 10^{-4}$. Two more iterations finds the residual minimum followed by disaster as $O_{12}^9 = 10^{-1}$. The improvement/disaster cycle repeats as iterations continue (since $\lambda = 0$). With a careful choice of iteration sequence, one can achieve a residual of 10^{-4} with good efficiency, but no higher accuracy is possible from an iterative BL without projection operators.

The final set of eigenvectors may be undercomplete, overcomplete, or both. An eigenvector may have been missed, and another eigenvector may have been duplicated. A treatment for linear dependence was given in Section 3.5. We account for possible undercompleteness by setting the desired number of vectors larger than the number needed by the application. (The idea is to force one additional iteration of the outer loop.) This is usually effective, since we are starting from the extreme ends of the spectrum. However, if a complete set of interior eigenvectors are needed, a considerably more sophisticated algorithm is required [10].

How costly is storing $\boldsymbol{C}$? Total memory requirements, imply that the PBL could solve a problem ten times larger than LAPACK for $m_1 = N/800$, including storage of $\boldsymbol{C}$ but no storage is required by the operator. If $\boldsymbol{C}$ were eliminated, m could be increased by 4/3. Thus, memory is not a compelling motive for implementing a sophisticated scheme for minimizing storage of or access to $\boldsymbol{C}$.

4. MANY-BODY THEORY OF MAGNETIC DIPOLES

The first examples are drawn from many-body quantum mechanics, specifically, clusters of magnetic dipoles. The eigenvectors can determine the relaxation processes related to nuclear magnetic resonance spectra. This example is sparse and results are given for $N \leq 16384$. Dense Hermitian examples are provided in Section 5.

The most important many-body problems are electron spin models, where spin is a good quantum number and only neighbors interact. These are many-body problems for which only one or two states need to be determined. When only a few states are needed, better methods are available. Quantum Monte Carlo appears to win by substituting statistical analysis for vector size.

In contrast, the magnetic dipole examples, require a great many states to describe spin dynamics of a lattice. For some of these examples, simple operators (e.g., Sz) do *not* commute with the Hamiltonian. For applied mathematicians, this exercise is *not* block diagonalizable. (More precisely, the unitary transformation is not a direct product; see Ref. [22, Section 4.8].) The ability to block-diagonalize the matrix affects studies of iterative algorithms. For this reason, the magnetic dipole examples are better than electron spin examples.

We report experimental results using our PBL method, and we report comparisons with the simple BL method. The results reported here and in Section 5 were computed on an IBM RS/6000 computer with full compiler optimization. The IBM ESSL library provided the BLAS routines, and an optimized LAPACK was built using default specifications.

4.1. *Clusters of Magnetic Dipoles*

The quantum mechanics of clusters of magnetic dipoles is treated. This is an exactly defined mathematical exercise. The Hamiltonian is fully described in chapter three of Slichter's text [23].

These examples are less sparse than electron spin exercises. All states differing by one or two spins are directly coupled, regardless of the component spins (i.e., Sz is not conserved). Furthermore, states are coupled even if the magnetic dipoles are not nearest neighbors. A significant number of states must be obtained to describe any NMR experiment. Physically, the M should be large enough to obtain the complete band of states contributing to the line shape of the signal.

The geometry consists of clusters of 12 and 13 protons having high symmetry with hexagonal closest packing. The central proton is either excluded or included. These have $N = 4096$ and $N = 8192$. Fourteen protons use cubic closest packing with $N = 16384$. All nearest neighbors are equidistant.

The geometric symmetry allows the Hamiltonian to be block-diagonalized, since it commutes with symmetry operators in the absence of the Zeeman terms. For $N = 8192$ and 16384, the number of nonzero matrix elements per thousand are 10.6 and 6.22. All eigenvalues are doubly degenerate.

Section 4.5 provides $N = 8192$ examples including Zeeman terms for a field *not* aligned with any symmetry axis. The geometry is the same, but the Hamiltonian cannot be block-diagonalized. If the Hamiltonian were blocked by

Sz, all diagonal blocks and the off-diagonal blocks, $\Delta Sz = \pm 1$ and $\Delta Sz = \pm 2$ are dense. There are 11.9 nonzero elements per thousand.

4.2. *Behavior of the PBL Algorithm*

To begin, we illustrate the detailed behavior of the algorithm for two examples. This provides a language and specific examples for further discussion.

All examples in this section use $m = 96$ and obtain the $M = 96$ most negative eigenpairs. (As described in Section 3.3, the algorithm has been successfully applied with $M > m$.) Unless otherwise noted, all examples use $\varepsilon_2 = 20\varepsilon_1$. Only ε_1 will be reported. All calculations use $\lambda = \boldsymbol{I}$, since the Zeeman terms are absent and M is relatively small.

The first example is $N = 8192$ with "high accuracy," $\varepsilon_1 = 5 \times 10^{-7}$. The algorithm requires three iterations. The first iteration, $k = 1$, sets $m_1 = M$ and requires $J_1 = 32$ BL recursions. Memory required for the BL recursions is slightly greater than $2m_1 N = 24$ Mb. The size of the Krylov space and the dimension of $\boldsymbol{T}^{\mathbf{1}}$ are $m_1 J_1 = 3072$. Storage of $\boldsymbol{T}^{\mathbf{1}}$ requires less than 3 Mb. In step 2, M eigenvectors of the band matrix are found.

Twenty-three eigenvectors are converged; $p_1 = 23$. In step 3, m_1 vectors are transformed back to the original space to provide 23 converged Ritz vectors, and 73 approximate vectors for $\boldsymbol{Q}_1^2$. Memory required is $m_1 J_1 M < 5$ Mb for $\boldsymbol{S}^1$ plus $MN = 12$ Mb for $\boldsymbol{C}$ (including $\boldsymbol{V}$).

The second iteration uses a block size of $m_2 = 73$ and requires $J_2 = 7$ BL recursions. The dimension of the Krylov space is 511. Converged eigenvectors number $p_2 = 44$. The final iteration uses $m_3 = 29$ block size and $J_3 = 10$ recursions. All eigenvectors are accepted, and the algorithm is finished.

The second example is $N = 16384$ with high accuracy. The PBL requires three outer iterations. The first iteration uses $m_1 = 96$ block size and $J_1 = 40$ recursions. The size of the Krylov space is 3840. Eighteen eigenvectors are converged; $p_1 = 18$. The second iteration uses $m_2 = 78$ and $J_2 = 10$, and $p_2 = 59$ eigenvectors are acceptable. The final iteration uses $m_3 = 19$ and $J_3 = 13$, and finds $p_3 = 19$.

Figure 1 illustrates the second example. The logarithm of the residual for each eigenvector is shown in order, starting with the most negative eigenvalue and increasing to the right. Example two appears as the middle solid curve. For each outer iteration, the BL produces a set of vectors whose residuals increase from left to right with a power law dependence. The saw tooth shape provides a graphic illustration of the three restarts.

4.3. *Accuracy*

Figure 1 shows three PBL runs of varying accuracy for $N = 16384$. The convergence parameters, ε_1 and ε_2, are read from the smallest and largest residuals. Convergence parameters used in this section are *not* chosen for practical reasons, but rather to illustrate the algorithm. Please refer to Section 5.2.

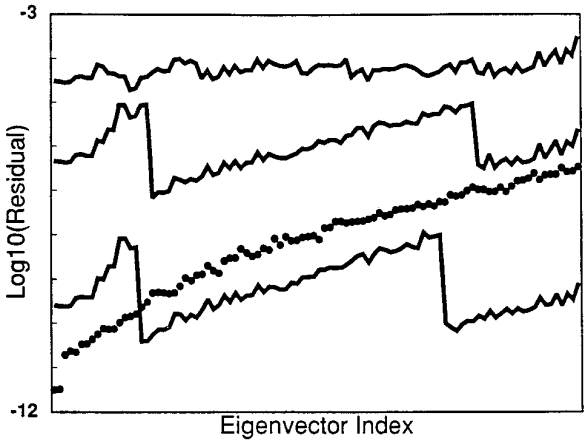

FIG. 1. Distribution of residuals for $N = 16384$ sorted by eigenvalue with most negative at left. Lines are PBL in order of accuracy: $\varepsilon = 5{*}10^{-5}$, $5{*}10^{-7}$, and $5{*}10^{-10}$. Points are BL with $\varepsilon = 5{*}10^{-11}$.

For the highest accuracy curve, ε_1 was three orders of magnitude smaller than example two. This curve appears identical, but is shifted by 3. Decreasing the ratio $\varepsilon_2/\varepsilon_1$ forces more restarts and more uniform residuals. The lowest accuracy curve demonstrates this by setting $\varepsilon_2/\varepsilon_1 = 2$, instead of 20. The eight restarts in this example are not seen, due to the uniformity of the residuals.

The figure demonstrates that any level of accuracy may be achieved reliably and uniformly. Furthermore, the least accurate run demonstrates that inaccurate eigenvectors do not cause difficulty in constructing the projection operator. Theoretically, a uniformly inaccurate residual is possible, and practically, this can be realized. Uniformly low accuracy approximations are useful when an approximate preconditioners are used. See Section 5.4. Alternatively, a low accuracy PBL could be followed by a Raleigh–Ritz procedure. See Section 3.5.

4.4. *Comparing BL, PBL, and LAPACK*

Why is an iterative BL better than a single BL recurrence? A simple BL recursion would involve one run through each step of the algorithm, without distinguishing converged and unconverged eigenvectors, $\varepsilon_2/\varepsilon_1 = \infty$. Extreme eigenvalues and regions of the spectrum with large gaps between the eigenvalues converge most quickly. Generally, the residuals are grossly nonuniform, range over many orders of magnitude, and increase with a power law dependence with increasing eigenvalue.

The dots in Fig. 1 are the residuals of the simple BL, and comparison should be made to the middle accuracy

TABLE II

The Performance of BL and PBL Algorithms

Example		$\Sigma J_k m_k$	CPU h
LAPACK	8192	na	(28.7)
BL	8192	4224	14.9
PBL	8192	3873	7.4
LAPACK	16384	na	(120.5)
BL	16384	5280	38.9
PBL	16384	4867	20.3

Note. Two magnetic dipole examples are used, the larger of which is also shown in Fig. 1. Performance is measured by the total number of Krylov vectors used and the actual CPU times for the full calculations. The *ZHBEVX* of step 2 dominates the BL timings. LAPACK timings are estimated using scaling arguments.

PBL curve, example 2. The power law dependence of the residuals is the same for both procedures, but the PBL is more uniform due to the restarts. Indeed, the BL residuals range over five orders of magnitude. The minimum residual ε_1 for the BL is four orders of magnitude smaller, but the figure shows that least accurate eigenvectors are only ten times more accurate than example 2. (The comparison is inadvertently and slightly biased.)

For similar accuracy criteria, the simple BL must iterate more, generating a larger $\boldsymbol{T}^{\boldsymbol{k}}$. Examples 1 and 2 of Section 4.2 are shown in Table II. The effort required by *ZHBEVX* scales as m^3J^2. The Krylov space in the simple BL is so large that step 2 dominates the CPU time. Table II shows that the PBL procedure is twice as fast, despite the total number of operator calls being only 10% smaller. Note that m_1J_1 for the larger PBL is 3840. The BL is estimated to be 1.89 times slower based on *ZHBEVX* alone. Using an SVL requires less space, but this ratio could be unchanged.

Comparison of BL with LAPACK can use scaling arguments. Actual timings are unfair since these examples are sparse. Assume that *ZHBEVX* scales as m^3J^2, and assume that constructing the Krylov space costs mJN^2. This is compared to N^3 for *ZHPTRD*. (Construction of eigenvectors is not included in either analysis.) For the two BL examples in Table II, the ratios of *ZHPTRD* to BL are estimated as 1.93 and 3.10. This is shown in Table II.

A logarithmic plot of the number of operator calls versus matrix size [21] shows that the performance of the PBL for this example appears to scale as $\log(N)$. This scaling is likely a result of the decreasing fraction of nonzero elements as N increases.

4.5. *Subspace Iteration*

The PBL is tested for ''band-by-band'' minimization. The Zeeman terms are included and the bands are the sets of single and double excitations.

The Zeeman terms are one thousand times larger than the dipolar terms, approximately the ratio for a proton lattice with 1 Bohr spacing in a 0.1 T field. These terms break the geometric symmetry since the field is not aligned with a symmetry axis. These examples are not block-diagonalizable, but these are computationally easier than the earlier examples, with large gaps between clusters of quasi-degenerate eigenvalues.

We compute the band of double excitations, so $M = m = 92$. For high accuracy set $\varepsilon_1 = 5 \times 10^{-6}$ and for low accuracy, set $\varepsilon_1 = 5 \times 10^{-3}$. Set $\varepsilon_2 = 2\varepsilon_1$. Eigenvectors of the Zeeman Hamiltonian provide an excellent set of initial vectors. For high accuracy, the PBL requires three iterations: obtaining the ground state, the band of single excitations, and the band of double excitations. The sizes of the Krylov spaces are 368, 368, and 552. The average residual is 3×10^{-6}, and the maximum is 4×10^{-6}.

For low accuracy, the PBL obtains both the ground state and first band on the first iteration. The Krylov space dimensions are 184 and 368. The average residual for the single and double excitations are 0.0075 and 0.0017.

A simple subspace iteration scheme is to set the initial $\boldsymbol{C}_1$ equal to the $p = 14$ Zeeman vectors for the ground state and first band. Otherwise, the algorithm is identical. Loop over execution of the PBL (triple iteration scheme) includes an extra step that redefines $\boldsymbol{C}_1$, p, and $m = M - p$. The second PBL assigns $\boldsymbol{C}_1$ to the $p = 78$ states from the first calculation of the second band. The third PBL assigns $\boldsymbol{C}_1$ to the $p = 14$ states obtained in the second calculation.

For the low accuracy calculation, each subspace iteration uses only one PBL iteration (i.e., PBL = BL). Except for the first, each subsequent PBL uses only $J = 2$. The average residual of the double excitations are 0.676023 and 0.676017 after the first and third PBL executions. However, no further improvement is seen after the fifth PBL. Clearly, a subspace iteration scheme must be more sophisticated.

5. DENSITY FUNCTIONAL THEORY

The second example is the density functional theory of diatomic beryllium and Be_{13} clusters, using plane waves and including all electrons. This example is a dense, complex Hermitian, with all the matrix elements nonzero. However, the high symmetry of the example generates a block-diagonalizable matrix. We include the ''Trotter product'' form of the quantum time propagator [13] as applied to plane wave density functional theory [24]. However, unlike section 4.5, the examples of Section 5 *can* be block-diagonalized.

Quantum mechanics often results in ''stiff'' eigensystems, with the eigenvalue gap varying as N (e.g., free waves and angular momenta). The kinetic energy dominates the uninteresting portion of the spectrum, resulting in eigenvalue gaps increasing linearly with the value. Yet, the singu-

larities at the ionic sites require large numbers of plane waves to converge. These are numerically difficult exercises.

The first section describes the model in sufficient detail that the exercise could be reproduced. An important discussion of the relationship between residuals and eigenvalue error includes practical means for choosing convergence parameters.

5.1. *Beryllium Clusters*

The example is constructed to be physically interesting and easily reproducible. These are contradictory goals. We simplify the physics, looking at a model for an isolated cluster, while preserving the major aspects of computational solid-state physics. The example does not include pseudo-potentials, Ewald sums, or Bloch waves. The explicit optimization of core states in a relatively large unit cell requires a large basis, making this example more numerically difficult than the typical electronic structure problem.

The beryllium atoms are located in a cube of length 10 a_0 (atomic units). All plane waves are chosen within an energy sphere corresponding to a kinetic energy cutoff of 19.34 Hartrees and $N = 4067$. The plane wave basis is not symmetry-adapted. The potential energy is the bare Coulomb interaction. Analytic form factors are used, implying that the electron density over all space interacts with the nuclei. The repulsion potential also uses these analytic form factors and a directly computed Fourier transform of the electron density. The diagonal matrix elements of the ionic and repulsion potentials are assumed to cancel exactly. The standard local exchange potential is used, together with a local correlation potential [25]. These are evaluated on a real space grid of size 80^3 and then Fourier transformed.

The self-consistent field (SCF) iterations are unstable due to polarization fluctuations of the weakly coupled core states. The best way to eliminate this difficulty is to explicitly block-diagonalize the Hamiltonian and separately solve for each symmetry species. Our focus is upon defining a reproducible example that mimics more complex, condensed-phase exercises. Focusing upon the applied mathematics instead of the physical exercise, we solve the polarization fluctuation difficulty in the usual manner, averaging the instantaneous electron density with the averaged density from the previous SCF iteration. Symmetry is not used, except that the averaged density is required to be inversion symmetric.

Initial vectors are very important in this example because of the SCF procedure. We use a set of fully converged states of the atom using the same basis. Five functions are shifted to each atomic center using structure factors. The resulting set is orthogonalized, sorted, and truncated by a Rayleigh–Ritz procedure applied to the kinetic plus ionic potentials (i.e., no density dependent terms).

The operator for this example is beautiful, a convolution plus the kinetic energy. However, since the plane wave basis is truncated and sorted, the convolution requires indirect addressing. Very little storage is required, as compared to a full matrix representation, but the elements of the potential vector are not accessed sequentially. Unrolling the inner loop over the block size provides a large improvement in performance, a factor of four for $m=10$ on our IBM RS6000 workstation.

5.2. *Interpretation of Convergence Criteria*

The convergence criterion, algorithm step 1i, uses the residual. However, the residual scales with eigenvalue scaling; dividing H by two cuts the residual in half. Thus, specification of the convergence parameters requires an interpretation of convergence that is independent of scaling.

The naive interpretation of the residual R_n is that it places error bars on the computed eigenvalue Λ_n. For isolated Λ_n the true eigenvalue lies in the interval. However, this grossly overestimates the eigenvalue error. We demonstrate a well-known theorem that places rigorous bounds on the eigenvalue error, and provides an interpretation of the convergence parameters ε_1 and ε_2 which is independent of scaling.

The rigorous bound for isolated Λ_n is given by the square of the residual divided by the gap between the true eigenvalue and the nearest eigenvalue to which it is coupled. See Ref. [4, Section 11-7]. The eigenvalue error Δ_n is approximated by R_n^2/γ, where $\gamma = |\Lambda_n\text{-}\Lambda_m|$. A relative error defined by R_n^2/γ^2 provides a convergence criterion independent of scaling. This relative error is interpreted as a measure of the uncertainty of the ordering of the spectrum. For density functional theory, this measures the uncertainty in orbital occupation assignments. So, a relative error exceeding one for any state implies complete uncertainty in assignments of occupations (if based solely upon the eigenvalue information). One could automate this convergence criterion by obtaining estimates of several eigenvalues and their residuals at steps 1h and 1i in the PBL algorithm.

We provide practical means for observing and interpreting the relationship between Λ_n, R_n, and Δ_n. A sensible choice of ε_1 and ε_2 can be made without adopting a more complicated convergence estimator. Diatomic beryllium, oriented along (111), provides an illustration.

A sample of eigenvalue errors and residuals is obtained from step 1i of the BL. The residuals associated with $\boldsymbol{S}_1^k$ during the BL recursion are immediately available. The corresponding eigenvalue $\Lambda_{p+1} = \Lambda_1^k$ is available at the end of the current BL recursion. The eigenvalue error is the

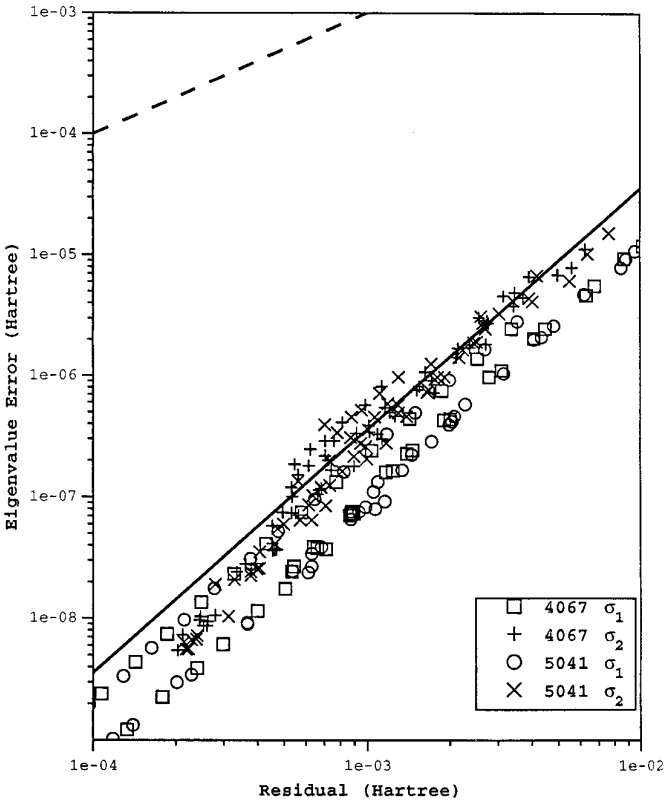

FIG. 2. Eigenvalue errors, residuals, and the choice of convergence parameters. The Be_2 example was fully SCF converged at extraordinarily high accuracy for $N = 4067$ and 5041 to obtain a large sample for analysis. The nature of the example dictates that $p = 1$ and $p = 3$ were obtained at every SCF iteration. These four sample sets are presented with different symbols. Yet, it is unnecessary to resolve individual points. The solid and dashed lines illustrate two theorems relating eigenvalue error to residual. This method of analysis permits a consistent choice for ε_1 and ε_2 for arbitrary scaling, αH.

difference between the converged Λ_{p+1} and the estimate obtained using $\boldsymbol{S}_1^k$ during the BL recursion. An unbiased sampling of pairs (R_{p+1}, Δ_{p+1}) is computed for fixed p after the BL recursion, and various p are obtained as the outer loop over k progresses.

In Fig. 2, the solid and dashed lines represent the two theorems relating the eigenvalue error to the residual. The solid line $\Delta_1 = R_1^2/\gamma$ illustrates the accurate interpretation of the residual. The naive interpretation says that the eigenvalue uncertainty must lie below the dashed line. Certainly, this is true. However, the weakness of this upper bound is apparent.

The solid line uses a single $\gamma = \Lambda_3 - \Lambda_1$ from the final SCF of $N = 5041$. (The first two states, σ_1 and σ_1*, are uncoupled.) Properly, each eigensystem (each SCF iteration) should be graphed separately with a different value for γ. On the logarithmic plot, this changes the position of the line without changing its slope. The slope accurately represents the data while its position is not perfect. For sets of related eigensystems, as in the example, only a rough estimate of the magnitude of γ is required.

The figure illustrates a consequence of the existence of matrices that commute with $\boldsymbol{H}$, allowing it to be block diagonalized. The plus signs and exes are sample sets taken from the second outer loop, $k = 2$ and $p = 2$. The projection operator forces the converged states to zero eigenvalues for these eigensystems. The corresponding gap is the eigenvalue difference between σ_2 and σ_3. This is five times larger than the apparent gap in the spectrum. The apparent gap would lead one to choose an ε_1 five times smaller than necessary. Conversely, one might be misled by this example to choose a large ε_1 for an example without symmetry.

Precise choices of ε_1 and ε_2 are nontrivial, requiring *a priori* knowledge of gaps in the eigenvalue spectrum. The figure demonstrates that only a rough estimate of the gaps is necessary for related eigensystems. However, one should not relate symmetric and nonsymmetric examples. An unrecognized ability to block-diagonalize the problem causes one to choose smaller parameters than necessary, which generates more work but no serious problems.

5.3. *Benchmark*

Figure 3 illustrates the benchmark calculation for diatomic beryllium. The performance of the algorithm is measured by the number of calls to the block operator, $m = 10$. The curve shows the convergence criterion for each step in the BL recursion. Literally, the abscissa measures CPU time linearly, in a machine-independent manner. The ordinate shows the progress towards converging unspecified eigenstates.

Interpreting the abscissa as time, the residual corresponds to the lowest eigenvalue that has not yet been converged. Restarts of the outer loop and the progression of SCF iterations appear as vertical jumps, nearly discontinuous breaks in the curve. Vertical dashed lines identify the beginnings of SCF iterations, and peaks without dashed lines correspond to the beginnings of PBL iterations. For instance, the calculation begins on the left, and four PBL iterations are required to obtain the eigenvectors for the first SCF iteration. Within the first SCF, the amount of time required by each PBL iteration varies. The first iteration is shown by the residual of the ground state, σ_1. The second PBL iteration within the first SCF iteration is shown by the residual of the third state, σ_2. (If only the ground state residual were shown, the curve would not vary during the subsequent PBL iterations.)

Our recommendation for the use of the PBL in an SCF procedure is to increase the accuracy as the SCF progresses. Horizontal dashed lines show the values of the convergence parameters ε_1 and ε_2. In this example, the convergence parameters are changed once, after four SCF

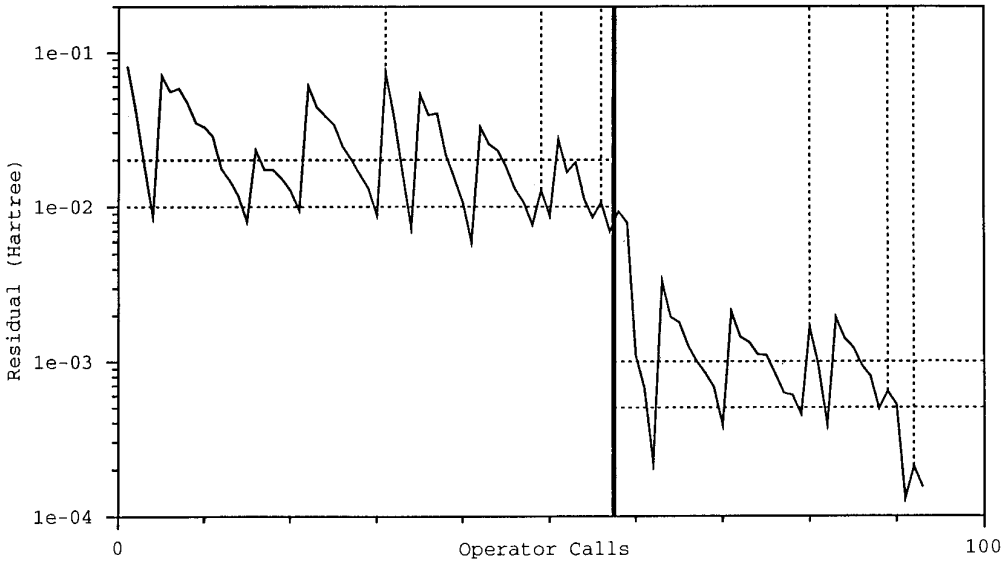

FIG. 3. Convergence during PBL and SCF iterations for Be_2 example. The curve is the convergence criteria, algorithm step 1i, for each call to the block operator. BL recursions stop when a specified accuracy is achieved. This is indicated by the curve crossing the lower horizontal line $R = \varepsilon_1$. Initial vectors for the BL have $R > \varepsilon_2$, the upper horizontal line. Vertical dashed lines indicate beginnings of SCF iterations.

iterations, as marked by the bold black line. The change in parameters was dictated by the total energy being converged within 0.001 Hartrees between the beginning and end of the fourth SCF iteration.

The ''preprocessor'' is the initial SCF using low accuracy eigenvectors and requesting moderate SCF convergence. The SCF procedure is restarted with higher accuracy using the results of the preprocessor as initial vectors. The PBL parameters are twenty times smaller. The convergence of the total energy during the final SCF is 10^{-6} Hartrees (50 parts per 10^9). Four SCF iterations are required using the high accuracy PBL. Note, the total energy converges faster than the individual eigenvalues during the SCF iterations. The ''history mechanism'' used to avoid polarization fluctuations is reinitialized when the SCF is restarted.

In the last two SCF iterations, the curve lies entirely below ε_2. The initial ground state eigenvector is sufficiently accurate to meet the convergence criteria. In other words, no PBL work is needed, but a minimum number of two BL recursions is required since $\delta m \neq 0$ in algorithm step 2a. The final two eigensystems are nearly identical.

The number of converged eigenvectors, M, obtained greatly affects the performance. Ten states were converged for each SCF in the preprocessor, but during the final SCF only four states were obtained. Throughout, the block size is fixed at 10. This makes good use of all the initial vectors, but in practice, one would reduce the block size to four after the preprocessor. The reasoning is that good initial vectors improve PBL performance, and the preprocessor is quite fast. So, $M = m$, and the 10 vectors comprising the initial block are all converged vectors from the previous SCF iteration. In the final SCF, only four states are converged, and each initial block consists of four converged vectors plus six unconverged vectors from the previous SCF.

The advantage of a large set of good initial vectors must be weighed against the cost of obtaining them. The cost outweighs the benefit unless the eigensystems are very closely related. No further analysis of block size or numbers of converged vectors is provided. Suffice it to say that this is the benchmark.

5.4. *Preconditioners and Spectral Transformations*

As shown in Fig. 3, the initial SCF can be converged with low accuracy vectors. Can these low accuracy eigenvectors be obtained more efficiently? In this section, we compare numerical methods based upon solving related eigensystems. We consider ''preconditioners'' which act upon the $\boldsymbol{H}$ and ''spectral transformations'' that are more complicated functionals of $\boldsymbol{H}$.

The shift transformation or resolvent (cf. Ref. [22]), $1/(\boldsymbol{H} - \Lambda)$, is well known. Eigenvalues near Λ appear at the extremes of the transformed spectrum, gaps between them are magnified, and uninteresting portions of the spectrum are made quasi-degenerate. For these reasons, the BL will converge more quickly. What is more important is that this method obtains interior eigenvalues. For quantum physics exercises, interior eigenvalues are not required. The effort required to form the inverse is often prohibitive. However, memory storage is the primary hindrance for exercises where a full matrix would not otherwise be needed.

TABLE III

Approximate Preconditioners and Spectral Transformations

Label	Type	Form	Function	
D0	Diagonal variable	$\alpha\mathbf{d}(\mathbf{H}\text{-}\beta)\mathbf{d}$	$\mathbf{d}^{-2} = \mathbf{h}\text{-}\beta\text{-}\Lambda$	$\Lambda = \mathrm{v}^*\mathbf{H}\mathrm{v}$
D1	Diagonal fixed		$\mathbf{d}^{-2} = \mathbf{h}\text{-}\beta\text{-}\Lambda$	$\Lambda = -20$
D2	Diagonal function	$\alpha\mathbf{d}\mathbf{H}\mathbf{d}$	$\mathbf{d}^{-2} = f(\mathbf{h}/\Lambda)$	$\Lambda = \mathrm{v}^*\mathbf{h}\mathrm{v}$
E0	Exponential	$\alpha\exp(\gamma\mathbf{h})\exp(2\gamma\mathrm{V})\exp(\gamma\mathbf{h})$		$\gamma = -.01$
E1	Exponential	$\alpha\mathbf{H}\exp(\gamma\mathbf{h})\exp(2\gamma\mathrm{V})\exp(\gamma\mathbf{h})$		$\gamma = -.02$
A1	Band approximation	$\alpha\mathbf{H}'$	$\mathbf{H}' = \mathrm{band}(\mathbf{H})$	
B0–B9	Band preconditioner	$\alpha\mathbf{L}^{-1}(\mathbf{H} - \beta)\,\mathbf{L}^{-1*}$	$\mathbf{H}' - \beta - \Lambda = \mathbf{L}\mathbf{L}^*$	
BB0,1	Band/band	$\alpha\mathbf{L}^{-1}(\mathbf{H}' - \beta)\,\mathbf{L}^{-1*}$		
C0–C3	Band matrix shift	$\alpha\mathbf{L}^{-1*}\mathbf{L}^{-1}$		
S0	Series approximation	$2\alpha\mathbf{B}^{-1}\text{-}\alpha\mathbf{B}^{-1}(\mathbf{H}\text{-}\beta\text{-}\Lambda)\mathbf{B}^{-1}$	$\mathbf{B} = \mathbf{H}'\text{-}\beta\text{-}\Lambda$	

Note. The low-accuracy preprocessor for density functional theory might benefit from altered forms of the operator. The forms are given and labeled. D2, E0, and E1 are specific to density functional theory. E0 and E1 are new. D2 is from Teter *et al.* [14] and D0 is the Davidson preconditioner [11, 12]. All others are general and obvious approximations to the shift operator or resolvent. $\mathbf{h} = \mathbf{H}_{gg} = |g|^2/2$ is a real diagonal matrix (kinetic energy only). Most forms include a spectral shift $\beta = 10$ so that the smallest eigenvalue of $\mathbf{H} - \beta$ exceeds -12.2 Hartrees. The $\mathbf{v}$ appearing in D0 and D2 is the initial vector indexed $p_k + 1$ and is changed at the beginning of each BL recursion indexed by k. The function f in D2 is from Ref. [14]. E0 and E1 use an approximate form of $\exp(\mathbf{h} + \mathbf{V})$; see Ref. [13]. Matrix $\mathbf{H}'$ is band matrix extracted from $\mathbf{H}$. Matrix $\mathbf{L}$ is lower triangular, obtained by Cholesky factorization using *ZPBTRF* of Ref. [2]. $\mathbf{L}^{-1}$ uses *ZTBTRS, op cit.* S0 is a truncation of a power series that converges if and only if $\mathbf{L}^{-1}(\mathbf{H} - \beta - \Lambda)\mathbf{L}^{-1*}$ has $|\lambda| < 1$ for all eigenvalues λ, which is certainly not true.

Approximations are often used. Table III defines several approximate preconditioners and spectral transformations studied in this paper.

The diagonal shift preconditioner is labeled D0 and is the Davidson preconditioner [12]. When Λ is an eigenvalue of H, the corresponding eigenvector could be obtained in the proper limit [11]. The method is extremely sensitive to the approximate eigenvalue $v^*\boldsymbol{H}v$ available at the beginning of the Lanczos recursion. This is a poor choice for block algorithms, and yet we test it with $m = 10$. The restarts of the BL are indexed by k, and for each, the initial vectors are different. The Λ used in D0 are the values available at the beginning of the PBL. For instance, if $k = 2$ and $p_1 = 3$, then Λ is the approximate values of the third initial vector. These values are not updated as k increases, since the updated initial vectors are less accurate (in our example).

Using a fixed shift, as in D1, is a conservative simplification of Davidson's approach. Unlike D0, it cannot obtain an eigenvector given an exact eigenvalue. However, when only poor eigenvalues are available, it is less severe a change to the original matrix, while retaining the other advantages. The fixed shift value of -20 is comparable to the most negative value of -12.2 (including the $\beta = 10$ spectral shift). In other preconditioners, we will use a shift of -50.

For density functional theory, a diagonal preconditioner, D2, introduced in Ref. [14] contributed significantly to the success of their conjugate gradient algorithm. The elements of $\boldsymbol{d}$ are a nontrivial function of the diagonal elements $\boldsymbol{H}_{nn}$ and the energy $\boldsymbol{v}_n^*\ \boldsymbol{h}\ \boldsymbol{v}_n$ with $\boldsymbol{h}$ being the diagonal of $\boldsymbol{H}$. Such functions are guaranteed to be positive definite.

An approximate exponential transformation, $\exp(-\beta\boldsymbol{H})$, was successful in Ref. [13] in difficult single electron exercises. We extend the transformation to density functional theory by numerically exponentiating the inverse Fourier transformation of the full potential, including the repulsion and exchange correlation terms. The result is a local potential $\exp(-\beta V)$ that is Fourier transformed and applied as a convolution. The resulting operator is applied alone, as a spectral transformation in E0, or as a preconditioner in E1.

The final preconditioners and spectral transforms make a band matrix approximation $\boldsymbol{H}'$ to the original $\boldsymbol{H}$. Appropriate choice of shift Λ guarantees a positive definite matrix, and Cholesky factorization provides the lower triangular matrix $\boldsymbol{L}$ with $\boldsymbol{H}' - \beta - \Lambda = \boldsymbol{L}\boldsymbol{L}^*$.

The band matrix approximation is used in five ways. For A1 the approximation is used alone. For B0 to B9, it is used as a preconditioner. In BB0 and BB1 the approximate preconditioner is combined with the approximation A1. In C0 to C3, a spectral transform computes an approximate matrix inverse. Finally, in S0 it uses an approximate inverse derived from a series approximation.

5.5. *Performance*

Eighteen preconditioners and spectral transforms are evaluated for the Be_2 example. The hope is to accelerate the computation of electronic states. These changes to the eigenvector problem alter the spectra and affect the performance of the PBL. However, each eigenvector problem is embedded in an SCF procedure, and the accuracy of the computed vectors affects the overall performance.

Table IV measures the performance and accuracy of the benchmark and 18 alternative formulations. Forms based upon a band matrix approximation depend upon the band width $2N_B + 1$. For Hermitian matrices, N_B is the number of subdiagonal bands (e.g., $N_B = 1$ for a tridiagonal). Additional storage is $(N_B + 1)N$. For $N_B = N/2$, the storage required for $\boldsymbol{L}$ equals the storage for a full matrix inverse, and the band matrix approximation becomes impractical.

A shift value of -20 is approximately the least negative value making all matrices diagonally dominant. This value increases when necessary, as in all $N_B = 2000$ examples. Indeed, both shifts, -20 and -50, are reset to -59 in the fifth and final SCF iteration of the preprocessor in all $N_B = 2000$ examples. Our use of diagonal dominance comes from the Gersgorin theorem (Ref. [22, Section 10.6]). Like the two interpretations of the residual, diagonal dominance is a weak bound on positive definiteness, and all Λ are too negative.

Scale factors α are extremely important in this study. The residual scales with arbitrary scaling of the matrix. See Section 5.3. We remove bias due to scaling implicit in each transformation. If our convergence criteria made use of calculated gaps in the spectra, this would be unnecessary. Since our convergence is based upon the weaker theorem, we must scale the matrices so that the gaps between the eigenvalues of interest are unchanged. In the context of Section 5.3, the eigenvalue error depends differently upon the residual for *each* transformation.

This scaling greatly affects the performance, but it attempts to achieve comparable accuracy for widely different eigenvector problems. For example, if C0 did not include $\alpha = -15816$, the convergence criteria (with unchanged parameters) would be met by every initial vector. C0 would appear to be perfectly efficient. However, the vectors for use in the postprocessor would be garbage.

For a fair comparison, we do not accept poor quality solutions from one transformation. If C0 were taken without scaling, then the convergence parameters for the other transformation should be weakened. In particular, the benchmark, A0, would perform much better for the calculation of lower quality vectors.

Performance is the objective. Q is the ratio of the cost of the transformed operator to the original. Thus, $Q = 1.0$ for the diagonal preconditioners; essentially no extra work is required. For the band approximations, we ignore the cost of the Cholesky factorization. Considering only the steps required explicitly in the operator, the band matrix inverse transformations C0, C1, and C2 are cheaper than their corresponding preconditioners, B0, B1, and B2.

The cost of applying the exponential operator is the same as applying $\boldsymbol{H}$, since both use the convolution. Thus, E1 is twice as expensive as E0. To avoid further escalation of cost the exponential is applied only once, and E1 is not Hermitian since the exponential is evaluated approximately. We have a good approximation, and so, this poses no problems in our examples.

The total cost C of reaching the same SCF-converged answer at the same accuracy is the weighted sum of the preprocessor plus the postprocessor. The alternatives are roughly ordered by total cost, but also they are grouped by technique and size.

All alternatives are more costly, because the quality of the preprocessor vectors is not sufficient for the postprocessor. For A1 and C2, the preprocessor is faster than the benchmark, but excessive numbers of iterations are required in the postprocessor. For such electronic structure problems, none of the transformations accelerates the overall process of reaching the final answer.

The residuals and overlaps indicate in more detail the failure of all the transformations. The residuals are for $\boldsymbol{H}$ (not the transformed operator) at the beginning of the postprocessor. The scaling should make these comparable, but the differences are substantial because of the SCF. The strengthening of PBL convergence parameters for the preprocessor would not improve these residuals. Weakening the parameters may make the residuals more uniform, but the SCF may be damaged. Certainly, weakening PBL convergence would favor the benchmark, which already performs very well. The five SCF steps in the preprocessor for the alternatives are the principal source of the errors. Perhaps, preprocessors with only one SCF step and weakened PBL convergence may show better performance.

In all cases, except D2, the core states are the problem. The valence states are reasonably accurate for all except D2, C0, C1, and C2. The accuracies of E0, E1, B0, and B1 are high. However, these are expensive. The poor accuracy of A1 was unexpected and contributes directly to the poor accuracy of BB0, BB1, C0, C1, and C2. The series approximation S0 is worse than its counterpart B3.

The overlaps measure the degree of self-consistency relative to the true Hamiltonian. C2 is a disaster, with the valence orbital of the preprocessor having no overlap with the true valence orbital. The result is the largest effort required in the postprocessor. The correlation between the overlaps and the efficiency of the postprocessor is stronger than the correlation with residuals (cf. BB0 and B2). Apparently, very high self-consistency is required for efficient postprocessing.

Finally, two pairs in Table IV prove that our scaling and our implementation of the transformations are correct. C0 and C1 have identical residuals and overlaps, indicating that the preprocessor vectors are nearly identical. Yet, these are very different calculations. The Λ, α, and the numbers I_{OP} are significantly different. Similarly, BB0 and BB1 are nearly identical except for Λ, α, and I_{OP}. The equivalence of these is theoretically required, since these

TABLE IV

Performance and Accuracy of Preconditioners and Transformations

	Parameters			Performance				Accuracy				Self-consistency			
	N_B	Λ	α	Q	I_{OP}	J_{OP}	C	Residuals*1000				Overlap*1000			
A0	4066	0	1.0	1.00	57	36	93	9	12	9	14	1000	1000	999	999
E0	4066	0	-43.8	1.00	55	46	101	266	267	48	18	998	998	999	999
E1	4066	0	0.746	2.00	41	47	129	165	164	35	32	1000	1000	1000	998
A1	2000	0	1.0	0.75	60	54	99	1767	1754	319	163	985	984	999	999
B0	2000	-50	61.4	1.75	57	65	165	373	383	81	68	994	992	984	973
B1	2000	-20	23.89	1.75	50	70	158	363	370	80	69	994	993	985	978
BB0	2000	-50	61.4	1.50	58	68	155	1769	1759	344	176	974	971	976	971
BB1	2000	-20	23.89	1.50	56	68	152	1768	1760	342	172	974	972	977	975
C0	2000	-50	-15816	0.75	104	110	188	2125	2084	582	513	765	744	807	605
C1	2000	-20	-10893	0.75	92	110	179	2125	2084	582	513	765	744	807	605
B2	600	-50	61.4	1.29	57	74	148	762	776	152	104	979	975	973	962
B3	600	-20	29.2	1.29	65	76	160	1024	1010	209	131	963	957	958	944
C2	600	-50	-3882	0.29	75	127	149	2380	2318	618	681	638	616	786	0
S0	600	-20	-1841	1.58	87	98	235	1731	1729	398	250	897	887	921	876
B4	20	-50	86.8	1.03	69	83	154	891	915	242	158	954	941	967	904
B5	20	-20	105.6	1.03	123	109	236	1607	1528	580	366	810	767	902	737
D0	0	na	46.4	1.00	118	96	214	1406	1381	365	346	870	854	829	643
D1	0	-20	40.1	1.00	67	94	161	1055	1029	263	206	936	924	921	817
D2	0	na	0.673	1.00	61	106	167	459	459	1396	1799	997	994	730	544

Note. Various alternative operators are evaluated for the low-accuracy preprocessor for the diatomic Beryllium example. The labels and parameters refer to the forms given in Table III. Performance of the preprocessor is measured by I_{OP}, the number of calls to the block operator, and Q, the times per operator call relative to the full operator **H**. The gross cost to reach the final answer is $C = QI_{OP} + J_{OP}$, where J_{OP} is the number of calls required to SCF-converge the high-accuracy, postprocessor. Accuracy of the preprocessor affects J_{OP} and the number of SCF iterations (which is roughly proportional to J_{OP}). Accuracy of the four vectors occupied at the beginning of the postprocessor are directly measured by their **H**-residuals. Overlaps measure the degree to which those vectors are self-consistent. N_B is the number of sub-diagonals, and Λ is fixed shift. Most scale factors α are chosen by requiring the $\sigma_2 - \sigma_1$ gap in the initial vectors to be approximately at 2.8 Hartrees for transformed operators. α for B0 and BB0 are taken from B2. A0 is the benchmark. "na" is "not applicable." Four vectors v_n from the preprocessor provide a density for construction of **H**. Residuals are $|\mathbf{H}\, v_n - \lambda_n v_n|$ without diagonalization or history averaging. **H** is diagonalized to obtain $\mathbf{w}_n$. Overlaps are $\Sigma_m \|\mathbf{w}_n{}^* \mathbf{v}_m\|^2$ with the sum running only over the four initial vectors used to construct **H**.

TABLE V

Performance for the Be_{13} Example

Method	*IOP*	*JOP*	max(R)	avg(R)
A0	99	96	0.033	0.017
E0	88	135	0.271	0.163
E1	75	126	0.277	0.165
B2	84	238	2.973	1.824
D0	201	259	1.351	1.009
D2	93	138	1.673	0.989

Note. The preconditioners and transformations correspond to those given in Tables III and IV. The scale factors are unchanged. Performance is measured by the numbers of calls, *IOP* and *JOP*, to the $m = 30$ block operator. Accuracy is measured by the maximum and average residuals of the $M = 26$ preprocessor vectors at the beginning of the postprocessor.

only use the band matrix $\boldsymbol{H}'$ with no knowledge of $\boldsymbol{H}$. The dependence of I_{OP} on Λ is as expected. If Λ were always the eigenvalue for the eigenvector being converged, these methods would be extremely fast. However, the postprocessor would be unchanged.

5.6. *Larger Examples*

To conclude, we apply selected techniques to a larger problem, a Be_{13} cluster with bond length of 2 a_0 in a face-centered cubic arrangement in the highest symmetry orientation. Most details are the same as Be_2, except that the preprocessor is fixed at 10 SCF iterations and the history mechanism uses a 1/4 ratio of new to old densities. Table V shows the performance and accuracy.

Again, none of the preconditioners and transformations are more efficient than the benchmark. Interestingly, most are faster in the preprocessor. The accuracies are comparable to those in Table IV. Again, the correlation between accuracy and JOP is not good. Overlaps are more difficult to analyze, but correlate better.

Finally, we use the PBL in the method A0 everyday in electronic structure calculations similar to, but bigger than, the Be_{13} cluster. We also use pseudo-potentials and have tested the method for Hartree–Fock calculations on molecules. Future publications of chemical and physical calculations from our group will use this procedure.

ACKNOWLEDGMENTS

This entire work was supported by the National Science Foundation, Presidential Young Investigator Award to FW. Computation is supported by startup funds provided FW by the Chemistry Department, and by an HESD award from IBM to FW.

REFERENCES

1. C. Lanczos, *J. Res. Nat. Bur. Stand.* **45,** 255 (1950).
2. E. Anderson, Z. Bai, C. Bischof, J. W. Demmel, J. J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. Mckenney, S. Ostrouchov, and D. Sorensen, *LAPACK Users' Guide* (SIAM, Philadelphia, 1992).
3. J. K. Cullum and R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vols. 1, 2* (Birkhauser, Boston, 1985).
4. B. N. Parlett, *The Symmetric Eigenvalue Problem* (Prentice–Hall, Englewood Cliffs, NJ, 1980).
5. Y. Saad, *Linear Algebra Appl.* **34,** 269 (1980).
6. Y. Saad, *Numerical Methods for Large Eigenvalue Problems* (Manchester Univ. Press, Manchester, 1992).
7. B. N. Parlett and D. S. Scott, *Math. Comput.* **33,** 217 (1979).
8. H. D. Simon, *Linear Algebra Appl.* **61,** 101 (1984).
9. T. Ericsson and A. Ruhe, *Math. Comput.* **35,** 1251 (1980).
10. R. G. Grimes, J. G. Lewis, and H. D. Simon, *SIAM J. Matrix. Anal. Appl.* **15,** 228 (1994).
11. R. B. Morgan and D. Scott, *SIAM J. Sci. Comput.* **4,** 585 (1993).
12. E. R. Davidson, *J. Comput. Phys.* **17,** 87 (1975).
13. F. Webster, P. J. Rossky, and R. A. Friesner, *Comput. Phys. Commun.* **63,** 494 (1991).
14. M. P. Teter, M. C. Payne, and D. C. Allan, *Phys. Rev. B* **40,** 12255 (1989).
15. M. P. Nightingale, V. S. Viswanath, and G. Muller, *Phys. Rev. B* **48,** 7696 (1993).
16. Note, we do not distinguish between minimization and maximization, since only the differences between eigenvalues determine convergence.
17. E. Dagotto and A. Moreo, *Phys. Rev. D* **31,** 865 (1985).
18. A. Nauts and R. E. Wyatt, *Phys. Rev. Lett.* **51,** 2238 (1983).
19. K. A. Gallivan, R. J. Plemmons, and A. H. Sameh, *SIAM Rev.* **32,** 54 (1990).
20. J. J. Dongarra, J. Du Croz, I. S. Duff, and S. Hammarling, *ACM Trans. Math. Software* **16** (1990).
21. A few conclusions are drawn from numerical experiments not reported in this paper.
22. P. Lancaster and M. Tismenetaky, *The Theory of Matrices* (Academic Press, New York, 1985).
23. C. P. Slichter, *Principles of Magnetic Resonance* (Springer-Verlag, Berlin, 1985).
24. G. C. Lo and F. Webster, in *Proceedings, Sixth SIAM Conf. on Parallel Processing for Scientific Computing, 1993,* edited by R. F. Sincovec *et al.,* p. 142.
25. Vosko, Wilk, and Nusair, *Can. J. Phys.* **10,** 123 (1980).