SIAM J. MATRIX ANAL. APPL.
Vol. 22, No. 2, pp. 602–616


# THICK-RESTART LANCZOS METHOD FOR LARGE SYMMETRIC EIGENVALUE PROBLEMS*

KESHENG WU† AND HORST SIMON†

**Abstract.** In this paper, we propose a restarted variant of the Lanczos method for symmetric eigenvalue problems named the thick-restart Lanczos method. This new variant is able to retain an arbitrary number of Ritz vectors from the previous iterations with a minimal restarting cost. Since it restarts with Ritz vectors, it is simpler than similar methods, such as the implicitly restarted Lanczos method. We carefully examine the effects of the floating-point round-off errors on stability of the new algorithm and present an implementation of the partial reorthogonalization scheme that guarantees accurate Ritz values with a minimal amount of reorthogonalization. We also show a number of heuristics on deciding which Ritz pairs to save during restart in order to maximize the overall performance of the thick-restart Lanczos method.

**Key words.** thick-restart, Lanczos eigenvalue method, partial reorthogonalization

**AMS subject classifications.** 65F15, 65F25

**PII.** S0895479898334605

## 1. Introduction.

Given an $n \times n$ matrix $A$, its eigenvalue $\lambda$ and the corresponding eigenvector $x$ are defined by $Ax = \lambda x$. If the matrix size is large and only a smaller number of eigenvalues are wanted, a projection-based method is usually used [16, 18]. These types of methods usually build orthogonal bases first and then perform the Rayleigh–Ritz projection to extract approximate solutions. There are some alternative projection methods, such as the harmonic Ritz value method [14], but the most significant difference among the projection eigenvalue methods is how they generate their bases. For this reason, most of the eigenvalue methods are named after their basis generation procedures.

When the matrix is symmetric, the Lanczos method (see Algorithm 1, [11, 16, 20]) is the most commonly used method. Other frequently used methods include the Arnoldi method (see Algorithm 2, [1, 20, 24]) and the Davidson method [6, 7, 23]. The Arnoldi method and the Lanczos method are mathematically equivalent on symmetric eigenvalue problems. The Lanczos method is used more frequently because it takes advantage of the fact that most coefficients $h_{j,i}$ computed in step (c) of Algorithm 2 are zero ($h_{j,i} = 0, j = 1, \ldots, i-2$), and the matrix formed from $h_{j,i}$ is symmetric ($\beta_{i-1} \equiv h_{i-1,i} = h_{i,i-1}$, $\alpha_i \equiv h_{i,i}$). This allows the Lanczos method to avoid a significant amount of arithmetic operations. The Davidson method offers more functionality, such as preconditioning, flexible restarting options, etc., but it also uses more arithmetic operations per iteration and more computer memory.

There are many different variations of the Lanczos method depending on factors such as restarting, reorthogonalization, storage schemes for the Lanczos vectors $q_i$, and

*Received by the editors February 26, 1998; accepted for publication (in revised form) by A. Greenbaum April 30, 2000; published electronically September 15, 2000. This work was supported by the Director, Office of Science, Office of Laboratory Policy and Infrastructure Management, of the U.S. Department of Energy under contract DE-AC03-76SF00098. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy. 
http://www.siam.org/journals/simax/22-2/33460.html

†Lawrence Berkeley National Laboratory/NERSC, Berkeley, CA 94720 (kwu@lbl.gov, hdsimon@lbl.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) $h_{i,i+1} = \|r_i\|$.

so on. This paper will discuss a restarted Lanczos method, and in the remainder of this section we will discuss the motivations for restarting and briefly review the commonly used restarting schemes. We plan to store the Lanczos vectors in a computer's main memory. This significantly reduces the scope of the discussion. The issues related to reorthogonalization and details of how to restart are discussed later in the paper.

In both Algorithms 1 and 2 a new vector $q_i$ is generated in each iteration. These vectors are needed when performing reorthogonalization and computing the Ritz vectors. Often a large number of iterations are needed to compute an eigenvalue. On most machines, there is not enough computer memory to store the Lanczos vectors. In addition, the number of arithmetic operations associated with the reorthogonalization and the Rayleigh–Ritz projection grows as the number of vectors increases. A restarted method avoids these difficulties by limiting the maximum number of vectors it generates at any time. When the maximum number is reached, a set of new starting vectors is computed and the method is restarted. Typically the restarted Lanczos method stores the Lanczos vectors in core, which allows fast access. Because the maximum number of vectors is usually modest, the arithmetic operations required by reorthogonalizations and the Rayleigh–Ritz projection are reasonably small. These features allow a restarted method to execute efficiently.

There are a number of ways to restart the Lanczos method. Since Algorithm 1 can start with only one vector, the most straightforward way is to use the Ritz vector if one eigenvalue is wanted. If more than one eigenvalue is wanted, we can lock the converged ones and combine the rest into one starting vector [18]. Typically, a restarted Lanczos method with one of these restarting schemes needs significantly more iterations to compute a solution than the nonrestarted version. Recently, there has been a number of significant developments in restarted methods. The implicit restarting scheme is a successful strategy that has been applied to both the Arnoldi method [24] and the Lanczos method [2]. Another successful technique is the dynamic thick-restart scheme [7, 10, 15, 25]. Because the thick-restart scheme uses Ritz pairs directly, it is also known as an explicit restarting scheme. The most commonly used thick-restart method is the thick-restart Davidson method. When used with identity preconditioner, this method is mathematically equivalent to the implicitly restarted Arnoldi method with exact shifts [25]. Since the implicit restart scheme does not restart with Ritz vectors, a separate postprocessing step is required to compute the final Ritz vectors when the users need eigenvectors. This postprocessing step is not needed with an explicit restarting scheme that keeps the latest Ritz vectors in the current Krylov subspace. The thick-restart procedure is only slightly different from the Rayleigh–Ritz projection, whereas the implicit restarting procedure is more complex. The implicitly restarted Arnoldi method is known to have stability concerns [13]; ex-

plicitly restarted methods do not have the same concerns. For these reasons, we set out to develop and study a thick-restart Lanczos method for eigenvalue problems in this paper.

**2. Thick-restart Lanczos algorithm.** We have briefly reviewed the features of the thick-restart procedure and the Lanczos method. In this section we will show how the two may be combined to form an eigenvalue method.

Before any restarting takes place, the restarted Lanczos method proceeds as described in Algorithm 1. Assume that $m$ iterations are allowed before restarting. After $m$ iterations, the vectors $q_i$ satisfy the Lanczos recurrence

$$AQ_m = Q_m T_m + \beta_m q_{m+1} e_m^T, \tag{2.1}$$

where $Q_m = [q_1, \ldots, q_m]$, $e_m$ is the last column of the identity matrix with $m$ columns, and $T_m \equiv Q_m^T A Q_m$ is an $m \times m$ symmetric tridiagonal matrix constructed from $\alpha_i$ and $\beta_i$, $t_{i,i} = \alpha_i, t_{i,i+1} = t_{i+1,i} = \beta_i$. Using the Rayleigh–Ritz projection, we can produce approximate solutions to the eigenvalue problem. Let $(\lambda, y)$ be a pair of eigenvalue and eigenvector, i.e., an eigenpair, of $T_m$; then $\lambda$ 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 residual of this approximate solution is defined to be $Ax - \lambda x$. For symmetric eigenvalue problems, the norm of this residual is a good indicator of the solution quality.

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, there is no apparent advantage to use any basis set other than the eigenvectors. If $Y$ are eigenvectors of $T_m$, the vectors saved during restart $\hat{Q}_k$ are Ritz vectors. To distinguish the quantities before and after restart, we denote the quantities after restart with a 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 relation

$$A\hat{Q}_k = \hat{Q}_k \hat{T}_k + \beta_m \hat{q}_{k+1} s^T, \tag{2.2}$$

where $\hat{q}_{k+1} \equiv q_{m+1}$ and $s \equiv Y^T e_m$. We recognize that this equation is an extension of (2.1) because the residual vector of every Ritz vector in $\hat{Q}_k$ is parallel to $\hat{q}_{k+1}$. In Algorithm 1, the Lanczos recurrence is extended one column at a time by augmenting the current basis with $q_{i+1}$. In the same spirit, we can augment the basis $\hat{Q}_k$ with $\hat{q}_{k+1}$.

To continue extending the basis, we follow the augment Krylov subspace method [3, 19] and use the Gram–Schmidt procedure to enforce the orthogonality of the whole basis. The expression for $\hat{q}_{k+2}$ is

$$\begin{aligned} \hat{\beta}_{k+1}\hat{q}_{k+2} = \hat{r}_{k+1} &\equiv (I - \hat{Q}_{k+1}\hat{Q}_{k+1}^T) A\hat{q}_{k+1} \\ &= (I - \hat{q}_{k+1}\hat{q}_{k+1}^T - \hat{Q}_k \hat{Q}_k^T) A\hat{q}_{k+1} \\ &= (I - \hat{q}_{k+1}\hat{q}_{k+1}^T) A\hat{q}_{k+1} - \hat{Q}_k \beta_m s. \end{aligned} \tag{2.3}$$

The scalar $\hat{\beta}_{k+1}$ in the above equation is equal to the norm of the right-hand side so that $\hat{q}_{k+2}$ has unit norm. Since the vector $\hat{Q}_k^T A \hat{q}_{k+1}$ is known ($= s$), we only need to

compute $\hat{\alpha}_{k+1}$ as in step (c) of Algorithm 1. The vector $\hat{q}_{k+2}$ can be computed by replacing 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, which produces an arrowhead matrix $\hat{T}_{k+1}$. The Lanczos recurrence relation (see (2.1)) is maintained after this step, 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 method, further steps of the restarted Lanczos algorithm can be carried out using three-term recurrence, as shown next.

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

$$\begin{aligned}
\hat{\beta}_{k+i}\hat{q}_{k+i+1} &= (I - \hat{Q}_{k+i}\hat{Q}_{k+i}^T)A\hat{q}_{k+i} \\
&= (I - \hat{q}_{k+i}\hat{q}_{k+i}^T - \hat{q}_{k+i-1}\hat{q}_{k+i-1}^T)A\hat{q}_{k+i} - \hat{Q}_{k+i-2}(A\hat{Q}_{k+i-2})^T\hat{q}_{k+i} \\
&= A\hat{q}_{k+i} - \hat{\alpha}_{k+i}\hat{q}_{k+i} - \hat{\beta}_{k+i-1}\hat{q}_{k+i-1}, \qquad (2.4)
\end{aligned}$$

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

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

The above formulas show how to continue the Lanczos iterations after the first restart. The derivation is based on the facts that the Lanczos vectors are orthogonal and that they satisfy the Lanczos recurrence. Since the vectors resulting from the above formulas also satisfy the same conditions, the procedure can be repeatedly restarted. It is clear that this restarted algorithm is cheaper than the straightforward versions of the augmented Krylov methods. If $k$ vectors $(\hat{Q}_k)$ are saved, the augmented Krylov subspace method needs the projection matrix $\hat{Q}_k^T A\hat{Q}_k$ in order to proceed. The crucial step here is determining how to generate $A\hat{Q}_k$. Usually one either explicitly multiplies $A$ and $\hat{Q}_k$ or computes $A\hat{Q}_k$ from stored $AQ_m$ of previous iterations. However, because of (2.2), $\hat{Q}_k^T A\hat{Q}_k$ is available without performing any matrix-vector multiplication with $A$ or storing $AQ_m$ in computer memory. Similar to the nonrestarted Lanczos method, using the Lanczos recurrence relation we can compute the residual norms of the approximate eigenpairs without explicitly computing the residual vectors. This allows us to measure the quality of the solutions efficiently.

The matrix $T_m$ is no longer tridiagonal after the first restart but can still be stored in an efficient manner. As mentioned before, the matrix $\hat{T}_k$ is diagonal, and its nonzero values can be stored as $\hat{\alpha}_1, \ldots, \hat{\alpha}_k$. The array $(\beta_m s)$ is of size $k$ and can be stored as $\hat{\beta}_1, \ldots, \hat{\beta}_k$. In short, the arrays $\hat{\alpha}_i$ and $\hat{\beta}_i$ are

$$\hat{\alpha}_i = \lambda_i, \qquad \hat{\beta}_i = \beta_m y_{m,i}, \qquad i = 1, \ldots, k, \qquad (2.5)$$

where $\lambda_i$ is the $i$th saved eigenvalue of $T_m$, the corresponding eigenvector is the $i$th column of $Y$, and $y_{m,i}$ is the $m$th element of the $i$th column. After restart the first $k$ basis vectors satisfy the following relation:

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

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

ALGORITHM 3.

The thick-restart Lanczos iterations starting with $k$ Ritz vectors and a residual vector $r_k$ such that $Aq_i = \alpha_i q_i + \beta_i q_{k+1}$, $i = 1, \ldots, k$, and $q_{k+1} = r_k/\|r_k\|$. The value $k$ may be zero, in which case $\alpha_i$ and $\beta_i$ 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 Algorithms 1 and 3 is in the initialization step. When $k$ is zero, the initialization step of the two algorithms are the same. Algorithm 3 can take on an arbitrary number of starting vectors, but Algorithm 1 cannot. When $k$ is greater than zero, the initialization step orthogonalizes $Aq_{k+1}$ against all existing $k+1$ vectors. In all subsequent steps, the same three-term recurrence is used for both Algorithms 1 and 3.

It is easy to see how an existing restarted Lanczos program can be converted to generate orthogonal bases using the above algorithm. To convert a complete eigenvalue program, the Rayleigh–Ritz projection step needs to be modified because the matrix $T_m$ is not tridiagonal in the new method. Our options 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, as in any other version of the Lanczos method, we can evaluate the residual norms of the approximate solutions and perform a convergence test [9, 18]. In addition, based on Ritz values and their residual norms, we can also decide which Ritz pairs to save. This allows us to compute only those Ritz vectors that are needed for restarting.

Following the same argument used to show that the implicitly restarted Arnoldi method is equivalent to the thick-restart Davidson method, it is easy to show that the thick-restart Lanczos method is mathematically equivalent to the implicitly restarted Lanczos method [27]. We derived the thick-restart Lanczos method by using the augmented Krylov subspace concept which may lead to bases that do not span Krylov subspaces. This equivalence property indicates that the bases built by this method in fact span Krylov subspaces, though the starting vectors for the Krylov sequences are not explicitly known. A corollary of this equivalence property is that the new method is a Krylov subspace method. Because of the equivalence relation, we expect the new method to be as effective as the implicitly restarted Lanczos method. One advantage of the new method is that it is simpler to implement as a computer program.

This concludes the description of the new algorithm. In the remainder of this paper, we will focus on two issues related to implementing the eigenvalue method on computers: how to maintain orthogonality among the Lanczos vectors, and which Ritz pairs to save when restarting.

TABLE 3.1
*Information about the test matrices used.*

| 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 |

**3. Orthogonality of the basis.** The above description of the thick-restart Lanczos method is accurate only when carried out in exact arithmetic. When implemented as a computer program, the round-off errors of floating-point arithmetic will cause the Lanczos vectors to lose orthogonality. A similar issue also exists for other variants of the Lanczos method and has been extensively studied before. The typical cure for loss of orthogonality is reorthogonalization through the Gram–Schmidt procedure. The commonly used reorthogonalization schemes are no reorthogonalization [4, 26], the selective reorthogonalization [17], the partial reorthogonalization [22], and the full reorthogonalization. In this section we will exam three of the four schemes, i.e., no reorthogonalization, full reorthogonalization, and partial reorthogonalization. We leave out the selective reorthogonalization because it has similar objectives to the partial reorthogonalization scheme, and the latter one was shown to be more effective [21].

If no reorthogonalization is performed, we avoid the arithmetic operations associated with reorthogonalization, and we need to access only the two most recent Lanczos vectors when building the basis. If eigenvectors aren't needed, there is no need to access the earlier Lanczos vectors. Not performing reorthogonalization may reduce both operation count and memory requirement of a nonrestarted Lanczos program. The same is not true for the thick-restart Lanczos method. The thick-restart Lanczos method cannot be implemented without storing the Lanczos vectors since they are an integral part of the restarting process. Not performing reorthogonalization in the thick-restart Lanczos method reduces only the arithmetic operations. In the nonrestarted Lanczos method, loss of orthogonality among the Lanczos vectors leads to spurious solutions, even though the spurious solutions are duplicate copies of the correct eigenvalues [9, 16]. To illustrate the issues related to loss of orthogonality in the thick-restart Lanczos method, we conduct some tests using three symmetric matrices listed in Table 3.1. These matrices are the largest symmetric matrices from the MatrixMarket web site[1] when the tests were conducted.

Figure 3.1 shows the orthogonality level of the bases built by the thick-restart Lanczos method without reorthogonalization. The horizontal axis indicates how many times the Lanczos method restarted, and the vertical axis is the Frobenius norm of $Q^TQ - I$, where $Q$ contains 20 Lanczos vectors. When vectors in $Q$ are orthogonal to machine precision, we would expect $\|Q^TQ - I\|_F$ to be on order of $10^{-15}$. As $\|Q^TQ - I\|_F$ becomes close to one, $Q$ is no longer a set of linearly independent vectors. Data in Figure 3.1 show that the loss of orthogonality progressively becomes worse after the first few restarts. Similar loss of orthogonality has been observed in the nonrestarted Lanczos method. Despite the loss of orthogonality, the nonrestarted Lanczos method can still compute the eigenvalues accurately. To see whether the thick-restarted Lanczos method behaves similarly, we conduct further tests. For con-

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

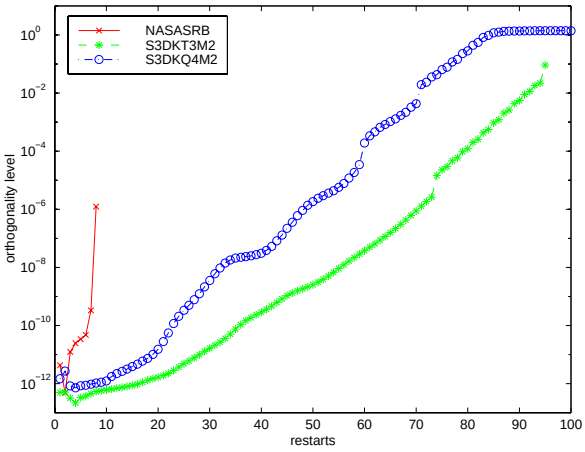


FIG. 3.1. *The orthogonality level* ($\|Q^TQ - I\|_F$) *of the bases built by the thick-restart Lanczos method without reorthogonalization* ($m = 20$).

venience of discussion, we define $\delta_\lambda$ and $\delta_r$ to measure the errors caused by the loss of orthogonality:

$$\delta_\lambda \equiv |\lambda - x^T Ax/x^T x|, \qquad \delta_r \equiv \|Ax - \lambda x\| - |\beta_m e_m^T y|.$$

These two quantities together will be called the floating-point errors in this paper. If the Rayleigh–Ritz projection is carried out using exact arithmetic on an exactly orthogonal basis, both $\delta_\lambda$ and $\delta_r$ are zero. If the Lanczos vectors are orthogonal to the machine precision ($\epsilon$), then the floating-point errors are on the order of $\epsilon\|A\|$.

Table 3.2 shows the five largest Ritz values and their corresponding $\delta_\lambda$ and $\delta_r$ computed by the thick-restart Lanczos method without reorthogonalization ($m = 20$). Since all values of $\delta_\lambda$ are close to $\epsilon\|A\|(= \epsilon\lambda_{max})$, these Ritz values are close to their exact values. Given that the Ritz values are accurate, $\delta_r$ indicates errors in the Ritz vectors. If the Ritz vectors are accurate, $\delta_r$ is expected to be close to $\epsilon\|A\|$. The values of $\delta_r$ in Table 3.2 are several orders of magnitude larger than $\epsilon\|A\|$, which indicates that the eigenvectors are not computed accurately. Similar characteristics are also present in the nonrestarted Lanczos method. What is also similar is that they both generate the same kind of spurious solutions. For example, the largest eigenvalue of S3DKQ4M2 is a simple eigenvalue; however, from Table 3.2, we see it is computed twice.

The most straightforward way to cure the loss of orthogonality problem is to perform full reorthogonalization. Since the full reorthogonalization maintains the orthog-

TABLE 3.2
*The five largest eigenvalues computed by the thick-restart Lanczos method without reorthogonalization.*

| NASASRB | | | S3DKT3M2 | | | S3DKQ4M2 | | |
|---|---|---|---|---|---|---|---|---|
| $\lambda$ | $\delta_\lambda$ | $\delta_r$ | $\lambda$ | $\delta_\lambda$ | $\delta_r$ | $\lambda$ | $\delta_\lambda$ | $\delta_r$ |
| 2648056755 | 1E-6 | 2E2 | 8798.436369 | 3E-11 | 7E-6 | 4601.653436 | 6E-11 | 2E-6 |
| 2647979344 | 1E-6 | 2E2 | 8796.715998 | 1E-11 | 7E-5 | 4601.653436 | 3E-11 | 7E-7 |
| 2634048615 | 4E-6 | 7E2 | 8794.143789 | 4E-11 | 1E-3 | 4600.851648 | 4E-11 | 3E-6 |
| 2633679289 | 1E-6 | 9E2 | 8793.936155 | 4E-11 | 7E-3 | 4599.515718 | 3E-12 | 2E-6 |
| 2606151408 | 3E-6 | 1E3 | 8792.317911 | 9E-12 | 8E-3 | 4598.281889 | 6E-11 | 2E-5 |

onality to the machine precision, reorthogonalization is necessary only if $r_{k+1}^T r_{k+1} < \alpha_{k+1}^2 + \sum_{i=1}^{k} \beta_i^2$ after step (1.d) or $r_i^T r_i < \alpha_i^2 + \beta_{i-1}^2$ after step (2.d) [28]. Usually it is necessary only to orthogonalize $r_i$ against $Q_i$ once [16]. If the norm of $r_i$ reduced significantly after the Gram–Schmidt procedure, it indicates that $r_i$ is almost a linear combination of the current basis $Q_i$. In other words, an invariant subspace has been found. The algorithm can be continued with any unit vector that is orthogonal to $Q_i$. This is a different form of restarting which happens very infrequently. In this case, $\beta_i$ should be set to zero. There are many different ways to implement the full reorthogonalization procedure, for example, always performing the Gram–Schmidt procedure once or twice, using a different criteria to determine when to invoke the Gram–Schmidt procedure [5], etc. The scheme we have selected above appears to be inexpensive and works well in tests.

The third reorthogonalization scheme is the partial reorthogonalization scheme which simulates loss of orthogonality using the $\omega$-recurrence and performs reorthogonalization only if the loss of orthogonality exceeds the user-specified limit. It is relatively easy to adopt the $\omega$-recurrence to the thick-restart Lanczos method [28]. Using the $\omega$-recurrence, we can monitor the loss of orthogonality and maintain the orthogonality to any reasonable level desired. Similar to the nonrestarted Lanczos method, it is easy to show that the thick-restart Lanczos method can generate accurate eigenvalues if the actual orthogonality level of the basis is no worse than $\sqrt{\epsilon}$ [28]. The partial reorthogonalization procedure for the thick-restart Lanczos method is very similar to that of the nonrestarted Lanczos method. One important caveat is that the last residual vector before restarting, $r_m$ or equivalently $q_{m+1} \equiv r_m/\|r_m\|$, must be orthogonal to the existing basis vectors to the machine precision. This does not mean that the all earlier vectors need to be orthogonal to machine precision; it merely means that the reorthogonalization process must be invoked in the last iteration before restarting [28]. More details on the partial reorthogonalization can be found in the appendix.

Figure 3.2 plots the orthogonality level of the Lanczos bases built by the thick-restart Lanczos method with the partial reorthogonalization ($m = 20$). The difference between the two curves is that the solid curve is generated with the modification that always reorthogonalizes $r_{20}$ while the dashed curve is generated without this modification. This extra reorthogonalization ensures that $r_{20}$ has no significant error in the directions of the vectors to be discarded during restarting. Since errors in those directions cannot be recovered in the future iterations, avoiding them improves the overall quality of the bases. The two tests shown in Figure 3.2 clearly demonstrate the importance of maintaining orthogonality of the last residual vector of a restarted loop. The figure also demonstrates that when the last residual vector is orthogonal, the orthogonality level of the whole basis can be maintained at a reasonable level.

The next set of tests will demonstrate that the thick-restart Lanczos method with partial reorthogonalization generates accurate solutions [28]. To do this, we compare the floating-point errors generated by the thick-restart Lanczos method with partial reorthogonalization and with full reorthogonalization. To verify the results, we also conduct the same test on an implementation of the implicitly restarted Lanczos method with full reorthogonalization (ARPACK [12]). Table 3.3 shows the results of this set of tests. The five largest eigenvalues of the three matrices are computed. Corresponding to each eigenvalue, there is a pair of $\delta_\lambda$ and $\delta_r$. The errors reported in the table are the maximum errors of the five pairs. In these tests, we have used a relatively small basis size ($m = 10$). The rationale for using a smaller basis size is that the floating-point errors might be larger because more iterations are needed.

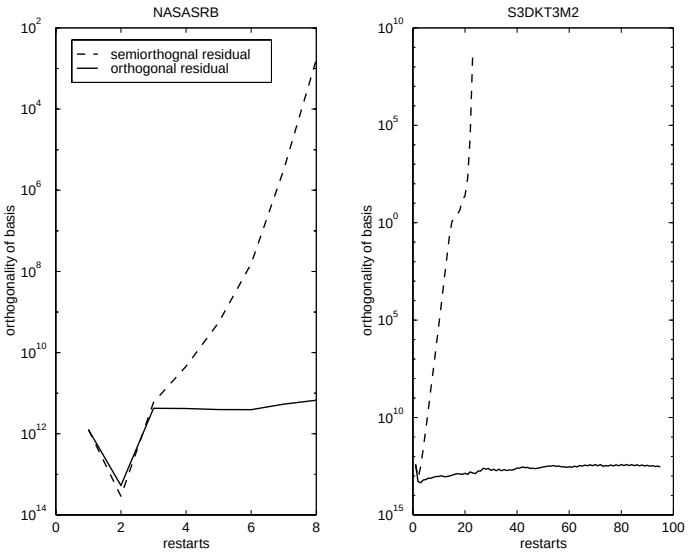


FIG. 3.2. *The orthogonality level* ($\|Q^TQ - I\|_F$) *of the bases prior to each restart.*

Indeed the floating-point errors are slightly larger than when the basis size is 20. The important point to note is that the errors of different methods are roughly the same. Most of the quantities in Table 3.3 are about $10 \sim 100\epsilon\|A\|$, which confirms that the partial reorthogonalization scheme can maintain a very good orthogonality level (see Figure 3.2) and generate accurate solutions.

It is possible for the orthogonality level in the Lanczos method with partial reorthogonalization to rise to the user-specified limit, typically $\sqrt{\epsilon}$, in which case $\delta_r$ would be on the order of $\sqrt{\epsilon}\|A\|$, though $\delta_\lambda$ remains on the order of $\epsilon\|A\|$. Table 3.4 shows an example where $\delta_r$ is significantly larger than $\epsilon\|A\|$. In this example, the smallest eigenvalues of NASASRB are computed. They are nine orders of magnitude smaller than the largest one, and the relative gap ratio is on the order of $10^{-10}$. It takes about 50,000 iterations for the thick-restart Lanczos method to reduce the residual norms of the five smallest Ritz values to $10^{-4}$. ARPACK reduces these residual norms to about $10^4$ using similar number of iterations and the same basis size. The performance difference is mainly due to the differences in the restarting strategies which will be discussed in next section. The size of $\|r\|$ does not affect the value of

TABLE 3.3

*The maximum floating-point errors of the five largest Ritz values computed using basis size (m)* 10*, where method* I *is the thick Lanczos method with partial reorthogonalization, method* II *is the thick-restart Lanczos method with full reorthogonalization, and method* III *is ARPACK.*

| | NASASRB | | | S3DKT3M2 | | | S3DKQ4M2 | | |
|---|---|---|---|---|---|---|---|---|---|
| | I | II | III | I | II | III | I | II | III |
| max $\delta_\lambda$ | 7E-6 | 2E-5 | 6E-6 | 1E-10 | 4E-11 | 2E-10 | 5E-10 | 4E-10 | 3E-11 |
| max $\delta_r$ | 1E-5 | 2E-6 | 3E-6 | 5E-11 | 2E-10 | 7E-11 | 6E-10 | 3E-10 | 4E-10 |
| MATVEC | 185 | 185 | 184 | 2269 | 2269 | 4459 | 5119 | 5119 | 3646 |
| restarts | 46 | 46 | 39 | 465 | 465 | 1310 | 1516 | 1516 | 1516 |
| time | 11.5 | 12.0 | 14.6 | 200 | 213 | 577 | 533 | 555 | 504 |

TABLE 3.4
*The maximum floating-point errors of NASASRB's five smallest Ritz values computed using basis size* 1000.

| | I | II | III |
|---|---|---|---|
| max $\delta_\lambda$ | 9E-7 | 2E-6 | 5E-7 |
| max $\delta_r$ | 7E-4 | 3E-6 | 4E-6 |
| MATVEC | 50471 | 50467 | 46761 |
| restarts | 51 | 51 | 61 |
| time | 9798(8PE) | 8546(8PE) | 8547(16PE) |

$\delta_r$. When full orthogonality is maintained, $\delta_r$ is on the order of $10^{-6}$ ($\sim 10\epsilon\|A\|$); see column II and III of Table 3.4. Because the smallest eigenvalues are much harder to compute than the largest ones, many large eigenvalues reach convergence before the smallest ones do. This provides ample opportunities for serious loss of orthogonality to occur. The actual orthogonality level is near $\sqrt{\epsilon}$ in this case.

Tables 3.3 and 3.4 also contain some performance information about the different methods. The CPU time reported is from a Cray T3E-900 at the National Energy Research Supercomputer Center.[2] The time in Table 3.3 is measured on two processors. The numbers of processors used for Table 3.4 are next to the time values. Since the two versions of the thick-restart Lanczos method use the same restarting strategy, the time differences are mainly due to the different reorthogonalization strategies. The results shown in Table 3.3 are representative of the typical case where using partial reorthogonalization saves execution time. With full reorthogonalization, the global reorthogonalization, i.e., the Gram–Schmidt procedure, is often invoked once per matrix-vector multiplication. With partial reorthogonalization, the Gram–Schmidt procedure is invoked only infrequently. The percentage of time saved is relatively small because the Gram–Schmidt procedure can be performed much faster than the parallel matrix-vector multiplications.

Table 3.4 shows an extreme case where using the partial reorthogonalization actually takes more time than using the full reorthogonalization. In this case, the partial reorthogonalization algorithm invokes the global reorthogonalization frequently, about once every three matrix-vector multiplications. Each time the global reorthogonalization is invoked, the Gram–Schmidt procedure is applied on the two most recent Lanczos vectors to make sure both vectors are orthogonal to the proceeding vectors to machine precision. Since there are some Lanczos vectors that are not orthogonal to machine precision against others, the Gram–Schmidt procedure might have to be repeated more times than in the case where all Lanczos vectors are orthogonal to machine precision [28]. All these make the partial reorthogonalization more expensive for solving this test problem.

In short, the loss of orthogonality among the Lanczos vectors has very similar effects on the thick-restart Lanczos method as on other variants of the Lanczos method. Using the partial reorthogonalization, the thick-restart Lanczos method can generate accurate eigenvalues but not always accurate eigenvectors. In most cases, using partial reorthogonalization reduces the CPU time compared to using full reorthogonalization. When implementing a general purpose Lanczos method, we would suggest using full reorthogonalization. The tests performed in the next section use only the thick-restart Lanczos method with full reorthogonalization.

[2]More information about the National Energy Research Supercomputer Center can be found at http://www.nersc.gov.

**4. Restarting strategies.** The thick-restart Lanczos method offers the flexibility of saving an arbitrary portion of the current basis during restarting. Given this capability, a crucial problem is determining how to take full advantage of it. A number of theoretical tools are available for analyzing restarting strategies used in the implicitly restarted Arnoldi method and the thick-restart Davidson method [2, 8, 15, 24]; however, the most successful strategies for deciding what to save are based on heuristics. For example, ARPACK uses a heuristic strategy based on basis size, number of eigenvalues converged, and so on. This section will briefly summarize our experiences of using three such heuristics.

The first restarting strategy attempts to maximize the reduction of residual norms at every step. This strategy is based on the one used in the dynamic thick-restart Davidson method [25]. It is implemented using a parameter called the effective gap ratio $\gamma$. In each step of the restarted Lanczos method, the residual norm is expected to reduce by a factor proportional to $e^{-\sqrt{\gamma}}$ [15]. To maximize the residual norm reduction, we need to maximize $\gamma$. If the eigenvalues of the matrix $A$ are $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$, the gap ratio for $\lambda_1$ is $(\lambda_2 - \lambda_1)/(\lambda_n - \lambda_1)$. When restarting to compute the smallest eigenvalue while saving the eigenvectors corresponding to $\lambda_2, \ldots, \lambda_k$, the effective gap ratio is [15, 25]

$$\gamma = (\lambda_{k+1} - \lambda_1)/(\lambda_n - \lambda_1).$$

When used in the Davidson method or the Lanczos method, the eigenvalues in the above definition are replaced with the computed Ritz values. Because the Ritz values may not be good approximations to their corresponding eigenvalues and because there are far fewer Ritz values than the eigenvalues $m \ll n$, the computed gap ratio $\gamma$ may be quite different from the actual gap ratio. Typically, when $k$ is large, say, $k \geq 2m/3$, the computed gap ratio $\gamma$ is significantly larger than the actual gap ratio. Another problem with maximizing $\gamma$ is that it is a monotonic function of $k$. The maximum value for $k$ is $m-1$. If this maximum value is used, $m-1$ Ritz pairs are saved during restarting, and the restarting procedure is invoked after every matrix-vector multiplication. Computing $m-1$ Ritz pairs takes a considerable number of arithmetic operations. Since the computed $\gamma$ is larger than its actual value, the reduction in residual norm is much less than expected. To reduce the number of Ritz pairs saved, the developers of the dynamic thick-restart Davidson method [25] enforce a limit on $k$, $k \leq m-3$. In our experiences, we found that reducing the size of $k$ often reduces the overall execution time. The following formula is found to be a reasonable choice, $k \leq \max(n_{eig}, (3m+2n_c)/5)$, where $n_{eig}$ is the number of eigenvalues to be computed and $n_c$ is the number of eigenvalues converged so far.

In the dynamic thick-restart Davidson method, instead of maximizing the residual norm reduction of each iteration, we choose to maximize the residual norm reduction of the whole restarted loop, i.e., maximize $\xi \equiv (m-k)\sqrt{\gamma}$. Alternatively, we can also choose to maximize $\mu \equiv (m-k)\gamma$. Both $\xi$ and $\mu$ are not monotonic functions of $k$; we actually need to search through all possible values of $k$ to find their maximums. Typically these strategies yield a smaller $k$ than maximizing $\gamma$. However, to avoid potentially choosing exceedingly large $k$, we also use the same restriction on $k$ as in the previous case.

Table 4.1 shows some examples of how the three restarting strategies work and compares them to a simple restarted Lanczos method (LANSO-locking) and the implicitly restarted Lanczos method implemented in ARPACK. It shows both iteration counts and time used to compute the five largest eigenvalues of the three test matrices.

TABLE 4.1
*Time and iterations needed to compute five largest eigenvalues of the test matrices ($m = 20$).*

| | NASASRB | | S3DKT3M2 | | S3DKQ4M2 | |
|---|---|---|---|---|---|---|
| | Iter. | Time | Iter. | Time | Iter. | Time |
| LANSO-locking | 2170 | 266 | 2678 | 464 | $> 5000$ | $> 1000$ |
| ARPACK | 145 | 23.0 | 973 | 233 | 961 | 257 |
| TRLAN max $\gamma$ | 88 | 14.0 | 693 | 195 | 784 | 232 |
| TRLAN max $\mu$ | 96 | 14.4 | 684 | 167 | 741 | 196 |
| TRLAN max $\xi$ | 92 | 12.9 | 691 | 165 | 757 | 191 |

The time reported is the number of seconds on a DEC alpha processor running at 450MHz. The simple restarted Lanczos method always restarts with the Ritz vector corresponding to the largest Ritz value that is not converged yet, and it locks the converged Ritz pairs. The program is implemented on top of the LANSO package maintained by Beresford Parlett of UC Berkeley. Table 4.1 shows that the three versions of the thick-restart Lanczos method use less time than the simple restarted Lanczos method and ARPACK on the test problems. The differences among the three versions of the thick-restart Lanczos method are relatively small.

This set of examples clearly demonstrates that the restarting strategy is important to the overall effectiveness of the eigenvalue methods. The strategies suggested here give reasonable performances compared to the existing strategies used in ARPACK. Some of the known techniques not discussed here include saving Ritz pairs from the opposite end of the spectrum and taking into account the residual norms when computing gap ratio. Our tests indicate that there are some advantages to using these techniques in combination with those described earlier in this section [29]. However, the modified strategies do not consistently outperform the simple ones shown in Table 4.1.

**5. Summary.** In this paper, we described an explicitly restarted Lanczos method for symmetric eigenvalue problems called the thick-restart Lanczos method. It is theoretically equivalent to the implicitly restarted Lanczos method. The main advantage of the new method is that it is simpler to use. We studied three different reorthogonalization schemes and found that the loss of orthogonality has similar effects on this restarted Lanczos method as on the original nonrestarted Lanczos method. In other words, without reorthogonalization, it usually generates accurate eigenvalues; with the partial reorthogonalization, it is guaranteed to generate accurate eigenvalues; only with the full reorthogonalization can it generate both accurate eigenvalues and accurate eigenvectors.

Through some examples, we also demonstrated the importance of employing an effective restarting strategy and suggested a number of restarting heuristics. Tests showed that these strategies are as effective as the best-known strategies.

**Appendix. Partial reorthogonalization.** This appendix gives more details on how to implement the partial reorthogonalization scheme in the thick-restart Lanczos method. We address three issues in three subsections: $\omega$-recurrence for the thick-restart Lanczos method, global reorthogonalization, and local reorthogonalization.

**A.1. Monitoring loss of orthogonality.** An important ingredient of the partial reorthogonalization is the $\omega$-recurrence for monitoring loss of orthogonality [22]. This subsection extends the $\omega$-recurrence for the thick-restart Lanczos method. The derivation of the recurrence is relatively straightforward. To start with, we will rewrite

(2.1), (2.3), and (2.4) to accommodate round-off errors during the actual computations:

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

In the above equations, $d_i$ represents the error associated with expressing $Aq_i$ in terms of other quantities ($\|d_i\| \le \epsilon \|Aq_i\|$).

The $\omega$-recurrence uses $\omega_{i,j} \equiv q_i^T q_j$ as the measure of loss of orthogonality. For symmetric matrices, we can use the relation $q_j^T Aq_i = q_i^T Aq_j$ and the above three equations to generate a recursion for $\omega_{i,j}$:

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

To use the recursion, we need to evaluate the quantities $\pm d_j^T q_i$. In our implementation of the $\omega$ recurrence, we simply replace $\pm d_j^T q_i$ with $\epsilon \|Aq_i\|$. The above set of equations can be used only when $i$ is greater than $k+1$. Among the first $k+2$ Lanczos vectors, $q_1, \ldots, q_{k+1}$ are not generated by the current Lanczos iterations, and only $q_{k+2}$ is computed in the current Lanczos iterations; see (2.3). Since the computation of $q_{k+2}$ explicitly involved all previous vectors, the decision of whether to perform reorthogonalization has been studied [5]. We need only apply the $\omega$-recurrence when computing $q_i$, $i > k+2$.

Let $W$ denote the matrix formed from $\omega_{i,j}$. One important point to note here is that the $(i+1)$st row of $W$ depends only on the two previous rows. This indicates that if the orthogonality levels of $q_{k+1}$ and $q_{k+2}$ are known, the above recurrence can be carried forward. In practice, we try to make $q_{k+1}$ and $q_{k+2}$ orthogonal to previous vectors to machine precision. This can prevent the loss of orthogonality among the first $k$ vectors from significantly affecting the orthogonality level of the new vectors computed later. Since $q_{k+1}$ after restarting is $q_{m+1}$ before restarting, this also explains why $q_{m+1}$, i.e., $r_m$, has to be computed accurately; see Figure 3.2. For similar reasons, when $q_i$ needs global reorthogonalization, we should orthogonalize both $q_i$ and $q_{i-1}$ [21].

**A.2. Global reorthogonalization.** The global reorthogonalization here refers to the process of applying the Gram–Schmidt procedure to explicitly orthogonalize $q_i$ against all previous vectors. This is necessary when $q_i$ has exceeded the user-specified limit on loss of orthogonality, say, $\|(\omega_{i,1}, \ldots, \omega_{i,i})\| \ge \sqrt{\epsilon}$. Using the $\omega$-recurrence to simulate the loss of orthogonality, when to invoke the global reorthogonalization procedure can be easily determined. Typically, if the Gram–Schmidt procedure is invoked, it may be repeated to ensure the desired orthogonality level is achieved. The

remainder of this subsection will briefly examine the decision of how many repetitions to use. To answer this question, we need to find out how effective the Gram–Schmidt procedure is and when to stop.

Let $z$ be an arbitrary vector and $Q$ be a set of Lanczos vectors that are nearly orthogonal; the process of repeatedly applying the Gram–Schmidt procedure can be written as $z_{(i)} = (I - QQ^T)z_{(i-1)}$, where $z_{(0)} \equiv z$. Define $w_{(i)} \equiv Q^T z_{(i)}$; the orthogonality level between $z_{(i)}$ and $Q$ can be measured by $\|w_{(i)}\|$. It is easy to see that $Q^T z_{(i)} = (I - Q^T Q)Q^T z_{(i-1)}$ and $\|w_{(i)}\| \leq \|I - Q^T Q\|\|w_{(i-1)}\|$. If $\|I - Q^T Q\| < 1$, repeating the Gram–Schmidt procedure will eventually reduce $\|w_{(i)}\|$ to a very small number. When the Gram–Schmidt procedure is carried out in exact arithmetic, $z_{(\infty)}$ is exactly the same as orthogonalizing $z$ against an exactly orthonormal basis of $Q$. When the Gram–Schmidt procedure is carried out in finite-precision arithmetic, it is likely to produce a $z_{(\infty)}$ that is orthogonal to $Q$ to machine precision. However, the solution may not be the same as in the exact arithmetic case.

Previously, it was argued that when $q_i$ needs reorthogonalization, it should be orthogonalized to machine precision [21]. We adopt the same stopping criteria for our global reorthogonalization. Orthogonalizing to machine precision can be expressed as requiring $\|w_{(i)}\| < \epsilon\|z_{(\infty)}\|$. Since $z_{(\infty)}$ is not computed, the above condition can be rewritten as $\|w_{(i)}\| < \epsilon\|z_{(i)}\|$. In the process of computing $z_{(i)}$, $w_{(i-1)}$ is computed. The above equation can be expressed in known quantities as

$$\|I - Q^T Q\|\|w_{(i-1)}\| < \epsilon\|z_{(i)}\|.$$

Using the relation between $\|w_{(i)}\|$ and $\|w_{(i-1)}\|$, we can estimate $\|I - Q^T Q\|$. The above stopping criteria can be implemented efficiently. This stopping test differs from earlier ones in that it takes into account of the orthogonality of $Q$ [5, 12]. This characteristic is important to the partial reorthogonalization since $Q$ may be of varying orthogonality level.

**A.3. Local reorthogonalization.** To implement a robust Lanczos method, the local orthogonality should always be maintained. When the global reorthogonalization is not necessary, we apply an explicit local reorthogonalization. This local reorthogonalization uses the Gram–Schmidt procedure to orthogonalize $q_{i+1}$ against $q_i$ and $q_{i-1}$. It needs to be done only once, but doing so has two important consequences. It ensures that $\omega_{i+1,i}$ and $\omega_{i+1,i-1}$ are on the order of $\epsilon$ and that $\alpha_i$ and $\beta_i$ are accurate to the machine precision. Both of these conditions are crucial ingredients in guaranteeing the accuracies of eigenvalues computed by the thick-restarted Lanczos method [28].

**Acknowledgments.** The authors would like to thank the referees for invaluable suggestions and comments.

REFERENCES

[1] W. E. Arnoldi, *The principle of minimized iteration in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951), pp. 17–29.

[2] D. Calvetti, L. Reichel, and D. Sorensen, *An implicitly restarted Lanczos method for large symmetric eigenvalue problems*, Electron. Trans. Numer. Anal., 2 (1994), pp. 1–21.

[3] A. Chapman and Y. Saad, *Deflated and Augmented Krylov Subspace Techniques*, Tech. Rep. UMSI 95/181, Minnesota Supercomputing Institute, University of Minnesota, Minneapolis, MN, 1995.

[4] J. Cullum and R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Theory*, Progress in Scientific Computing 3, Birkhäuser Boston, Boston, MA, 1985.

[5] 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 (1976), pp. 772–795.
[6] E. R. Davidson, *The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices*, J. Comput. Phys., 17 (1975), pp. 87–94.
[7] E. R. Davidson, *Super-matrix methods*, Comput. Phys. Comm., 53 (1989), pp. 49–60.
[8] G. de Samblanx, *Filtering And Restarting Projection Methods for Eigenvalue Problems*, Ph.D. thesis, Katholieke Universiteit Leuven, Heverlee, Belgium, 1998.
[9] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, PA, 1997.
[10] D. R. Fokkema, G. L. G. Sleijpen, and H. A. van der Vorst, *Jacobi-Davidson style QR and QZ algorithms for the reduction of matrix pencils*, SIAM J. Sci. Comput., 20 (1999), pp. 94–125.
[11] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.
[12] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK User's Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, SIAM, Philadelphia, PA, 1998. ARPACK Software is available at http://www.caam.rice.edu/software/ARPACK/.
[13] R. B. Lehoucq and D. Sorensen, *Deflation techniques for an implicitly restarted Arnoldi iteration*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 789–821.
[14] R. B. Morgan, *Computing interior eigenvalues of large matrices*, Linear Algebra Appl., 156 (1991), pp. 289–309.
[15] R. B. Morgan, *On restarting the Arnoldi method for large nonsymmetric eigenvalue problems*, Math. Comp., 65 (1996), pp. 1213–1230.
[16] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Classics Appl. Math. 20, SIAM, Philadelphia, PA, 1998.
[17] B. N. Parlett and D. Scott, *The Lanczos algorithm with selective orthogonalization*, Math. Comp., 33 (1979), pp. 217–238.
[18] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press, Manchester, UK, 1993.
[19] Y. Saad, *Analysis of Augmented Krylov Subspace Techniques*, Tech. Rep. UMSI 95/175, Minnesota Supercomputing Institute, University of Minnesota, Minneapolis, MN, 1995.
[20] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing, Boston, MA, 1996.
[21] H. D. Simon, *The Lanczos Algorithm for Solving Symmetric Linear Systems*, Ph.D. thesis, University of California, Berkeley, CA, 1982.
[22] H. D. Simon, *The Lanczos algorithm with partial reorthogonalization*, Math. Comp., 42 (1984), pp. 115–136.
[23] 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 (1996), pp. 401–425.
[24] D. S. Sorensen, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385.
[25] A. Stathopoulos, Y. Saad, and K. Wu, *Dynamic thick restarting of the Davidson, and the implicitly restarted Arnoldi methods*, SIAM J. Sci. Comput., 19 (1998), pp. 227–245.
[26] L.-W. Wang and A. Zunger, *Large scale electronic structure calculations using the Lanczos method*, Computational Materials Science, 2 (1994), pp. 326–340.
[27] K. Wu, *Preconditioned Techniques for Large Eigenvalue Problems*, Ph.D. thesis, University of Minnesota, Minneapolis, MN, 1997. An updated version also appears as Tech. Rep. TR97-038 at the Computer Science Department, University of Minnesota, Minneapolis, MN.
[28] K. Wu and H. Simon, *Thick-Restart Lanczos Method for Symmetric Eigenvalue Problems*, Tech. Rep. 41412, Lawrence Berkeley National Laboratory, Berkeley, CA, 1998.
[29] K. Wu and H. D. Simon, *Dynamic Restarting Schemes for Eigenvalue Problems*, Tech. Rep. LBNL-42982, Lawrence Berkeley National Laboratory, Berkeley, CA, 1999.