LBNL-41412

Thick-Restart Lanczos Method for Symmetric Eigenvalue Problems[†]

Kesheng Wu[‡]and Horst Simon[‡]

February 12, 1998

Abstract

For real symmetric eigenvalue problems, there are a number of algorithms that are mathematically equivalent, for example, the Lanczos algorithm, the Arnoldi method and the unpreconditioned Davidson method. The Lanczos algorithm is often preferred because it uses significantly fewer arithmetic operations per iteration. To limit the maximum memory usage, these algorithms are often restarted. In recent years, a number of effective restarting schemes have been developed for the Arnoldi method and the Davidson method. This paper describes a simple restarting scheme for the Lanczos algorithm. This restarted Lanczos algorithm uses as many arithmetic operations as the original algorithm. Theoretically, this restarted Lanczos method is equivalent to the implicitly restarted Arnoldi method and the thick-restart Davidson method. Because it uses less arithmetic operations than the others, it is an attractive alternative for solving symmetric eigenvalue problems.

1 Introduction

Given an $n \times n$ matrix A, its eigenvalue λ^* and the corresponding eigenvector x^* are defined by the following relation,

$$Ax^* = \lambda^* x^*.$$

The Lanczos algorithm for symmetric eigenvalue problems appears in many textbooks [13, Section 9.1.2] [26] [29, Section 6.6]. It solves the eigenvalue problem by first building an orthonormal basis, see Algorithm 1, then forming an approximate solution (λ, x) using the Rayleigh-Ritz projection [13, 26, 28]. It is a simple yet effective algorithm for finding extreme eigenvalues and corresponding eigenvectors. Because it only accesses the matrix through matrix-vector multiplications, it is commonly used when the matrices are too large to store in computer memory or not explicitly available. There are a number of algorithms that have

[†]This work was supported by the Director, Office of Energy Research, Office of Laboratory Policy and Infrastructure Management, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

[‡]Lawrence Berkeley National Laboratory/NERSC, Berkeley, CA 94720. Email addresses: {kewu, simon}@nersc.gov.

ALGORITHM **1** *The Lanczos iterations starting with r_0. Let $\beta_0 = \|r_0\|$, and $q_0 = 0$.*

```
For $i = 1, 2, \ldots,$
    (a) $q_i = r_{i-1}/\|r_{i-1}\|$,
    (b) $p = Aq_i$,
    (c) $\alpha_i = q_i^T p$,
    (d) $r_i = p - \alpha_i q_i - \beta_{i-1} q_{i-1}$,
    (e) $\beta_i = \|r_i\|$.
```

ALGORITHM **2** *The Arnoldi iterations starting with r_0.*

```
For $i = 1, 2, \ldots$
    (a) $q_i = r_{i-1}/\|r_{i-1}\|$,
    (b) $p = Aq_i$,
    (c) $h_{j,i} = q_j^T p$, $j = 1, \ldots, i$,
    (d) $r_i = p - \sum_{j=1}^{i} h_{j,i} q_j$,
    (e) $h_{i,i+1} = \|r_i\|$.
```

similar characteristics, for example, the Arnoldi method, see Algorithm 2 [1, 16, 30, 36], and the Davidson method [5, 9, 10, 18, 34, 43]. In fact, for symmetric problems, both the Arnoldi method and the unpreconditioned Davidson method are mathematically equivalent to the Lanczos method. However, the Lanczos algorithm uses fewer arithmetic operations per step because it explicitly takes advantage of the symmetry of the matrix and avoids computing dot-products that are zero. In exact arithmetic, the values of $h_{i,j}, j = 1, \ldots, i-2$, in step *(c)* of Algorithm 2 are zero. The Lanczos algorithm only computes $\alpha_i (\equiv h_{i,i})$. Because $h_{i-1,i} = h_{i,i-1}$, the Lanczos algorithm uses $\beta_{i-1} (\equiv h_{i-1,i})$ in place of $h_{i,i-1}$. Steps *(c)* and *(d)* of Algorithms 1 and 2 are known as the orthogonalization steps. Clearly, the orthogonalization step of the Lanczos algorithm is much cheaper.

When the algorithms are implemented using floating-point arithmetic, the expected zero dot-products are no longer zero. This phenomenon is related to loss of orthogonality among the Lanczos vectors. It was shown that the orthogonality level, $\omega_{i,j} \equiv q_i^T q_j$, can be as large as $\sqrt{\epsilon_u}$, where ϵ_u is the unit round-off error, the Lanczos algorithm will still produce accurate eigenvalue approximations [31]. In most implementations of the Arnoldi method and the Davidson method, full orthogonality is maintained, i.e., the orthogonality level among the basis vectors is roughly ϵ_u. In order to maintain the desired level of orthogonality, re-orthogonalization is commonly used. To achieve stricter orthogonality, usually more re-orthogonalization is needed. The Lanczos algorithm is more tolerant of loss of orthogonality, therefore it can perform less re-orthogonalization compared to other methods. These are the additional advantages of the Lanczos algorithm.

A number of re-orthogonalization schemes are commonly used with the Lanczos algorithm, for example, without re-orthogonalization [6, 7, 41, 42], with full re-orthogonalization, with selective re-orthogonalization [23], or with partial re-orthogonalization [31]. Among these schemes, the Lanczos method without re-orthogonalization is the simplest and it could be implemented with the least amount of computer memory. However, it has the drawback of producing extra copies of eigenvalues known as the spurious eigenvalues. The Lanczos method with full re-orthogonalization can avoid spurious eigenvalues but it often takes more time than the Lanczos methods that only maintain semi-orthogonality. Both selective re-orthogonalization and partial re-orthogonalization maintain semi-orthogonality among the Lanczos vectors. The partial re-orthogonalization scheme will be used later because it is more effective in predicting the orthogonality level and reducing the number of re-orthogonalizations in tests [31].

One effective way of using the Lanczos algorithm is to use it with a shift-and-invert operator, i.e., replacing A with $(A-\delta I)^{-1}$ in Algorithm 1. This scheme allows one to compute the eigenvalues near δ by applying the Lanczos algorithm to compute the extreme eigenvalues of $(A-\delta I)^{-1}$. When δ is close to the wanted eigenvalues of A, the shift-and-invert Lanczos method is often able to find the solutions in a very small number of steps. This scheme is often used to compute interior eigenvalues of A which cannot be easily computed by applying the Lanczos algorithm on A. The shift-and-invert Lanczos method needs to apply the operator $(A-\delta I)^{-1}$ on the Lanczos vectors. This is usually accomplished by solving a linear system, $(A-\delta I)p = q_i$. Often these linear systems can only be solved by a direct LU factorization based method because the solutions have to be fairly accurate. There are many cases where the matrices are not available or the factorizations are too large to be computed or stored. In addition, matrix factorization programs for new computing environments are sometimes not available. Because of these reasons, the shift-and-invert scheme is only available on some computing platforms. The Arnoldi method and the Davidson method can both take advantage of approximate solutions to the above linear systems, in other words, they can be preconditioned. Similar to preconditioning for linear system solutions, good preconditioners are usually problem specific, and effective preconditioners for eigenvalue problems are not widely available. For these reasons the Lanczos algorithm is still one of the favorite methods for finding eigenvalues.

Usually, the minimum number of Lanczos iterations required to compute a given eigenvalue is unknown until the eigenvalue is actually found. To control the maximum memory needed to store the Lanczos vectors, the Lanczos algorithm is restarted. Recently, a number of successful restarting schemes have been reported, for example, the implicit restarting scheme of the implicitly restarted Arnoldi method [36] and the thick-restart scheme of the dynamic thick-restart Davidson method [37, 43]. The implicitly restarted Arnoldi method for the symmetric eigenvalue problem is mathematically equivalent to the *implicitly restarted Lanczos method*. A commonly used implementation of the implicitly restarted Arnoldi method is ARPACK [35]. However, in the current implementation, ARPACK does not explicitly take advantage of the symmetry of the matrix. In this paper we will study a restarted Lanczos method for symmetric eigenvalue problems that takes advantage of the symmetry. Currently, the implicitly restarted Arnoldi method is the most effective method for many eigenvalue problems. We believe a good restarted Lanczos method could be more competitive by avoiding some arithmetic operations and reducing the overall solution time.

The main goal of this paper is to develop a restarted version of the Lanczos algorithm. We are also interested in extending the partial re-orthogonalization technique to minimize the amount of arithmetic operations and in understanding how the orthogonality of the Lanczos vectors affect the accuracy of the solutions. The remainder of paper is structured as follows: Section 2 describes the thick-restart Lanczos method appropriate for implementation in exact arithmetic. Section 3 examines how the loss of orthogonality occurs and extends the ω-recurrence to measure the orthogonality level. Section 4 discusses when to perform global re-orthogonalization and shows that repeating the Gram-Schmidt procedure can restore the orthogonality in the new Lanczos vectors even though the existing Lanczos vectors might have considerable loss of orthogonality. Section 5 shows how the accuracy of coefficients α_i and β_i of the Lanczos algorithm are related to the orthogonality of the Lanczos vectors. Based on these relations, we also give conditions for performing local re-orthogonalization in the Lanczos algorithm. Section 6 shows how to evaluate a number of unknown quantities to develop an effective re-orthogonalization procedure. Section 7 discusses a number of implementation details, examines the error caused by restarting with the Ritz vectors that are not fully orthogonal to each other, and shows the results of applying the restarted Lanczos methods on a number of real-world problems. In Section 8, we summarize the paper, suggest how to implement an effective restarted Lanczos method, and discuss possible improvements to the current implementations used in the tests.

2 Thick-restart Lanczos iteration

In this section, we first review the restarting schemes used with the Arnoldi method and the Davidson method and then describe a restarted Lanczos algorithm in exact arithmetic.

Our interest in restarted Lanczos algorithm is in large part sparked by the success of the implicitly restarted Arnoldi method [16, 35, 36]. Eigenvalue methods like the Arnoldi method and the Lanczos method build their basis progressively. The more steps they take, the more basis vectors they generate. Normally, there is no way to predict how many steps is needed for an eigenvalue to converge. Therefore, there is no way to foresee the storage requirement needed to store the basis vectors. One simple scheme to address this concern is to restart these methods. The restarted eigenvalue methods build their basis as usual until the given workspace is filled up. At which point, if the approximate solutions still need improvement, a restarting strategy is adopted to reduce the space occupied and to make room to build new basis vectors. Restarted methods take more matrix-vector multiplications to converge compared to their nonrestarted counterparts. Recently developed restarting schemes have significantly increased the efficiency of the restarted methods. However, there has not been a restarted eigenvalue method designed specifically for real symmetric eigenvalue problems. The Lanczos method is the most efficient method for symmetric eigenvalue problems in exact arithmetic. Thus it is a good method to restart. Let's first review the restarting schemes available.

Since the Lanczos method naturally starts with one vector, see Algorithm 1, one of the most straightforward restarting schemes is to reduce the whole basis into one vector and start the new Lanczos iterations with it. If only one eigenvalue is needed, we can choose to restart with the Ritz vector. If more than one eigenvalue is wanted, we may add all wanted Ritz vectors together to form one starting vector, or use a block version of the Lanczos algorithm that has the same block size as the number of eigenvalues wanted [28]. These options are simple to implement but not nearly as effective as the more sophisticated ones such as the implicit restarting scheme [36] and the thick-restart scheme [38, 43]. What makes the implicit restarting scheme more efficient are the following,

i. It retains a large portion of the basis. Through carefully arrange the algorithm, it also avoids recomputing the projection matrix in the Rayleigh-Ritz projection.

ii. It can restart with an arbitrary number of starting vectors.

iii. It can use arbitrary shifts to enhance the quality of the starting vectors.

The implicit restarting scheme with exact shifts, i.e., using the unwanted Ritz values as the shifts, is equivalent to restarting with Ritz vectors for symmetric eigenvalue problems [19, 38, 43]. There are a number of restarting schemes for the Davidson method that use Ritz vectors [10, 20, 40]. From the reports, we see that the ability to reuse a large portion of the previous basis greatly enhances their overall efficiencies [19, 38, 43]. Using the exact shifts is generally a good choice. A number of researchers have shown that the *optimal* shifts are Leja points [2]. Significant reduction in the number of matrix-vector multiplications can be achieved by using Leja points as shifts when the maximum basis size is very small, say less than 5. However, when the maximum basis size is larger than 10, using the Leja shifts is not much different from using the exact shifts in the implicitly restarted Arnoldi method. When moderate basis size is used, we can use the Ritz vectors to restart as in the thick-restart scheme. The thick-restart scheme has the first two advantages of the implicit restarting scheme. It is simple to implement and just as effective as the implicit restarting scheme in most cases [38]. Based on this observation, we will design a restarted Lanczos method that restarts with Ritz vectors, i.e., a thick-restart Lanczos method.

Now that we have decided to use Ritz vectors to restart, we can proceed to derive the restarted Lanczos method. Assume that the maximum number of Lanczos step can be taken before restart is m. After m steps of Algorithm 1, the Lanczos vectors satisfy the following Lanczos recurrence,

$$AQ_m = Q_m T_m + \beta_m q_{m+1} e_m^T, \tag{1}$$

where $Q_m = [q_1, \ldots, q_m]$, e_m is the last column of the identity matrix I_m, and $T_m = Q_m^T A Q_m$ is an $m \times m$ symmetric tridiagonal matrix constructed from α_i and β_i as follows,

$$T_m = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{m-2} & \alpha_{m-1} & \beta_{m-1} \\ & & & \beta_{m-1} & \alpha_m \end{pmatrix}.$$

Using the Rayleigh-Ritz projection, we can produce approximate solutions to the eigenvalue problem. Let (λ, y) be an eigenpair of T_m, then λ is an approximate eigenvalue of A, and $x = Q_m y$ is the corresponding approximate eigenvector. They are also known as the Ritz value and the Ritz vector. The result of this approximate solution is defined to be $Ax - \lambda x$. For symmetric eigenvalue problems, the norm of this result is often used as the measure of the quality of the approximation.

When restarting, we first determine an appropriate number of Ritz vectors to save, say k, then choose k eigenvectors of T_m, say Y, and compute k Ritz vectors, $\hat{Q}_k = Q_m Y$. The following derivation can be carried out by assuming Y to be any orthonormal basis of a k-dimensional invariant subspace of T_m. Since the matrix T_m is symmetric and there is no apparent advantage to using a different basis set, we will only use Ritz vectors in the restarted Lanczos algorithm. To distinguish the quantities before and after restart, we denote the quantities after restart with a hat ($\hat{}$). For example, the projected matrix T_m after restart is $\hat{T}_k \equiv Y^T T_m Y$. Since we have chosen to restart with Ritz vectors, the matrix $\hat{T}_k$ is diagonal and the diagonal elements are the Ritz values. Immediately after restart, the new basis vectors satisfy the following relation,

$$A\hat{Q}_k = \hat{Q}_k \hat{T}_k + \beta_m \hat{q}_{k+1} s^T, \tag{2}$$

where $\hat{q}_{k+1} = q_{m+1}$ and $s = Y^T e_m$. We recognize that this equation is an extension of Equation 1. One crucial feature of the Lanczos recurrence is maintained here, i.e., the residual vectors of the basis $\hat{Q}_k$ are in one direction. In Algorithm 1, the Lanczos recurrence is extended one column at a time by augmenting the current basis with q_{m+1}. In the same spirit, we can augment the basis $\hat{Q}_k$ with $\hat{q}_{k+1}$. If there is no relation between $\hat{Q}_k$ and $\hat{q}_{k+1}$, we can build an augmented Krylov subspace from $\hat{q}_{k+1}$ [4, 27]. The thick-restart Arnoldi method and the thick-restart Davidson method generate basis of the same subspace when no preconditioning is used. These two methods are shown to be equivalent to the implicitly restarted Arnoldi method [38, 43]. Because the Arnoldi method and the Lanczos method are mathematically equivalent to each other, this restarted Lanczos method is also equivalent to the implicitly restarted Arnoldi method with the exact shifts. Based on this equivalence, the basis generated after restarting is another Krylov subspace even though we do not know the starting vector.

To compute the new Lanczos vectors after restart we can use the Gram-Schmidt procedure as in the Arnoldi algorithm, see Algorithm 2. Fortunately a cheaper alternative exists because of the symmetry of the matrix. Let's first look at how to compute $\hat{q}_{k+2}$. Based on the Gram-Schmidt procedure, the expression for $\hat{q}_{k+2}$ is as follows,

$$\begin{aligned} \hat{\beta}_{k+1}\hat{q}_{k+2} &= (I - \hat{Q}_{k+1}\hat{Q}_{k+1}^T)A\hat{q}_{k+1} \\ &= (I - \hat{q}_{k+1}\hat{q}_{k+1}^T - \hat{Q}_k\hat{Q}_k^T)A\hat{q}_{k+1} \\ &= (I - \hat{q}_{k+1}\hat{q}_{k+1}^T)A\hat{q}_{k+1} - \hat{Q}_k\beta_m s. \end{aligned} \tag{3}$$

The above equation uses the fact that $\hat{Q}_k^T A\hat{q}_{k+1} = \beta_m s$, which is due to Equation 2 and the orthogonality of the Lanczos vectors. The scalar $\hat{\beta}_{k+1}$ in the above equation is equal to the norm of the right-hand side so that $\hat{q}_{k+2}$ is normalized. This equation shows that $\hat{q}_{k+2}$ can be computed more efficiently than a typical step of the Arnoldi method. Since the vector $\hat{Q}_k^T A\hat{q}_{k+1}$ is known, we only need to compute $\hat{\alpha}_{k+1}$ as in step *(c)* of Algorithm 1 and replace step *(d)* with $\hat{r}_{k+1} = \hat{p} - \hat{\alpha}_{k+1}\hat{q}_{k+1} - \sum_{j=1}^{k} \beta_m s_j \hat{q}_j$, where $\hat{p} = A\hat{q}_{k+1}$. While computing $\hat{q}_{k+2}$, we also extended the matrix $\hat{T}_k$ by one column and one row,

$$\hat{T}_{k+1} = \begin{pmatrix} \hat{T}_k & \beta_m s \\ \beta_m s^T & \hat{\alpha}_{k+1} \end{pmatrix},$$

where $\hat{\alpha}_{k+1} = \hat{q}_{k+1}^T A\hat{q}_{k+1}$. Obviously, the Lanczos recurrence relation, Equation 1, is maintained after restart, more specifically, $A\hat{Q}_{k+1} = \hat{Q}_{k+1}\hat{T}_{k+1} + \hat{\beta}_{k+1}\hat{q}_{k+2}e_{k+1}^T$, where $\hat{\beta}_{k+1} = \|\hat{r}_{k+1}\|$. Even though $\hat{T}_{k+1}$ is not tridiagonal as in the original Lanczos algorithm, it does not affect the restarted Lanczos recurrence as we will show next.

After we have computed $\hat{q}_{k+i}$ $(i > 1)$, to compute the next basis vector $\hat{q}_{k+i+1}$, we again go back to the Gram-Schmidt procedure, see Equation 3,

$$\begin{aligned}
\hat{\beta}_{k+i}\hat{q}_{k+i+1} &= (I - \hat{Q}_{k+i}\hat{Q}^T_{k+i})A\hat{q}_{k+i} \\
&= (I - \hat{q}_{k+i-1}\hat{q}^T_{k+i-1} - \hat{q}_{k+i}\hat{q}^T_{k+i} - \hat{Q}_{k+i-2}\hat{Q}^T_{k+i-2})A\hat{q}_{k+i} \\
&= (I - \hat{q}_{k+i}\hat{q}^T_{k+i} - \hat{q}_{k+i-1}\hat{q}^T_{k+i-1})A\hat{q}_{k+i} - \hat{Q}_{k+i-2}(A\hat{Q}_{k+i-2})^T\hat{q}_{k+i} \\
&= A\hat{q}_{k+i} - \hat{\alpha}_{k+i}\hat{q}_{k+i} - \hat{\beta}_{k+i-1}\hat{q}_{k+i-1},
\end{aligned}$$

where $\hat{\alpha}_{k+i}$ is $\hat{q}^T_{k+i}A\hat{q}_{k+i}$ by definition and $\hat{\beta}_{k+i}$ is the norm of the right-hand side. The above equation is true for any i grater than 2. From this equation we see that computing $\hat{q}_{k+i}$ $(i > 2)$ requires the same amount of work as in the original Lanczos algorithm, see Algorithm 1. The matrix $\hat{T}_{k+i} \equiv \hat{Q}^T_{k+i}A\hat{Q}_{k+i}$ can be written as follows,

$$\hat{T}_{k+i} = \begin{pmatrix}
\hat{T}_k & \beta_m s & & & \\
\beta_m s^T & \hat{\alpha}_{k+1} & \hat{\beta}_{k+1} & & \\
 & \hat{\beta}_{k+1} & \hat{\alpha}_{k+2} & \hat{\beta}_{k+2} & \\
 & & \ddots & \ddots & \ddots \\
 & & & \hat{\beta}_{k+i-1} & \hat{\alpha}_{k+i}
\end{pmatrix}.$$

This restarted Lanczos iteration also maintains the recurrence relation described in Equation 1. Since the Lanczos recurrence relation is satisfied by the restarted Lanczos algorithm, the above equations for computing $\hat{q}_{k+i}$ are not only true after restarting the initial Lanczos iterations, they are true after every restart. The recurrence relation is a three-term recurrence except the first step after restart. Therefore most of the Lanczos vectors can be computed as efficiently as in the original Lanczos algorithm. In addition, using the Lanczos recurrence relation we can estimate the residual norms of the approximate eigenpairs cheaply.

The thick-restart Lanczos algorithm is basically specified by the above equations. Next we will discuss one detail concerning the storage of T_m. As mentioned before, if Y is a collection of eigenvectors of T_m, the matrix $\hat{T}_k$ is diagonal and the diagonal elements can be stored as first k elements of $\hat{\alpha}_i$, $i = 1, ..., k$. The array $(\beta_m s)$ is of size k, it can be stored in the first k elements of $\hat{\beta}_i$, $i = 1, ..., k$. In summary, the arrays $\hat{\alpha}_i$ and $\hat{\beta}_i$ are as follows when restarted,

$$\hat{\alpha}_i = \lambda_i, \qquad \hat{\beta}_i = \beta_m y_{m,i}, \qquad i = 1, \ldots, k, \tag{4}$$

where λ_i is the ith saved eigenvalue of T_m, the corresponding eigenvector is the ith column of Y, and $y_{m,i}$ is the mth element of ith column. At restart the first k basis vectors satisfy the following relation,

$$A\hat{q}_i = \hat{\alpha}_i\hat{q}_i + \hat{\beta}_i\hat{q}_{k+1}.$$

It is easy to arrange the algorithm so that $\hat{q}_i$ and q_i are stored in the same memory in a computer. The hat is dropped in the following algorithm.

ALGORITHM **3** *Restarted Lanczos iterations starting with k Ritz vectors and corresponding residual vector r_k satisfying $Aq_i = \alpha_i q_i + \beta_i q_{k+1}$, $i = 1, \ldots, k$, and $q_{k+1} = r_k/\|r_k\|$. The value k may be zero, in which case, α_i and β_i are uninitialized, and r_0 is the initial guess.*

1. **Initialization**.

   (a) $q_{k+1} = r_k/\|r_k\|$,
   (b) $p = Aq_{k+1}$,
   (c) $\alpha_{k+1} = q_{k+1}^T p$,
   (d) $r_{k+1} = p - \alpha_{k+1}q_{k+1} - \sum_{i=1}^{k} \beta_i q_i$,
   (e) $\beta_{k+1} = \|r_{k+1}\|$,

2. **Iterate**. *For* $i = k+2, k+3, \ldots$,

   (a) $q_i = r_{i-1}/\beta_{i-1}$,
   (b) $p = Aq_i$,
   (c) $\alpha_i = q_i^T p$,
   (d) $r_i = p - \alpha_i q_i - \beta_{i-1}q_{i-1}$,
   (e) $\beta_i = \|r_i\|$.

The difference between Algorithm 1 and 3 is in the initialization step. In most implementations of Algorithm 1, the first iteration of the algorithm differs only slightly from the subsequent steps. At the first iteration, step *(d)* is modified to be $r_1 = p - \alpha_1 q_1$. In Algorithm 3, the difference between the first iteration and the subsequent iterations is similar, i.e., a different number of SAXPY operations are performed. It should be fairly easy to modify an existing Lanczos program based on Algorithm 1 into a restarted version. To convert a complete eigenvalue program to use the above restarted Lanczos algorithm, the Rayleigh-Ritz projection step needs to be modified because the matrix T_m is not tridiagonal in the above restarted Lanczos algorithm. Some of the options to deal with T_m include treating it as a full matrix, treating it as a banded matrix, and using Givens rotations to reduce it to a tridiagonal matrix. After deciding what to do, we can use an appropriate routine from LAPACK or EISPACK to find all eigenvalues and eigenvectors of T_m. At this point, the restarted Lanczos eigenvalue program performs convergence tests as in nonrestarted versions.

If we have not found all wanted eigenvalues, we will restart the Lanczos algorithm. The main decision here is what Ritz pairs to save and how many. Many simple choices are described in literatures [16, 35, 37, 38, 43]. Most of them save a number of Ritz values near the wanted ones. For example, if we want to compute n_d largest eigenvalue of a matrix, we may choose to save $k = n_c + \min(n_d + n_d, m/2)$ largest Ritz values, where n_c is the number of converged eigenvalues. Usually, we also require that $k < m - 3$. Once k is decided, we can prepare all the necessary data to restarted Algorithm 3, $Q_m Y \rightarrow Q_k$, $q_{m+1} \rightarrow q_{k+1}$ and $\alpha_i, \beta_i, i = 1, \ldots, k$ can be computed from Equation 4.

3 Loss of orthogonality

In exact arithmetic, the Lanczos vectors are orthogonal to each other. When the Lanczos algorithm is implemented in floating-point arithmetic, the Lanczos vectors may lose orthogonality after a relatively small number of steps [6, 12, 25]. If no extra work is done to maintain the orthogonality, extraneous copies of eigenvalues will be computed by the Lanczos eigenvalue method [6, 41, 42]. To avoid the spurious solutions and manage the loss of orthogonality, there are a number of schemes to predict the orthogonality level, determine what orthogonality level can be tolerated, and recover the orthogonality when severe loss of orthogonality is detected. Without restart, the Lanczos vectors will lose their orthogonality when some eigenvalues reach convergence [22, 25]. In this case, if the orthogonality level is less than $\sqrt{\epsilon_u}$, the computed eigenvalues are as accurate as if the Lanczos vectors are accurately orthogonal [31, 32, 33]. There are a number of software packages that take advantage of this fact to reduce arithmetic operations [14] [15, Section 10.6] [17] [26]. In the next three sections, we will discuss similar issues for the restarted Lanczos algorithm. In particular, this section will extend the ω-recurrence to monitor the loss of orthogonality [31]. In the next two sections, we will discuss how to maintain the desired level of orthogonality among the Lanczos vectors.

The errors that contribute to the loss of orthogonality can be roughly separated into two parts, the local round-off errors of orthogonalization step *(2.d)* and the global error propagated through the ω-recurrence [14, 31, 32]. In this section, we will first analyze the round-off errors of local orthogonalization, specifically, the error of evaluating step *(2.d)* of Algorithm 3, then derive the ω-recurrence to capture the propagation of error in the restarted Lanczos algorithm.

Before discussing the local loss of orthogonality, let's define a few quantities that will be used. Following the notation of partial re-orthogonalization [14, 31], let $\omega_{i,j} \equiv q_i^T q_j$. If the Lanczos vectors are computed using exact arithmetic, $\omega_{i,j}$, $i \neq j$, are zero. If the vectors are computed using floating-point arithmetic, $\omega_{i,j}, i \neq j$, will not be exactly zero. Estimating the values of $\omega_{i,j}$ is the objective of the ω-recurrence. The orthogonality level of the Lanczos recurrence is ϵ_q if the following is true,

$$\omega_{i,j} < \epsilon_q, \qquad i \neq j. \tag{5}$$

We say that full orthogonality is maintained if $\epsilon_q = \epsilon_u\sqrt{n}$. The Lanczos basis is semi-orthogonal if $\epsilon_q \leq \sqrt{\epsilon_u}$. The following analyses are valid with $\epsilon_q = \sqrt{\epsilon_u}$. In fact, if we use semi-orthogonality with a specific orthogonality level, the orthogonality level is assumed to be $\sqrt{\epsilon_u}$. There are certain advantages to using a smaller ϵ_q. For example, using $\epsilon_q = 10^{-4}\sqrt{\epsilon_u}$ was shown to minimize the time needed to solve some linear systems using the Lanczos method [31].

In the regular Lanczos method, see Algorithm 1, q_{i+1} is computed from Aq_i by orthogonalizing it against q_i and q_{i-1}. The vector norm before orthogonalization is $\|Aq_i\| = \sqrt{\alpha_i^2 + \beta_{i-1}^2 + \beta_i^2}$. The vector is named r_i after the orthogonalization, $r_i = Aq_i - \alpha_i q_i - \beta_{i-1}q_{i-1}$. We define β_i and q_{i+1} as follows

$$\beta_i = \|r_i\|, \qquad q_{i+1} = r_i/\beta_i.$$

When the above computations are carried out in floating-point arithmetic, β_i and q_{i+1} are not computed exactly. We denote their error by d_i, i.e., $d_i = Aq_i - \alpha_i q_i - \beta_{i-1}q_{i-1} - \beta_i q_{i+1}$. Later we will show that α_i, β_{i-1} are computed accurately. Assuming Aq_i, q_i, and q_{i-1} are accurate, the size of d_i can be bounded as follows,

$$\|d_i\| \;\leq\; \epsilon_u \, \|Aq_i + \alpha_i q_i + \beta_{i-1}q_{i-1}\|$$

$$\leq \quad 3\epsilon_u(\sqrt{\alpha_i^2 + \beta_{i-1}^2 + \beta_i^2} + |\alpha_i| + \beta_{i-1}). \tag{6}$$

The above formula is a rough estimate of the local round-off error d_i. The actual error can be larger because Aq_i, q_i, and q_{i-1} are not accurate in actual computations. We usually don't know the error in computing Aq_i because the matrix-vector multiplication may be implemented in any form. The Lanczos vectors are not accurate since we only maintain semi-orthogonality. Even though the above error bound is not reliable, we will see later that it does not significantly affect the estimate orthogonality level because local round-off errors are relatively small. In fact, in many implementations of partial re-orthogonalization, the local round-off errors are ignored [14, 31].

In preparation to extend the ω-recurrence to the restarted Lanczos iterations, we list the Lanczos recurrence with local round-off errors for the restarted Lanczos algorithm [14, 31].

$$\begin{aligned}
Aq_i &= \alpha_i q_i + \beta_i q_{k+1} + d_i, && (i \leq k), && (7)\\
Aq_{k+1} &= \alpha_{k+1} q_{k+1} + \sum_{j=1}^{k} \beta_j q_j + \beta_{k+1} q_{k+2} + d_{k+1}, && && (8)\\
Aq_i &= \alpha_i q_i + \beta_{i-1} q_{i-1} + \beta_i q_{i+1} + d_i, && (i > k+1). && (9)
\end{aligned}$$

The process of evaluating $\omega_{i,j} = q_i^T q_j$ for restarted Lanczos algorithm closely follows the process used to derived the ω-recurrence for non-restarted Lanczos algorithm [14, 31]. The computation of q_{k+2} at step *(1.d)* of Algorithm 3 is a full Gram-Schmidt procedure, see also Equations 3 and 8. The loss of orthogonality in q_{k+2} is treated as a special case in the coming section. This section will discuss the orthogonality level of $q_i (i > k+2)$. Using the identity $q_j^T A q_i = q_i^T A q_j$, we can derive the following recurrence for the restarted Lanczos iteration for the $(i+1)$st row of W_m for $i > k+2$,

$$\begin{aligned}
\beta_i \omega_{i+1,j} &= (\alpha_j - \alpha_i)\omega_{i,j} + \beta_j \omega_{i,k+1} - \beta_{i-1}\omega_{i-1,j} + d_j^T q_i - q_j^T d_i, \qquad (j \leq k), && (10)\\
\beta_i \omega_{i+1,k+1} &= (\alpha_i - \alpha_{k+1})\omega_{i,k+1} + \sum_{j=1}^{k} \beta_j \omega_{i,j} + \beta_{k+1}\omega_{i,k+2} - \beta_{i-1}\omega_{i-1,k+1} \\
&\quad + d_{k+1}^T q_i - q_{k+1}^T d_i, && (11)\\
\beta_i \omega_{i+1,j} &= (\alpha_j - \alpha_i)\omega_{i,j} + \beta_j \omega_{i,j+1} + \beta_{j-1}\omega_{i,j-1} - \beta_{i-1}\omega_{i-1,j} + d_j^T q_i - q_j^T d_i, \\
&\qquad (k+1 < j < i-1), && (12)\\
\beta_i \omega_{i+1,i-1} &= (\alpha_{i-1} - \alpha_i)\omega_{i,i-1} + \beta_{i-2}\omega_{i,i-2} + \beta_i(\omega_{i,i} - \omega_{i-1,i-1}) + d_{i-1}^T q_i - q_{i-1}^T d_i && (13)\\
\beta_i \omega_{i+1,i} &= \alpha_i(1 - \omega_{i,i}) - \beta_{i-1}\omega_{i,i-1} + q_i^T d_i. && (14)
\end{aligned}$$

Equation 14 is directly obtained by multiplying q_i^T on both sides of Equation 9 from the left. In the above equations, we have used the fact that $\omega_{i,j} = \omega_{j,i}$ to rearrange the expressions so that the new row of W can be computed from two previous rows. The above equations contain d_i on the right-hand sides. Since d_i is an unknown vector, we will need to estimate it value in practice. For the convenience of theoretical discussion we assume $\omega_{i,j}$ can be evaluated accurately. We will discuss how to properly estimate $\omega_{i,j}$ in section 6.

The ω-recurrence simulates the loss of orthogonality in the Lanczos iterations. With it, we can extend the partial re-orthogonalization scheme to the restarted Lanczos algorithm.

Before discussing the details of the partial re-orthogonalization scheme, we introduce a few more notations that will be used later.

After m steps of the Lanczos iterations, let L_m be the Cholesky factor of W_m and N_m be an orthonormal basis of column space of Q_m defined as follows, $N_m = Q_m L_m^{-T}$, where L_m^{-T} is the inverse of L_m^T. Let $\tilde{L}_m = L_m - I$, because the diagonal elements of W_m are one by construction, we have the following,

$$\begin{aligned} W_m &= L_m L_m^T = (\tilde{L}_m + I)(\tilde{L}_m + I) = \tilde{L}_m \tilde{L}_m^T + \tilde{L}_m + \tilde{L}_m^T + I \\ &\sim \tilde{L}_m + \tilde{L}_m^T + I. \end{aligned}$$

This shows that the off-diagonal elements of the Cholesky factor L_m are approximately the off-diagonal elements of W_m and the diagonal elements are approximately one,

$$l_{i,j} \sim \omega_{i,j}, \qquad j < i. \tag{15}$$

It is easy to see that the error in this approximation is $\tilde{L}_m \tilde{L}_m^T$. This leads to a second-order approximation of $l_{i,j}$ as follows,

$$\begin{aligned} l_{i,j} &\sim \omega_{i,j} - \sum_{p<j} \omega_{i,p}\omega_{j,p} \qquad j < i, & (16) \\ l_{i,i} &\sim 1 - \frac{1}{2}\sum_{j<i} \omega_{i,j}^2. & (17) \end{aligned}$$

Without restarting, N_m should be exactly what would be computed had the Lanczos iterations been carried out in exact arithmetic. The above decomposition of W_m and Equations 15 – 17 facilitate the study of re-orthogonalization criteria. Depending on how many vectors are involved in the process, we will define two re-orthogonalization schemes, the global re-orthogonalization and the local re-orthogonalization. The global re-orthogonalization scheme explicitly orthogonalizes r_i against all previous Lanczos vectors using the Gram-Schmidt procedure. The local re-orthogonalization scheme only orthogonalizes r_i against q_i and q_{i-1}. Because the global re-orthogonalization contains the operations performed in the local re-orthogonalization, we will discuss the condition for global re-orthogonalization first. Our orthogonalization criteria are established to guarantee accurate α_i and β_i as in the non-restarted version [31].

4 Global re-orthogonalization

In this section, we give the criteria for performing global re-orthogonalization, show that repeating the Gram-Schmidt procedure can restore the orthogonality, and establish a termination criterion for stopping the repetition. Similar to what is done in partial re-orthogonalization, the global re-orthogonalization is applying the Gram-Schmidt procedure to make sure the new Lanczos vector is orthogonal to all existing Lanczos vectors. Our criteria for performing global re-orthogonalization ensure that the norms of the residual vectors r_i are computed accurately. To achieve this, we use 2-norm rather than the infinity norm as in previous partial re-orthogonalization implementations [14, 31]. The study of how the Gram-Schmidt procedure restores orthogonality also clarifies the criteria for performing re-orthogonalization for the Arnoldi method and the Davidson method [8, 10, 16, 30, 39].

In the normal iterations of Algorithm 3, the orthogonality level of q_{i+1} $(i > k+1)$ can be computed after step *(2.e)* using Equations 10 – 14. We use $\omega_{i+1,j}$ to decide whether to perform global re-orthogonalization. Using the definitions of L_{i+1} and N_{i+1}, the vector q_{i+1} is the following linear combination of the exact Lanczos vectors, $q_{i+1} = \sum_{j\le i+1} l_{i+1,j} n_j$, and similarly, $r_i = \beta_i \sum_{j\le i+1} l_{i+1,j} n_j$. If r_i is computed exactly, it should be parallel to n_{i+1}. The error in $r_i^T r_i$ due to loss of orthogonality is $\beta_i^2 \sum_{j\le i} l_{i+1,j}^2$. If

$$\sum_{j\le i} l_{i+1,j}^2 \le \epsilon_u, \tag{18}$$

then the norm of r_i would be the same as if $l_{i+1,j}, j = 1,\ldots,i$ are zero. Since β_i is taken to be the norm of r_i, $\sum_{j\le i+1} l_{i+1,j}^2$ must be one. If $\sum_{j\le i} l_{i+1,j}^2 \le \epsilon_u$, $l_{i+1,i+1}$ is indistinguishable from one in floating-point arithmetic. There are a number of cases where Equation 18 guarantees the norm of r_i can be computed accurately. Since $r_i = Aq_i - \alpha_i q_i - \beta_{i-1} q_{i-1} (i > k+1)$, the following is true,

$$\beta_i = n_{i+1}^T r_i = n_{i+1}^T A q_i = (A n_{i+1})^T q_i.$$

The above equation uses the fact that $n_i^T q_j = 0, i > j$. The same orthogonality relation also hold for $i = k+1$. Assuming there is an exact Lanczos algorithm such that $An_{i+1} = \alpha_{i+1}^* n_{i+1} + \beta_i^* n_i + \beta_{i+1}^* n_{i+2} (i > k)$, where $\alpha_i^* = n_i^T A n_i$ and $\beta_i^* = \|An_i - \alpha_i^* n_i - \beta_{i-1}^* n_{i-1}\|$, we will have $\beta_i = \beta_i^* n_i^T q_i = \beta_i^* l_{i,i}$. Without restarting, the implicit Q theorem guarantees that N_m is the same as the Lanczos vectors computed to full accuracy. Therefore the above assumption is satisfied. If Equation 2 is satisfied accurately, it is easy to see that N_m should be the same as if the restarted Lanczos algorithm is carried out to full accuracy. In both cases, satisfying Equation 18 ensures that the norm of r_i can be computed accurately.

Based on the above observation, we suggest performing global re-orthogonalization if $\sum_{j\le i} l_{i+1,j}^2 > \epsilon_u$. Since the ω-recurrence does not directly produce $l_{i,j}$, we use Equation 15 to restate this condition as follows,

$$\|w_{i+1}\|_2 = \sqrt{\sum_{j=1}^{i} \omega_{i+1,j}^2} > \epsilon_q, \tag{19}$$

To ensure the error in the norm of r_i is computed accurately we need to have $\epsilon_q \le \sqrt{\epsilon_u}$. Typically, we the Gram-Schmidt procedure to perform the global re-orthogonalization, $r_i = (I - Q_i Q_i^T) r_i$, and we assume q_{i+1} is accurately orthogonal to all previous Lanczos vectors afterward [14] [15, Section 10.6] [17, 31]. To ensure the orthogonality is fully restored, often the re-orthogonalization routine allows the Gram-Schmidt procedure to be applied more than once. The effectiveness of this scheme has been studied in the case of QR factorization where full orthogonality is desired [8]. The remainder of this section will show that the Gram-Schmidt procedure is effective even when the existing basis Q_i is semi-orthogonal.

Let's orthogonalize an arbitrary vector z against a semi-orthogonal basis Q_i, the result of the first Gram-Schmidt orthogonalization is $z_{(1)} = (I - Q_i Q_i^T) z$. The dot-products between Q_i and the new vector are as follows,

$$w_z^{(1)} \equiv Q_i^T z_{(1)} = (I - Q_i^T Q_i) Q_i^T z = (I - W_i) Q_i^T z.$$

If we repeat the orthogonalization procedure, i.e., computing $z_{(2)}$ as follows, $z_{(2)} = (I - Q_iQ_i^T)z_{(1)}$, then the dot-products between Q_i and $z_{(2)}$ is

$$w_z^{(2)} \equiv Q_i^T z_{(2)} = (I - W_i)Q_i^T z_{(1)} = (I - W_i)w_z^{(1)}.$$

More generally, let $z_{(0)} \equiv z$ and $z_{(j+1)} = (I - Q_iQ_i^T)z_{(j)}$, we know the following equations are true,

$$\begin{aligned} Q_i^T z_{(j+1)} &= (I - W_i)Q_i^T z_{(j)}, \\ \|w_z^{(j+1)}\| &\leq \|I - W_i\|\|w_z^{(j)}\|. \end{aligned} \tag{20}$$

When the basis Q_i is semi-orthogonal, the eigenvalues of $(I - W_i)$ can be bounded by $\pm i\epsilon_q$ using Gershgorin Theorem [13, Theorem 7.2.1]. The spectral radius $\rho = \|I - W_i\|$ is bounded by $i\epsilon_q$. Usually, this spectral radius is much smaller than one ($i \ll 1/\epsilon_q$), which indicates that the sequence $w_z^{(j)}$, $j = 0, 1, \ldots$, converges to zero. Denote the limit of $z_{(j)}$ by $z_\perp$. It is relatively straightforward to evaluate the value of $z_\perp$.

By repeating the Gram-Schmidt procedure, we are essentially computing $z_\perp$ by the following summation,

$$z_\perp = z + \sum_{j=0}^{\infty}(z_{(j+1)} - z_{(j)}).$$

It can be reformulated using $z_{(j+1)} - z_{(j)} = -Q_iQ_i^T z_{(j)} = -Q_i(I - W_i)^i Q_i^T z_{(0)}$ and $\sum_{i=0}^{\infty} A^i = (I - A)^{-1}$,

$$\begin{aligned} z_\perp &= z - Q_i\left(\sum_{i=0}^{\infty}(I - W_i)^i\right)Q_i^T z \\ &= z - Q_i(I - (I - W_i))^{-1}Q_i^T z \\ &= (I - N_iL_i^T(L_iL_i^T)^{-1}L_iN_i^T)z \\ &= (I - N_iN_i^T)z. \end{aligned} \tag{21}$$

This equation shows that the ultimate result of repeatedly orthogonalizing against a semi-orthogonal basis Q_i leads to the same answer as orthogonalizing against an orthogonal basis of Q_i. This means that repeatedly applying the Gram-Schmidt procedure on r_i can compute it to full accuracy. If the Lanczos recurrence starts with one initial vector, the implicit Q theorem guarantees that after re-orthogonalization, r_i is the same as if all previous Lanczos vectors were computed accurately. When there is more than one starting vector, as in the thick-restart Lanczos iterations, repeating the Gram-Schmidt procedure will still make sure that r_i is accurately orthogonal to all current basis vectors. However, after global re-orthogonalization, it is not clear whether the vector r_i is the same as if we had computed all the basis vectors accurately. We will show that maintaining semi-orthogonality is enough to produce accurate solutions in section 7.

The derivation of Equation 21 is based on repeating the Gram-Schmidt procedure forever. In practice a small number of iterations are enough. Next we will study when to terminate the global re-orthogonalization process. The goal of global re-orthogonalization is to make sure q_{i+1} is fully orthogonal to Q_i. Since the orthogonalization is directly applied on r_i ($q_{i+1} = r_i/\|r_i\|$), we define the error measure as:

$$w_z^{(j)}/\|z_\perp\| \equiv Q_i z_{(j)}/\|z_\perp\| = (I - W_i)^j Q_i^T z_{(0)}/\|z_\perp\| = (I - W_i)^j w_z^{(0)}/\|z_\perp\|.$$

If $\|w_z^{(0)}\|/\|z_\perp\|$ is large, that is, the input vector z is almost a linear combination of Q_i, more iterations will be required. For convenience of discussion, let's denote the ratio $\|w_z^{(0)}\|/\|z_\perp\|$ as ζ. Usually, we can assume that ζ is not larger than $1/\epsilon_u$. The round-off errors of floating-point operations to compute $Q_i Q_i^T z_{(0)}$ can be as large as $\epsilon_u \|w_z^{(0)}\|$, if $\|z_\perp\|$ happens to be smaller than $\epsilon_u \|w_z^{(0)}\|$, the round-off errors will dominate $z_{(1)}$. In this case, $z_{(i)}$ does not converge to $z_\perp$, even though $z_{(i)}$ is perpendicular to the existing basis. The error d_i in Equation 9 is on the round-off error level, i.e., Equation 6 may still be true. However, $\|r_i\|$ is no longer β_i. In practice, we set β_i to zero if the norm of r_i is very small, e.g., $\|r_i\| < \epsilon_u \sqrt{n(\alpha_i^2 + \beta_{i-1}^2)}$. There are two cases where the round-off error will not overtake $z_\perp$, if the actual round-off error happens to be much smaller than the estimate value given above, or the round-off error lies in the column space of Q_i. In these cases, the Gram-Schmidt procedure may be repeated up to four times. If more re-orthogonalization is needed, we declare that the current r_i is a zero vector and we may choose either to terminate the Lanczos algorithm or continue with a different starting vector.

When orthogonality of Q_i is maintained to full accuracy, say, $\epsilon_q = \epsilon_u \sqrt{n}$, $\rho = i\epsilon_u\sqrt{n}$, two Gram-Schmidt iterations should achieve full orthogonality in most cases,

$$\|w_z^{(2)}\|/\|z_\perp\| < i^2 n \epsilon_u^2 \zeta < i^2 n \epsilon_u \qquad (\zeta < 1/\epsilon_u).$$

This confirms the observations made previously [8].

When only semi-orthogonality is maintained, $z_{(j)}$ still converges to $z_\perp$. However, it converges more slowly because of larger ρ, $\rho \le i\epsilon_q$. Given the same vector z, about twice as many steps may be required in order to achieve the same accuracy. If ζ is $1/\epsilon_u$, four orthogonalization iterations may be needed to achieve full accuracy,

$$\|w_z^{(4)}\|/\|z_\perp\| < i^4 \epsilon_u^2 \zeta = i^4 \epsilon_u, \qquad (\epsilon_q = \sqrt{\epsilon_u}).$$

In some implementations of the Lanczos algorithm with re-orthogonalization, e.g., [14] [31], when it is necessary to perform global re-orthogonalization on r_i, global re-orthogonalization is also performed on q_i. This is done to make sure that the future Lanczos vectors are computed from accurate starting vectors. We will adopt the same strategy. However, we may save arithmetic operations by performing different numbers of Gram-Schmidt iterations on the two vectors. Since q_i is semi-orthogonal, applying the Gram-Schmidt procedure once is enough to make it fully orthogonal to previous Lanczos vectors. In addition, there is no need to re-evaluate β_{i-1}. To make r_i fully orthogonal to Q_i, more iterations might be needed. The above analysis shows that four iterations should be enough for most cases, but four iterations of the Gram-Schmidt procedure require a significant amount of arithmetic operations. In actual implementations, we need to determine when to terminate the orthogonalization loop dynamically. The loss of orthogonality after j Gram-Schmidt iterations is $\|w_z^{(j+1)}\|/\|z_\perp\|$, which can be approximated as, $\|w_z^{(j+1)}\|/\|z_\perp\| \sim \|I - W_i\|\|w_z^{(j)}\|/\|z_{(j+1)}\|$. If the following condition is true,

$$\|I - W_i\|\|w_z^{(j)}\|/\|z_{(j+1)}\| > \epsilon_u \sqrt{n}, \tag{22}$$

then the Gram-Schmidt procedure is repeated. In this test, the norm of $w_z^{(j)}$ is cheap to compute, but computing the norm of $z_{(j+1)}$ requires a global sum operation on parallel computers, which might be expensive. However, when z is r_i, it is generally necessary to recompute the norm of r_i to update the value of β_i. Thus the above error estimate can be computed with almost no extra operations.

If the Gram-Schmidt procedure has been applied more than four times, we regard the vector r_i as a zero vector. If not enough eigenvalues have been computed, then the Lanczos iteration can be continued by setting β_i to zero and replacing q_{i+1} with an arbitrary unit vector that is orthogonal to Q_i.

After a global re-orthogonalization, the loss of orthogonality is assumed to be $\epsilon_u\sqrt{n}$ as in most implementations of the partial re-orthogonalization scheme [14, 31].

Equation 19 is only applicable to the regular iterations of Algorithm 3. When computing q_{k+2}, it is clear that $\omega_{k+2,j}$ cannot be computed using only the last two rows of W_{k+1}. In addition, the local re-orthogonalization and global re-orthogonalization are identical at this stage of the Algorithm. Instead of using Equation 19 as the criterion for global re-orthogonalization, we need to derive a criterion based on Equation 22. The condition we use is,

$$\rho\|Q_{k+1}^T z_{(j-1)}\| > \epsilon_u\sqrt{n}\|z_{(j)}\|, \qquad (\rho = \|I - W_{k+1}\|) \tag{23}$$

where $z_{(0)} = r_{k+1}$. When the basis Q_{k+1} is maintained to full orthogonality, the above inequality is often approximated as [8, 16, 35],

$$\|Q^T z_{(j-1)}\| > \|z_{(j)}\|. \tag{24}$$

We have dropped the subscript to Q to indicate that it can be an arbitrary size. The underlying assumption of this test is that $\rho = \epsilon_u\sqrt{n}$, which is roughly true if the basis size is relatively small.

If full orthogonality is desired, we suggest only using global re-orthogonalization. In this case, there is no need to evaluate the ω-recurrence and the previous re-orthogonalization criterion can be simplified as follows,

$$\sqrt{\alpha_i^2 + \beta_{i-1}^2} \geq \|r_i\|. \tag{25}$$

We may still use Equation 24 to decide whether to stop the global re-orthogonalization.

5 Local re-orthogonalization

We will now consider the condition for performing local re-orthogonalization in the cases where global re-orthogonalization is not needed. Based on the results of previous studies [24], our local re-orthogonalization will repeat step *(2.d)* of algorithm 3, i.e., orthogonalizing r_i against both q_i and q_{I-1}. To establish the criteria for local re-orthogonalization, we will start by looking at the formulas for computed α_i and β_i. When semi-orthogonality is maintained, it was shown that the errors in α_i and β_i are on the order of $\epsilon_u\|A\|$ without restart [31, 32]. In this section we will demonstrate when the same is true for restarted Lanczos algorithm.

Since the global re-orthogonalization is not necessary, i.e., $\sum_{j\le i}\omega_{i+1,j}^2 \le \epsilon_u$, the diagonal element of L_{i+1} should be accurately one. Using this fact, we can write q_i as,

$$q_i = n_i + \sum_{j<i} l_{i,j} n_j. \tag{26}$$

The computed α_i is given by the following formula,

$$\begin{aligned}
\alpha_i &= (n_i + \sum_{j<i} l_{i,j} n_j)^T A (n_i + \sum_{j<i} l_{i,j} n_j), \\
&= n_i^T A n_i + 2 n_i^T A \sum_{j<i} l_{i,j} n_j + (\sum_{j<i} l_{i,j} n_j)^T A (\sum_{j<i} l_{i,j} n_j) \\
&= \alpha_i^* + 2 l_{i,i-1}\beta_{i-1}^* + \sum_{j=1}^{i-1} \alpha_j^* l_{i,j}^2 + \\
&\quad 2\sum_{j=1}^{k} \beta_j^* l_{i,j} l_{i,k+1} + 2\sum_{j=k+1}^{i-2} \beta_j^* l_{i,j-1} l_{i,j}.
\end{aligned}$$

This derivation uses the fact that $i > k+1$. The quantities α_i^* and β_i^* are exact values of the computed α_i and β_i, i.e., $\alpha_i^* = n_i^T A n_i$ and $\beta_i^* = \|An_i - \alpha_i^* n_i - \beta_{i-1}^* n_{i-1}\|$. To reduce the error in α_i we need to first reduce the size of $2l_{i,i-1}\beta_{i-1}^*$ because all other error terms are much smaller. Using Equation 16, we see that $l_{i,i-1} \sim \omega_{i,i-1} - \sum_{j<i-1} \omega_{i,j}\omega_{i-1,j}$. This shows that in order to reduce the error in α_i, we need to make sure $\omega_{i,i-1}$ is small. When computing α_i at step *(2.c)* of Algorithm 3, we have computed q_i already. At this point of the algorithm, in order to reduce $\omega_{i,i-1}$, we would have to recompute q_i which means another matrix-vector multiplication is needed to re-evaluate α_i. One alternative to this scheme is to make sure r_{i-1} is orthogonal to q_{i-1} which can be done by local re-orthogonalization.

To establish the condition for local re-orthogonalization, we evaluate $\omega_{i+1,i}$ using Equation 14 after step *(2.e)* of Algorithm 3. Assuming that $q_i^T d_i$ is negligible, $\omega_{i+1,i}$ can be approximated as $\omega_{i+1,i} = -\beta_{i-1}\omega_{i,i-1}/\beta_i$. If $\beta_{i-1}/\beta_i > 1$, than $\omega_{i+1,i}$ would be larger than $\omega_{i,i-1}$. We will orthogonalize r_i against q_i if the following is true,

$$\beta_{i-1} > \beta_i. \tag{27}$$

Using Equation 26 and step *(2.e)* of Algorithm 3, we can express r_i and q_{i+1} as,

$$r_i = \beta_i (n_{i+1} + \sum_{j\le i} l_{i+1,j} n_j), \quad q_{i+1} = n_{i+1} + \sum_{j\le i} l_{i+1,j} n_j.$$

Let $r_i^{(1)}$ be the residual vector after local re-orthogonalization, i.e., $r_i^{(1)} = r_i - q_i q_i^T r_i$. It is easy to show the following,

$$r_i^{(1)} = \beta_i \left(n_{i+1} + (l_{i+1,i} - \omega_{i+1,i}) n_i + \sum_{j<i} (l_{i+1,j} - \omega_{i+1,i} l_{i,j}) n_j \right).$$

The orthogonality between q_i and $q_{i+1}^{(1)} \equiv r_i^{(1)}/\|r_i^{(1)}\|$ is

$$q_i^T q_{i+1}^{(1)} = l_{i+1,i} - \omega_{i+1,i} + \sum_{j<i} (l_{i+1,j} - \omega_{i+1,j} l_{i,j}) l_{i,j} = -\omega_{i+1,i} \sum_{j<i} l_{i,j} l_{i,j}.$$

The above derivation uses the fact that $\omega_{i+1,i} = l_{i+1,i} + \sum_{j<i} l_{i+1,j} l_{i,j}$. If semi-orthogonality is maintained, $\sum_{j<i} l_{i+1,j} l_{i,j} \leq \frac{1}{2}(\sum_{j<i} l_{i+1,j}^2 + \sum j < i l_{i,j}^2) \sim \frac{1}{2}(\sum_{j\leq i} \omega_{i+1,j}^2 + \sum j < i \omega_{i,j}^2) < \epsilon_q^2$. It shows that after orthogonalizing against q_i only once, the vector r_i is fully orthogonal to q_i. There is no need to repeat the orthogonalization against q_i. After this local orthogonalization, we set $w_{i+1,i}$ to $\epsilon_u \sqrt{n}$ as in previous implementations of the ω-recurrence [14, 31]. The loss of orthogonality of $q_{i+1}^{(1)}$ against other Lanczos vectors is

$$q_j^T q_{i+1}^{(1)} = \omega_{i+1,j} - \omega_{i+1,i} l_{i+1,j}^2, \quad j < i.$$

Since the changes are small for $\omega_{i+1,j}, j < i$, we choose not to modify them. Another point to note is that after orthogonalizing against q_i, there is no need to recompute β_i because the dot-product $r_i^T r_i$ cannot be computed accurately enough to capture the change.

After one local re-orthogonalization, r_i is accurately orthogonal to q_i. The limitation on the actual value of $\omega_{i+1,i}$ is the round-off error of computing r_i. Assuming $\omega_{i+1,i} = \epsilon_u \sqrt{n}$, the error in α_{i+1} due to loss of orthogonality is $2\beta_i \epsilon_u \sqrt{n}$. The round-off error in the dot-product operation used to compute α_{i+1} is bounded by $n\epsilon_u |q_{i+1}|^T |Aq_{i+1}|$ [13, Section 2.4.5]. If β_i is not significantly larger than the magnitude of α_{i+1}, then the error caused by loss of orthogonality would be no larger than the round-off error of computing α_{i+1}.

There are many ways to implement the Lanczos algorithm [21, 26]. In particular, there are at least two ways of computing β_i. One is to use $\beta_i = \|r_i\|$ as in Algorithm 3. The other is to compute β_i as a coefficient of Gram-Schmidt orthogonalization, $\beta_i = q_i^T A q_{i+1}$. The first formula computes the sub-diagonal elements $(h_{i,i+1})$ of $T_m = Q_m^T A Q_m$, and the second formula computes the super-diagonal elements $(h_{i+1,i})$ of T_m, see Algorithm 2. The matrix T_m is symmetric because A is symmetric. In order to maintain the symmetry of the matrix T_m, the β_i values computed from the two formulas should be the same. To distinguish the two β_i values, we define $\beta_i^{(-)} = \|r_i\|$ and $\beta_i^{(+)} = q_i^T A q_{i+1}$. Because the matrix A is symmetric and $Aq_i = \alpha_i q_i + \beta_{i-1} q_{i-1} + \beta_i q_{i+1}$, we can establish the following relation between the two values,

$$\beta_i^{(+)} = \alpha_i \omega_{i+1,i} + \beta_{i-1} \omega_{i+1,i-1} + \beta_i^{(-)} \omega_{i+1,i+1}.$$

The value of $\omega_{i+1,i+1}$ is one if semi-orthogonality is maintained. The difference between the two β_i values is

$$\beta_i^{(+)} - \beta_i^{(-)} = \alpha_i \omega_{i+1,i} + \beta_{i-1} \omega_{i+1,i-1}.$$

This shows that both $\omega_{i+1,i}$ and $\omega_{i+1,i-1}$ have to be small in order to maintain symmetry of the computed T_m [24]. In some implementations, local re-orthogonalization only orthogonalizes r_i against q_i [14]. Extended local re-orthogonalization [24, 26] [15, Section 10.6], i.e., orthogonalizing r_i against both q_i and q_{i-1}, is clearly important to a successful implementation of the Lanczos method. We choose to orthogonalize r_i against q_i and q_{i-1} if the following is true,

$$|\alpha_i \omega_{i+1,i}| + |\beta_{i-1} \omega_{i+1,i-1}| > \epsilon_u n \|r_i\|. \tag{28}$$

The right-hand side of the above inequality is the round-off error of computing the norm of r_i [13, Section 2.4.5]. If the difference between two versions of β_i is large enough to be computed in the formula $\beta_i = \|r_i\|$, then the extended local re-orthogonalization should be performed. We have shown that orthogonalizing r_i against q_i reduces $\omega_{i+1,i}$ to below ϵ_u.

Similarly, orthogonalizing r_i against q_{i-1} should reduce $\omega_{i+1,i-1}$ below ϵ_u as well. In both case, applying the Gram-Schmidt procedure once is enough.

The goal of this criterion is to ensure the projected matrix T_m is symmetric to the full precision. In other words, we want to compute β_i to full precision of floating-point arithmetic. After the extended local re-orthogonalization, if the norm of r_i is not much smaller than α_i and β_{i-1}, then β_i is exactly the norm of r_i. If the norm of r_i is very small compared to α_i or β_{i-1}, then the local re-orthogonalization is needed to reduce the size of $\omega_{i+1,i}$ and $\omega_{i+1,i-1}$. Based on previous experience, we set β_i to zero if the following is true, $\|r_i\| < \epsilon_u\sqrt{n(\alpha_i^2 + \beta_{i-1}^2)}$.

In most cases, the difference between $\beta_i^{(+)}$ and $\beta_i^{(-)}$ is an overestimate of the error in β_i computed using $\beta_i = \|r_i\|$. When discussing the global re-orthogonalization criteria, we have pointed out two cases where our criteria should ensure that the norm of r_i can be computed accurately. This also partly explains why the formula $\beta_i = \|r_i\|$ is favored over $\beta_i = q_i^T A q_{i+1}$ in most implementations of the Lanczos algorithm [14] [15, Section 10.6] [17, 31].

The above local re-orthogonalization criteria, Equations 27 and 28, are based on the assumption that N_m are exact Lanczos vectors. Assuming that the Lanczos method is started with a nontrivial vector at the beginning, then Equation 2 is satisfied accurately at every restart. The restarted Lanczos algorithm with above global and local re-orthogonalization schemes should generate accurate $\alpha_i, \beta_i, i = k+2, \ldots, m$. Therefore it also generates accurate eigenvalue approximations.

Equation 2 can be satisfied to different accuracies depending on how the Lanczos algorithm is actually implemented. The most straightforward implementation is to apply the full re-orthogonalization using the criterion described in Equation 15. As we have argued before, to maintain full orthogonality, a significant amount of re-orthogonalization work is required. On the other hand, the partial re-orthogonalization with the above global and local re-orthogonalization criteria may not be able to guarantee accurate solutions. We may need to perform additional orthogonalization in order to compute accurate eigenvalues.

Normally, the Lanczos iterations are started with one initial vector. Before the first restart, there is an orthonormal basis N_m, such that the Lanczos recurrence, Equation 1, is satisfied accurately [31],

$$AN_m = N_m T_m + \beta_m n_{m+1} e_m^T \tag{29}$$

Through the analysis in section 4, we know that n_{m+1} can be computed accurately without first computing N_m. This can be achieved by applying global re-orthogonalization on r_m. After the re-orthogonalization, the above equation can be replaced with $AN_m = N_m T_m + \beta_m q_{m+1} e_m^T$. At restart, we set $\hat{T}_k = Y^T T_m Y$, where Y consists of k selected eigenvectors of T_m. The corresponding Lanczos vectors should be $\hat{Q}_k^* = N_m Y$. We can compute $\hat{Q}_k^*$ by explicitly computing $W_m = Q_m^T Q_m$ and perform Cholesky factorization on W_m, after which, the new starting Lanczos vectors can be computed as

$$\hat{Q}_k^* = Q_m L_m^{-T} Y.$$

We may choose to first evaluate $L_m^{-T} Y$ then perform the linear combination operation. This process correctly restores the orthogonality of the starting vectors $\hat{Q}_k$. If we have computed q_{m+1} to full accuracy, then Equation 2 is satisfied accurately. If $k \ll n$, the costs of Cholesky

name	N	NNZ	description
NASASRB	54870	2677324	shuttle rocket booster structure from NASA
S3DKT3M2	90449	3753461	cylindrical shell, uniform triangular mesh
S3DKQ4M2	90449	4820891	cylindrical shell, quadratic elements

Table 1: Information about the test matrices used.

factorization and the triangular solution are relatively small. The main cost of this correction scheme is in computing W_m. Since computing W_m is roughly half as expensive as performing global re-orthogonalization for every Lanczos vector, for many eigenvalue problems using partial re-orthogonalization with this correction scheme might be more efficient than maintaining full orthogonality by performing global re-orthogonalization at every step.

In the above correction scheme, one of the steps is to make sure r_m is fully orthogonal to the existing Lanczos vectors. This generates an orthogonal residual vector for the Lanczos iterations rather than a semi-orthogonal residual vector. Making this vector as accurate as possible is important because after restarting the new vectors are computed from this one. Orthogonalizing only r_m does not guarantee the accuracy of Equation 2. However, because the new vectors are computed from an accurate starting vector, loss of orthogonality before restart is not amplified after the restart. Figure 1 shows the comparison of the orthogonality of the bases computed with and without orthogonalizing r_m. In both examples, the orthogonality of the bases with orthogonal residual vectors are almost constant, but the orthogonality of the bases with semi-orthogonal residual vectors increases quickly as more restarts are used. It is clear from this small experiment that maintaining an orthogonal residual when restarting is effective in reducing error propagation.

If the relation described in Equation 2 is not satisfied, we can implement the algorithm by first generating the Lanczos basis with $\hat{q}_{k+1}$ as the starting vector. After $m-k$ steps of the Lanczos algorithm, we appended $\hat{Q}_k$ to the current Lanczos basis to form an augmented Krylov subspace [4, 27]. Mathematically, this is equivalent to Algorithm 3. This variation also has the advantage of allowing any number of arbitrary vectors to be appended at the end. The drawback is that k matrix-vector multiplications have to be performed in order to complete the Rayleigh-Ritz projection for the k vectors in $\hat{Q}_k$. Using Algorithm 3, to build up the Lanczos basis of size m, only $m-k$ matrix-vector multiplications are needed. Therefore we prefer to use Algorithm 3. Later we will see that the restarted Lanczos algorithm with partial re-orthogonalization computes solutions that are as accurate as the variant with full re-orthogonalization.

6 Effective re-orthogonalization procedure

An effective re-orthogonalization procedure needs to evaluate $\omega_{i,j}$ accurately and restore orthogonality in the least amount of time. This section will address these two issues based on a number of numerical experiments. In these experiments, we try to find the five largest eigenvalues and their corresponding eigenvectors. The test matrices include NASASRB and 14 other test matrices from various sources, such as magnetohydrodynamics, structure

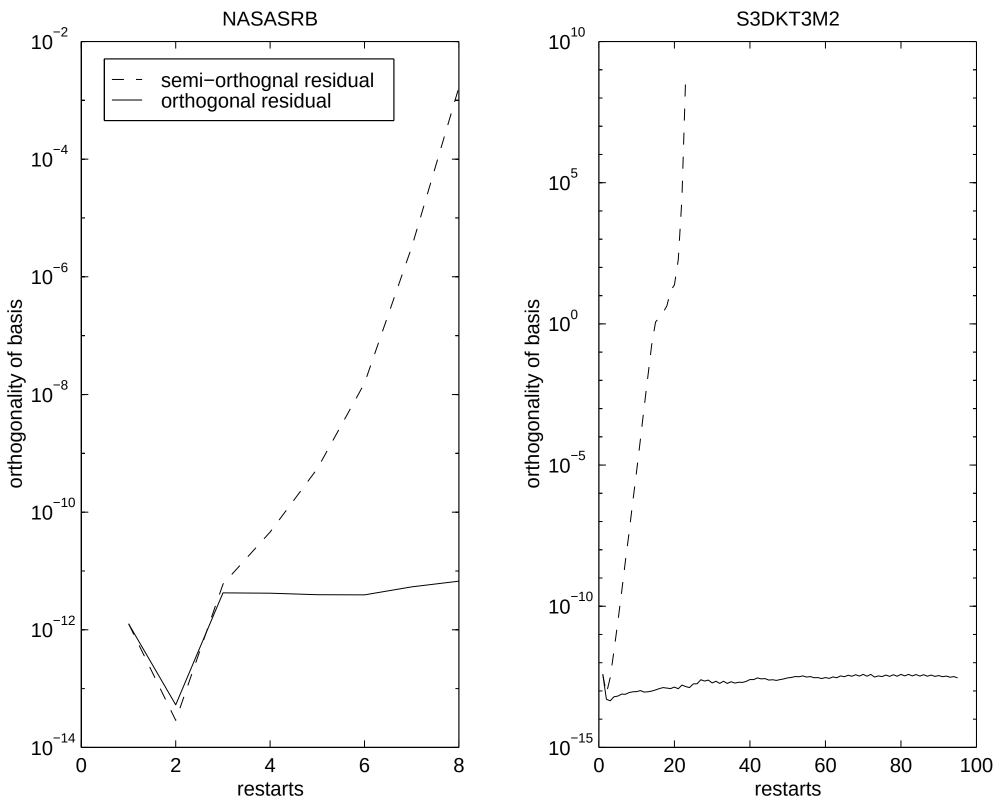

Figure 1: The orthogonality level ($\|Q^TQ - I\|_F$) of the bases prior to each restart.

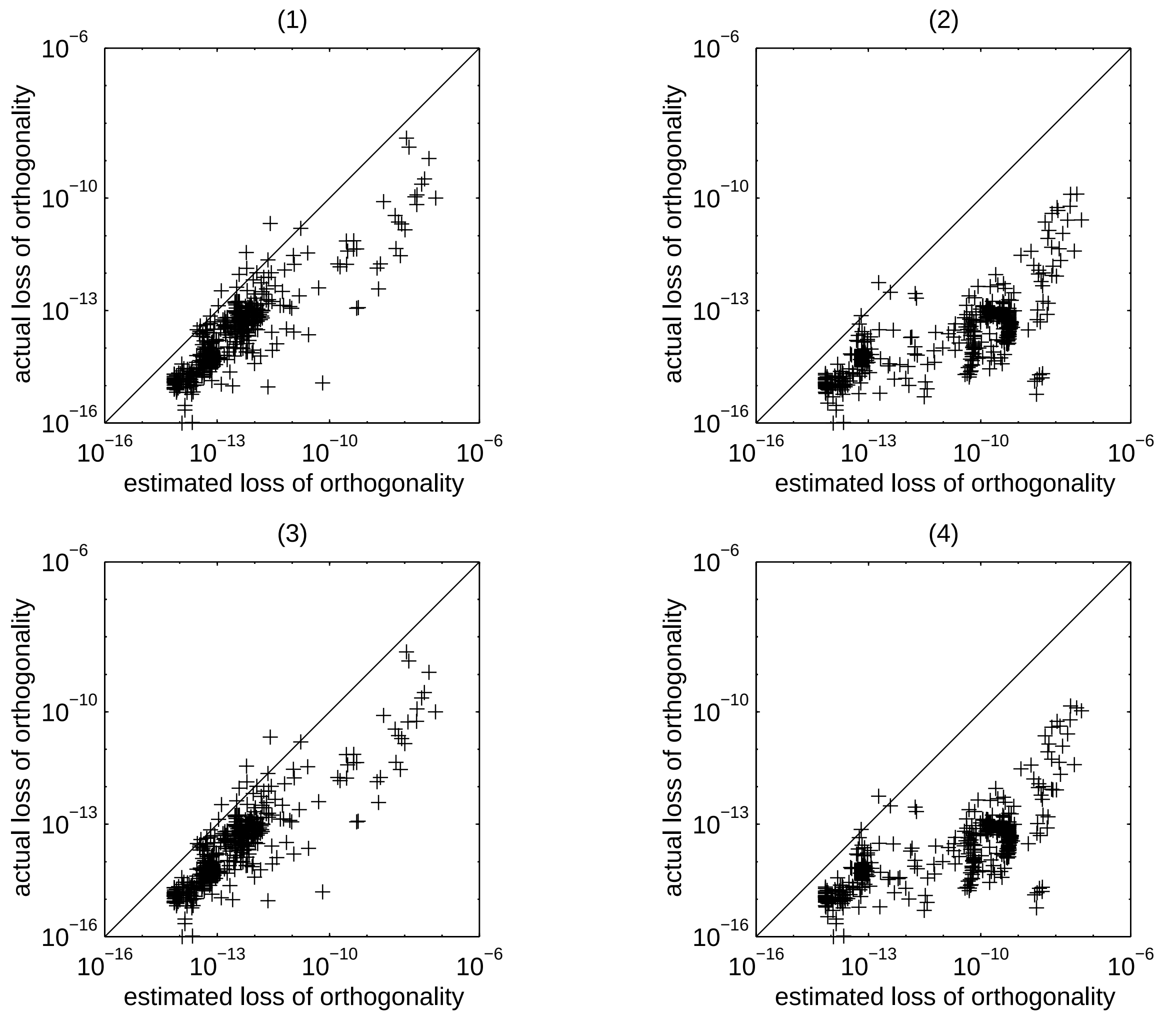

Figure 2: Estimated orthogonality level versus actual orthogonality level: (1) signed ω-recurrence, ignoring $\xi_{i,j}$; (2) absolute ω-recurrence, ignoring $\xi_{i,j}$; (3) signed ω-recurrence, adding $\|d_i\|$; (4) absolute ω-recurrence, adding $\|d_i\|$.

simulation, *Ab Initio* simulation of electronic properties, oceanographic model, and so on. Information about three matrices that will also be used later are listed in Table 1. All test matrices are in Harwell-Boeing format [11] and the three listed in the above table are among the largest symmetric matrices available from MatrixMarket[1] at the time of this test.

To accurately estimate $\omega_{i,j}$ using Equations 10 – 14, we need to estimate a number of terms. The first one is $\xi_{i,j} \equiv q_j^T d_i - q_i^T d_j$. In the previous implementations of partial re-orthogonalization [14, 31], this term was assumed to be zero. These terms are related to local round-off errors, which are only a small part of the loss of orthogonality. By definition of $d_i = Aq_i - \alpha_i q_i - \beta_{i-1} q_{i-1} - \beta_i q_{i+1}$, we expect the primary components of d_i to be in the direction of q_i, q_{i-1} and q_{i+1}. This again confirms that most elements of $\xi_{i,j}$ are zero. We can bound the magnitude of $q_{i-1}^T d_i$ and $q_i^T d_i$ by $\|d_i\|$. On this issue, we tested two options,

[1]MatrixMarket URL: `http://math.nist.gov/MatrixMarket/`.

ignoring $\xi_{i,j}$ in the ω-recurrence, or using $\|d_i\|$ to replace $\xi_{i,i-1}$ and $q_i^T d_i$ in Equations 13 and 14. The testing results are shown in Figure 2. We will discuss them after we explain the second set of plots in the figure.

We can use Equations 10 – 14 as they are, with the plus and minus signs, or we can overestimate $\omega_{i,j}$ and use only absolute values, for example, changing Equation 12 to

$$\beta_i \omega_{i+1,j} = |\alpha_j - \alpha_i|\omega_{i,j} + \beta_j|\omega_{i,j+1}| + \beta_{j-1}|\omega_{i,j-1}| + \beta_{i-1}|\omega_{i-1,j}| + |\xi_{i,j}|.$$

The argument for using the absolute value in the recurrence is that we don't know the sign of $\xi_{i,j}$ or the sign of $\omega_{2,1}$ which is used to start the ω-recurrence. Using absolute values in the recurrence formulas will provide a correct upper bound on the orthogonality level. If we don't use absolute values, the estimated orthogonality level can be smaller than the actual orthogonality level. It is possible that a Lanczos vector would need to be re-orthogonalized but the ω-recurrence predicts that the orthogonality level is within tolerance. If the actual orthogonality level is worse than $\sqrt{\epsilon_u}$, the computed eigenvalues could be wrong. This is a case that should be avoided.

Figure 2 compares the estimated loss of orthogonality by ω-recurrence versus the actual loss of orthogonality. The data is collected from all 15 test problems. The basis size is 20. Only one starting vector is provided to the Lanczos routine. The vector is $[1, 1, \ldots, 1]^T$. The Lanczos method was able to find the five largest eigenvalues of all 15 test problems. There are above 550 data points in each plot. Each data point is collected when a global re-orthogonalization is performed, most of which are performed when the loss of orthogonality is not severe, e.g., step *(1.d)* of Algorithm 3.

One obvious observation from Figure 2 is that using absolute values in ω-recurrence often causes the estimated orthogonality level to be significantly larger than the actual orthogonality level. When the ω-recurrence is used as shown in Equations 10 – 14, the estimated orthogonality level is still larger than the actual orthogonality level in most cases, see plots (1) and (3). However, the data points in plots (1) and (3) are closer to the diagonal than those in plots (2) and (4). The signed ω-recurrence predicts 15 cases of severe loss of orthogonality for all test problems. Among the 15 predicted cases of severe loss of orthogonality, in only one case the actual orthogonality level is beyond the tolerance. The ω-recurrence with absolute values predicts 32 cases of severe loss of orthogonality, none of which are true. By having the estimated orthogonality level closer to the actual value, we reduce the total number of re-orthogonalizations needed. The data shown in Figure 2 are collected with the maximum basis size 20. If the basis size is larger, the ω-recurrence with absolute values will cause even more unnecessary global re-orthogonalizations.

From Figure 2, we see that adding local round-off error to the ω-recurrence does not significantly change the estimated value of $\omega_{i,j}$. There is no noticeable difference between plots (1) and (3) or (2) and (4). Normally, the values of $\|d_i\|$ are relatively small compared to the other terms in the ω-recurrence. However, since it does not cost many arithmetic operations to compute, we decided to keep it to deal with the cases where the local round-off error happens to dominate.

We should avoid underestimating the orthogonality levels. However, Figure 2 reveals a number of cases where the predicted orthogonality levels are lower than their actual values. There are ten cases when using the signed ω-recurrence and two cases when using the ω-

recurrence with absolute values. In all of these cases, the actual orthogonality levels are much smaller than the tolerance, therefore it did not affect the accuracy of the solutions. The signed ω-recurrence may underestimate the loss of orthogonality. The experiments show that even the ω-recurrence with absolute values may underestimate the loss of orthogonality as well. The first reason for this is that we do not have a proper estimate for the local round-off error. The local round-off errors are generally small compared to other terms in ω-recurrence. However, since we reset the ω-recurrence to $\epsilon_u\sqrt{n}$ after global re-orthogonalizing, we essentially ignored local round-off error d_i associated with existing Lanczos vectors. The orthogonality levels of the next few Lanczos vectors are estimated as if all previous Lanczos vectors are computed accurately. In these cases, loss of orthogonality due to the local round-off errors $\xi_{i,j}$ may be larger than the other terms in the ω-recurrence. The error bound on computing dot-product of two unit vectors is roughly $\epsilon_u n$, but we estimate it to be $\epsilon_u\sqrt{n}$ through out our computations [14] [15, Section 10.6] [17, 31]. This may also cause the orthogonality level to be underestimated. Usually the local round-off errors are small and $\epsilon_u\sqrt{n}$ can reasonably represent the actual orthogonality level after global re-orthogonalization, therefore we do not significantly underestimate the orthogonality levels.

During the global re-orthogonalization, we use Equation 22 to determine whether to stop. We need to evaluate $\rho = \|I - W_i\|$. One obvious upper bound on ρ is $i\epsilon_q$. However, this upper bound is too large to be used effectively, see Figure 3 plots (2) and (4). Next we will give a different way of estimating ρ. In the implementations of the ω-recurrence, the estimated loss of orthogonality of q_{i+1} is saved in memory which can be used to estimate ρ, $\bar{\rho} = \sum_{j=1}^{i} |\omega_{i+1,j}|$. If two consecutive orthogonalizations are performed on a vector, we can have yet another estimate of ρ. After the first Gram-Schmidt procedure in a global re-orthogonalization, two consecutive orthogonalizations have been performed. We can approximate ρ as $\underline{\rho} = \|Q_i^T r_i\| / \sqrt{\alpha_i^2 + \beta_{i-1}^2}$. This estimate of ρ is an under-estimate. In our implementations, the harmonic average of $\bar{\rho}$ and $\underline{\rho}$ is used, i.e.,

$$\rho = \sqrt{\bar{\rho}\underline{\rho}}. \tag{30}$$

Immediately after restarting, the criterion for re-orthogonalization, Equation 22, contains ρ. In this case, we cannot use the above formula to estimate $\bar{\rho}$ because $\omega_{k+2,j}$ are not known. We set $\bar{\rho}$ to the previous estimate of ρ in our implementations of the partial re-orthogonalization scheme.

Figure 3 compares the effect of ρ on the estimated orthogonality levels. We either compute ρ with Equation 30 or set it to its upper bound $\rho = i\epsilon_q$. The two plots on the left use the estimate ρ as described by Equation 30 and the two on the right use upper bound $i\epsilon_q$ instead. The data shown here are collected from all 15 test problems as in Figure 2. In fact, plot (1) in Figure 3 is exactly the same as plot (3) of Figure 2. The data points in the two right plots are clearly more concentrated toward the lower right corners, which indicates that using upper bound for ρ cause the orthogonality levels to be significantly overestimated. In this set of tests, changing the actual value of ρ directly influences the loss of orthogonality estimate of r_{k+1} but does not cause extra re-orthogonalization.

The value of ρ computed using Equation 30 may be less than the actual spectral radius of W_i, which can cause the global re-orthogonalization procedure to terminate prematurely,

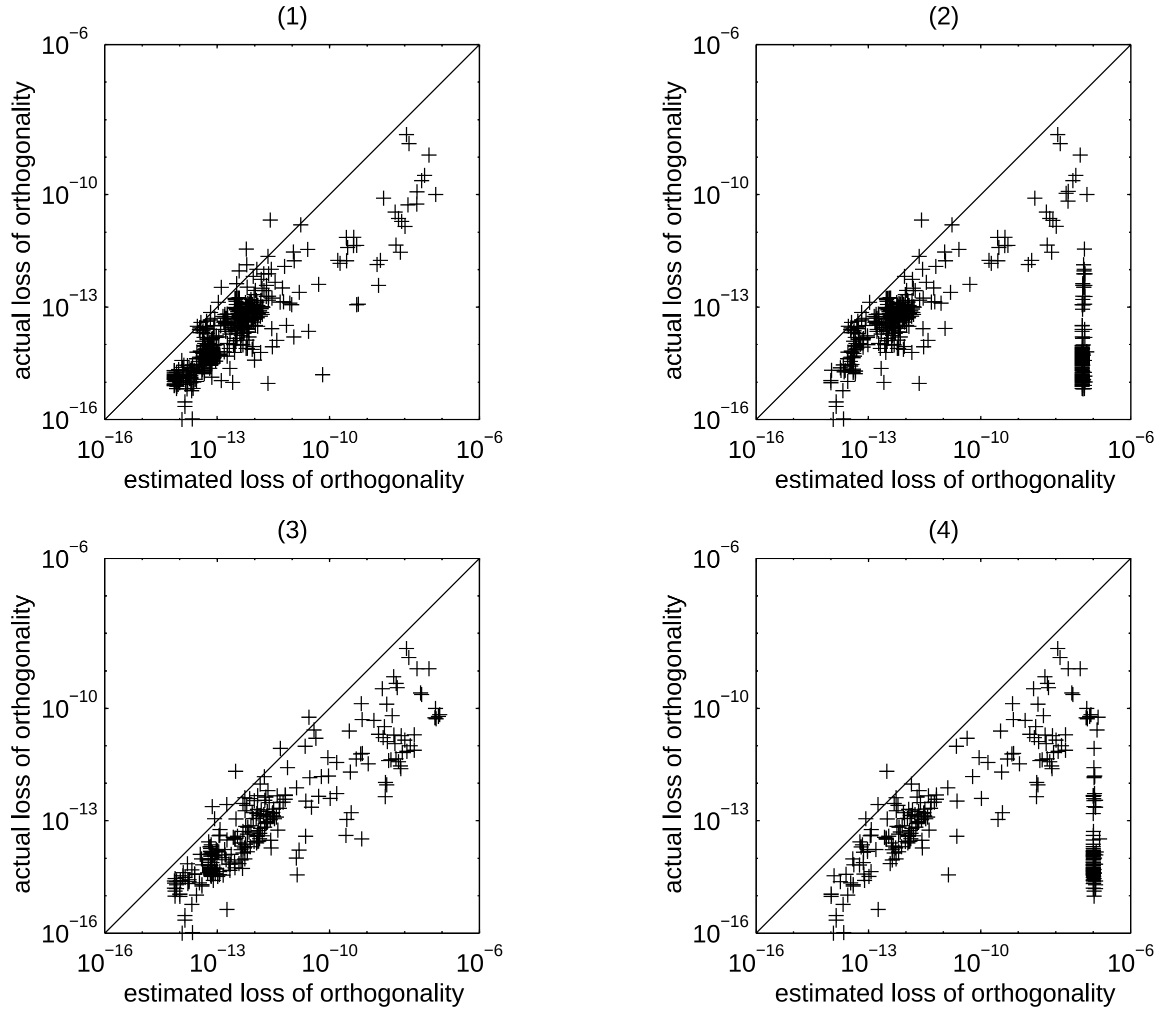

Figure 3: Estimated loss of orthogonality versus actual loss of orthogonality: (1) basis size 20, estimate ρ; (2) basis size 20, $\rho = i\epsilon_q$; (3) basis size 40, estimate ρ; (4) basis size 40, $\rho = i\epsilon_q$.

see Equation 22. Fortunately, in our tests the estimates were close enough so that no adverse effect was detected.

In summary, our implementations of the partial re-orthogonalization use signed ω-recurrence and add $\|d_i\|$ to $\omega_{i+1,i-1}$ and $\omega_{i+1,i}$ to account for the local round-off errors. When determining whether to stop the global re-orthogonalization, see Equation 22, we estimate the norm of $\|W - I\|$ using Equation 30.

7 Errors in solutions

In previous sections we have presented a restarted Lanczos algorithm and studied how orthogonality of the Lanczos vectors affects the accuracies of the coefficients α_i and β_i. This section will look at how loss of orthogonality affects the accuracy of the solutions. Given that the Lanczos eigenvalue method is iterative in nature, the accuracy of the estimated errors used in the convergence test is of particular interest. This section will also show the actual errors of three different implementations of the restarted Lanczos method on a handful of test problems.

At the end of Section 5, we described two schemes of computing accurate α_i and β_i using the restarted Lanczos algorithm, the full re-orthogonalization or the partial re-orthogonalization with correction. We have also shown that the partial re-orthogonalization (PRO) scheme which orthogonalizes the last residual vector r_m can maintain orthogonality that is close to ϵ_u, see Figure 1. Thus it should be able to generate accurate solutions as well. We have three schemes for implementing the restarted Lanczos algorithm. In the two cases where α_i and β_i are computed accurately, we understand that the Lanczos recurrence relation, Equation 1, is satisfied to $O(\epsilon_u\|A\|)$ accuracy. Next we will estimate the errors when only partial re-orthogonalization is used.

Before the first restart in the Lanczos method with partial re-orthogonalization, Equation 29 is satisfied. If we restart with $\hat{Q}_k = Q_m Y$, Equation 2 is not satisfied, i.e., $A\hat{Q}_k \neq \hat{Q}_k\hat{T}_k + \beta_m q_{m+1} e_m^T Y$. Since T_m is accurate when the Lanczos vectors are semi-orthogonal, $\hat{T}_k = Y^T T_m Y$ is the same as what is expected when the Lanczos vectors are computed to full accuracy. The error in $\hat{Q}_k$ can be measured as follows,

$$\begin{aligned} Z &\equiv (I - \hat{Q}_k^{*T}\hat{Q}_k^*)\hat{Q}_k = (I - N_m Y Y^T N_m^T) N_m L_m Y \\ &= N_m (I - YY^T) L_m^T Y = N_m (I - YY^T)\tilde{L}_m^T Y, \end{aligned}$$

where $\tilde{L}_m$ is the off-diagonal part of L_m, $\tilde{L}_m = L_m - I$. If Y is a basis of an invariant subspace of L_m^T, then this error is zero. Unless L_m is reducible to smaller triangular matrices, the only invariant subspaces of L_m^T is the space spanned by e_m. Normally L_m would not be reducible. Therefore the error in $\hat{Q}_k$ is roughly on the order of $\|\tilde{L}_m\| \sim m\epsilon_q$.

The error Z is reflected in the solutions as discrepancies between estimated and actual residual norms. Let λ, y be an eigenpair of T_m, i.e., $T_m y = \lambda y$. The residual of Ritz pair $\lambda, Q_m y$ is

$$\begin{aligned} r &= AQ_m y - \lambda Q_m y = AN_m L_m^T y - \lambda N_m L_m^T y, \\ &= N_m (T_m - \lambda I) L_m^T y + \beta_m q_{m+1} e_m^T L_m^T y, \\ &= N_m (T_m - \lambda I) L_m^T y + \beta_m q_{m+1} e_m^T y. \end{aligned} \tag{31}$$

If the Lanczos vectors are exactly orthogonal, the residual for the Ritz pair $(\lambda, Q_m y)$ is the second term of above equation, $r = \beta_m q_{m+1} e_m^T y$. The residual due to loss of orthogonality is $r_q = N_m(T_m - \lambda I)L_m^T y$, its norm is bounded as follows,

$$\|r_q\| \leq \|N_m\|\|T_m - \lambda I\|\|L_m^T\|\|y\| \leq (\lambda_{\max}(T_m) - \lambda_{\min}(T_m))m\epsilon_q.$$

One way we might use this error bound is to compute the maximum ϵ_q. If we want to compute the residual norm of the computed Ritz pairs to be less than τ, we would like to have $\|r\| = |\beta_m e_m^T y| > \|r_q\|$. The intention is to make sure the residual norm estimate is accurate until the Ritz pair is declared as converged. The above condition can be written as $\tau > \|r_q\|$ which leads to the following condition of tolerance on the orthogonality levels,

$$\epsilon_q < \frac{\tau}{m(\lambda_{\max}(A) - \lambda_{\min}(A))}.$$

In most of the examples, we attempt to compute the largest eigenvalues to the following accuracy $\|r\| \leq 10^{-8}|\lambda|$ using a basis size of 10 – 20. In these cases, we choose $\epsilon_q = 10^{-8}/m$.

We will test three implementations of the restarted Lanczos algorithm: using partial re-orthogonalization (PRO) without correction, using partial re-orthogonalization with correction, and maintaining full orthogonality. The three test matrices are shown in Table 1. The examples compute the five largest eigenvalues and their corresponding eigenvectors. The convergence tolerance is set to $\|r\| \leq 10^{-8}|\lambda|$. Aside from the three different restarted Lanczos methods, we also applied PARPACK to solve the same test problems. The PARPACK program is based on ARPACK version 2.1 dated 3/19/97. The above convergence test is also used in PARPACK for symmetric eigenvalue problems[2]. The eigenvalues are computed and their corresponding errors are shown in Tables 2 – 5. The tables list the eigenvalues computed (λ_i), the differences between the computed eigenvalues and the Rayleigh quotients of the computed eigenvectors ($\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$), the estimated residual norms ($|\beta_m e_m^T y|$), and the actual residual norms ($\|Ax_i - \lambda_i x_i\|$). The tables also show the number of matrix-vector multiplications (MATVEC), the number of restarts, and the number of global re-orthogonalizations (re-orthog) used by the four different methods.

The data in Tables 2 – 5 are designed to show errors in the computed solutions. Our goal is to measure how much error is introduced by loss of orthogonality. Ideally, we would like to compare the quantities computed by the Lanczos programs against those computed in exact arithmetic. Since we cannot produce the exact solution easily, we resolve to use the following two quantities to measure the errors due to loss of orthogonality. As shown in Equation 31, when there is loss of orthogonality, the actual residuals of the computed Ritz pairs would be different than the predictions that were based on exact arithmetic. Thus one measure indication of errors due to loss of orthogonality is the difference between the predicted residual norms and the actual residual norms. Loss of orthogonality affects the accuracies of α_i and β_i. In exact arithmetic, α_i and β_i would be accurate and the Ritz values computed as the eigenvalues of T_m would be equal to the Rayleigh quotients of their corresponding Ritz vectors. In Section 5, we have shown that the coefficients α_i and β_i can

[2]Additional information on PARPACK and ARPACK can be found at `http://www.caam.rice.edu/-software/ARPACK`.

thick-restart Lanczos with PRO
MATVEC: 90, restarts: 8, re-orthog: 16

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	2648056755.2108874	1.43E-06	4.92E-01	4.92E-01
2	2647979344.2127838	-1.91E-06	7.48E-01	7.48E-01
3	2634048614.9912028	4.29E-06	1.41E+00	1.41E+00
4	2633679289.2081308	3.81E-06	2.32E+00	2.32E+00
5	2606151408.4051986	9.54E-07	7.22E+00	7.22E+00

thick-restart Lanczos with PRO and correction
MATVEC: 90, restarts: 8, re-orthog: 16

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	2648056755.2108870	1.91E-06	4.92E-01	4.92E-01
2	2647979344.2127848	-1.43E-06	7.48E-01	7.48E-01
3	2634048614.9912038	1.43E-06	1.41E+00	1.41E+00
4	2633679289.2081327	3.34E-06	2.32E+00	2.32E+00
5	2606151408.4052033	-4.29E-06	7.22E+00	7.22E+00

thick-restart Lanczos with full re-orthogonalization
MATVEC: 90, restarts: 8, re-orthog: 88

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	2648056755.2108898	-9.54E-07	4.92E-01	4.92E-01
2	2647979344.2127857	-1.91E-06	7.48E-01	7.48E-01
3	2634048614.9912052	4.77E-07	1.41E+00	1.41E+00
4	2633679289.2081337	0.00E+00	2.32E+00	2.32E+00
5	2606151408.4051962	1.91E-06	7.22E+00	7.22E+00

PARPACK(20)
MATVEC: 157, restarts: 11, re-orthog: 156

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	2648056755.2108912	-3.34E-06	7.11E-08	4.56E-06
2	2647979344.2127795	3.34E-06	8.02E-09	6.88E-06
3	2634048614.9912057	0.00E+00	5.39E-07	3.21E-06
4	2633679289.2081347	-9.54E-07	9.91E-07	3.35E-06
5	2606151408.4052033	-2.86E-06	8.46E-01	8.46E-01

Table 2: The largest eigenvalues of NASASRB and their errors (basis size 20).

thick-restart Lanczos with PRO
MATVEC: 185, restarts: 46, re-orthog: 92

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\|\beta_m e_m^T y\|$	$\|\|Ax_i - \lambda_i x_i\|\|$
1	2648056755.2108932	-5.25E-06	7.69E-05	7.57E-05
2	2647979344.2127895	-5.72E-06	1.19E-04	1.28E-04
3	2634048614.9912024	3.34E-06	3.68E-03	3.68E-03
4	2633679289.2081351	-4.77E-07	6.56E-03	6.57E-03
5	2606151408.4051919	6.68E-06	1.06E+01	1.06E+01

thick-restart Lanczos with PRO and correction
MATVEC: 185, restarts: 46, re-orthog: 92

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\|\beta_m e_m^T y\|$	$\|\|Ax_i - \lambda_i x_i\|\|$
1	2648056755.2108908	-2.86E-06	7.69E-05	7.87E-05
2	2647979344.2127919	-8.11E-06	1.19E-04	1.19E-04
3	2634048614.9912133	-9.06E-06	3.68E-03	3.68E-03
4	2633679289.2081370	-1.43E-06	6.56E-03	6.56E-03
5	2606151408.4051905	8.58E-06	1.06E+01	1.06E+01

thick-restart Lanczos with full re-orthogonalization
MATVEC: 185, restarts: 46, re-orthog: 182

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\|\beta_m e_m^T y\|$	$\|\|Ax_i - \lambda_i x_i\|\|$
1	2648056755.2108812	8.58E-06	7.69E-05	7.67E-05
2	2647979344.2127852	-4.29E-06	1.19E-04	1.19E-04
3	2634048614.9911947	1.00E-05	3.68E-03	3.68E-03
4	2633679289.2081351	1.43E-06	6.56E-03	6.56E-03
5	2606151408.4051809	1.72E-05	1.06E+01	1.06E+01

PARPACK(10)
MATVEC: 184, restarts: 39, re-orthog: 183

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\|\beta_m e_m^T y\|$	$\|\|Ax_i - \lambda_i x_i\|\|$
1	2648056755.2108836	4.29E-06	5.37E-06	1.18E-05
2	2647979344.2127848	-1.91E-06	6.08E-07	3.33E-06
3	2634048614.9912062	-1.91E-06	7.10E-05	7.10E-05
4	2633679289.2081342	4.77E-07	1.32E-04	1.32E-04
5	2606151408.4051933	5.72E-06	1.63E+01	1.63E+01

Table 3: The largest eigenvalues of NASASRB and their errors (basis size 10).

thick-restart Lanczos with PRO

MATVEC: 2269, restarts: 465, re-orthog: 930

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	8798.4363691285525	2.00E-11	9.15E-14	5.03E-11
2	8796.7159980617544	-4.18E-11	4.57E-11	7.73E-11
3	8794.1437886041895	3.09E-11	1.53E-06	1.53E-06
4	8793.9361552127648	2.18E-11	4.28E-06	4.28E-06
5	8792.3179110199835	-9.82E-11	5.66E-05	5.66E-05

thick-restart Lanczos with PRO and correction

MATVEC: 2269, restarts: 465, re-orthog: 930

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	8798.4363691285835	-2.00E-11	9.10E-14	4.76E-11
2	8796.7159980616507	5.46E-11	4.82E-11	9.21E-11
3	8794.1437886042440	-3.64E-12	1.63E-06	1.63E-06
4	8793.9361552127575	2.36E-11	4.54E-06	4.54E-06
5	8792.3179110199835	-1.22E-10	6.05E-05	6.05E-05

thick-restart Lanczos with full re-orthogonalization

MATVEC: 2269, restarts: 465, re-orthog: 2263

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	8798.4363691285998	-3.82E-11	1.01E-13	7.72E-11
2	8796.7159980618926	-1.71E-10	4.67E-11	1.99E-10
3	8794.1437886042459	-5.46E-12	1.61E-06	1.61E-06
4	8793.9361552127393	4.18E-11	4.51E-06	4.51E-06
5	8792.3179110199180	-4.00E-11	6.13E-05	6.13E-05

PARPACK(10)

MATVEC: 4459, restarts: 1310, re-orthog: 4458

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	8798.4363691285907	-2.73E-11	4.20E-19	6.45E-11
2	8796.7159980616543	4.18E-11	1.12E-15	8.83E-11
3	8794.1437886042349	0.00E+00	4.18E-09	4.17E-09
4	8793.9361552128848	-1.13E-10	7.75E-10	7.85E-10
5	8792.3179110193978	1.87E-10	8.65E-05	8.65E-05

Table 4: The five largest eigenvalues of S3DKT3M2 and their errors (basis size 10).

thick-restart Lanczos with PRO
MATVEC: 5119, restarts: 1516, re-orthog: 3029

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	4601.6534362483071	-1.02E-10	8.76E-27	1.11E-10
2	4600.8516475995593	-4.64E-11	6.47E-19	1.91E-10
3	4599.5157181857421	-5.38E-10	7.79E-17	5.72E-10
4	4598.2818892741634	1.26E-10	3.64E-12	1.69E-10
5	4597.6462278967929	-2.30E-10	4.50E-05	4.50E-05

thick-restart Lanczos with PRO and correction
MATVEC: 5113, restarts: 1514, re-orthog: 3025

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	4601.6534362481598	4.64E-11	1.77E-26	4.77E-11
2	4600.8516475997794	-2.64E-10	6.79E-19	3.27E-10
3	4599.5157181855975	-3.97E-10	8.09E-17	4.37E-10
4	4598.2818892741143	1.76E-10	3.74E-12	2.05E-10
5	4597.6462278967292	-1.83E-10	4.59E-05	4.59E-05

thick-restart Lanczos with full re-orthogonalization
MATVEC: 5119, restarts: 1516, re-orthog: 5105

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	4601.6534362481489	5.64E-11	1.64E-26	7.08E-11
2	4600.8516475997076	-1.94E-10	6.47E-19	2.67E-10
3	4599.5157181855830	-3.78E-10	7.79E-17	4.17E-10
4	4598.2818892742534	4.09E-11	3.64E-12	7.63E-11
5	4597.6462278967347	-1.77E-10	4.50E-05	4.50E-05

PARPACK(10)
MATVEC: 3646, restarts: 1516, re-orthog: 3645

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	4601.6534362481807	2.27E-11	1.20E-17	3.22E-11
2	4600.8516475994866	2.09E-11	5.66E-15	3.87E-11
3	4599.5157181852219	-2.27E-11	6.18E-11	6.79E-11
4	4598.2818892742935	4.55E-12	1.91E-06	1.91E-06
5	4597.6462278966010	3.46E-11	4.54E-05	4.54E-05

Table 5: The five largest eigenvalues of S3DKQ4M2 and their errors (basis size 10).

be computed in a number of cases. We would like to verify the claim by measuring the difference between the Ritz values and the Rayleigh quotients. The reference quantities are computed using floating-point arithmetic as well. We consider the differences as zero if they are less than the errors in the reference quantities. The error in multiplying A with a unit vector is $\epsilon_u\|A\|$. The upper bound of error in a residual norm explicitly computed using $\|r\| = \|Ax - \lambda x\|$ is no less than $\epsilon_u\|A\|$. Therefore, we consider them zero if the differences between the predicted residual norms and the actual residual norms are smaller than $\epsilon_u\|A\|$. Similarly, the error in computing a Rayleigh quotient $x_i^T A x_i / x_i^T x_i$ is roughly $\epsilon_u\|A\|$ and we consider them zero if the differences between the Ritz values and the Rayleigh quotients $(\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i)$ are smaller than $\epsilon_u\|A\|$.

Let's first compare the predicted residual norms and the actual residual norms. Table 2 shows results from solving the NASASRB test problem with basis size of 20. This example only needs a handful of restarts. We notice a difference between predicted residual norms $(|\beta_m e_m^T y|)$ and the actual computed residual norms for PARPACK. The differences are about 10^{-6}, which are close to $\epsilon_u\|A\|$. This confirms that PARPACK maintains full orthogonality among the basis vectors. The data shown in Table 3 are the results of solving the NASASRB test problem with basis size 10, which requires about 40 restarts. In this case, all four methods solved the eigenvalue problem to about the same accuracy. The discrepancies between predicted residual norms and computed residual norms are on the order of 10^{-5} for thick-restart Lanczos with PRO and PARPACK, which is 14 orders of magnitude smaller than the norm of the matrix. The two other methods have a maximum discrepancy of about 10^{-6}. The discrepancies for the S3DKT3M2 test problem, see Table 4, are mostly on the order of 10^{-11}, which is again 14 orders of magnitude smaller than the matrix norm. The discrepancies for the restarted Lanczos method with full orthogonality are slightly larger than the others, as large as 10^{-10}. The discrepancies between the estimated residual norms and the actual residual norms for S3DKQ4M2, see Table 5, are on the order of 10^{-10} for the three restarted Lanczos variants and 10^{-11} for PARPACK. Overall, the discrepancies are about 2 – 3 orders of magnitude larger than $\epsilon_u\|A\|$ no matter whether the eigenvalue routine maintains full orthogonality or semi-orthogonality.

The second indicator of error is the difference between the computed Ritz value and the Rayleigh quotient, $(\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i)$. In Table 2, these differences are on the order of 10^{-6} for most eigenvalues computed by all four methods. The same is true in Table 3, except for two entries for the thick-restart Lanczos method with full re-orthogonalization where the differences are about 10^{-5}. In S3DKT3M3 example, see Table 4, the differences between Ritz values and Rayleigh quotients are mostly on the order of 10^{-11} with a few exceptions of 10^{-10}. In this case, the thick-restart Lanczos method with partial re-orthogonalization and correction (TRLan PRO-c) has slightly smaller differences than others. In S3DKQ4M2 example, see Table 5, the differences are predominantly on the order of 10^{-10}, except PARPACK has slightly smaller differences between the Ritz values and the Rayleigh quotients.

Overall, there is no indication that the solutions computed by the thick-restart Lanczos method with partial re-orthogonalization are inferior to the other two restarted Lanczos schemes. The errors in the computed solutions are about the same sizes for PARPACK as well.

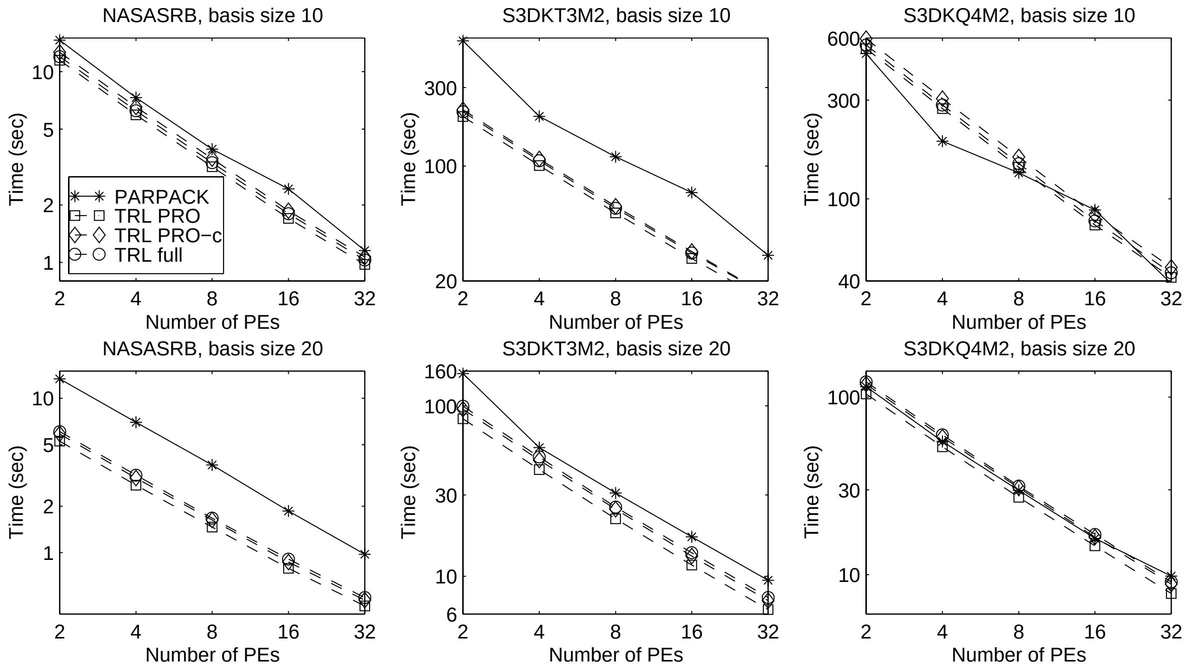

Figure 4: Time spent to solve the three test problems on Cray T3E900.

Tables 2 – 5 also show the number of matrix-vector multiplications used by the four methods tested and Figure 4 shows the timing results on the test problems. The timing results were collected on the T3E 900 at National Energy Research Scientific Computing Center (NERSC)[3]. The three restarted Lanczos methods are implemented with the same restarting scheme which attempts to mimic the restarting scheme of PARPACK. The numbers of matrix-vector multiplications used by the three restarted Lanczos methods are the same most of the time, however they are different from that of PARPACK. The differences in the restarting schemes are the main causes for these dramatic differences in the numbers of matrix-vector multiplications. When similar numbers of matrix-vector multiplications are used, the thick-restart Lanczos method with partial re-orthogonalization uses less time than the other methods. PARPACK implements the Arnoldi algorithm described in Algorithm 2 which has a more expensive orthogonalization step than that of Algorithm 3 and it performs global re-orthogonalization at almost every step. Overall, it takes more time than the restarted Lanczos implementations tested when a similar number of matrix-vector multiplications are used. In most cases, the time spent by the Lanczos with partial re-orthogonalization and correction is about the same as the one with full re-orthogonalization.

In many practical applications, the user often wants to know the smallest eigenvalue of a matrix. The smallest eigenvalues of the three test matrices shown in Table 1 need considerably more matrix-vector multiplications than the largest ones. For example, without restart, only 75 matrix-vector multiplications are needed to compute the five largest eigenvalues of NASASRB, but 11,600 matrix-vector multiplications are needed to compute the smallest eigenvalue. Table 6 shows the results of using the restarted methods to find the smallest

[3]General information located at `http://www.nersc.gov/`.

thick-restart Lanczos with PRO
MATVEC: 50471, restarts: 51, re-orthog: 14713
time: 9798 seconds on 8 PEs

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	4.7434129039704969	8.27E-07	2.32E-15	5.47E-04
2	5.0304972052890955	4.37E-07	2.00E-15	4.20E-04
3	57.269882264578108	8.91E-07	2.47E-07	6.81E-04
4	59.325145999622919	-1.18E-07	8.06E-10	4.76E-04
5	114.48805504571098	3.42E-08	1.37E-04	1.17E-03

thick-restart Lanczos with full re-orthogonalization
MATVEC: 50467, restarts: 51, re-orthog: 50423
time: 8546 seconds on 8 PEs

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	4.7434139714076551	-2.40E-07	3.87E-15	1.83E-06
2	5.0304978087926671	-1.66E-07	3.34E-15	1.85E-06
3	57.269882851333911	3.04E-07	3.98E-07	1.09E-06
4	59.325143838635135	2.04E-06	1.30E-09	3.02E-06
5	114.48805524461864	-1.65E-07	2.27E-04	2.27E-04

PARPACK
MATVEC: 46761, restarts: 61, re-orthog: 46760
time: 8547 seconds on 16 PEs

i	λ_i	$\frac{x_i^T A x_i}{x_i^T x_i} - \lambda_i$	$\lvert\beta_m e_m^T y\rvert$	$\lVert Ax_i - \lambda_i x_i\rVert$
1	388.43347600844902	-5.14E-07	6.99E+03	6.99E+03
2	1961.8238927546274	-1.94E-07	1.69E+04	1.69E+04
3	9890.9668426024327	-7.96E-09	5.92E+04	5.92E+04
4	18693.850673534958	4.62E-08	8.49E+04	8.49E+04
5	34854.798171100148	-2.88E-07	5.84E+04	5.84E+04

Table 6: The five smallest eigenvalues of NASASRB and their errors (basis size 1,000).

eigenvalues of NASASRB. A basis size of 1,000 is used in this test because smaller basis sizes are ineffective in finding the desired solutions. Using basis size of 100, the smallest Ritz values computed by the four restarted methods are all larger than ten. Using basis size of 1,000, the restarted Lanczos method with partial re-orthogonalization and the restarted Lanczos method with full re-orthogonalization declare the smallest four eigenvalues converged after about 50,000 matrix-vector multiplications. On the Cray T3E at NERSC, the longest jobs are limited to four hours long and different numbers of processors are used to allow our tests to finish. The time and the number of processors used are shown in Table tb:srb1000. The PARPACK routine is stopped after 60 restarts and its results are used to give a reference to the error in Ritz values. A more effective scheme of computing the smallest eigenvalue of NASASRB is a shift-and-invert Lanczos algorithm. However, the main reason to study this example is to show that the restarted Lanczos method can find solutions to *difficult* eigenvalue problems. Even in this difficult case, the restarted Lanczos method with partial re-orthogonalization generates accurate eigenvalues.

In Table 6, the differences between Ritz values λ_i and the Rayleigh quotients of their corresponding Ritz vectors are on the order of $\epsilon_u \|A\|$. In fact the differences are roughly 16 orders of magnitudes smaller than the matrix norm which are smaller than the same differences when computing the largest eigenvalues, see Tables 2 and 3. The differences for the Lanczos method with partial re-orthogonalization are about the same size as the difference for PARPACK. This again shows that the Ritz values computed by the restarted Lanczos methods are accurate.

The discrepancies between the estimated residual norms and the computed residual norms are as large as 10^{-3} for the restarted Lanczos method with partial re-orthogonalization, which is clearly larger than 10^{-6}, the discrepancies for the Lanczos method with full re-orthogonalization. The value of 10^{-6} is close to $\epsilon_u \|A\|$ for this test matrix. The discrepancies of 10^{-3} are considerably larger than $\epsilon_u \|A\|$, which indicates that the Ritz vectors computed by the Lanczos method with partial re-orthogonalization are not as accurate as the Ritz vectors computed by the Lanczos method with full re-orthogonalization. In previous examples, the difference between the Lanczos method with partial re-orthogonalization and the Lanczos method with full re-orthogonalization were too small to be noticed. By comparing the discrepancies from the two variants of the restarted Lanczos method, we see that the orthogonality level of bases generated with partial re-orthogonalization is roughly three order of magnitude worse than the orthogonality level of bases generated with full re-orthogonalization. The bases generated with partial re-orthogonalization are still semi-orthogonal, but the orthogonality level is worse than in previous examples.

The number of re-orthogonalizations performed by the Lanczos method with partial re-orthogonalization is 14,713 in this example. Subtracting two global re-orthogonalizations per restart due to the structure of the algorithm, there are 14,611 re-orthogonalization due to severe loss of orthogonality. When seeking the largest eigenvalues of NASASRB, there is almost no re-orthogonalization due to severe loss of orthogonality. The large number of re-orthogonalizations is detrimental to the overall performance because the Gram-Schmidt procedure has to be invoked frequently to restore the orthogonality. Overall, the total time used by the Lanczos method with full re-orthogonalization is 8,546 seconds, which is almost 15% less than the time used by the Lanczos method with partial re-orthogonalization, 9,798 seconds. Since the number of matrix-vector multiplications are very close and the number of restarts are the same, the difference in re-orthogonalization should be the main cause of the time difference. Another side effect of frequent occurrences of severe loss of orthogonality is the above-mentioned worsening orthogonality level.

From the above tests we also notice that the maximum basis size has a strong influence on the time it takes to find the desired solutions. For example, when the maximum basis size is ten the restarted versions of Lanczos method used more than 5,000 matrix-vector multiplications and almost 600 seconds on two processors of the T3E to find the five largest eigenvalues of S3DKQ4M2. When the maximum basis size increases to 20, the number of matrix-vector multiplications decreases to just above 800 and execution time reduces to around 100 seconds on two processors. Had we not restarted, the number of Lanczos steps required would be 617. The 617 Lanczos vectors can fit on four processors of the T3E used and it takes 160 seconds on two processors. In this case, a basis size of 20 seems to be effective, but a basis size of 10 causes an excessive number of matrix-vector multiplications to be used.

8 Summary

The main goal of this paper is to study how to implement an effective restarted Lanczos method for real symmetric eigenvalue problems. The emphasis is on how to maintain semi-orthogonality and to reduce the amount of arithmetic operations. The method generated from this study is the thick-restart Lanczos method with partial re-orthogonalization. It restarts with Ritz vectors and computes the last residual vector of the Lanczos iterations accurately before restarting. This restarted Lanczos algorithm is as inexpensive as the original Lanczos algorithm and it computes accurate eigenvalues without maintaining full orthogonality among the Lanczos vectors. In most examples shown, the quality of the solutions found by the restarted Lanczos method with partial re-orthogonalization is the same as that of the Lanczos method with full re-orthogonalization.

We have noticed a case where the partial re-orthogonalization scheme did not produce solutions with the same accuracies as the full re-orthogonalization scheme. In this case, the Ritz values computed are still accurate, but the differences between estimated residual norms and actual residual norms are larger for the Lanczos method with partial re-orthogonalization. In this case, we try to compute the smallest eigenvalues of NASASRB. Using the Lanczos method without restart, roughly 2,000 eigenvalues on the high-end of the spectrum converged before the smallest one reached convergence. Each time an eigenvalue reaches convergence, a Lanczos vector may lose orthogonality. In this case, the restarted Lanczos method will encounter severe loss of orthogonality frequently and have to perform global re-orthogonalization frequently. Overall, due to the large numbers of global re-orthogonalizations in the partial re-orthogonalization scheme, it is more efficient to simply use the full re-orthogonalization for this particular problem.

From our experiments, we see that the differences in execution time due to various re-orthogonalization schemes are consistent in most cases. However, the differences due to the re-orthogonalization schemes are often not as important as other factors, for example, the maximum basis size and the restarting scheme. Our experiments indicate that differences in the maximum basis sizes can significantly alter the time required to find the solution for a given eigenvalue problem and a small change in number of Ritz vectors saved at restart can also change the total execution time significantly. These issues are not unique to restarted Lanczos methods. They are common to all restarted methods [2, 5, 10, 36, 37, 43]. We only showed the results of using one very simple restarting scheme. Since the Lanczos method can provide estimated residual norms cheaply, it may be used to enhance the restarting schemes and speed-up the overall solution process.

In the process of developing an efficient restarted Lanczos method, we also studied how the orthogonality is related to the accuracy of projected matrix and how repeating the Gram-Schmidt procedure restores the orthogonality. The criteria for terminating the re-orthogonalization process can also be used in other algorithms where orthogonality is required.

9 Acknowledgments

The authors want to express our sincere gratitude to Professor Beresford Parlett for bringing to our attention the extended local re-orthogonalization concept which is adopted in Section 5. The authors would also like to thank Dr. Xiaoye Li and Dr. Osni Marques for reviewing the early drafts of this paper.

This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Energy Research of the U.S. Department of Energy.

References

[1] W. E. Arnoldi. The principle of minimized iteration in the solution of the matrix eigenvalue problem. *Quarterly of Applied Mathematics*, 9:17–29, 1951.

[2] J. Baglama, D. Calvetti, and L. Reichel. Iterative methods for the computation of a few eigenvalues of a large symmetric matrix. *BIT*, 36:400–421, 1996.

[3] J. D. Brown, M. T. Chu, D. C. Ellison, and R. J. Plemmons, editors. *Proceedings of the Cornelius Lanczos International Centenary conference*, Philadelphia, PA, 1993. SIAM.

[4] A. Chapman and Y. Saad. Deflated and augmented Krylov subspace techniques. Technical Report UMSI 95/181, Minnesota Supercomputing Institute, Univ. of Minnesota, 1995.

[5] M. Crouzeix, B. Philippe, and M. Sadkane. The Davidson method. *SIAM J. Sci. Comput.*, 15:62–76, 1994.

[6] J. Cullum and R. A. Willoughby. *Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Theory*, volume 3 of *Progress in Scientific Computing*. Birkhauser, Boston, 1985.

[7] J. Cullum and R. A. Willoughby. *Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Programs*, volume 4 of *Progress in Scientific Computing.* Birkhauser, Boston, 1985.

[8] J. W. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization. *Math. Comp.*, 30(136):772–795, October 1976.

[9] Ernest R. Davidson. The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices. *J. Comput. Phys.*, 17:87–94, 1975.

[10] Ernest R. Davidson. Super-matrix methods. *Computer Physics Communications*, 53:49–60, 1989.

[11] I. S. Duff, R. G. Grimes, and J. G. Lewis. Sparse matrix test problems. *ACM Trans. Math. Soft.*, pages 1–14, 1989.

[12] G. H. Golub and D. P. O'Leary. Some history of the Conjugate Gradient and Lanczos algorithms: 1948-1976. *SIAM Review*, 31:50–102, 1989.

[13] G. H. Golub and C. F. van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, MD 21211, third edition, 1996.

[14] R. G. Grimes, J. G. Lewis, and H. D. Simon. A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. *SIAM J. Matrix Anal. Appl.*, 15(1):228–272, 1994.

[15] Thomas J. R. Hughes. *The Finite Element Method.* Prentice-Hall, 1987.

[16] Richard B. Lehoucq. *Analysis and implementation of an implicitly restarted Arnoldi iteration.* PhD thesis, Rice University, 1995.

[17] Osni A. Marques. BLZPACK: Description and user's guide. Technical Report TR/Pa/95/30, CERFACS, 1995. Lastest source code available at URL `http://www.nersc.gov/research/SCG/Osni/marques_software.html`.

[18] R. B. Morgan and D. S. Scott. Generalizations of Davidson's method for computing eigenvalues of sparse symmetric matrices. *SIAM J. Sci. Statist. Comput.*, 7:817–825, 1986.

[19] Ronald B. Morgan. On restarting the Arnoldi method for large nonsymmetric eigenvalue problems. *Mathematics of Computation*, 65(215):1213–1230, July 1996.

[20] C. W. Murray, S. C. Racine, and E. R. Davidson. Improved algorithms for the lowest eigenvalues and associated eigenvectors of large matrices. *J. Comput. Phys.*, 103(2):382–389, 1992.

[21] C. C. Paige. Computational variants of the Lanczos method for eigenproblem. *Journal of the Institute for Mathematics and its Applications*, 10:373–381, 1972.

[22] C. C. Paige. Accuracy and effectiveness of the Lanczos algorithm for symmetric matrix. *Lin. Alg. Appl.*, 34:235–258, 1976.

[23] B. N. Parlett and D. Scott. The Lanczos algorithm with selective orthogonalization. *Math. Comp.*, 33:217–2388, 1979.

[24] Beresford N. Parlett. *The rewards for maintaining semi-orthogonality among Lanczos vectors*, 1991.

[25] Beresford N. Parlett. Do we fully understand the symmetric Lanczos algorithm yet? In Brown et al. [3], pages 93–107.

[26] Beresford N. Parlett. *The symmetric eigenvalue problem.* SIAM, Philadelphia, PA, 1998.

[27] Y. Saad. Analysis of augmented Krylov subspace techniques. Technical Report UMSI 95/175, Minnesota Supercomputing Institute, Univ. of Minnesota, 1995.

[28] Yousef Saad. *Numerical Methods for Large Eigenvalue Problems.* Manchester University Press, 1993.

[29] Yousef Saad. *Iterative Methods for Sparse Linear Systems.* PWS publishing, Boston, MA, 1996.

[30] Miloud Sadkane. A block Arnoldi-Chebyshev method for computing the leading eigenpairs of large sparse unsymmetric matrices. *Numer. Math.*, 64(2):181–193, 1993.

[31] Horst D. Simon. *The Lanczos algorithm for solving symmetric linear systems.* PhD thesis, University of California, Berkeley, 1982.

[32] Horst D. Simon. Analysis of the symmetric Lanczos algorithm with reorthogonalization methods. *Lin. Alg. Appl.*, 61:101–131, 1984.

[33] Horst D. Simon. The Lanczos algorithm with partial reorthogonalization. *Math. Comp.*, 42:115–136, 1984.

[34] G. L. G. Sleijpen and H. A. van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 17(2), 1996.

[35] D. Sorensen, R. Lehoucq, P. Vu, and C. Yang. *ARPACK: an implementation of the Implicitly Restarted Arnoldi iteration that computes some of the eigenvalues and eigenvectors of a large sparse matrix*, 1995.

[36] D. S. Sorensen. Implicit application of polynomial filters in a K-step Arnoldi method. *SIAM J. Matrix Anal. Appl.*, 13(1):357–385, 1992.

[37] A. Stathopoulos, Y. Saad, and K. Wu. Dynamic thick restarting of the Davidson and the implicitly restarted Arnoldi methods. Technical Report UMSI 96/123, University of Minnesota Supercomputer Institute, 1996.

[38] A. Stathopoulos, Y. Saad, and K. Wu. Thick restarting of the Davidson method: an extension to implicit restarting. In T. Manteuffel, S. McCormick, L. Adams, S. Ashby, H. Elman, R. Freund, A. Greenbaum, S. Parter, P. Saylor, N. Trefethen, H. van der Vorst, H. Walker, and O. Wildlund, editors, *Proceedings of Copper Mountain Conference on Iterative Methods*, Copper Mountain, Colorado, 1996.

[39] Andreas Stathopoulos and Charlotte Fischer. A Davidson program for finding a few selected extreme eigenpairs of a large, sparse, real, symmetric matrix. *Computer Physics Communications*, 79(2):268–290, 1994.

[40] J. H. van Lenthe and P. Pulay. A space-saving modification of Davidson's eigenvector algorithm. *J. Comput. Chem.*, 11:1164–1168, 1990.

[41] L.-W. Wang and A. Zunger. Large scale electronic structure calculations using the Lanczos method. *Computational Materials Science*, 2:326–340, 1994.

[42] F. Webster and G.-C. Lo. Projective block Lanczos algorithm for dense, Hermitian eigensystems. *J. Comput. Phys.*, pages 146–161, 1996.

[43] Kesheng Wu. *Preconditioned Techniques for Large Eigenvalue Problems.* PhD thesis, University of Minnesota, 1997. An updated version also appears as Technical Report TR97-038 at the Computer Science Department.