A Shift Selection Strategy for Parallel Shift-Invert Spectrum Slicing in Symmetric Self-Consistent Eigenvalue Computation

DAVID B. WILLIAMS–YOUNG, Lawrence Berkeley National Laboratory
PAUL G. BECKMAN, University of Chicago
CHAO YANG, Lawrence Berkeley National Laboratory

The central importance of large scale eigenvalue problems in scientific computation necessitates the development massively parallel algorithms for their solution. Recent advances in dense numerical linear algebra have enabled the routine treatment of eigenvalue problems with dimensions on the order of hundreds of thousands on the world's largest supercomputers. In cases where dense treatments are not feasible, Krylov subspace methods offer an attractive alternative due to the fact that they do not require storage of the problem matrices. However, demonstration of scalability of either of these classes of eigenvalue algorithms on computing architectures capable of expressing excessive parallelism is non-trivial due to communication requirements and serial bottlenecks, respectively. In this work, we introduce the SISLICE method: a parallel shift-invert algorithm for the solution of the symmetric self-consistent field (SCF) eigenvalue problem. The SISLICE method drastically reduces the communication requirement of current parallel shift-invert eigenvalue algorithms through various shift selection and migration techniques based on density of states estimation and k-means clustering, respectively. This work demonstrates the robustness and parallel performance of the SISLICE method on a representative set of SCF eigenvalue problems and outlines research directions which will be explored in future work.

CCS Concepts: •**Mathematics of computing → Mathematical software performance;** •**Applied computing → Mathematics and statistics;**

Additional Key Words and Phrases: Eigenvalues, Parallel Eigenvalue Algorithms, Self-Consistent Field, Shift-Invert Spetrum Slicing

ACM Reference format:
David B. Williams–Young, Paul G. Beckman, and Chao Yang. 20xx. A Shift Selection Strategy for Parallel Shift-Invert Spectrum Slicing in Symmetric Self-Consistent Eigenvalue Computation. *ACM Trans. Math. Softw.* 0, 0, Article 0 (March 20xx), 34 pages.
DOI: xx.xxxx/xxxxxx

1 INTRODUCTION

Large-scale symmetric eigenvalue problems arise in many types of scientific computation (Yang 2005). In the case of electronic structure calculations based on the Hartree–Fock approximation or Kohn-Sham density functional theory, a large symmetric nonlinear eigenvalue problem must be iteratively solved through what is known as the self-consistent field (SCF) procedure (Szabo and Ostlund 2012). Typical methods to solve the so-called SCF eigenvalue problem require the partial diagonalization of a sequence of matrix pencils where each pencil of the sequence is generated using a subset of the eigenvectors of the previous pencil. The SCF problem is considered solved

DOI: xx.xxxx/xxxxxx

when convergence of the sequence is achieved, i.e. the change in the matrix pencil (or equivalently, the desired eigenvectors of said pencil) between two subsequent iterations of the SCF procedure falls below a specified threshold. The repeated diagonalization required by the SCF procedure is often the computational bottleneck in large scale electronic structure calculations (Banerjee et al. 2018; Jay et al. 1999; Shepard 1993), especially in cases where a large number of computational resources are available. As such, methods must be developed to efficiently solve this class of nonlinear eigenvalue problem on modern, massively-parallel computing architectures.

One of the hallmarks of the SCF procedure is that the desired eigenpairs need not be computed to full accuracy before convergence is reached. However, as the sequence progresses, an increasing level of accuracy in the desired eigenpairs is needed to ensure convergence to the proper SCF solution. Although the spectrum of the matrix pencils may change quite a bit in the first few iterations of the SCF procedure, this change becomes progressively smaller as convergence is reached. This feature makes the design and implementation of algorithms for solving the SCF eigenvalue problem somewhat different from traditional algorithms for solving an eigenvalue problem of a fixed matrix.

In this work, we consider the partial diagonalization of n_e eigenpairs of a converging sequence of symmetric matrix pencils, $(A^{(i)}, B)$, of dimension N,

$$A^{(i)}X^{(i)} = BX^{(i)}\Lambda^{(i)}, \tag{1}$$

where $i \in \mathbb{Z}^+$ is a sequence index, $A^{(i)} \in \mathbb{R}^{N\times N}$ is symmetric and $B \in \mathbb{R}^{N\times N}$ is symmetric positive definite (SPD). $X^{(i)} \in \mathbb{R}^{N\times n_e}$ and $\Lambda^{(i)} \in \mathbb{R}^{n_e\times n_e}$ are the eigenvectors and the diagonal matrix of eigenvalues corresponding to the desired eigenpairs of $(A^{(i)}, B)$, respectively. We denote the eigenvalues as $\Lambda^{(i)}_{pq} = \delta_{pq}\lambda^{(i)}_p$ and will refer to the increment of i as an *SCF iteration*. Further, we will make the following assumptions about the sequence of matrix pencils:

- We assume that $A^{(i+1)}$ depends in some (possibly non-linear) way on $(X^{(i)}, \Lambda^{(i)})$.
- As the SCF iterations progress, we assume $A^{(i)}$ converges toward a matrix A, but are not concerned with how this convergence is achieved other than the requirement that the convergence is not chaotic and the desired eigenpairs of $(A^{(i)}, B)$ must be computed to progressively higher accuracy as this convergence occurs.
- We assume that the desired eigenpairs of each matrix pencil in the sequence are contiguous within the desired spectral region bounded by $\lambda^{(i)}_{\min}$ and $\lambda^{(i)}_{\max}$.

In cases where n_e is relatively small ($O(< 1,000)$) compared to N, or when $(A^{(i)}, B)$ is sparse or structured, iterative algorithms such as the implicitly restarted Lanczos algorithm (Lehoucq et al. 1998), the Jacobi-Davidson algorithm (Sleijpen and Van der Vorst 2000; Stathopoulos and McCombs 2007), and the locally optimal block preconditioned conjugate gradient (LOBPCG) algorithm (Knyazev 2001) are often very effective. If the matrix vector multiplication, $y \longleftarrow A^{(i)}x$, can be implemented efficiently on a large number of computational cores, one may obtain the desired eigenpairs of matrices with much larger dimensions (e.g., millions or even billions) in a matter of minutes if not less.

In cases where n_e is a considerable fraction of N or when n_e is larger than thousands or tens of thousands, iterative algorithms become less efficient partly due to the need to solve a projected dense eigenvalue problem as a part of the Rayleigh-Ritz procedure via some dense eigensolver. Dense eigensolvers such as those available in the LAPACK (Anderson et al. 1999), ScaLAPACK (Blackford et al. 1997) and ELPA (Marek et al. 2014) libraries are also often used to perform a full diagonalization of each $(A^{(i)}, B)$. Recent advances in dense numerical linear algebra have made it possible to perform full diagonalizations for matrices of dimension $O(10,000) - O(100,000)$ in a few wall clock minutes using thousands to tens of thousands computational cores. However, making further improvements when even more computational resources (e.g. GPU accelerators) become available appears to be difficult due to the communication requirement of existing parallel algorithms.

In this report, we present the SISLICE method: a parallel symmetric eigensolver based on shift-invert spectrum slicing for the solution of the SCF eigenvalue problem described in Eq. (1). In spectrum slicing methods, the eigenspectrum of the problem of interest is divided into several subintervals (spectral slices) such that the eigenvalues within each slice may be computed simultaneously. This approach eliminates the Rayleigh-Ritz bottleneck and increases the potential for concurrency in a parallel implementation. The notion of spectrum slicing is well documented in the literature for many classes of eigenvalue problems (Bai et al. 2000; Saad 2011). While the basic idea behind spectrum slicing is relatively simple, its practical implementation on a large number of computational resources is non-trivial.

One of the key practical considerations in the development of a spectral slicing method is the choice of method for computing eigenvalues within each independent spectral slice. For this purposed, the SISLICE method employs the shift-invert subspace iteration. Despite its simplicity, the shift-invert subspace iteration is particularly attractive because the convergence of the method is sufficiently fast for approximate eigenpairs in the spectral neighborhood of a target shift. Further, it is robust and relatively easy to implement.

The primary practical issue of spectral slicing addressed in this work is the development of a scheme to partition the spectrum into spectral slices that contain an appropriate number of eigenvalues without knowing how the eigenvalues are distributed in advance. In general, the spectrum may be partitioned by the selection of the target shifts used in the shift-invert subspace iteration. In early work by Grimes, *et al.* (Grimes et al. 1994), this shift selection was performed in a sequential manner to balance the cost of the sparse matrix factorizations and triangular back substitutions required to solve the sequence of shifted linear systems. Given an initial shift, this sequential spectrum slicing process uses a block shift-invert Lanczos iteration to compute one spectral slice at a time. As the shift-invert Lanczos iterations progress, convergence of the Ritz pairs is monitored and a new shift is selected when convergence is deemed to have stagnated. A similar technique was used in the development of the SIPs method of Zhang, *et al.* (Zhang et al. 2007). The SIPs method is a parallel spectrum slicing method which generates spectral slices and target shifts in a dynamically scheduled parallel framework where the number of required spectral slices are assumed to be much larger than the number of processors or process groups available to solve the eigenvalue problem. This parallel slicing strategy was also adopted for density functional calculations in (Campos and Roman 2012). The selection of shifts was not discussed in (Aktulga et al. 2014) where a multiple shift-invert Lanczos method was compared with a contour integral based eigensolver called FEAST (Polizzi 2008).

The shift selection strategy used in SISLICE is designed for parallel spectrum slicing to be performed on computing platforms with extensive computational resources (i.e. in terms of node and processor counts). To do this, we select all of the target shifts which partition a desired part of the spectrum at once, such that the shift-invert subspace iterations associated with different shifts or spectral slices may be carried out simultaneously. A key feature of this shift selection strategy is the use of spectral density estimation (also known as the density of states, or DOS) to approximate the distribution of eigenvalues. In this work, this DOS estimation is obtained from a Lanczos-based procedure described in (Lin et al. 2016) at the beginning of the slicing procedure as is described in (Li et al. 2016). Given a DOS estimation, we are able to place more shifts in spectral regions which have many tightly spaced eigenvalues that are not well separated from the rest of the spectrum, and fewer shifts in regions that have isolated clusters with small radii. Through this procedure, we may

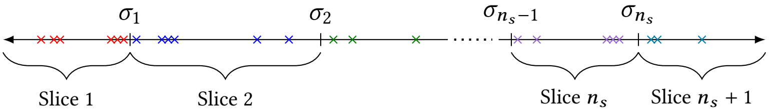


Fig. 1. Partitioning the spectrum of interest into several slices or subintervals which may be computed simultaneously.

ensure that the shift-invert subspace iterations are able to converge rapidly (in a few iterations) for all spectral slices. Further, we introduce an eigenvalue clustering strategy which allows us to refine shift placement and track eigenvalue migration throughout the SCF procedure as $A^{(i)}$ converges to a fixed matrix. Because the matrix sequence $A^{(i)}$ can change significantly in early SCF iterations when it is far from converged, some of the selected shifts resulting from the analysis of approximate eigenvalues in the previous SCF iteration may not be optimal. Consequently, some of these shifts may need to be deleted and new shifts may need to be inserted to ensure no eigenvalue is missed and all eigenvalues within the spectral region of interest can be computed efficiently by the shift-invert subspace iteration. We will discuss how this can be implemented in conjunction with a spectral slice validation scheme.

The SISLICE method is designed to minimize communication overhead and thus improve parallel scalability at the expense of performing more local calculations. This strategy takes into account the recent trend in high performance computing platforms on which floating point operations become cheaper due to the emergence of multicore processors and accelerators while data movement remains costly. In SISLICE, we compute more approximate eigenpairs than the number eigenvalues within a spectral slice. This redundancy does not necessarily increase time to solution if there is an abundance of computational resources which can accommodate such redundancy. However, it makes the validation of eigenpairs easier and more efficient to implement. In particular, we will show that in SISLICE it is not necessary to check mutual orthogonality of approximate eigenvectors obtained in different spectral slices. As a result, our validation scheme does not require moving vectors across different nodes or processor groups, which is often costly. This key feature enables SISLICE to scale to more than tens of thousands of processors.

This paper is organized as follows. Section 2 briefly reviews the salient aspects of shift-invert spectrum slicing and the spectral slice validation scheme used by the SISLICE method. Sections 3 and 4 examine the practical issues of spectrum slicing, such as shift selection, parallel load balance, etc., and how the SISLICE method aims to resolve them. Section 5 provides a series of numerical experiments which exhibit the performance and robustness of the proposed SISLICE method, and some additional improvements to the SISLICE method which we will implement in the future are discussed in Sec. 6.

2 SHIFT-INVERT SPECTRUM SLICING

Algorithm 1 depicts the general framework with which the SISLICE method will perform the sequence of pencil diagonalizations required for the SCF eigenvalue problem. At each SCF iteration, the general strategy adopted by the SISLICE method is to partition the spectral region of interest of a matrix pencil $(A^{(i)}, B)$ into subintervals which may be treated independently. These spectral subintervals will be referred to as *spectral slices* in this work. As the SCF iterations progress, the span of the desired eigenvectors between subsequent iterations approaches invariance, thus the approximate eigenvectors obtained from a particular SCF iteration may be used as a *best guess* approximation (initial guess) for the subsequent iteration. Because the considered eigenvalue

Algorithm 1: Shift-invert spectrum slicing for computing n_e eigenpairs of a sequence of matrix pencils $(A^{(i)}, B)$ as they converge to a pencil (A, B).

```
  Input  : $(A^{(0)}, B)$, number of desired eigenpairs $n_e$, number of slices $K$
  Output : $X \in \mathbb{R}^{N \times n_e}$ and diagonal matrix $\Lambda \in \mathbb{R}^{n_e \times n_e}$ which describe the desired $n_e$ eigenpairs
           of the converged $(A, B)$.

  for $i = 0, 1, 2, \ldots$ do
1     Partition the spectral region of interest for $(A^{(i)}, B)$ into $K$ slices by selecting $n_s = K - 1$
        spectral shifts;
2     Choose starting guesses for eigenvectors within each slice;
3     Use shift-invert subspace iteration to obtain approximate eigenpairs within each slice;
4     Validate the computed eigenvalues $\longrightarrow (X, \Lambda)$;
5     Compute the matrix $A^{(i+1)}$ using $(X, \Lambda)$;
      if $A^{(i+1)}$ converged then
6         return $(X, \Lambda)$;
      end
  end
```

problem is symmetric, its real-valued eigenspectrum may be partitioned into $n_s + 1$ spectral slices by selecting a set of points $\{\sigma_j \mid \sigma_j \in \mathbb{R}\}_{j=1}^{n_s}$ as shown in Fig. 1. A particular σ_j will be referred to as a *spectral shift*, and each spectral slice will be bounded on either side by either a spectral shift or $\lambda_{\min}$ ($\lambda_{\max}$) for slice 1 ($n_s + 1$), respectively. As such, the problem of computing the eigenpairs within a particular spectral slice amounts to computing approximate eigenpairs in the neighborhood of the shifts which form its boundary and validating those eigenpairs against some well defined criteria. We note here that the SISLICE method treats n_s as a static quantity throughout the SCF procedure. This constraint is adopted primarily for considerations regarding load balance in a distributed parallel computing environment (see Sec. 4 for details). In this work, we use the shift-invert subspace iteration to compute eigenpairs near a spectral shift. The validation of eigenpairs takes into account the shifted matrix inertia as well as the accuracy of the computed eigenpairs. Specific details regarding the selection of spectral shifts are given in Sec. 3. In this section, we review the salient aspects of the shift-invert subspace iteration and slice validation scheme used by the SISLICE method given a set of spectral shifts.

We should note that the algorithm outlined in Alg. 1 can be used to compute desired eigenpairs of a fixed matrix pencil also. When the matrix pencil (A, B) is fixed, we obviously do not need to perform the update in Step 5, however; we may improve the efficiency of the shift-invert subspace iteration by repartitioning the spectral region of interest (Step 1) using previously computed eigenvalue approximations as the reference. The application of Alg. 1 to a fixed eigenvalue problem is particularly attractive for large, sparse matrix pencils with $N > O(100,000)$ where sparse matrix factorizations are possible, but direct eigenvalue decomposition is impractical on currently available computer hardware.

2.1 The Shift–Invert Subspace Iteration

In this subsection, we examine the salient aspects of the shift-invert subspace iteration as it pertains to a particular SCF iteration, i.e. the partial diagonalization of a single element of the SCF sequence.

Algorithm 2: The Shift-Invert Subspace Iteration: SISubIt($A, B, V_{(0)}, \sigma, m$)

```
Input :Symmetric matrices $A, B \in \mathbb{R}^{N \times N}$ with $B$ being SPD, a target shift $\sigma \in \mathbb{R}$, the number
       of eigenpairs to be computed $k$, an initial guess of the eigenvectors $V_{(0)} \in \mathbb{R}^{N \times k}$, and a
       number of subspace iterations $M$
Output:$(X, \Lambda)$ which approximates $k$ eigenpairs of $(A, B)$ in the spectral neighborhood of $\sigma$.

$V \longleftarrow$ CholeskyQR($V_{(0)}, B$);
$(L, D) \longleftarrow LDL^T$ factorization of $A - \sigma B$;
for $m$ = 1,2,...,$M$ do
    $V_{(m)} \longleftarrow L^{-T} D^{-1} L^{-1} B V_{(m-1)}$;
    $V_{(m)} \longleftarrow$ CholeskyQR($V_{(m)}, B$);
end
return $(X, \Lambda) \longleftarrow$ RayleighRitz($A, B, V_{(M)}$);
```

For convenience, we denote this particular pencil as (A, B). Approximate eigenpairs of (A, B) corresponding to eigenvalues in the neighborhood of a shift, σ, may be obtained through the shift-invert subspace iteration (Bai et al. 2000; Saad 2011). For the purposes of this work, we assume that σ is distinct from any eigenvalue of (A, B). The shift-invert subspace iteration may be viewed as the power iteration applied to the shift-invert transformed system given by,

$$\xi(A, B, \sigma) = (A - \sigma B)^{-1} B. \tag{2}$$

$\xi(A, B, \sigma)$ may easily be shown to be self-adjoint with respect to the B-inner product, i.e.

$$\langle x, \xi(A, B, \sigma) y \rangle_B = x^T B (A - \sigma B)^{-1} B y = \langle \xi(A, B, \sigma) x, y \rangle_B, \quad \forall x, y \in \mathbb{R}^N. \tag{3}$$

Under ξ, a particular eigenpair (x, λ) of (A, B) admits the following map,

$$Ax = \lambda Bx \quad \mapsto \quad \xi(A, B, \sigma) x = \mu B x, \tag{4}$$

where

$$\mu = \frac{1}{\lambda - \sigma}. \tag{5}$$

As λ approaches σ, the magnitude of μ rapidly grows larger. Thus, eigenvalues of (A, B) in the spectral neighborhood of σ are mapped to the extremal eigenvalues of $(\xi(A, B, \sigma), B)$ while leaving their corresponding eigenvectors unchanged. As such, application of the power iteration to a k dimensional subspace $V \in \mathbb{R}^{N \times k}$,

$$V_{(m+1)} = \mathrm{orth}(\xi(A, B, \sigma) V_{(m)}, B), \tag{6}$$

converges rapidly towards the invariant subspace corresponding to the k largest eigenvalues of $(\xi(A, B, \sigma), B)$, or equivalently, those corresponding to the k eigenvalues of (A, B) closest to σ. Here, $m \in \mathbb{Z}^+$ is the shift-invert subspace iteration index and V is denoted with a subscript to distinguish itself from the SCF iteration of Eq. (1). One possible implementation of Eq. (6) is given in Alg. 2.

There are a few alternatives to the shift-invert subspace iteration which include:

- The shift-invert Lanczos method. This method requires solving a sequence of linear systems of equations with a single right-hand side. Although this method has a faster convergence rate than the shift-invert subspace iteration, the dependency among these linear systems makes it difficult to achieve good parallel scalability.

- The Davidson or LOBPCG methods applied to $(A - \sigma B)^{-1}B$. This method also has a faster convergence rate than the shift-invert subspace iteration, but it is more difficult to implement in a stable manner due to the numerical rank deficiency of the basis of the search space from which approximations of the desired eigenpairs are extracted (Duersch et al. 2018).
- Contour integral based methods (Polizzi 2008; Sakurai and Sugiura 2003; Tang and Polizzi 2014). These methods requires solving a number of complex-shifted linear systems of equations. The parallelization requires taking into account the partition of the spectrum and the number of quadrature points used to approximate the contour integral representation of the spectral projection operator.
- Polynomial filtering based methods. When A is very sparse, and B is the identity, we can use a subspace or Lanczos iteration applied to $p(A)$ to compute eigenpairs within a spectral slice, where $p(\cdot)$ is a bandpass polynomial filter that amplifies spectral components associated with eigenvalues within that slice (Li et al. 2016). This approach only requires the multiplication of A with vectors. However, an extremely high degree polynomial may be required when the spectral slice is not well separated from the rest of the spectrum and the size of the slice is small.

When some of the above methods are carefully implemented, they can potentially outperform the shift-invert subspace iteration. However, such implementation is far from trivial, especially on many-core distributed parallel computing platforms. By this rationale, the development of the SISLICE method employs the shift-invert subspace iteration for the evaluation of eigenpairs within a particular spectral slice.

At the highest level, Alg. 2 consists of three key computational subtasks. One of the subtasks is used to construct a B-orthonormal basis of a subspace spanned by columns of V. There are several options for performing this task. We employ the Cholesky QR procedure for this purpose, as outlined in Alg. 3. Although Cholesky QR is not the most accurate procedure for constructing the B-orthonormal subspace, it is computationally efficient and easy to parallelize due to its use of dense matrix-matrix multiplication and Cholesky factorization. For the purposes of this work, we have found the Cholesky QR procedure to provide a reasonable compromise between accuracy and efficiency. The second subtask applies Eq. (2) to the subspace. When A and B are dense, we employ the Bunch-Kaufman (LDL^T) factorization implemented in LAPACK or ScaLAPACK to decompose the shifted matrix $A - \sigma B$ and repeatedly apply forward and backward substitutions to solve the sequence of shifted linear systems required by the shift-invert subspace iteration. In addition to the fact that the LDL^T factorization is very efficient and highly parallelizable, the diagonal matrix, D, of the factorization may be further used in the validation of eigenpairs within a spectral slice (see Sec. 2.2 for details). When A and B are sparse, we may use a symmetric sparse solver such as those implemented in MUMPS (Amestoy et al. 2001, 2006), PARDISO (Schenk and Gärtner 2002, 2006; Schenk et al. 2000), symPACK (Bachan et al. 2019, 2017), or SuperLU (Li 2005) to solve the shifted linear systems. The third subtask is used to perform a subspace rotation to extract approximate eigenpairs from the subspace $V_{(M)}$ after the subspace iterations is terminated. As the eigenvectors of the shift-invert transformed system are invariant under ξ, approximate eigenpairs of (A, B) with eigenvalues in the spectral neighborhood of σ may be extracted through the Rayleigh-Ritz procedure outlined in Alg. 4 using the subspace obtained from iteration of Eq. (6). These approximate eigenpairs are referred to as Ritz pairs.

Remark that the algorithm outlined in Alg. 2 is the simplest version of the shift-invert subspace iteration. Several modifications can be made to improve the performance and robustness of the algorithm, especially in the context of a convergent sequence of matrix pencils which must be

Algorithm 3: The Cholesky QR Procedure: CholeskyQR(V, B)

```
Input  : $V \in \mathbb{R}^{N \times k}$, $B \in \mathbb{R}^{N \times N}$ with $B$ SPD
Output : $Z$ such that $Z^T B Z = I$
$Y \leftarrow V^T B V$
$LL^T \leftarrow Y$ (Cholesky factorization)
return $Z \leftarrow V L^{-T}$
```

treated in Eq. (1). For example, the number of columns of V in Eq. (6) need only be *at least* k to obtain approximations for k eigenpairs. In practice, the convergence rate of Eq. (6) in obtaining the k *desired* eigenpairs may be drastically improved by choosing a trial vector space which is several times larger than k (see Sec. 5.4 for examples). Further, we may exploit the fact that $(X^{(i)}, \Lambda^{(i)})$ need not be computed to full accuracy until the SCF iterations of Eq. (1) is nearly converged. The rate at which $V_{(m)}$ approaches X in Eq. (6) largely depends on the choice of initial guess $V_{(0)}$. If the distance between $V_{(0)}$ and X (as measured in terms of subspace angle) is sufficiently small, convergence may be achieved in only a few subspace iterations. As $A^{(i)}$ converges to A, the change in the eigensystem between $(A^{(i)}, B)$ and $(A^{(i+1)}, B)$ becomes sufficiently small such that the distance between $X^{(i)}$ and $X^{(i+1)}$ is also small. Thus, Eq. (6) may be seeded with $X^{(i)}$ to obtain $X^{(i+1)}$ in these last few SCF iterations to enable faster convergence. This assumption is typically most valid in the last few SCF iterations, though this seeding procedure will be demonstrated to be effective throughout the SCF procedure in Sec. 5.

For each spectral shift selected to partition the spectral region of interest, σ_j, we will associate a subspace obtained by performing a set of shift-invert subspace iterations using that shift, $V_j \in \mathbb{R}^{N \times k}$. From each V_j, we may compute a set of Ritz pairs, (Λ_j, X_j), which approximate k eigenpairs of (A, B) in the spectral neighborhood of σ_j. From each (Λ_j, X_j), we may compute a set of residuals, $R_j = AX_j - BX_j\Lambda_j \in \mathbb{R}^{N \times k}$, from which we may evaluate a vector of residual norms, $r_j \in \mathbb{R}^k$, as the 2-norm of the columns of R_j. The tuple $(\sigma_j, \Lambda_j, X_j, r_j)$ will be referred to as the j-th spectral probe throughout the remainder of this work and will be occasionally denoted SP(σ_j). We note for clarity that one need not consider both V_j and X_j simultaneously due to the fact that they admit identical linear spans (X_j is simply a rotation of V_j). As X_j contains more useful information related to the eigensystem of (A, B) than V_j, V_j is typically discarded in favor of X_j for eigenvalue calculations.

Algorithm 4: The Rayleigh–Ritz Procedure: RayleighRitz(A, B, V)

```
Input  : Symmetric $A, B \in \mathbb{R}^{N \times N}$ with $B$ SPD, $V \in \mathbb{R}^{N \times k}$ with $V^T B V = I$,
Output : $(X, \Lambda)$ which approximate $k$ eigenpairs of $(A, B)$ spanned by the columns of $V$.
$Y \leftarrow V^T A V$
Solve $YZ = Z\Lambda$ (Diagonalization)
$X = VZ$
return $(X, \Lambda)$
```

2.2 Validation of Spectral Slices

In the SISLICE method, the approximate eigenpairs associated with a particular slice (σ_j, σ_{j+1}) are obtained by analyzing the Ritz pairs that are computed from the spectral probes defined by the spectral shifts σ_j and σ_{j+1}, which we denote by $\mathrm{SP}(\sigma_j)$ and $\mathrm{SP}(\sigma_{j+1})$, respectively. The Ritz values obtained from $\mathrm{SP}(\sigma_j)$ can potentially overlap with those obtained from $\mathrm{SP}(\sigma_{j+1})$. It is also possible that σ_j and σ_{j+1} are too far apart that a number of desired eigenvalues are not captured by either $\mathrm{SP}(\sigma_j)$ or $\mathrm{SP}(\sigma_{j+1})$. Thus, a key aspect in the development of a robust spectrum slicing method is to provide a mechanism to select approximate eigenpairs within a spectral slice from the Ritz pairs of its associated probes as to avoid double counting and detect any missing or spurious eigenpairs, if present. Such selected eigenpairs will be referred to as being *validated*.

To select candidates for validation within the spectral slice (σ_j, σ_{j+1}), we examine the Ritz values computed from the spectral probes $\mathrm{SP}(\sigma_j)$ and $\mathrm{SP}(\sigma_{j+1})$ that are within (σ_j, σ_{j+1}). We choose a point τ between σ_j and σ_{j+1}, e.g., $\tau = (\sigma_j + \sigma_{j+1})/2$ and select all Ritz values obtained from $\mathrm{SP}(\sigma_j)$ that are in (σ_j, τ), and those from $\mathrm{SP}(\sigma_{j+1})$ that are in (τ, σ_{j+1}) as validation candidates. A graphical representation of this candidate selection process is depicted in Fig. 2. For the the spectral slices at both ends of the desired spectral region of interest, validation candidates are selected as those Ritz values that are in $[\lambda_{\min}, \sigma_1)$ and $(\sigma_{n_s+1}, \lambda_{\max}]$, respectively.

The duplication of eigenvalues can be checked by measuring the mutual orthogonality of the corresponding eigenvectors. However, such a scheme would require comparing Ritz vectors computed by different spectral probes. In a parallel implementation in which different spectral probes are mapped to different processor groups, this scheme would require excessive data communication. We choose to check duplication or missing eigenvalues by simply comparing the number of validation candidates with the exact eigenvalue count that can be obtained from the factorization $L_j D_j L_j^T = A^{(i)} - \sigma_j B$ for each spectral shift. By making use of Sylvester's inertia theorem (Sylvester 1852), we are able to ascertain the exact number of eigenvalues within the slice bounded by (σ_j, σ_{j+1}) by taking the difference between the number of negative diagonal elements of D_{j+1} and D_j, respectively.

If the number of validation candidates is equal to the expected number of eigenpairs within that slice, we view each of the candidates as a reasonable approximation to a true eigenpair and consider it to be validated.

If the number of validation candidates is less than the expected number of eigenvalues in (σ_j, σ_{j+1}), there are true eigenpairs within this spectral slice that are not captured by either $\mathrm{SP}(\sigma_j)$ or $\mathrm{SP}(\sigma_{j+1})$. Thus, either more shift-invert subspace iterations need to performed or a spectral shift must be added somewhere within (σ_j, σ_{j+1}) to ensure that all desired eigenpairs are accounted for. We examine the specifics of this shift addition in Sec. 3.3.

If the number of candidates exceeds the expected number of eigenvalues within (σ_j, σ_{j+1}), a number of the validation candidates are likely duplicated copies between adjacent spectral probes. In this case, we select the candidates with the smallest residual norms to be validated. This strategy follows from the assumption that Ritz values that approximate eigenvalues closer to a spectral shift tends to converge faster. That is, if θ_j and θ_{j+1} are both approximations to the same eigenvalue λ that lies in (σ_j, τ), but are obtained from two different spectral probes $\mathrm{SP}(\sigma_j)$ and $\mathrm{SP}(\sigma_{j+1})$ respectively, the residual norm associated with $\theta^{(i)}$ is likely to be smaller because λ is closer to σ_j than to σ_{j+1}.

3 SHIFT SELECTION AND MIGRATION

In this section, we examine the selection of the set of spectral shifts $\{\sigma_j\}_{j=1}^{n_s}$ which partition the spectral region of interest into $n_s + 1$ spectral slices. There are are three primary topics related to this selection which are explicitly treated in this work:

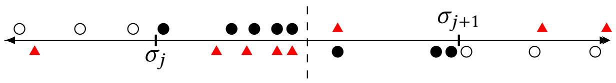


Fig. 2. Scheme for selection of validation candidates for the spectral slice bounded by (σ_j, σ_{j+1}). The points with markers above the axis are Ritz values computed from the SP(σ_j), and those below the axis are those computed from SP(σ_{j+1}). The vertical dashed line denotes the midpoint of the spectral slice (τ). The filled black circles represent the validation candidates for the slice, while the open circles represent Ritz values which may belong to other spectral slices depending on the placement of σ_{j-1} and σ_{j+2}. The red triangles represent Ritz values which are not considered for validation for the spectral slice.

- As the distribution of eigenvalues in the spectral region of interest is not known *a priori*; we must obtain some sort of estimation of this distribution to properly select spectral shifts as to maximize the amount of useful computation and avoid shift placement in spectral regions which do not contain eigenvalues. In this work, we utilize a Lanczos approximation for the so-called density of states (DOS) of the considered matrix pencil to obtain this estimation.
- As the Lanczos DOS approximation does not provide an exact description of the desired eigenvalue distribution, it is often possible to obtain a better set of shift placements given a more accurate description of the eigenvalues of interest. Further, for the sequence of diagonalizations given in Eq. (1), the eigenspectrum is expected to change as the SCF iterations progress towards convergence. As such, we must develop a method that iteratively refines the placement of the spectral shifts throughout the SCF procedure to maintain accuracy and load balance throughout the calculation. In this work, we examine the use of eigenvalue clustering between successive SCF iterations for this task.

3.1 Density of States

Consider the eigenvalue decomposition of the pencil (A, B) in Eq. (1). The exact density of states (DOS) for (A, B) is given by

$$y(\lambda) = \sum_{j=1}^{N} \delta(\lambda - \lambda_j), \tag{7}$$

where $\delta(\cdot)$ is the Dirac delta distribution. For any particular λ, the spectral density in the neighborhood of λ is given by $y(\lambda)\mathrm{d}\lambda$. Thus, for an interval $[a, b] \subset \mathbb{R}$, we may define a quantity

$$\gamma(a, b) = \int_a^b y(\lambda)\mathrm{d}\lambda, \tag{8}$$

which provides an eigenvalue count on $[a, b]$. Using this definition, we may define the cumulative density of states (CDOS) which returns the number of eigenvalues of (A, B) below a certain value

$$c(\lambda') \equiv \gamma(-\infty, \lambda') = \int_{-\infty}^{\lambda'} y(\lambda)\mathrm{d}\lambda. \tag{9}$$

Clearly, the construction of the exact DOS and CDOS requires a full diagonalization of (A, B), which is something we are trying to avoid. In this section we examine the estimation of the DOS as a linear combination of smooth functions associated with a set of Ritz pairs obtained from the Lanczos procedure.

3.1.1 The Lanczos Density of States Approximation. Consider the k-step B–orthogonal Lanczos algorithm (Saad 2011) which produces a factorization of (A, B) of the form

$$B^{-1}AV_k = V_kT_k + f_ke_k^T, \tag{10}$$

with

$$V_k^TBV_k = I_k, \tag{11a}$$

$$V_k^TBf_k = 0_k. \tag{11b}$$

Here $I_k \in \mathbb{R}^{k\times k}$ is the k-by-k identity matrix, $0_k \in \mathbb{R}^k$ is a vector of zeros, and $T_k \in \mathbb{R}^{k\times k}$ is the symmetric tridiagonal Lanczos matrix. $e_k \in \mathbb{R}^N$ is the kth column of the $N \times N$ identity matrix.

Suppose (θ_j, g_j) is an eigenpair of T_k. The Ritz pair (θ_j, V_kg_j) is considered an approximation to an eigenpair of (A, B) and $(e_1^Tg_j)^2$ is an approximation to the spectral density of eigenvalues in the neighborhood of θ_j (Li et al. 2016; Lin et al. 2016). To obtain the spectral density at points other than θ_j, i.e. the DOS, we may convolve these approximate eigenpairs with smooth functions such as Gaussians or Lorentzians centered at the approximate eigenvalues. In this work, we consider expansion in terms of Gaussian functions such that the approximate DOS may be written as

$$y(\lambda) = N\sum_{j=1}^{k}\zeta_j^2 \exp\left(-\frac{(\lambda-\theta_j)^2}{\nu_j}\right), \qquad \zeta_j = e_1^Tg_j. \tag{12}$$

ν_j is a length parameter which determines the width of the Gaussian. For the purposes of this work, we choose ν_j so that the Gaussian $\exp[-(\lambda-\theta_j)^2/\nu_j]$ nearly vanishes some distance d_j away from θ_j. The parameter d_j is chosen to be either the maximum or average of $\theta_j - \theta_{j-1}$ and $\theta_{j+1} - \theta_j$. Some safeguard is used to prevent d_j from becoming too small when eigenvalues are tightly clustered. Due to the linearity of $y(\lambda)$, we may obtain a closed form expression for the corresponding CDOS as

$$c(\lambda) = \frac{N\sqrt{\pi}}{2}\sum_{j=1}^{k}\zeta_j^2\sqrt{\nu_j}\left(\operatorname{erf}\left(\frac{\lambda-\theta_j}{\sqrt{\nu_j}}\right)+1\right), \tag{13}$$

where $\operatorname{erf}(\cdot)$ is the error function defined as

$$\frac{\mathrm{d}\operatorname{erf}(x)}{\mathrm{d}x} = \frac{2}{\sqrt{\pi}}e^{-x^2}. \tag{14}$$

3.1.2 Selection of Shifts from the Density of States. We now discuss how to partition the spectrum into different slices and select shifts for spectral probes based on the DOS and CDOS obtained from a Lanczos algorithm. When the CDOS of (A, B) increases gradually in a nearly continuous fashion (see Fig. 9), our strategy is to partition CDOS uniformly into a number of intervals with approximately k eigenvalues per interval, and use either the bisection or Newton's algorithm to find the roots of

$$y(\omega) = kj, \;\; j = 1, 2, ..., n_e/k,$$

which will form the endpoints of the spectral intervals. The shift for each probe can be chosen to be the midpoint of the interval or a DOS weighted average of a set of uniformly sampled points ω_i within the interval, i.e.,

$$\sigma = \frac{\sum_{i=1}^{m}\omega_iy(\omega_i)}{\sum_{i=1}^{m}y(\omega_i)}.$$

This strategy does not work well when the spectrum has clusters of eigenvalues with large gaps in between. These clusters and gaps can be revealed from the DOS estimation as isolated peaks or sharp increases in the CDOS (see Fig. 8). In this case, we should partition the spectrum in such a way that each isolated cluster is contained in one interval.

We can locate spectral clusters by identifying local maximizers of the DOS. However, this needs to performed at an appropriate spectral resolution. Evaluating DOS on a very fine spectral grid may produce too many "artificial" local maximizers that are introduced by the inexact nature of the estimated DOS. Evaluating DOS on a very coarse grid may result in missing an important local maximizer (hence a cluster).

We use the following strategy to adaptively refine the DOS so that well separated spectral clusters can be identified, the bounds of the intervals that contain these clusters can be properly defined, and the shifts assigned to spectral probes for seeking eigenvalues within these intervals can be properly selected.

(1) Using the lower and upper bounds λ_l, λ_u of the spectrum returned from the Lanczos procedure, we can evaluate the DOS at a set of uniformly distributed points

$$\omega_i = \lambda_l + (\lambda_u - \lambda_l)/(n-1)i,$$

for $i = 0, 1, \ldots, n-1$.

(2) We identify local maximizers among $y(\omega_i)$, where $y(\omega)$ is defined by (12).

(3) Between two local maximizers $\hat{\omega}_{j-1}$ and $\hat{\omega}_j$, we find the local minimizer of the DOS,

$$\mu_j = \text{argmin}_{\omega \in (\hat{\omega}_{j-1}, \hat{\omega}_j)} y(\omega).$$

A local minimizer of the DOS between $\hat{\omega}_j$ and $\omega_{\widehat{j+1}}$ can be found also. These local minimizers define the bounds of the interval (l_j, u_j) that contains the cluster centered at $\hat{\omega}_j$. Note that $u_j = l_{j+1}$.

(4) We estimate the number of eigenvalues within the jth interval by computing $c(u_j) - c(l_j)$, where $c(\cdot)$ is the CDOS defined in (13).

(5) If the estimated eigenvalue count for the jth interval is smaller than a threshold (e.g., 1 or 2), and if the interval does not contain a Ritz value, we simply delete that interval, and adjust the bounds of the adjacent intervals.

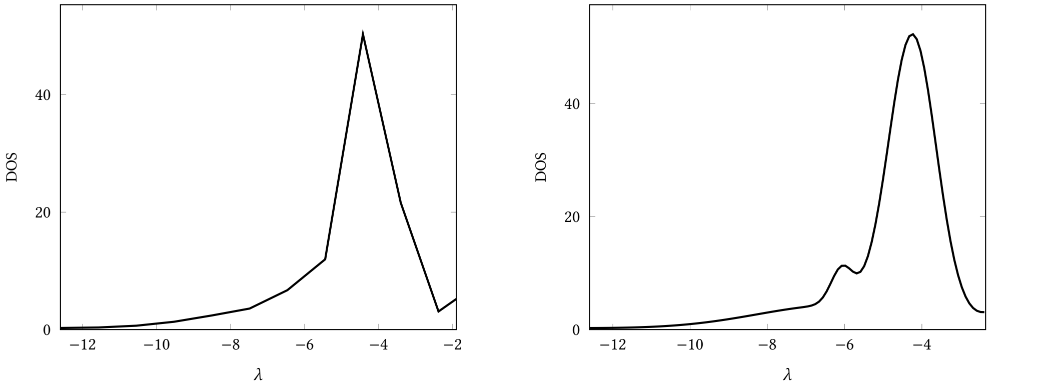
(6) If the estimated eigenvalue count m for the jth interval is larger than a preselected threshold (e.g., 50), we refine the spectral grid within (l_j, u_j) and evaluate $y(\omega)$ at $\omega_j^i = l_j + (u_j - l_j)/(m-1)i$, for $i = 0, 1, 2, \ldots.m-1$. We identify additional clusters within (l_j, u_j) by finding the local maximizers within this interval, and using the procedure described in step 3 to determine the bounds of new intervals and a new shift within each new interval. Fig. 3 shows that an additional cluster around -6.0 is identified when the third cluster shown in Fig. 8 is refined. The bounds of the refined cluster are also adjusted.

(7) Steps 6 is repeated until no new local maximizer can be found in each interval. Fig. 4 shows all new clusters identified by the above process at the upper end of the spectrum of the Silane matrix.

(8) When no new cluster can be found, we go through each cluster and partition the a cluster uniformly into smaller intervals if the estimated number of eigenvalues within that cluster is larger than a threshold c and when

$$\max\left\{\frac{\sigma_j - l_i}{\sigma_j - \sigma_{j-1}}, \frac{u_i - \sigma_i}{\sigma_{j+1} - \sigma_j}\right\} > r, \tag{15}$$

for a prescribed ratio tolerance r. In practice, we often choose $c = 30$ and $r = 0.6$, although these parameters can be adjusted by the user.



(a)

(b)

Fig. 3. Refining the resolution of the DOS in the third cluster shown in Fig. 8a reveals an additional cluster near -6.0 (b). The refinement also allows us to set tighter bounds for refined clusters.

3.2 Shift Refinement and Eigenvalue Clustering

Due to the limited number of Ritz values that can be extracted from the Lanczos method in the spectral interior, it is possible that the initial selection of spectral shifts produced by a Lanczos DOS estimation procedure is far from optimal. In particular, shifts may be placed in spectral regions devoid of eigenvalues. Another possible scenario is that an insufficient number of shifts are placed in regions that contain a disproportionately large number of eigenvalues. An illustration of this issue is given on the axis labeled “DOS” in Fig. 5. However, shift misplacement can be incrementally corrected in subsequent SCF iterations by using a clustering algorithm to partition previously computed eigenvalue approximations and refine the shift selection.

For each SCF iteration, we obtain a set of eigenpair approximations for $(A^{(i)}, B)$. Thus, for $i > 0$, we have available to us a set of approximate eigenvalues for $(A^{(i-1)}, B)$. In the early SCF iterations, when the change in eigensystem between two subsequent iterations is relatively large, it is possible that the shifts selected for $(A^{(i-1)}, B)$ would not be appropriate for the slicing of the

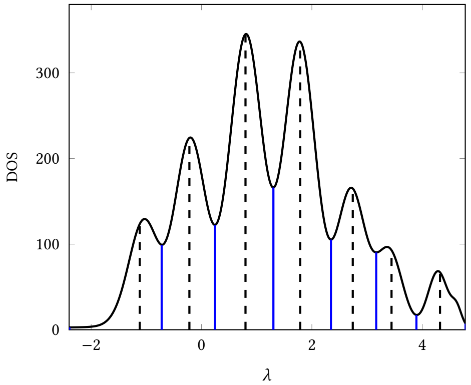


Fig. 4. Refined DOS at the upper end of the spectrum of the Silane matrix. The blue solid lines mark the bounds of all clusters, and the black dashed lines mark the shift positions in each cluster.

spectrum for $(A^{(i)}, B)$. We discuss a strategy to determine this (dis)similarity and strategies for subsequent shift selection in Sec. 3.3. However, if the eigenvalues of $(A^{(i)}, B)$ are sufficiently close to those of $(A^{(i-1)}, B)$, then we may use the approximated eigenvalues of $(A^{(i-1)}, B)$ as a reference to determine the spectral shift placement for the spectrum slicing of $(A^{(i)}, B)$. Due to the localized nature of the shift-invert spectral transformation described in Eq. (2), rapid convergence of the shift-invert subspace iteration is achieved when spectral shifts are placed centrally in clusters of eigenvalues. Thus, we may determine more optimal shift placement by identifying spectral clusters from the computed eigenvalues $(A^{(i-1)}, B)$ and placing shifts in the centroids of these clusters for determination of the eigenpairs of $(A^{(i)}, B)$.

To identify spectral clusters, we employ the k-means clustering algorithm (Lloyd 1982). Algorithm 5 depicts how the k-means algorithm generates a fixed number of clusters and their centroids from a set of properly ordered approximate eigenvalues. At the i-th SCF iteration for $i > 0$, we use Algorithm 5 to identify n_s clusters from the validated eigenpairs obtained from the $(i-1)$-st iteration. The centroids of the clusters may then be used in the generation of the i-th set of spectral shifts. As the SCF procedure converges, the centroids of the clusters will also converge to a particular set of spectral shifts. An illustration of this convergence behavior is given in Fig. 5.

Algorithm 5: Ordered K-Means Clustering: KMeans$(K, X, \{c_k\})$

```
Input  : Number of desired clusters $K$, ordered data $X = \{x_i\}_{i=1}^{N}$, initial guess of centroids,
         $\{c_k\}_{k=1}^{K}$.
Output : A set of $K$ clusters, $\{(X_k, c_k) \,|\, X_k \subset X\}_{k=1}^{K}$ such that $X = \bigcup_{k=1}^{K} X_k$ and $X_k \cap X_j = \emptyset$
         for $k \neq j$.

Sort elements of $X$ in non-decreasing order;
repeat
    Initialize $X_k = \emptyset, \forall k$;
    $k \leftarrow 1$;
    $j \leftarrow 1$;
    for $i$ = 1:N do
        if $x_i > c_k$ and $k < K$ then
            $j \leftarrow$ first $j$ s.t. $x_i \leq c_j$;
            $k \leftarrow \min(K, j+1)$;
        end
        $m \leftarrow \frac{1}{2}(c_k + c_j)$;
        if $x_i < m$ then
            $X_j = X_j \cup \{x_i\}$;
        else
            $X_k = X_k \cup \{x_i\}$;
        end
    end
    Update centroids: $c_k = \frac{1}{|X_k|} \sum_{x \in X_k} x, \forall k$;
until $\{(X_k, c_k)\}$ is unchanging;
return $\{(X_k, c_k)\}$;
```

Although the k-means clustering problem is generally NP-hard, we do not necessarily need to obtain a globally optimal solution to the clustering problem in order to identify appropriate spectral shifts. Our objectives are to identify eigenvalue clusters and to partition nearly uniformly distributed eigenvalues into slices of roughly equal size. In general, determination of n_s clusters is a drastic over clustering of the Ritz values. However, k-means clustering usually results in equal sized clusters, even in the case of over clustering. Due to the fact that n_s is relatively small, obtaining clusters from this data using k-means may be achieved with negligible cost.

The k-means algorithm is an iterative procedure initialized with a set of guesses to cluster centroids. The choice of these initial guesses can have a significant effect on the convergence of the algorithm and the quality of cluster centroids it produces. In the SISLICE method, these guesses are usually taken to be the spectral shifts used in the previous SCF iteration. However, if the previous spectral shifts are generated from the DOS shift selection strategy, it is possible that the k-means algorithm can converge to a suboptimal solution if the centroids are initialized with these shifts. To address this issue, we employ the k-means++ (Arthur and Vassilvitskii 2007) cluster initialization strategy to improve initial guesses of the centroids prior to the k-means clustering process.

Assuming that the eigenvalue distribution does not change drastically throughout the SCF procedure, k-means also allows the SISLICE method to track shift migration between SCF iterations as shown in the axes labeled "Update" in Fig. 5. Until convergence is reached, shift migration is performed using the validated eigenpairs of the previous SCF iteration. As Alg. 5 converges to the local optimum nearest to the initial guess, this choice of guess allows for the k-means shift migration strategy to converge to a single set of shifts as the SCF converges. It is important that the clustering is performed on validated eigenpairs to avoid over sampling of spectral regions for which the Ritz pairs of adjacent spectral probes overlap or contain spurious Ritz values. We demonstrate the efficacy of this migration scheme in Sec. 5.

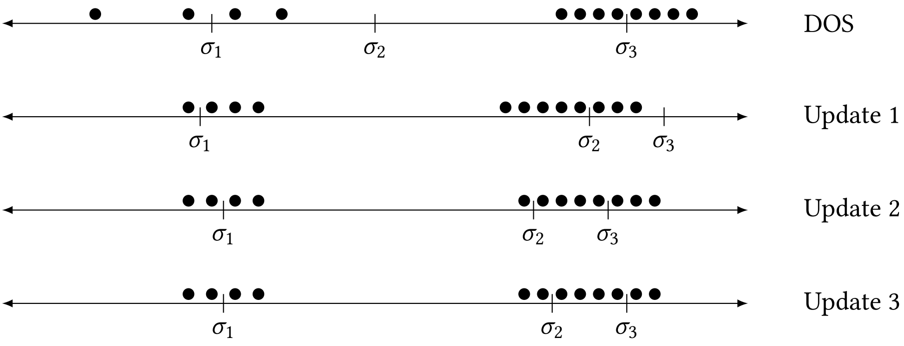


Fig. 5. A graphical representation of the shift migration process throughout the SCF procedure. The SCF iterations progress from top to bottom with the filled circles representing the validated Ritz values at that iteration. σ_1, σ_2, σ_3 represent the spectral shifts used to obtained the Ritz values at each iteration. The shifts for the first SCF iteration are representative of a typical DOS-shift selection scheme where shifts are chosen both in regions without eigenvalues as well as regions with a disproportionately large number of eigenvalues due to inaccuracy in the DOS approximation. At each subsequent SCF iteration, new shifts are chosen via k-means clustering of the Ritz pair obtained from the previous iteration.

Once a clustering of the validated Ritz values has been obtained, we may use the centroids of those clusters to generate the set of spectral shifts for the next SCF iteration. Instead of creating new spectral probes, we would like to reuse the Ritz vectors produced by the existing spectral probes as the initial guesses to the desired eigenvectors in the subsequent shift-invert subspace iteration to improve convergence. Therefore, in the SISLICE method, we update the shifts of the existing spectral probes based on the clustering information rather than starting completely from scratch. To update the shift for each spectral probe, we form a mapping between the eigenvalue clusters and spectral probes such that each cluster is mapped to the probe with which it has maximal overlap.

In the case that this map is bijective, the spectral shift associated with the spectral probe is taken to be the centroid of its associated cluster. However, it is possible, especially in the early SCF iterations when the eigenspectrum undergoes considerable change, that this map is not bijective. As a result, there is some ambiguity as to how to best update the probes which have no preimage under this mapping.

The case of non-bijective maps between clusters and spectral probes is typically a symptom of poor shift selection in the previous SCF iteration resulting in some spectral probes picking up only a few validated Ritz values while others capturing a disproportionate number of validated Ritz values. As the validated Ritz values are separated by the k-means algorithm into different clusters, validated Ritz values retrieved from one spectral probe may be separated into several clusters resulting in several clusters being mapped to the same spectral probe. (See the schematic illustration in Fig. 6.) In the meantime, the few validated Ritz values obtained by a poorly placed spectral probe SP(σ_j) may be placed into a cluster that gets mapped to a different spectral probe SP(σ_j), leaving SP(σ_j) without any cluster to map to.

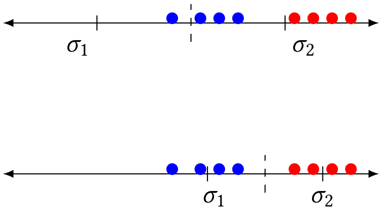


Fig. 6. A schematic illustration of how multiple cluster may be mapped to the same probe, and how an old probe may be deleted and a new probe can be inserted. The blue and red dots are two clusters of approximate eigenvalues that are both mapped to the same spectral probe centered at σ_2. After the mapping between clusters and the previous spectral probes is established. The probe centered at σ_1 is deleted because no cluster is mapped to it. A new problem centered at a new shift σ_1 is inserted, and σ_2 is also moved to the right.

When several clusters are mapped to the same spectral probe, we merge these clusters into a single cluster. The resulting centroid of the merged Ritz values is taken to be the spectral shift of the associated spectral probe. In the case that the mapped cluster contains too many eigenvalues, a probe may be inserted to ensure proper load balance.

If a spectral probe is not mapped to any cluster, it is simply deleted. However, since the number of spectral shifts (and thus the number of spectral probes) is fixed throughout the SCF procedure, when a probe is removed, another probe must be added to maintain this fixed number of shifts. We choose to add a probe to a cluster that contains the largest number of Ritz values. To add such a spectral probe to such a cluster, we need to break up the cluster first into two clusters. In practice, this may be achieved by performing a 2-means clustering of the Ritz pairs in the largest cluster. One of the clusters is mapped to the original spectral probe mapped to the cluster before it was broken up. The shift associated with that cluster is replaced with the centroid of the new smaller cluster. The other cluster is mapped to the newly added spectral probe. In addition to setting the shift of the probe to the centroid of the new cluster, we also need to copy (or send in a distributed memory implementation) the Ritz vectors associated with the Ritz values in this cluster to the added spectral probe. This process of breaking up a large cluster and adding a new spectral probe is repeated until the desired number of spectral shifts and probes is obtained. The specifics regarding probe insertion in a distributed infrastructure are discussed Sec. 4.

3.3 Missing Eigenvalues

Because $A^{(i)}$ can change significantly from $A^{(i-1)}$ in early SCF iterations, a shift selection scheme based on the clustering of approximate eigenvalues of $A^{(i-1)}$ may not be optimal for computing eigenpairs of $A^{(i)}$. In particular, it is possible that spectral probes constructed from the suboptimal selection of target shifts miss some eigenvalues. The spectral slices in which these missing eigenvalues reside can be identified in the validation process described in Sec. 2.2.

When missing eigenvalues are detected, we perform a new DOS estimation on $(A(i), B)$ with an appropriate resolution to place new shifts in spectral slices that contain missing eigenvalues. New spectral probes are created to recompute approximate eigenvalues within these newly created spectral slices. This is a costly step because the next SCF cycle cannot start until all missing eigenvalues are accounted for. An example of this state of affairs is demonstrated in Sec. 5.3.

To reduce the likelihood of missing eigenvalues resulting from shift misplacement, we can monitor the convergence of SCF for drastic changes in the spectrum by comparing the partial traces of the system matrices within the subspace spanned by the previously validated eigenvectors. For example, given the metric

$$\eta(V, A) = \mathrm{Tr}\left(V^T A V\right), \tag{16}$$

if the difference between $\eta(X^{(i-1)}, A^{(i-1)})$ and $\eta(X^{(i-1)}, A^{(i)})$ is larger than some specified threshold, then the spectra of the matrices may be deemed to be sufficiently dissimilar.If this is found to be the case, then it would be beneficial to use the DOS shift selection strategy discussed in Sec. 3.1.2. Although this strategy does not completely eliminate the possibility of missing eigenvalues (because individual eigenvalues can move around without affecting the trace of the $A^{(i)}$), it may help reduce that possibility and the cost associated with generating new probes to seek the missing eigenvalues.

4 PARALLEL IMPLEMENTATION

The algorithmic subtasks described in the previous sections have been constructed in such a way as to allow for maximal concurrency in the slicing of the spectral region of interest: each of the spectral probes may be constructed independently of any other spectral probe. As the construction of the spectral probes through shift-invert subspace iterations constitutes the bulk of the work in the SISLICE method, this task independence should lead to scalable performance. The slice validation scheme outlined in Sec. 2.2 would require some level of synchronization between independent computing units. Further, the shift insertion and deletion schemes outlined in Sec. 3.2 would require some data to be copied from some processors/nodes to others. However, these communication and synchronization overheads are generally small as we will see in the next section.

In this section, we outline the salient aspects of the parallel implementation of the SISLICE method.

We note that the parallelization discussed here focuses exclusively on the parallel execution of spectral probes. An additional level of finer grain parallelism exists within each spectral probe. If matrices A and B can be replicated and stored on each single many-core compute node, a hybrid-parallelism scheme utilizing both shared-memory and message passing parallelism may be achieved through exploitation of optimized implementations of threaded BLAS and LAPACK (such as those found in Intel(R) MKL, IBM(R) ESSL, OpenBLAS, BLIS, ATLAS, cuBLAS, etc) within a particular MPI rank. While we do not treat this level of parallelism explicitly in this section, its leverage is trivial on modern computing architectures and is implied for the numerical experiments in Sec. 5. If A and B are too large to be stored on a single compute node, then the factorizations and linear system solves required for each probe may be performed using ScaLAPACK in the case of dense matrices, or a distributed sparse solver such as SuperLU, symPACK, MUMPS, or PARDISO in the case of sparse matrices. We should note that the parallel scalability of LDL^T factorization and back substitutions for solving triangular systems with multiple right hand sides is generally much better than that can be achieved in a dense eigensolver.

These distributed calculations for each spectral probe may take place on a subset of the total number of MPI ranks, allowing leverage of massive parallelism on large computing clusters. We do not treat this particular parallelism scheme in this work, but it has been discussed at length in other related work (Keçeli et al. 2016; Zhang et al. 2007).

4.1 Spectral Probe Distribution and Synchronization

The SISLICE method is designed for taking advantage of computer systems that have a large amount of computational resource in terms of compute nodes and cores within each node. In an ideal scenario, the number of spectral probes should match the number of nodes (or groups of nodes) so that all probes can be executed simultaneously. The optimal number of nodes (or group of nodes) that should be used to perform the computation can be determined by the spectrum partition and shift selection scheme discussed in Sec. 3.1. One can query such information from the solver in a separate analysis run prior to running the SISLICE solver.

When the number of computational nodes is less than the number of spectral probes, a round-robin distribution of probes to nodes can be used, i.e., $\text{SP}(\sigma_j)$ may be mapped to the $(j \bmod n_r)$th MPI rank, where n_r is the number of MPI ranks. In this case, the computation is not load balanced if n_r does not divide n_s.

Once the spectral probes have been constructed, each $\text{SP}(\sigma_j)$ contains a set of Ritz pairs which approximate the eigenpairs in the neighborhood of σ_j. However, the slice validation scheme described in Sec. 2.2 requires knowledge of Ritz pair information from adjacent probes. If each of the adjacent spectral probes has been constructed on a different MPI rank (or group of MPI ranks), the validation scheme requires some level of communication / synchronization of Ritz pair information between the MPI ranks. However, the validation scheme only requires knowledge of the Ritz values and associated residual norms to validate the spectral slices. The Ritz vectors are not explicitly required.

If the Ritz values and residual norms were only to be used in the slice validation scheme, their synchronization could be further limited to only the neighboring ranks of the owner of a particular spectral probe. However, because the entire set of validated Ritz values is used in updating spectral shifts through k-means clustering (as described in Sec. 3.2), it is useful to synchronize this information across all of the MPI ranks. While a distributed implementation of k-means clustering is possible, the fact that each spectral probe only accounts for a relatively small number of validated Ritz pair would require excessive communication to perform the clustering. Because the storage requirement of the Ritz values and residual norms is negligible relative to the Ritz vectors, this synchronization scheme poses no storage overhead. In the implementation of the SISLICE method discussed in this work, the set of spectral probes is represented as a replicated data structure on each MPI rank which is synchronized according to Fig. 7 at each SCF iteration. As the Ritz values and residual norms are replicated across each MPI rank, the tasks of slice validation and shift updates may also be replicated to avoid communication. We note for clarity that in the case of random initialization of the clustering problem through e.g. k-means++, the Ritz value clustering may still be replicated through the use of pseudo random number generation using the same seed value. The scalability of this distribution and synchronization scheme is demonstrated in Sec. 5.5.

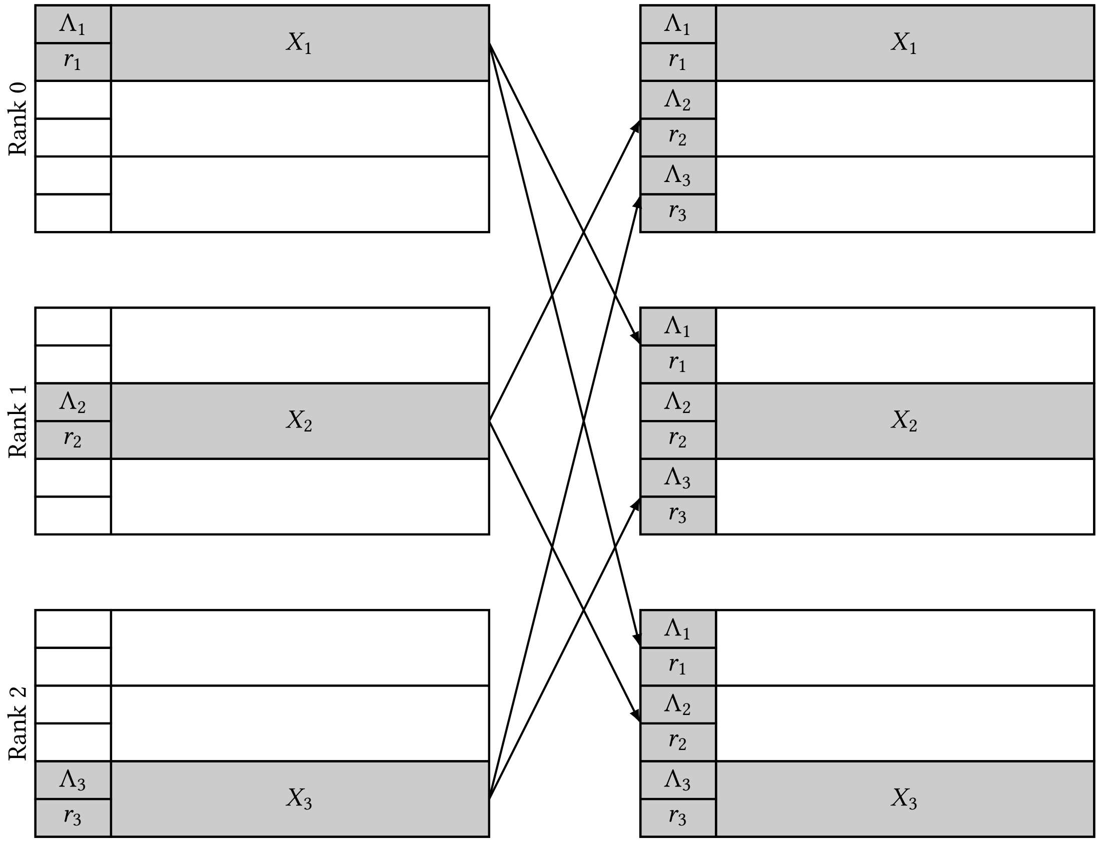


Fig. 7. Spectral probe distribution and synchronization scheme for the SISLICE method for an example case of 3 spectral probes distributed among 3 MPI ranks. The data-structure which holds the spectral probe information is replicated among the MPI ranks with each of the ranks constructing its probe locally. After the probes have been constructed, each rank broadcasts its Ritz values and residual norms to the other MPI ranks, thus synchronizing this data across the system. This synchronization process does not broadcast the Ritz vectors as to avoid excessive communication of large amounts of data.

4.2 Spectral Probe Insertion and Removal

As was discussed in Sec. 3.2, occasionally shift selection and migration schemes employed by the SISLICE method yield suboptimal shift placement leading to redundant spectral probes and probes which are responsible for a disproportionate number of validated eigenpairs. This is typically the case in the early SCF iterations due to the crude DOS approximation by the Lanczos procedure described in Sec. 3.1. The presence of redundancies in the spectral probes leads to a load imbalance which should be avoided to ensure scalability on large, distributed computing systems. For the purposes of this section, the term "load balance" should be thought of as balance of *useful* work. Technically speaking, even in the case of redundancies in the spectral probes, the computational work performed for each spectral probe will always be roughly the same given that the number of subspace iterations performed and subspace dimensions are uniform across all probes. Thus this work is always “balanced”. However, we want to ensure that each rank is performing a roughly equal amount of useful work (in the sense of yielding a roughly equal number of validated Ritz pairs) rather than wasting valuable computational resources in spectral regions where it is not needed.

As the number of spectral probes is fixed in the SISLICE method, removal of a spectral probe necessitates the insertion of a spectral probe to balance the work in another spectral region. This probe removal necessarily leads to a load imbalance if the work was balanced in the previous SCF iteration. As discussed in Sec. 3.2, probes are inserted so as to break up large clusters of validated Ritz values such that they effectively span multiple probes after the subsequent shift-invert subspace iterations. In a distributed computing environment, care must be taken to ensure probe insertion is performed in such a way as to balance the work between independent computing ranks while avoiding a large communication overhead. As the k-means clustering is replicated on each rank, probe insertion may also be effectively replicated with only minimal communication. For each probe to be inserted, the determination of the two new spectral shifts is replicated on each rank. The new probe which is to be inserted is assigned to the rank with the least amount of work (thus ensuring load balance). Once this decision has been made, the probe whose shift has been moved through this procedure communicates its Ritz vector data to the newly inserted probe to allow its reuse in the subsequent SCF iteration. The cost of this point-to-point communication is relatively small in practice and may be overlapped with the determination and communication of other probe insertions.

In the case when shifts must be inserted due to missing eigenvalues within a particular spectral slice as described in Sec. 3.3, we may leverage the fact that the computation is done in parallel to our advantage. It may be the case that the DOS shift insertion strategy yielded several shifts in the spectral region that contains the missing eigenvalues. Rather than have processors or processor groups sit idle while the missing eigenpairs are obtained sequentially, the newly inserted probes may be distributed in the same manner as the initial probe distribution. Due to the fact that missing eigenvalues are typically a symptom of poorly placed shifts, not of too few shifts, inserting probes will not yield more useful probes than n_s, i.e. if a probe had to be inserted to resolve missing eigenpairs, it is typically the case that some probes did not produce validated eigenpairs. However, even if each of the probes from the first round of subspace iterations produced validated eigenpairs, the mapping scheme between eigenvalue clusters and spectral probes will preclude the possibility of yielding more than n_s probes for the subsequent SCF iteration. This is due to the fact that the SISLICE method obtains n_s clusters regardless of the number of probes used to produce the validated eigenpairs in the previous SCF iteration. Thus, even in the case of probe insertion in the previous SCF iteration, the SISLICE method ensures load balance is maintained in subsequent SCF iterations.

5 NUMERICAL EXPERIMENTS

In this section, we report a set of numerical experiments which demonstrate the effectiveness of the proposed shift selection technique for computing all or a subset of eigenvalues of a matrix pencil or a sequence of matrix pencils. We examine two limiting cases of eigenvalue distribution shown in Figs. 8 and 9.

The Silane test case (Fig. 8, $N = 1109$) is an all-electron density functional theory calculation using a Gaussian basis set. Its spectrum exhibits a number of isolated eigenvalue clusters at lower eigenvalues and a more uniform distribution at larger eigenvalues. The isolated eigenvalue clusters at low eigenvalues are a common feature in all-electron density functional calculations. All matrices related to the Silane test case in this work were obtained using the NWChemEx software package (Kowalski et al. [n.d.]).

The Graphene test case (Fig. 9, $N = 9360$) is a density functional theory calculation using pseudo-potentials for the core electrons. As such, its spectrum does not contain isolated clusters. The more uniform nature of the spectrum is a common feature in nearly all pseudo-potential based density functional calculations. All matrices related to the Graphene test case in this work were obtained using the SIESTA software package (Soler et al. 2002).

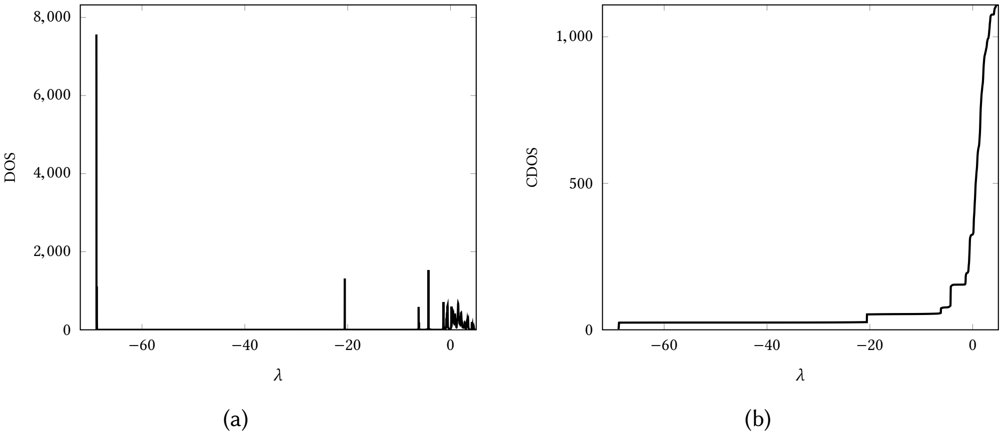


Fig. 8. Lanczos DOS (a) and CDOS (b) for the entire spectrum of Silane ($N = 1109$). Silane exhibits a number of isolated eigenvalue clusters (spikes in the CDOS) lower in the spectrum and a more uniform distribution at larger eigenvalues. The DOS and CDOS calculations were performed using 100 Lanczos iterations with the converged matrix pencil.

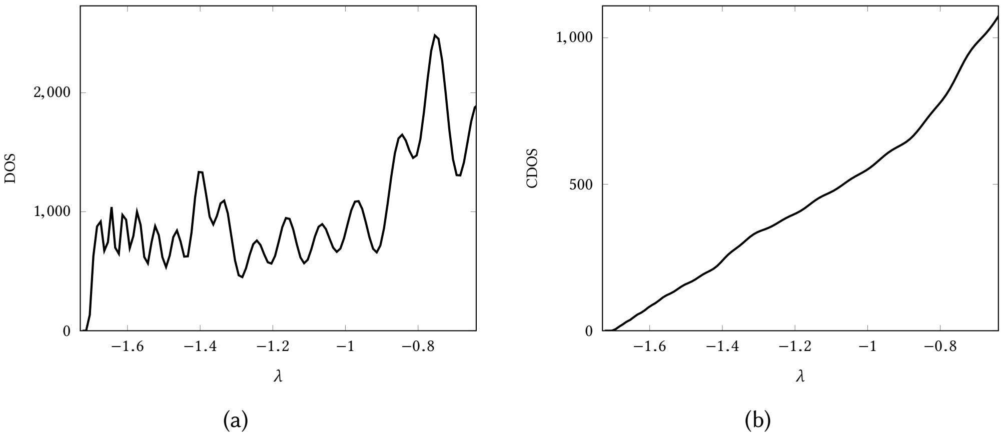


Fig. 9. Lanczos DOS (a) and CDOS (b) for the lowest 1000 eigenvalues of Graphene ($N = 9360$). Graphene exhibits a nearly uniform eigenvalue distribution throughout its entire spectrum. The DOS and CDOS calculations were performed using 100 Lanczos iterations with the converged matrix pencil.

5.1 Shift Selection for a Fixed Matrix Pencil

To demonstrate how the shift selection strategy enables rapid convergence of the SISLICE method as the SCF procedure approaches convergence, i.e. when the matrix pencils change very little between SCF iterations, we examine the convergence of eigenpairs for fixed matrix matrix pencils in this section. This experiment allows us to gauge the effectiveness of the shift selection strategy when more accurate estimation of the desired eigenvalues becomes available in successive SCF iterations.

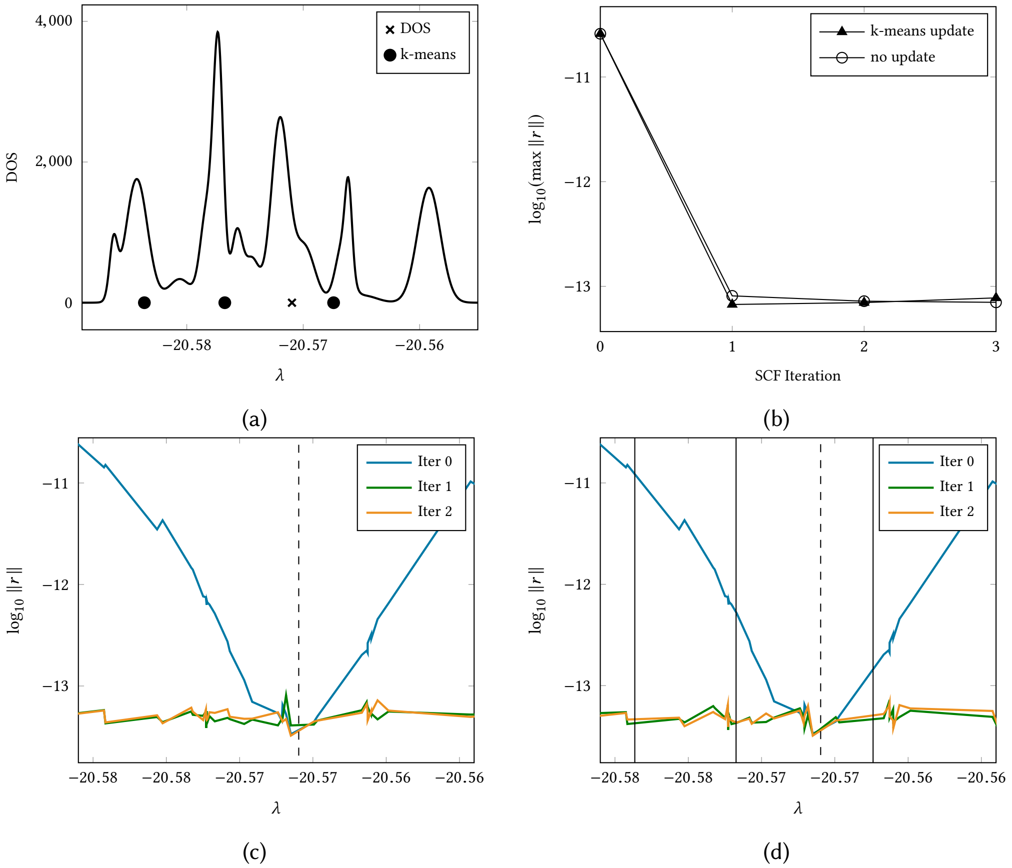
To simplify our exposition, we examine a set of representative spectral windows for the aforementioned test cases using the converged matrix pencil (A, B) as the representative eigenvalue problem. Even though the matrix pencil does not change, an artificial SCF procedure is carried out and a new set of shifts may be chosen after a fixed number of subspace iterations have been performed. This procedure can be viewed as a generalized (block) Rayleigh quotient iteration.

All calculations in this section were performed using a probe basis dimension of $k = 100$ and 4 shift invert subspace iterations per SCF iteration. The SCF iteration is considered converged if the maximum of the residual norms associated with all validated approximate eigenpairs is below the threshold of 10^{-13}. In both of the presented cases, we observe the expected monotonic convergence of the eigenvalues within individual spectral slices once shift migration has been performed.

Silane. For the case of Silane, SISLICE was applied to perform a full diagonalization using 100 shifts so that $n_e/n_s \approx 11$. We examine two representative spectral windows for this test case, $C_1 = [-20.59, -20.55]$ (Fig. 10) and $C_2 = [-0.9, -0.39]$ (Fig. 11). The C_1 window represents a dense, isolated cluster of eigenvalues, while the eigenvalues in C_2 are embedded in a dense region of eigenvalues. Due to the different distribution characteristics of these two spectral windows, the convergence behavior of the eigenpairs within these windows are different. However, because these windows are not treated separately in the sense of the larger eigenvalue calculation, SCF iterations are performed until convergence is reached across the spectrum. Further, in this test case, DOS-based shift selection yielded 25 useless probes that were not well placed, i.e. probes which did not produce any validated eigenvalues after the validation scheme outlined in Sec. 2.2 was applied. These probes were redistributed in the subsequent iterations via the method outlined in Sec. 3.2.

Because eigenvalues in C_1 are well separated from the rest of the spectrum, the convergence of the subspace iteration is rapid. Using the DOS based shift partitioning, a single shift is placed just below $\lambda = -20.57$ to account for the 37 eigenvalues in the immediate vicinity. In the first SCF iteration, the eigenvalues near the selected shift converged much more rapidly than those further away. After the first SCF iteration, k-means eigenvalue clustering yielded 3 clusters of ~ 12 eigenvalues with centroids shown in Fig. 10a. Convergence for this spectral window is achieved within 2 SCF iterations both with and without the k-means shift update, with all eigenvalues converging at roughly the same rate notwithstanding their distance to the nearest shift. We can also see that for the case of this isolated cluster, k-means clustering yielded no noticeable effects on residual convergence.

In contrast, the convergence of approximate eigenvalues in C_2 is less rapid due to the fact that there exist eigenvalues both immediately below and above the eigenvalues in this spectral window. DOS based shift selection and spectrum partitioning placed 8 evenly spaced shifts to account for the 141 eigenvalues within this window. After the first SCF iteration, k-means clustering revealed a non-uniform distribution of eigenvalues within this window, yielding 12 clusters of ~ 11 eigenvalues. Convergence rates for the eigenvalues in this window vary considerably based on their distance to their nearest shift. Convergence across the entire spectral window is achieved within 4 SCF iterations with the k-means update and 6 SCF iterations without the update. Thus, for this cluster, the k-means shift update yielded a discernible improvement in the residual convergence of the approximate eigenpairs.



(a)

(b)

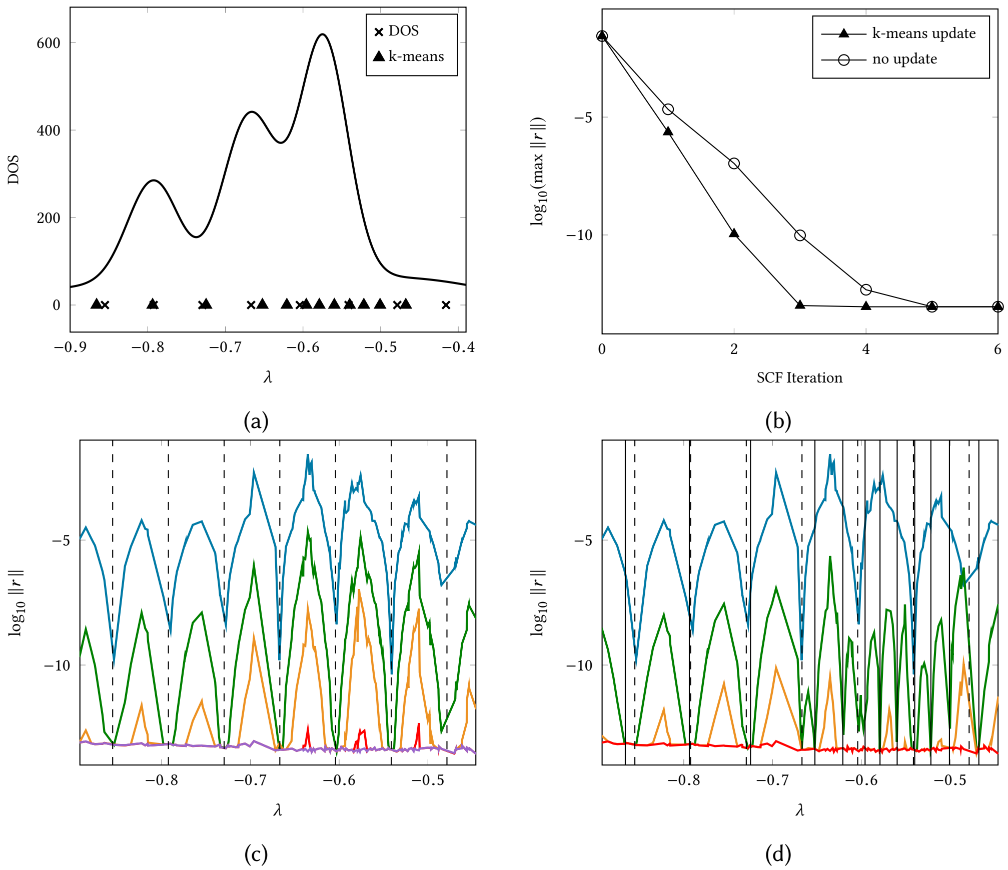
(c)

(d)

Fig. 10. Isolated cluster of Silane eigenvalues [−20.59, −20.55] (37 eigenvalues). (a) Initial Lanczos DOS along with DOS shift placement and k-means update. (b) Convergence behavior of the largest residual norm in the spectral window both with and without the k-means shift update. Overall residual convergence for the SISLICE method within the spectral window with (d) and without (c) k-means shift update. Converges in 2 SCF iterations both with and without k-means shift update.

Graphene. For the case of Graphene, SISLICE was applied to perform a partial diagonalization of the lowest 1000 eigenvalues using 100 shifts to obtain $n_e/n_s \approx 10$. As can be seen in Fig. 9a, the eigenvalue distribution for Graphene is approximately uniform. As such, the DOS based shift selection produced uniformly distributed shifts along the entire spectral window, yielding no useless probes. We examine the eigenvalue interval $C = [-1.4, -1.3]$ as a representative example of the convergence behavior for this test case.

Within C, the Graphene test case admits 93 eigenvalues in a roughly uniform distribution. As such, the DOS based shift partitioning places 8 evenly spaced shifts in this spectral window so that $n_e/n_s \approx 11$. Within each spectral probe, convergence is more rapid near the shifts than further away. The k-means shift update simply migrates the shifts without any appreciable changes to the shift spacing, i.e. the k-means result yields 8 clusters of ∼ 11 Ritz values with centroids of roughly



(a)

(b)

(c)

(d)

Fig. 11. Embedded cluster of Silane eigenvalues [−0.9, −0.39] (141 eigenvalues) (a) Initial Lanczos DOS along with DOS shift placement and k-means update. (b) Convergence behavior of the largest residual norm in the spectral window both with and without the k-means shift update. Overall residual convergence for the SISLICE method within the spectral window with (d) and without (c) k-means shift update. Converges in 4 SCF iterations with k-means shift update and 6 iterations without shift update.

equal spacing. Convergence for this spectral window is achieved within 3 SCF iterations both with and without the k-means shift update. We note that in the second SCF iteration, the convergence is slightly worse with the k-means selected shift than the DOS selected shifts. However, as both methods yield convergence in the same number of SCF iterations overall, we do not believe this discrepancy to be problematic in practice.

5.2 Shift Selection for a Converging Matrix Pencil Sequence

In this section, we examine how our shift selection strategy enables the SISLICE method to efficiently compute eigenpairs of pregenerated, convergent sequences of matrix pencils obtained from a true SCF procedure. Such a test allows us to gauge the ability of the SISLICE method to solve true SCF eigenvalue problems. To determine the efficacy of the shift selection and migration strategy, we examine both the convergence of the residuals produced by the SISLICE method and the change of the true eigenvalues throughout the SCF procedure itself. The latter is possible because these

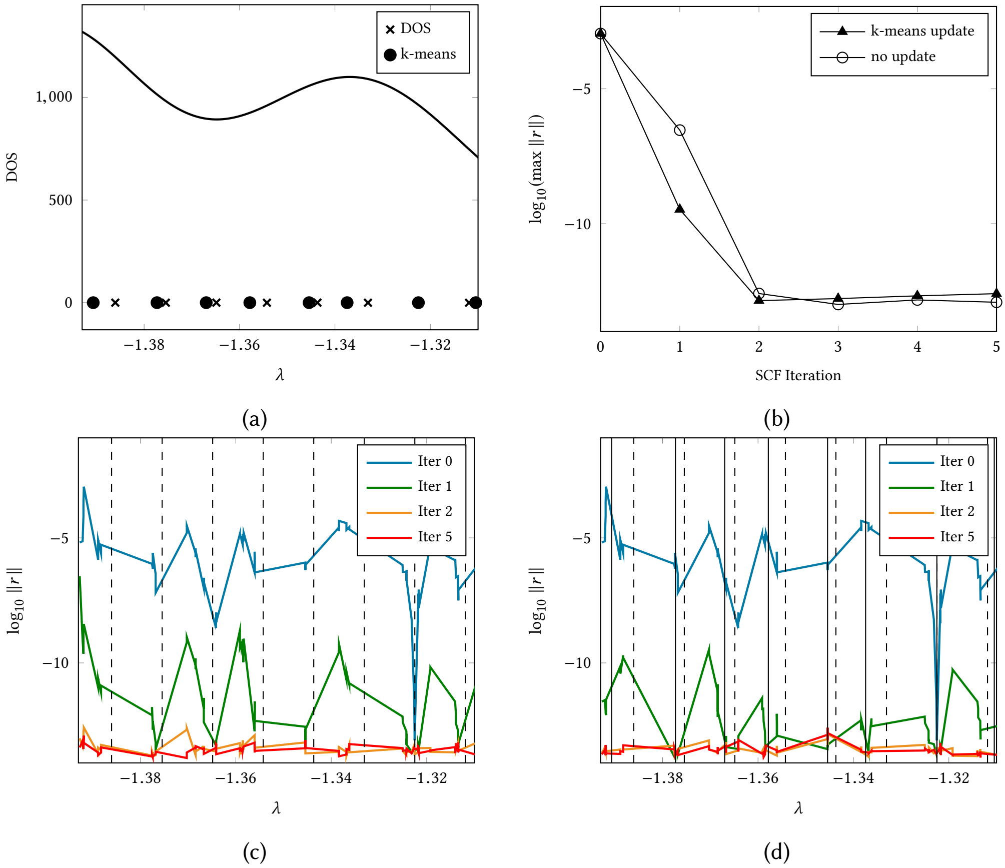


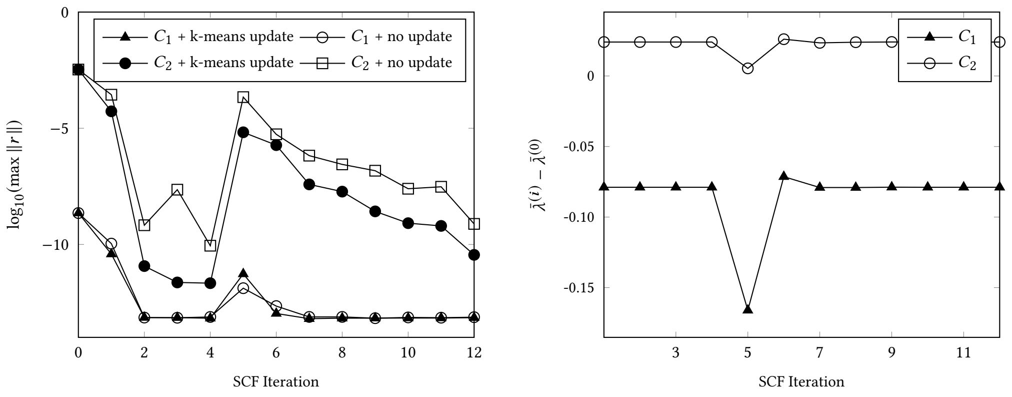
Fig. 12. Graphene eigenvalue cluster [−1.4, −1.3] (93 eigenvalues). (a) Initial Lanczos DOS along with DOS shift placement and k-means update. (b) Convergence behavior of the largest residual norm in the spectral window both with and without the k-means shift update. Overall residual convergence for the SISLICE method within the spectral window with (d) and without (c) k-means shift update. Converges in 3 SCF iterations both with and without k-means shift update.

matrices are pregenerated, thus we have access to the exact eigenvalues of these matrices as a reference to compare the convergence of the SISLICE method. Further, as was examined in the previous section, we perform analogous comparisons of the SISLICE method both with and without k-means updates to the spectral shifts throughout the SCF procedure.

Silane. The Silane SCF procedure converged within 13 iterations in the NWChemEx software package. It is the nature of this particular test case (and is typical of all-electron density functional theory calculation) that the spectrum is separated into well defined clusters throughout the entire SCF procedure. For this reason, we are able to examine the same eigenvalue clusters as discussed in the previous section for this test case. The convergence behavior of the SISLICE method applied to this test case is given in Fig. 13.

Much like the results presented in Sec. 5.1, we see a significant difference in the convergence behavior between the two clusters. Due to the isolated nature of C_1, convergence of the subspace iteration is rapid. Despite changes in the eigenvalues resulting from the changes in the matrix pencil in early SCF iterations, the shift selection strategy we developed is able to track this change, construct and move spectral probes to obtain eigenvalues within C_1 at convergence. The convergence for C_2 is much less rapid, as was also the case in the previous experiments due to the lack of a large separation between eigenvalues within C_2 and the rest of the spectrum. Furthermore, the change in the eigenvectors in C_1 is much less than those in C_2, thus they provide excellent initial guesses for subsequent SCF iterations. The eigenvectors in C_2 undergo a much more drastic change, but it can be seen in Fig. 13a that this change becomes less as the SCF converges.

Note that the convergence of the SCF for the Silane test case is not smooth; there is a large change in the average eigenvalue for the two examined clusters at the fifth SCF iteration. This is not an uncommon feature in the SCF procedure for density functional calculations. There is an analogous change in the residual norms for the SISLICE method at the same SCF iteration. The reason for this is two-fold. In the case where k-means clustering is used to migrate the shifts between SCF iterations, the fact that the clustering is performed using the validated eigenpairs from the *previous* SCF iteration yielded a non-optimal placement for the 5th iteration. However, because the change in residual norms is present also for the experiment without k-means shift updates, the shift migration is not the only reason for this change. The large change in average eigenvalue for this SCF iteration is also accompanied by a change in corresponding eigenvectors within these spectral windows. Thus, the validated eigenvectors from the previous SCF iteration are also not optimal choices for the initial guess to seed the subspace sequence at this iteration. Remark that the increase in residual norm is in fact less for the C_2 cluster with k-means shift updates, indicating that the shift migration strategy is beneficial for this cluster even when the shifts are placed non-optimally.

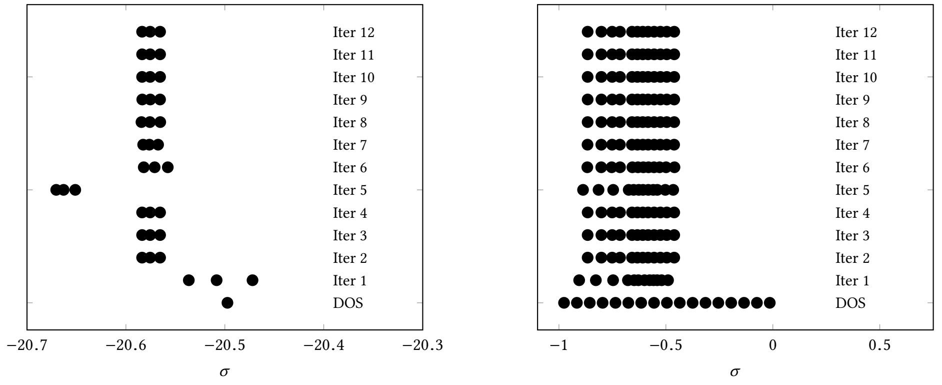


(a)

(b)

Fig. 13. Convergence of the SISLICE method applied to the Silane SCF procedure for two representative spectral windows. (a) Comparison of the convergence behavior of the largest residual norm in the respective spectral windows both with and without k-means shift updates throughout the SCF procedure. (b) The change in average eigenvalue from the initial average of the two spectral windows respectively.

Due to the fact that Silane admits well-defined (and trackable) clusters in its spectrum, we are also able to examine the shift migration within these clusters in Fig. 14. The largest change in shift placement occurred between the first and second SCF iterations, the former of which was produced by the DOS strategy. Because the characteristic of these clusters is largely unchanging throughout the SCF procedure, we are able to see that the k-means shift update remains visually unchanging with the exception of the fifth SCF iteration. Remark that the k-means shift selection strategy was able to track the change in eigenvalues in this iteration and subsequently recover to a reasonably static set of shifts in the following SCF iterations.



(a) (b)

Fig. 14. Shift migration for the SISLICE method in the Silane SCF procedure for the C_1 (a) and C_2 (b) eigenvalue clusters.

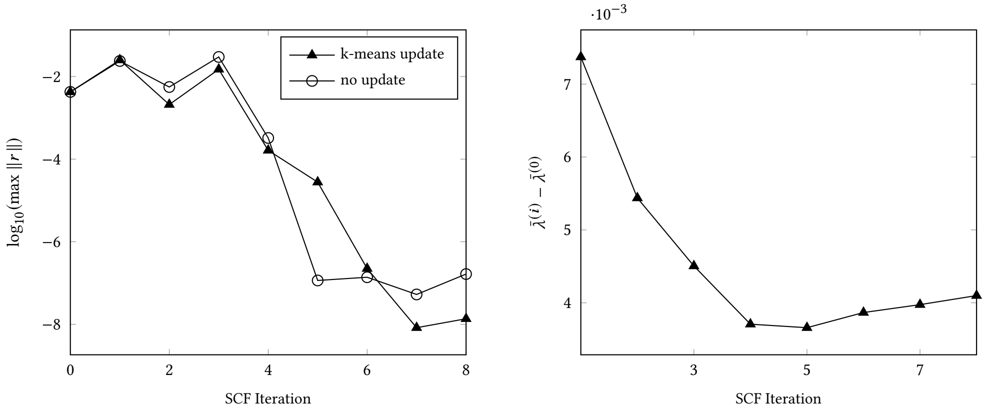
Graphene. The Graphene SCF procedure converged within 10 iterations in the SIESTA software package. Convergence results for the SISLICE method applied to this problem may be found in Fig. 15. Unlike the Silane test case, the homogeneity of the eigenvectors for the Graphene test case makes tracking eigenvalue clusters throughout the SCF procedure impractical. The eigenpairs in a particular spectral interval at one SCF iteration are not likely to be of the same character in the subsequent iterations until convergence is reached. As such, we examine the convergence globally across all of the 1000 eigenpairs obtained desired for this test case.

Unlike the Silane test case, the SCF convergence for Graphene is smooth. This smooth SCF convergence is mirrored in the monotonic convergence of the SISLICE method as the SCF approaches convergence. When the SCF procedure yielded large changes in the underlying spectrum, i.e. the first 3 iterations, the error produced by the SISLICE method was larger as the bases from the previous SCF iteration were not as good of an initial guess as they were in the later iterations. After the fourth SCF iteration, the spectrum is only undergoing small changes and the SISLICE method exhibits rapid convergence. As was the case for the previous numerical experiments with Graphene, no discernible difference between DOS and k-means shifts is exhibited. For example, at the fifth SCF iteration, the DOS shifts produced more accurate results whereas at the seventh, the k-means shifts produced more accurate results.

Due to the fact that the SCF underwent large spectral changes in the early SCF iterations, the extent to which the shifts were able to be usefully updated using the spectrum of the previous matrix pencils was limited. As such, shifts needed to be inserted per the prescription in Sec. 4.2. In the following subsection, we examine an example of this insertion for the Graphene test case.

5.3 Missing Eigenvalues and Probe Insertion

For the Graphene example, we found that some eigenvalues were missed in the second SCF iteration due to a poorly placed shift produced by the k-means clustering of approximate eigenvalues obtained in the first SCF iteration. The red crosses in Sec. 4.2(a) show all eigenvalues within the spectral



(a) (b)

Fig. 15. Convergence of the SISLICE method applied to the Graphene SCF procedure for the lowest 1000 eigenvalues. (a) Comparison of the convergence behavior of the largest residual norm in both with and without k-means shift updates throughout the SCF procedure. (b) The change in average eigenvalue from the initial average of the lowest 1000 eigenvalues.

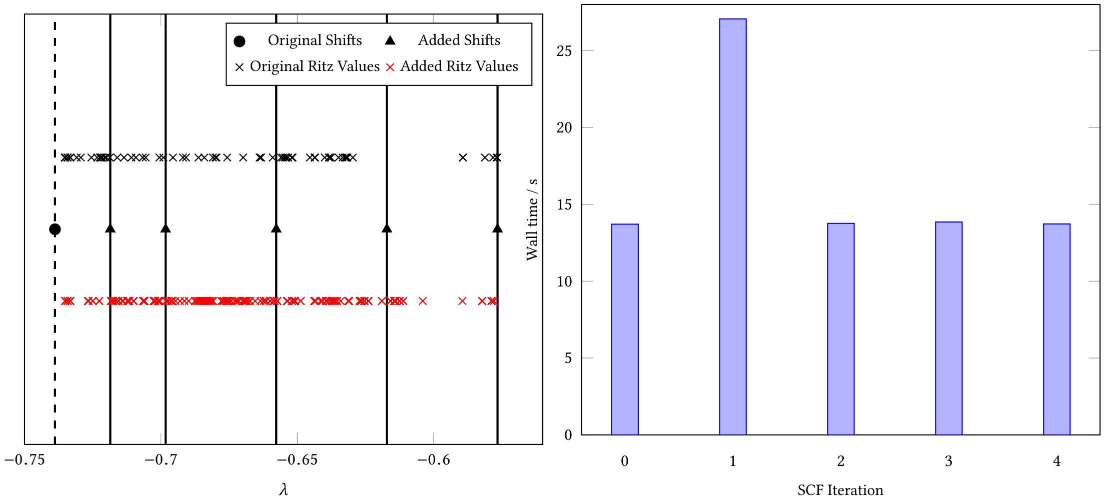
slice $[-0.75, -0.56]$ that contains the missing eigenvalues. The black crosses mark the locations of the approximate eigenvalues that were found by the spectral probe associated with the poorly placed target shift to the left of this interval (marked by a solid black circle).

In this case, after performing an DOS estimate of the matrix pencil updated in the second SCF iteration as suggested in Sec. 3.3, we constructed five new probes whose target shifts were placed at the positions marked by the black vertical lines (with arrows) in Sec. 4.2(a). After new target shifts were selected from the k-means clustering of the computed eigenvalues, some of the inserted probes were mapped to the new shifts and some of them were deleted in the third SCF iterations. In all subsequent SCF iterations, no missing eigenvalue was detected, and SCF convergence was achieved in 10 iterations.

While the inserted new probes captured all the missing eigenvalues, there is an associated cost/penalty for this insertion as shown in Fig. 16b(b). Because the insertion of new probes essentially amounts to a recalculation of part of the spectrum in the second SCF iteration, the wall clock time required to complete that iteration was doubled. However, we should point out that this type of probe insertion is rare in our experiments. Because it only occurs in early SCF iterations, the extra cost is typically amortized over the remainder of the SCF procedure.

5.4 Probe Basis Dimension

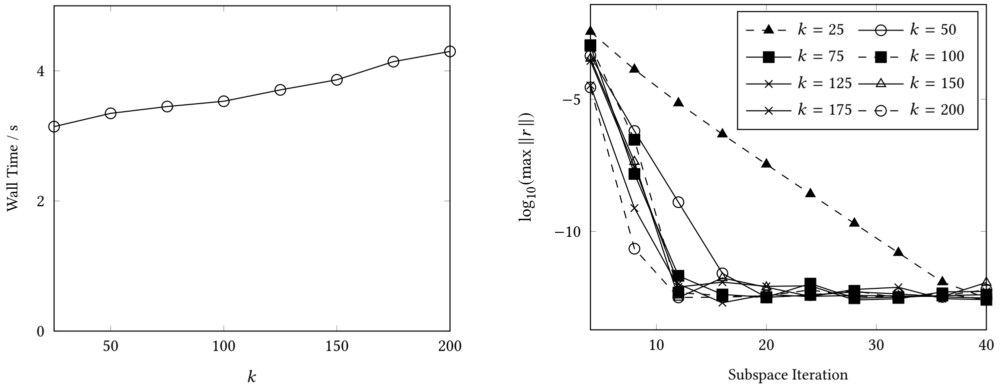
As was discussed in Sec. 2.1, the dimension of the basis used for the shift-invert subspace iterations need only be *at least* the number of desired eigenpairs in the neighborhood of a particular spectral shifts. In practice, the basis dimension should be chosen to be slightly larger to enable faster convergence. However, as the basis dimension increases, the computational time required to perform the shift invert subspace iterations also increases due to the need to solve linear systems with a larger number of right hand sides. In Fig. 17 we examine the effects of basis dimension on the convergence of the subspace iterations as well as on the computational time required to perform the subspace iterations for the Graphene test case.



(a)

(b)

Fig. 16. Example shift insertion for the Graphene test case. The last slice of this test case was determined to have missing eigenvalues per the slice validation scheme at SCF iteration 1. (a) shows the positions of the inserted shifts and the new slices and eigenvalues produced by this insertion. (b) shows the computational required to perform the first 5 SCF iterations with this insertion.



(a)

(b)

Fig. 17. The effects of the probe basis dimension on timings and convergence in the SISLICE method. Results were obtained using the Graphene test case with 100 spectral shifts. Timings (a) were obtained with Intel(R) MKL on 32 Intel(R) Haswell threads. The convergence (b) is tracked as the largest residual norm for the first spectral slice as a function of subspace iteration.

Figure 17a shows the increase in computational time required to perform the subspace iterations as a function of basis dimension. All timing results were obtained using the Haswell partition of the Cori Supercomputer (2x16 Intel(R) Xeon(TM) Processor E5-2698 v3 at 2.3 GHz) using Intel(R) MKL to solve the linear systems and are representative of a single set of 4 subspace iterations. It is clear that even with a 10 fold increase in the basis dimension, the effect on overall timing for the subspace iterations is negligible.

Figure 17b tracks the convergence of a particular spectral slice as a function of the number of subspace iterations. Due to the nearly uniform distribution of the eigenvalues within the Graphene's spectrum, this slice is representative of the entire spectrum. The shift placement was chosen such that each spectral probe is responsible for $\sim$ 10 validated eigenpairs. Unlike the effects on timing, the convergence of the examined spectral slice is sensitive to the basis dimension; with small basis dimensions (e.g. 25) yielding very suboptimal convergence results and large basis dimensions (e.g. 100-200) yielding much faster convergence. In practice, we have found that choosing $k \approx 10n_e/n_s$ yields sufficiently fast convergence in most cases.

5.5 Parallel Scalability

In this section, we examine the parallel scaling behavior of the proposed SISLICE method. All timing results were obtained using the Haswell partition of the Cori Supercomputer (2x16 Intel(R) Xeon(TM) Processor E5-2698 v3 at 2.3 GHz).

Figure 18 demonstrates the strong scaling of a single SCF iteration of the SISLICE method using the Graphene test case. The calculation was performed using 64 shifts and 4 shift-invert subspace iterations per shift on a basis of $k = 100$. The distribution was chosen such that each independent set of LDL^T factorizations and shift-invert subspace iterations utilized 32 threads using the Intel(R) MKL library. The wall time is given in seconds and includes the timings for the LDL^T factorization, shift-invert subspace iterations, Rayleigh-Ritz calculation and probe synchronization (see Sec. 4.1). To demonstrate the effects of the probe synchronization on the overall time, timings are shown with and without the probe synchronization. If we neglect the probe synchronization, we see linear strong scaling. This is to be expected as there is no communication between MPI ranks except for the synchronization. If we include the synchronization, we begin to see the overhead of this procedure at 32 nodes. However, it is clear from this example that this overhead manifests as a prefactor rather than effecting the overall scaling.

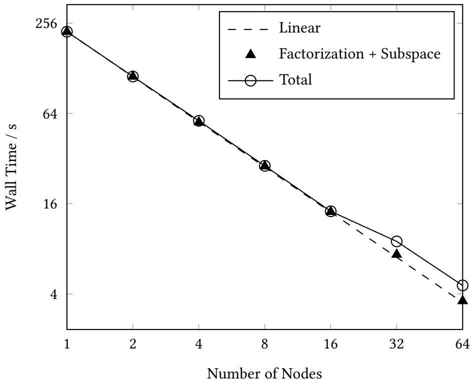


Fig. 18. Strong scaling of the SISLICE method for a single SCF iteration of the Graphene test case using 64 spectral shifts and 4 shift-invert subspace iterations per shift on a basis of $k = 100$. The open circles represent the overall wall time to complete a single SCF iteration (including probe synchronization) of the SISLICE method. The triangles represent the wall time to perform the shift-invert subspace iterations in parallel. The dashed line represents linear scaling.

Figure 19 demonstrates the strong scaling of the SISLICE method compared to ScaLAPACK and ELPA on a large number of processors. For this purpose, we examine the Si10H16 matrix from the University of Florida Sparse Matrix Collection (Davis and Hu 2011) ($N = 17077$) which is representative of a finite difference, pseudo-potential density functional theory calculation using the PARSEC software package (Kronik et al. 2006). Unlike the previously examined Silane and Graphene test cases, Si10H16 is sparse; yielding only 87592 non-zero elements (99.7% zero). As such, we employ the SuperLU_DIST distributed-memory parallel linear solver to perform the shift-invert subspace iterations for this test case. For this example, we compare the full diagonalization of the Si10H16 matrix using ScaLAPACK and ELPA with the partial diagonalization of the lowest 1000 eigenvalues using the SISLICE method ($n_s = 100$, 4 subspace iterations, $k = 100$). A block size of MB = 128 was used for all ScaLAPACK and ELPA diagonalizations. The distribution was chosen such that threads were not utilized for intranode parallelism, i.e. a 1:1 processor-to-MPI rank ratio. Timings for ScaLAPACK and ELPA represent the utilization of the entire process grid to perform the diagonalization. Timings for the SISLICE method utilize a subset of the processors to perform the shift invert subspace iterations (64 / 256 ranks for the 8x8 / 16x16 grids, respectively) and include the time required to perform the probe synchronization. As can clearly be seen, both ScaLAPACK and ELPA drastically outperform the SISLICE method using a small number of computational resources. However, both the strong scaling of ScaLAPACK and ELPA stagnates for large numbers of processors, whereas the SISLICE method continues to exhibit linear scaling. With a very large number of processors (25,600), the SISLICE method using a 16x16 process grid outperforms the best ELPA time by a factor of 2.3x.

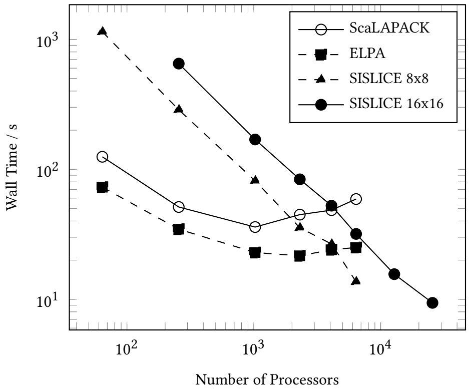


Fig. 19. Strong scaling comparison of ScaLAPACK / ELPA direct diagonalization and the SISLICE method using SuperLU for the Si10H16 test case. The ScaLAPACK and ELPA calculations were performed using a square process grid with a distribution blocking factor of MB = 128. Timings for ScaLAPACK and ELPA represent the full diagonalization using all available processors. The SISLICE calculations were performed using 100 shifts with 4 subspace iterations per shift, and a basis of dimension $k = 100$. SuperLU was used to perform the distributed-memory parallel factorization and linear system solves for the shift-invert subspace iterations on a square subset of the process grid: 8x8 and 16x16, respectively. Timings for the SISLICE calculations include the times required to perform the distributed-memory parallel factorization, linear solves, local Rayleigh-Ritz calculation and spectral probe synchronization.

6 CONCLUSION

In this work, we have developed the SISLICE method: a robust and efficient parallel shift-invert spectrum slicing strategy for self-consistent symmetric eigenvalue computation. The novelty of the SISLICE method is in its shift selection and migration strategies which allow for only minimal communication requirements in its distributed-memory parallel implementation. Like all spectrum slicing methods, the SISLICE method partitions a spectral region of interest into non-overlapping intervals which are then treated independently. However, unlike previous and contemporary slicing methods which rely on sequential shift placement to partition the spectral region of interest, the SISLICE method utilizes DOS estimates to form the entire set of spectral slices at once. This strategy allows for maximal concurrency with minimal communication overhead. As the desired eigenvalues of the considered matrix pencils are dynamic throughout the SCF procedure, the SISLICE method employs a shift migration strategy based on k-means clustering which allows for tracking of the relevant eigenvalues throughout the SCF procedure without the need to recompute the costly DOS estimation at each iteration.

We have demonstrated the robustness and parallel efficiency of the SISLICE method for a representative set of SCF eigenvalue problems commonly encountered in electronic structure theory in Sec. 5. In particular, we have demonstrated that the k-means shift migration yielded noticeable convergence improvements in spectral regions with a highly irregular distribution of eigenvalues (such as the one exhibited for the Silane test case).

From the perspective of performance, the SISLICE method was demonstrated to exhibit linear strong scaling for medium to large problem dimensions up to tens of thousands of processors. This scaling behavior far exceeds those exhibited by distributed direct eigensolvers such as those in ScaLAPACK and ELPA, despite the latter being more performant at low processor counts. Further, we have demonstrated that the main communication requirement, the synchronization of Ritz values and residual norms across the distributed network, yields only a small prefactor in terms of the overall computational time and does not yield a noticeable change in overall scaling.

Despite the demonstrated success of the proposed SISLICE method, there are several topics which were not addressed in this work that should be addressed to fully demonstrate the effectiveness of the method in real applications. The first is the integration of the SISLICE method into an actual SCF code such as NWChemEx, SIESTA, etc. While our results have demonstrated the usefulness of the SISLICE method for pregenerated matrix sequences, the accuracy of the eigenvectors at any particular SCF iteration will influence the overall convergence of the SCF. This topic will be addressed in future work.

Another topic which should be explored is the integration of the SISLICE method with other approximate eigenvalue schemes, such as polynomial filtering, etc. This is of particular interest for spectra which exhibit similar characteristics as the all-electron Silane test case which admits several isolated eigenvalue clusters in the lower region of the spectrum. Due to the isolated nature of these clusters, they would likely be better addressed by polynomial filtering, whereas the larger "clusters" of eigenvalues higher in the spectrum are well addressed by SISLICE. This topic will also be addressed in future work.

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) program. (D. B. W. and C. Y.), and by the Berkeley Lab Undergraduate Research (BLUR) Program, managed by Workforce Development & Education at Berkeley Lab (P. B.). This research was also partially supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration. The authors thank the National Energy Research Scientific Computing (NERSC) center for making computational resources available to them. They would also like to thank Wei Hu for providing the Graphene test matrices generated from the SIESTA software and Ajay Panyala for helping generate the matrix pencils for the Silane example.

REFERENCES

H. M. Aktulga, L. Lin, C. Haine, E. G. Ng, and C. Yang. 2014. Parallel Eigenvalue Calculation based on Multiple Shift-invert Lanczos and Contour Integral based Spectral Projection Method. *Parallel Comput.* 40, 7 (2014), 195.

P. R. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent. 2001. A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling. *SIAM J. Matrix Anal. Appl.* 23, 1 (2001), 15–41.

P. R. Amestoy, A. Guermouche, J.-Y. L'Excellent, and S. Pralet. 2006. Hybrid scheduling for the parallel solution of linear systems. *Parallel Comput.* 32, 2 (2006), 136–156.

E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. 1999. *LAPACK Users' Guide* (third ed.). Society for Industrial and Applied Mathematics, Philadelphia, PA. https://doi.org/10.1137/1.9780898719604

David Arthur and Sergei Vassilvitskii. 2007. K-means++: The Advantages of Careful Seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '07)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1027–1035. http://dl.acm.org/citation.cfm?id=1283383.1283494

John Bachan, Scott B Baden, Steven Hofmeyr, Mathias Jacquelin, Amir Kamil, Dan Bonachea, Paul H Hargrove, and Hadia Ahmed. 2019. UPC++: A High-Performance Communication Framework for Asynchronous Computation. In *Proceedings of the 33rd IEEE International Parallel & Distributed Processing Symposium (to appear)*.

John Bachan, Dan Bonachea, Paul H Hargrove, Steve Hofmeyr, Mathias Jacquelin, Amir Kamil, Brian van Straalen, and Scott B Baden. 2017. The UPC++ PGAS library for exascale computing. In *Proceedings of the Second Annual PGAS Applications Workshop*. ACM, 7.

Zhaojun Bai, James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst. 2000. *Templates for the solution of algebraic eigenvalue problems: a practical guide*. SIAM.

Amartya S. Banerjee, Lin Lin, Phanish Suryanarayana, Chao Yang, and John E. Pask. 2018. Two-Level Chebyshev Filter Based Complementary Subspace Method: Pushing the Envelope of Large-Scale Electronic Structure Calculations. *Journal of Chemical Theory and Computation* 14, 6 (2018), 2930–2946. https://doi.org/10.1021/acs.jctc.7b01243 arXiv:https://doi.org/10.1021/acs.jctc.7b01243 PMID: 29660292.

L. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. Whaley. 1997. *ScaLAPACK Users' Guide*. Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898719642 arXiv:https://epubs.siam.org/doi/pdf/10.1137/1.9780898719642

Carmen Campos and Jose E Roman. 2012. Strategies for spectrum slicing based on restarted Lanczos methods. *Numerical Algorithms* 60, 2 (2012), 279–295.

Timothy A. Davis and Yifan Hu. 2011. The University of Florida Sparse Matrix Collection. *ACM Trans. Math. Softw.* 38, 1, Article 1 (Dec. 2011), 25 pages. https://doi.org/10.1145/2049662.2049663

J. Duersch, M. Shao, C. Yang, and M. Gu. 2018. A Robust and Efficient Implementation of LOBPCG. *SIAM Journal on Scientific Computing* 40, 5 (2018), C655–C676. https://doi.org/10.1137/17M1129830 arXiv:https://doi.org/10.1137/17M1129830

Roger G Grimes, John G Lewis, and Horst D Simon. 1994. A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. *SIAM J. Matrix Anal. Appl.* 15, 1 (1994), 228–272.

Laurent O. Jay, Hanchul Kim, Yousef Saad, and James R. Chelikowsky. 1999. Electronic structure calculations for plane-wave codes without diagonalization. *Computer Physics Communications* 118, 1 (1999), 21 – 30. https://doi.org/10.1016/S0010-4655(98)00192-1

Murat Keçeli, Hong Zhang, Peter Zapol, David A Dixon, and Albert F Wagner. 2016. Shift-and-invert parallel spectral transformation eigensolver: Massively parallel performance for density-functional based tight-binding. *Journal of computational chemistry* 37, 4 (2016), 448–459.

A. Knyazev. 2001. Toward the Optimal Preconditioned Eigensolver: Locally Optimal Block Preconditioned Conjugate Gradient Method. *SIAM Journal on Scientific Computing* 23, 2 (2001), 517–541. https://doi.org/10.1137/S1064827500366124 arXiv:https://doi.org/10.1137/S1064827500366124

Karol Kowalski, Edoardo Apra, Ray Bair, Colleen Bertoni, Jeffery S. Boschen, Eric J. Bylaska, Wibe A. de Jong, Jr. Thom Dunning, Niri Govind, Robert J. Harrison, Kris Keipert, Sriram Krishnamoorthy, Erdal Mutlu, Ajay Panyala, Ryan M. Richard, T. P. Straatsma, Edward F. Valeev, Hubertus J. J. van Dam, Álvaro Vázquez-Mayagoitia, David B. Williams-Young, Chao

Yang, and Theresa L. Windus. [n.d.]. NWChemEx — computational chemistry for the exascale era. *Chem. Rev.* ([n. d.]), In Preparation.

Leeor Kronik, Adi Makmal, Murilo L Tiago, MMG Alemany, Manish Jain, Xiangyang Huang, Yousef Saad, and James R Chelikowsky. 2006. PARSEC–the pseudopotential algorithm for real-space electronic structure calculations: recent advances and novel applications to nano-structures. *Physica Status Solidi (B)* 243, 5 (2006), 1063–1079.

Richard B Lehoucq, Danny C Sorensen, and Chao Yang. 1998. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods.* Vol. 6. Siam.

Ruipeng Li, Yuanzhe Xi, Eugene Vecharynski, Chao Yang, and Yousef Saad. 2016. A Thick-Restart Lanczos algorithm with polynomial filtering for Hermitian eigenvalue problems. *SIAM Journal on Scientific Computing* 38, 4 (2016), A2512–A2534.

Xiaoye S. Li. 2005. An Overview of SuperLU: Algorithms, Implementation, and User Interface. *ACM Trans. Math. Software* 31, 3 (September 2005), 302–325.

Lin Lin, Yousef Saad, and Chao Yang. 2016. Approximating spectral densities of large matrices. *SIAM review* 58, 1 (2016), 34–65.

Stuart Lloyd. 1982. Least squares quantization in PCM. *IEEE transactions on information theory* 28, 2 (1982), 129–137.

Andreas Marek, Volker Blum, Rainer Johanni, Ville Havu, Bruno Lang, Thomas Auckenthaler, Alexander Heinecke, Hans-Joachim Bungartz, and Hermann Lederer. 2014. The ELPA library: scalable parallel eigenvalue solutions for electronic structure theory and computational science. *Journal of Physics: Condensed Matter* 26, 21 (2014), 213201.

E. Polizzi. 2008. Density-matrix-based algorithm for solving eigenvalue problems. *Phys. Rev. B* 79 (2008), 115112.

Yousef Saad. 2011. *Numerical methods for large eigenvalue problems: revised edition.* Vol. 66. Siam.

T. Sakurai and H. Sugiura. 2003. A projection method for generalized eigenvalue problems using numerical integration. *J. Comput. Appl. Math.* 159 (2003), 119–128.

Olaf Schenk and Klaus Gärtner. 2002. Two-level dynamic scheduling in PARDISO: Improved scalability on shared memory multiprocessing systems. *Parallel Comput.* 28, 2 (2002), 187–197.

Olaf Schenk and Klaus Gärtner. 2006. On fast factorization pivoting methods for sparse symmetric indefinite systems. *Electronic Transactions on Numerical Analysis* 23, 1 (2006), 158–179.

Olaf Schenk, Klaus Gärtner, and Wolfgang Fichtner. 2000. Efficient sparse LU factorization with left-right looking strategy on shared memory multiprocessors. *BIT Numerical Mathematics* 40, 1 (2000), 158–176.

Ron Shepard. 1993. Elimination of the Diagonalization Bottleneck in Parallel Direct-SCF Methods. *Theoretica Chimica Acta* 84, 4 (01 Jan 1993), 343–351. https://doi.org/10.1007/BF01113273

Gerard LG Sleijpen and Henk A Van der Vorst. 2000. A Jacobi–Davidson iteration method for linear eigenvalue problems. *SIAM review* 42, 2 (2000), 267–293.

José M Soler, Emilio Artacho, Julian D Gale, Alberto García, Javier Junquera, Pablo Ordejón, and Daniel Sánchez-Portal. 2002. The SIESTA method for ab initio order-N materials simulation. *Journal of Physics: Condensed Matter* 14, 11 (2002), 2745.

A. Stathopoulos and J. R. McCombs. 2007. Nearly optimal preconditioned methods for Hermitian eigenproblems under limited memory. Part II: Seeking many eigenvalues. *SIAM J. Sci. Comput.* 29, 5 (2007), 2162–2188.

James Joseph Sylvester. 1852. XIX. A demonstration of the theorem that every homogeneous quadratic polynomial is reducible by real orthogonal substitutions to the form of a sum of positive and negative squares. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 4, 23 (1852), 138–142.

Attila Szabo and Neil S Ostlund. 2012. *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory.* Courier Corporation.

P. T. P. Tang and E. Polizzi. 2014. FEAST as a subspace iteration eigensolver accelerated by approximate spectral projection. *SIAM J. Matrix Anal. Appl.* 35 (2014), 354–390.

Chao Yang. 2005. Solving large-scale eigenvalue problems in SciDAC applications. *Journal of Physics: Conference Series* 16 (jan 2005), 425–434. https://doi.org/10.1088/1742-6596/16/1/058

Hong Zhang, Barry Smith, Michael Sternberg, and Peter Zapol. 2007. SIPs: Shift-and-invert parallel spectral transformations. *ACM Transactions on Mathematical Software (TOMS)* 33, 2 (2007), 9.