Dealing With Linear Dependence during the Iterations of the Restarted Block Lanczos Methods

J. Baglama*

March 22, 2005

Dedicated to Richard Varga on the occasion of his 70th birthday

Abstract

The Lanczos method can be generalized to block form to compute multiple eigenvalues without the need of any deflation techniques. The block Lanczos method reduces a general sparse symmetric matrix to a block tridiagonal matrix via a Gram-Schmidt process. During the iterations of the block Lanczos method an off-diagonal block of the block tridiagonal matrix may become singular, implying that the new set of Lanczos vectors are linearly dependent on the previously generated vectors. Unlike the single vector Lanczos method, this occurrence of linearly dependent vectors may not imply an invariant subspace has been computed. This difficulty of a singular off-diagonal block is easily overcome in non-restarted block Lanczos methods, see Golub and Underwood [12], and Ruhe [30]. The same schemes applied in non-restarted block Lanczos methods can also be applied in restarted block Lanczos methods. This allows the largest possible subspace to be built before restarting. However, in some cases a modification of the restart vectors is required or a singular block will continue to reoccur. In this paper we examine the different schemes mentioned in [12, 30] for overcoming a singular block for the restarted block Lanczos methods, namely the restarted method reported in [12] and the Implicitly Restarted Block Lanczos (IRBL) method developed by Baglama et al. [3]. Numerical examples are presented to illustrate the different strategies discussed.

block Lanczos method, eigenvalues, implicit restarting, singular block, polynomial acceleration.

65F15

*Department of Mathematical Sciences, Ball State University, Muncie, Indiana 47306-0490. E-mail: jbaglama@math.bsu.edu.

1 Introduction

The eigenvalue problem

$$Ax = \lambda x, \tag{1}$$

where $A \in \mathbf{R}^{n\times n}$ is a large sparse symmetric matrix is an important computational problem. The task of finding a few extreme eigenvalues and associated eigenvectors arises in many applications, such as in structural engineering to determine vibrational frequencies, or in chemistry for the configurations of molecules. An impressive calculation by Lehoucq et al. [18] for finding the vibrational states of a four-atom molecule required computing the fifty-two smallest eigenvalues of a symmetric matrix on the order of $2 \cdot 10^6$. A calculation involving multiple eigenvalues by Baglama et al. [3] required computing eigenvectors associated with the smallest eigenvalues of a symmetric matrix on the order of $3 \cdot 10^5$. The eigenvectors were then used to determine solution paths of the minimal energy configuration of liquid crystals. Eigenvalue problems are becoming larger and more complicated and the need for robust and efficient algorithms has spurred considerable research aimed at improving existing methods or at developing alternative methods; see, e.g., [1, 2, 3, 6, 9, 12, 14, 23, 22, 24, 26, 27, 28, 31, 33, 34, 35] and references therein. Research has generated numerous public domain codes for finding a few eigenvalues and associated eigenvectors of a large sparse symmetric matrix; for instance, ARPACK developed by Lehoucq et al. [21], BLZPACK developed by Marques [22], and LASO2 developed by Scott [32]. See [4] for a survey on public domain codes. The key to developing robust and efficient public code is the detail to implementation issues. The goal of this paper is to discuss the implementation issue of encountering linearly dependent Lanczos vectors in the restarted block Lanczos methods.

The Lanczos method [16] can be generalized to block form to compute multiple or close eigenvalues without the need of any deflation techniques. There are many implementations of the block Lanczos method; see e.g., Chatelin [7], Cullum and Willoughby [9], Golub and Underwood [12], Grimes et al. [14], Parlett [26], Ruhe [30], Scott [32], and Ye [36]. All block methods generate an orthonormal basis for the block Krylov subspace,

$$\begin{aligned} \mathbf{K}_j^b(A, v_1, \ldots, v_r) \;=\; & \operatorname{span}\{v_1, Av_1, \ldots, A^{j-1}v_1, \\ & \quad v_2, Av_2, \ldots, A^{j-1}v_2, \\ & \qquad \vdots \\ & \quad v_r, Av_r, \ldots, A^{j-1}v_r\}, \end{aligned} \tag{2}$$

where $v_1, v_2, \ldots, v_r \in \mathbf{R}^n$.

Similarly to the basic Lanczos method, the basic block Lanczos method encounters the difficulties of a large storage requirement for the Krylov subspace basis and the low accuracy of the computed approximate eigenvalues and eigenvectors due to loss of orthogonality. To overcome these difficulties one can

periodically restart the block Lanczos method with starting vectors that have strong components in the direction of the eigenvectors associated with the desired eigenvalues. Golub and Underwood [12] suggest restarting with the desired Ritz vectors. A generalization of the Implicitly Restarted Lanczos (IRL) method described by Calvetti et al. [6] and Sorensen [34] to block form, the Implicitly Restarted Block Lanczos (IRBL) method by Baglama et al. [3] can be regarded as a curtailed block QR algorithm for the symmetric eigenvalue problem. The IRBL method is an efficient method for restarting the block Lanczos algorithm. It is well suited for the computation of multiple or very close eigenvalues and requires very little storage requirement when both eigenvalues and associated eigenvectors are required.

The block Lanczos methods introduce another difficulty, that is not present in the single vector Lanczos methods. When a vector $A^l v_i$, $l \leq j-1$ and $1 \leq i \leq r$, in (2) is a linear combination of the other Krylov vectors, this does not necessarily imply that an invariant subspace has been found. It only implies that no additional information can be obtained from $A^l v_i$ and any A multiple of $A^l v_i$. Therefore, a modification of the block Lanczos method is required to continue to be able to build a basis. This is considered a favorable breakdown in the single vector Lanczos case yielding an invariant subspace with eigenvalues of the Lanczos matrix coinciding with eigenvalues of the matrix A, see [13, p. 478]. Breakdown due to linearly dependent Krylov vectors without converged eigenvectors does not occur frequently. Grimes et al. [14] state that they have never seen this breakdown occur in their code. However, this does not mean that we should ignore it.

One way of handling a breakdown in a restarted block Lanczos method is to stop the algorithm when the breakdown occurs, check if any eigenvectors have converged, and modify the starting vectors by introducing a random vector. However, this is not very advantageous in a restarted method since this will produce a much smaller Krylov subspace for that iteration, yielding a poor approximation for the next restart vectors. Also, the introduction of a random vector causes undesired eigenvector components to be present, hence, slows down convergence. Therefore, we suggest handling this breakdown as one would in a non-restarted block Lanczos method, i.e., use a scheme that will allow the restarted block Lanczos method to continue to build the Lanczos basis to the set maximum size before restarting. This will allow the best approximation for the next iteration.

In this paper, we will investigate two simple schemes for handling a breakdown. Both schemes will allow the Lanczos basis to be built to the user set maximum size. Also, we will show when a random vector must be used to replace a restart vector to avoid a reoccurrence of the breakdown.

This paper is organized as follows. In Section 2, we review a block Lanczos method and describe the different strategies for overcoming the breakdown in the non-restarted block Lanczos methods. Section 3 investigates the different strategies for overcoming the breakdown for the restarted block Lanczos

methods when restarting with Ritz vectors. In Section 4, we review the IRBL method and investigate strategies for overcoming the breakdown when using this method. Illustrative numerical examples are displayed in Section 5, and concluding remarks are found in Section 6.

2 Block Lanczos Method

For this discussion we use a version of the block Lanczos method developed by Ruhe [30] that yields an algorithm that is quite similar to the original Lanczos algorithm. When the block size is equal to 1, Ruhe's implementation coincides with the usual Lanczos process. A particular advantage of the block Lanczos algorithm is the possibility to multiply a group of vectors by a matrix A using level 3 BLAS [10] matrix-matrix multiplication subroutines. Ruhe's version does not allow for the use of level 3 BLAS. However, this can be remedied by performing r matrix-vector multiplications every r steps.

In Ruhe's implementation, the vectors in the Krylov subspace bases generated are orthogonalized sequentially. This implementation is attractive for this discussion, since the breakdown can be easily identified by computing the Euclidean vector norm $\|\cdot\|$ of a single vector (see step 14) in Algorithm 2.

Block Lanczos (Ruhe's Variant) Algorithm

1.) Choose r initial orthonormal vectors $V_{1r} = [v_1, \ldots, v_r]$ and set $\{t_{ij}\}_{i,j=1}^{mr} = 0$.
2.) For $j = r, r+1, \ldots, mr$ do:
3.) $k = j - r + 1$
4.) Compute $f_1 = Av_k$
5.) For $i = 1, k-1$ do:
6.) $f_1 = f_1 - t_{ik} v_i$
7.) Enddo
8.) For $i = k, \ldots, j$ do:
9.) $t_{i,k} = f_1^T v_i$
10.) If $i \neq k$ $t_{k,i} = t_{i,k}$
11.) $f_1 = f_1 - t_{ik} v_i$
12.) Enddo
13.) If $j < mr$ then
14.) $t_{j+1,k} = \|f_1\|$
15.) $v_{j+1} = \frac{f_1}{\|f_1\|}$
16.) $t_{k,j+1} = t_{j+1,k}$
17.) Endif
18.) Enddo
19.) For $j = 2, \ldots, r$ do:
20.) $k = mr - r + j$
21.) $f_j = Av_k$
22.) For $i = 1, k-1$ do:

23.) $f_j = f_j - t_{ik} v_i$
24.) Enddo
25.) For $i = k, mr$ do:
26.) $t_{i,k} = f_j^T v_i$
27.) If $i \neq k$ then $t_{k,i} = t_{i,k}$
28.) $f_j = f_j - t_{ik} v_i$
29.) Enddo
30.) Enddo
31.) Set $F_r = [f_1, \ldots, f_r]$, $V_{mr} = [v_1, \ldots, v_{mr}]$, $T_{mr} = \{t_{ij}\}_{i,j=1}^{mr}$.

An application of m steps of Algorithm 2 yields

$$AV_{mr} = V_{mr} T_{mr} + F_r E_r^T, \tag{2}$$

with $V_{mr}^T V_{mr} = I_m$, $V_{mr}^T F_r = 0$, and

$$T_{mr} = \begin{pmatrix} D_1 & B_1^T & & & & 0 \\ B_1 & D_2 & B_2^T & & & \\ & B_2 & D_3 & B_3^T & & \\ & & & \ddots & & \\ & & & \ddots & \ddots & B_{m-1}^T \\ 0 & & & & B_{m-1} & D_m \end{pmatrix}, \tag{3}$$

where $D_j \in \mathbf{R}^{r \times r}$ are $r \times r$ blocks, and $B_j \in \mathbf{R}^{r \times r}$ are upper triangular blocks.

The loss of orthogonality among the basis vectors is generally observed after a number of iterations. This causes redundant copies of eigenpairs to emerge. To secure the orthogonality of the basis vectors a reorthogonalization step is performed when necessary. The remedies used to secure orthogonality in the single vector Lanczos method can be used in Algorithm 2, see [25, 26] for a discussion on maintaining orthogonality in the Lanczos methods. Results in [5, 26, 29] show that at most one reorthogonalization is required.

The values $t_{j+1,k} = \|f_1\|$, $j = r, \ldots, mr-1$, $k = 1, \ldots, mr-r$ (at step 14 in Algorithm 2) correspond to the diagonal elements in the off-diagonal blocks $B_1, B_2, \ldots, B_{m-1}$ in (3). Therefore, an off-diagonal block is singular if and only if $t_{j+1,k} = \|f_1\| = 0$. This implies that the next Lanczos vector f_1 becomes linearly dependent on the previous j Lanczos vectors, $v_1, \ldots, v_j$. Suppose ϵ is suitably small. We say a breakdown occurs when $r > 1$ and

$$\|f_1\| < \epsilon. \tag{4}$$

This may or may not imply that an eigenvector has converged. This only implies that no additional information can be obtained from $A^l v_k$, where $1 \leq l \leq m-1$ and any A multiple of $A^l v_k$.

Algorithm 2 needs to be modified in order to handle the breakdown and continue to build the Lanczos basis. There are two possible remedies when $\|f_1\| < \epsilon$ and $r > 1$:

Option I
Set $t_{k,j+1} = t_{j+1,k} = 0$ and generate a random unit vector f_1, such that $f_1^T v_i = 0$, for $i = 1, \ldots j$. Then set $v_{j+1} = f_1/\|f_1\|$ and continue. That is, insert steps $14a - 14g$ into Algorithm 2.

Block Lanczos Algorithm With Option I

$\vdots$
13.) If $j < mr$ then
14.) $t_{j+1,k} = \|f_1\|$
14a.) if $\|f_1\| < \epsilon$
14b.) Set $t_{k,j+1} = t_{j+1,k} = 0$
14c.) Let f_1 be a random vector
14d.) for $i = 1, \ldots, j$ do:
14e.) $f_1 = f_1 - (f_1^T v_i) v_i$
14f.) Enddo
14g.) Endif
15.) $v_{j+1} = \frac{f_1}{\|f_1\|}$
16.) $t_{k,j+1} = t_{j+1,k}$
17.) Endif
$\vdots$

Option II
Set $t_{k,j+1} = t_{j+1,k} = 0$, reduce the block size and go to the next iteration. This is the strategy that Ruhe suggested in [30] for the non-restarted block Lanczos method. Termination occurs when $r = 1$ and $\|f_1\| < \epsilon$, and yields an invariant subspace.
Insert steps $14a - 14g$ into Algorithm 2.

Block Lanczos Algorithm With Option II

$\vdots$
13.) If $j < mr$ then
14.) $t_{j+1,k} = \|f_1\|$
14a.) if $\|f_1\| < \epsilon$
14b.) Set $t_{k,j+1} = t_{j+1,k} = 0$
14c.) If r=1 Goto step 31.
14d.) $r = r - 1$
14e.) $j = j - 1$

14f.) Goto step 17.
14g.) Endif

15.) $v_{j+1} = \frac{f_1}{\|f_1\|}$
16.) $t_{k,j+1} = t_{j+1,k}$
17.) Endif
⋮

Both options I and II discard the vector f_1. In Option I the vector is replaced by a random vector (step 14c in Algorithm 2), and in Option II the reduction of block size (step 14d in Algorithm 2), implies that v_{j+1} is replaced in the next iteration. Both of these options can be implemented in the restarted block Lanczos method so as to continue to build the Lanczos basis.

3 Restarted Block Lanczos Method

One method of restarting is with approximate Ritz vectors.

Restarted Block Lanczos Algorithm
1.) Let k be the number of desired eigenvalues.
2.) Choose r initial orthonormal vectors $V_{1r} = [v_1, \ldots, v_r]$.
3.) Choose m, the maximum number of block Lanczos steps.
4.) Compute $AV_{mr} = V_{mr}T_{mr} + F_r E_r^T$ by Algorithm 2.
5.) Compute k eigenvalues and eigenvectors of T_{mr}; $\{\theta_i, y_i\}_{i=1}^r$.
6.) Check convergence.
7.) Deflate any converged eigenvectors; stop if all eigenvalues have been found.
8.) Compute new starting vectors. $V_{1r} = [v_1 = V_{mr}y_1, \ldots, v_r = V_{mr}y_r]$.
9.) Goto step 4.

Suppose a breakdown occurs during step 4 of Algorithm 3 and no eigenvectors converged. If Option II was used to continue Algorithm 2, then in the next iteration of Algorithm 3 a breakdown will occur again during step 4. This can be illustrated as follows.

Let r be the block size and m the number of Lanczos steps to be applied. Let

$$AV_l = V_l T_l + F_{\hat{r}} E_{\hat{r}}^T \tag{5}$$

be the block Lanczos decomposition and assume that a breakdown has occurred and Option II was used during step 4 of Algorithm 3. Then $\hat{r} < r$, $l \leq mr$, $V_l \in \mathbf{R}^{n\times l}$ and $T_l \in \mathbf{R}^{l\times l}$. Before the breakdown the block tridiagonal matrix in the block Lanczos decomposition had bandwidth $2r+1$. After the breakdown, T_l in (5) has bandwidth $2\hat{r}+1$.

Compute new starting vectors $V_{1r} = [v_1^+ = V_l y_1, \ldots, v_r^+ = V_l y_r]$, where $\{\theta_i, y_i\}_{i=1}^r$ are r eigenpairs of T_l. Then

$$
\begin{array}{c}
Av_1^+ = AV_l y_1 = (V_l T_l + F_{\hat{r}} E_{\hat{r}}^T) y_1 = \theta_1 v_1^+ + F_{\hat{r}} E_{\hat{r}}^T y_1 \\
Av_2^+ = AV_l y_2 = (V_l T_l + F_{\hat{r}} E_{\hat{r}}^T) y_2 = \theta_2 v_2^+ + F_{\hat{r}} E_{\hat{r}}^T y_2 \\
\vdots \\
Av_r^+ = AV_l y_r = (V_l T_l + F_{\hat{r}} E_{\hat{r}}^T) y_r = \theta_r v_r^+ + F_{\hat{r}} E_{\hat{r}}^T y_r
\end{array} \tag{6}
$$

and $\text{span}\{v_1^+, \ldots, v_r^+, Av_1^+, \ldots, Av_r^+\} \subseteq \text{span}\,\{v_1^+, \ldots, v_r^+, F_{\hat{r}} E_{\hat{r}}^T\}$. The latter set contains $r + \hat{r} < 2r$ vectors. Thus, $v_1^+, \ldots, v_r^+, Av_1^+, \ldots, Av_r^+$ must be linearly dependent and a breakdown in Algorithm 2 will occur when creating the first off-diagonal block. Also, a similar argument applies if we were to stop Algorithm 2 when the breakdown occurs. However, if Option I is used to continue Algorithm 2, then the residual matrix $F_{\hat{r}}$ in the Lanczos decomposition (5) will contain r vectors instead of $\hat{r} < r$ vectors. This will allow the next iteration of Algorithm 3 to continue, in general, without a breakdown. Therefore, we suggest using this implementation to handle a breakdown in a restarted block Lanczos methods which restarts with a combination of Ritz vectors. If no eigenvectors have converged, this implementation does not require a modification to the starting vectors.

However, neither Option I nor Option II can be implemented in the IRBL method without replacing a restart vector with a random vector. This is shown in the following section.

4 Implicitly Restarted Block Lanczos (IRBL) Method

We review the IRBL method presented in [3]. It generalizes the IRL method discussed in [6, 34]. Let m steps of the block Lanczos method produce the block Lanczos decomposition (2).

Let $z \in \mathbf{R}$ and determine the QR factorization $T_{mr} - zI_{mr} = QR$, where $Q, R \in \mathbf{R}^{mr \times mr}$, $Q^T Q = I_{mr}$, and R is upper triangular. We refer to z as a shift. Multiplying equation (2) by Q from the right-hand side we obtain

$$
A(V_{mr} Q) = (V_{mr} Q)(Q^T T_{mr} Q) + F_{mr} E_{mr}^T Q. \tag{7}
$$

Let $T_{mr}^+ = Q^T T_{mr} Q$. Then T_{mr}^+ is a symmetric block tridiagonal matrix with the same bandwidth as T_{mr}. The matrix Q in the QR factorization of $T_{mr} - zI_{mr}$ is a generalized upper Hessenberg matrix, whose lower triangular part has bandwidth r. After applying the $m-1$ shifts $z_1, z_2, \ldots, z_{m-1}$, we obtain

$$
AV_{mr}^+ = V_{mr}^+ T_{mr}^+ + F_r E_r^T Q^+, \tag{8}
$$

where $V_{mr}^+ = [v_1^+, \ldots, v_{mr}^+] = V_{mr} Q^+$, $Q^+ = Q_1 \cdots Q_{m-1}$, $T_{mr}^+ = (Q^+)^T T_{mr} Q^+$, and Q_j denotes the orthogonal matrix associated with the shift z_j. Introduce

the partitioning

$$T_{mr}^{+} = \left(\begin{array}{c|ccc} T_r^+ & B_r^T & 0 \cdots & 0 \\ \hline B_r & & & \\ 0 & & & \\ \vdots & & T_{mr-r}^+ & \\ 0 & & & \end{array}\right), \tag{9}$$

where $T_r^+ \in \mathbf{R}^{r\times r}$, $B_r \in \mathbf{R}^{r\times r}$ is upper triangular, and $T_{mr-r}^+ \in \mathbf{R}^{(mr-r)\times(mr-r)}$. Equate the first r columns on the right-hand side and left-hand side of (8). We then obtain

$$AV_r^+ = V_r^+ T_r^+ + F_r^+, \tag{10}$$

where $V_r^+ = [v_1^+, v_2^+, \ldots, v_r^+]$ and $F_r^+ = [v_{r+1}^+, \ldots, v_{2r}^+]B_r + F_r E_r^T Q^+ I_{n\times r}$. The mth shift z_m is applied according to

$$V_r^{++} = F_r^+ + V_r^+(T_r^+ - z_m I_r). \tag{11}$$

Introduce $\hat{R} = (R_m^r R_{m-1}^r \ldots R_1^r)^{-1}$, where R_j^r is the first r columns and r rows of the upper triangular matrix R_j in the QR factorization of $T_{mr} - z_j I$. Then

$$V_r^{++} = \psi_m(A)V_r\hat{R}, \tag{12}$$

where ψ_m is a polynomial of degree m with zeros $z_1, \ldots, z_m$.

Formula (12) shows that we can multiply the initial matrix V_r for the block Lanczos method by an accelerating polynomial in A of degree m without evaluating any matrix-vector products with the matrix A, in addition to those matrix-vector products that were computed during m steps of the block Lanczos method.

The choice of accelerating polynomial ψ_m, i.e., the choice of the shifts $z_1, \ldots, z_m$, is discussed in [3] and [19], see also [1, 2, 6, 34]. One seeks to choose shifts z_j so that the range V_r^{++} is in, or close to, an invariant subspace of A associated with all or a subset of the desired eigenvalues of A.

Having computed V_r^{++} in the manner outlined, we orthonormalize the columns of V_r^{++}, and denote the orthonormal matrix so obtained by V_r. The block Lanczos process is now restarted with the initial matrix V_r.

If a breakdown occurs and if Option I or Option II are used to continue Algorithm 2, then a breakdown will occur in the next iteration when restarting with the matrix V_r^{++}. Suppose $A^{i+1}v_k$, $1 \le k \le r$, $i+1 \le m$, is a linear combination of the vectors in the Krylov subspace, i.e., $A^{i+1}v_k = \sum_{n=1}^{p} \alpha_n A^l v_j$, $0 \le l \le i$, and $1 \le j \le r$. Multiply equation (12) from the right-hand side by the axis vector e_k and from the left-hand side by A^{i+1},

$$A^{i+1}v_k^{++} = A^{i+1}\psi_m(A)V_r\hat{R}e_k = A^{i+1}\psi_m(A)[r_{1k}v_1 + \ldots + r_{kk}v_k] \tag{13}$$

where $[r_{1k}, \ldots, r_{kk}]^T$ is the kth column of $\hat{R}$ and

$$\begin{aligned} A^{i+1}v_k^{++} &= \psi_m(A)[A^{i+1}(r_{1k}v_1 + \ldots + r_{k-1k}v_{k-1}) + r_{kk}A^{i+1}v_k] \\ &= \psi_m(A)[r_{1k}A^{i+1}v_1 + \ldots + r_{k-1k}A^{i+1}v_{k-1} + r_{kk}\textstyle\sum_{n=1}^{p}\alpha_n A^l v_j] \\ &= A^{i+1}[r_{1k}\psi_m(A)v_1 + \ldots + r_{k-1k}\psi_m(A)v_{k-1}] + r_{kk}\textstyle\sum_{n=1}^{p}\alpha_n A^l \psi_m(A)v_j. \end{aligned}$$

Notice that if Option I is used, then the random vector introduced at step 14c in Algorithm 2 or any A multiple of the random vector does not appear in the linear combination of $A^{i+1}v_k$, or in $A^{i+1}v_k^{++}$. Also, it follows from (12) that

$$\begin{array}{rcl} v_1^{++} &=& \psi_m(A)r_{11}v_1 \\ v_2^{++} &=& \psi_m(A)[r_{12}v_1 + r_{22}v_2] \\ &\vdots& \vdots \\ v_r^{++} &=& \psi_m(A)[r_{1r}v_1 + r_{2r}v_2 + \ldots + r_{rr}v_r], \end{array} \tag{14}$$

where $r_{11}, \ldots, r_{rr}$ are nonvanishing since $\hat{R}$ is assumed to be invertible. This implies that the vectors $\psi_m(A)v_j$, $1 \le j \le r$, can be written as linear combinations of the vectors $v_1^{++}, \ldots, v_r^{++}$. Hence, $A^{i+1}v_k^{++}$ can be written as a linear combination of the previous vectors and a breakdown will occur.

This shows that if a breakdown occurs while using the IRBL method then a random vector must replace a starting vector. A similar argument holds if we stop Algorithm 2 when the breakdown occurs. Considering that the convergence of the IRBL method depends on applying an accelerating polynomial to dampen part of the spectrum, stopping Algorithm 2 early will result in a lower degree polynomial applied during that iteration. Hence, we should apply either Option I or II, so that we get the largest degree possible for the accelerating polynomial. This is especially important if Ritz values are used as shifts in the IRBL method since early termination will produce poor approximate Ritz values.

If a breakdown occurs and an eigenvector has not converged, then we need to modify our starting vectors. Therefore, if a breakdown occurs while creating vector v_{j+1} in Algorithm 2, then before restarting we should replace the starting vector v_k, $k = j \bmod(r) + 1$ in V_r^{++} with a random vector. The addition of a random vector will introduce undesired eigenvector components, slowing down convergence. To help prevent the introduction of undesired eigenvector components the random vector can be projected onto the Krylov subspace already computed before replacing the starting vector v_k.

To reduce computation flops, Option II can be used so that a reduction in block size may result and more shifts can be applied during that iteration. This is illustrated in Example 2 in Section 5. However, it is not always possible to take advantage of a reduced block size and application of more shifts. Therefore, we suggest using Option I with replacing the starting vector that caused the

breakdown with a random vector.

5 Numerical Experiments

This section presents a few examples that illustrate how the different strategies presented in this paper can be used to overcome a singular block in a restarted method. The numerical examples presented here have a breakdown occur in the first iteration. All numerical experiments were carried out in MATLAB using double precision arithmetic, i.e., with approximately 16 significant digits. The stopping criterion used in all examples is

$$\|Ax - x\theta\| = \|(AV_{mr} - V_{mr}T_{mr})y\| = \|F_r E_r^T y\| \leq \text{TOL} \tag{15}$$

where $AV_{mr} = V_{mr}T_{mr} + F_r E_r^T$ is a block Lanczos decomposition, θ is an eigenvalue of the matrix T_{mr}, y is an associated eigenvector, $x = V_{mr}y$, and TOL is a chosen tolerance. For all numerical experiments, the value of ϵ in (4) which determines when a breakdown has occurred is chosen to be $\sqrt{\text{macheps}}\|A\|$. Where macheps is the machine epsilon, $\approx 2.2 \cdot 10^{-16}$, and $\|A\|$ is the largest singular value of A. The latter is approximated by $\|T_{mr}\|$.

Example 1. Let $A \in \mathbf{R}^{100\times 100}$ be the matrix obtained by discretizing the 2-dimensional negative Laplace operator on the unit square by the standard 5-point stencil with Dirichlet boundary conditions. Algorithm 3 will be used with block size $r = 2$, $m = 5$, and initial vectors

$$\begin{aligned} v_1 &= rand_1, \\ v_2 &= A^2 \cdot v_1, \end{aligned} \tag{16}$$

where $rand_1$ is a random vector and v_2 is chosen so that a breakdown occurs in the first iteration. This choice of starting vectors will cause the second off-diagonal block B_2 of the block tridiagonal matrix T_{mr} (3) to be singular. TOL is set to 10^{-6}. We are looking for the 3 smallest eigenvalues.

Deflation Procedure	# Matrix-Vector Products	Reoccurrence of Breakdown	Missed Multiple Eigs.
Option I	170	No	No
Option II	90	Yes	Yes

When applying Option I, the breakdown occurs only in the first iteration in the second off-diagonal block B_2. However, when applying Option II, the breakdown occurred in the second off-diagonal block B_2 in the first iteration, and in the first off-diagonal block B_1 in every iteration thereafter. We also missed one of the multiple eigenvalues due to the early reduction of the block size.

Example 2. Let $A = \text{diag}\{a_{11}, a_{22}, \ldots, a_{100,100}\}$ have entries

$$a_{ii} = \begin{cases} \frac{1}{100}, & \text{if } 1 \le i \le 3, \\ \\ \frac{i^2}{100}, & \text{if } 4 \le i \le 100. \end{cases}$$

The IRBL method will be used with block size $r = 3$, $m = 5$, shifts chosen as Leja points [3], and initial vectors

$$\begin{array}{l} v_1 = rand_1, \\ v_2 = rand_2, \\ v_3 = A^3 \cdot v_1, \end{array} \tag{17}$$

where $rand_1$ and $rand_2$ are random vectors and v_3 is chosen so that a breakdown occurs in the first iteration. This choice of starting vectors will cause the third off-diagonal block B_3 of the block tridiagonal matrix T_{mr} (3) to be singular. TOL is set to 10^{-8}. We are looking for the 3 smallest eigenvalues.

Deflation Procedure	# Matrix-Vector Products	Reoccurrence of Breakdown	Missed Multiple Eigs.
Option I (random starting vector)	555	No	No
Option I	675	Yes	Yes
Option II (random starting vector)	390	No	No
Option II	525	Yes	Yes

When applying Option I without replacing starting vector v_3 with a random vector in any iteration, the breakdown occurs in the third off-diagonal block B_3 in the first iteration, and in the third off-diagonal block B_3 in at least half of the iterations afterwards. This breakdown did not reoccur in every iteration due to round-off errors. When applying Option I and replacing starting vector v_3 with a random vector in the second iteration, the breakdown occurred only in the first iteration in the third off-diagonal block B_3.

Applying Option II gave the following structure for the 15×15 block tridiagonal matrix,

$$\begin{pmatrix} D_1 & B_1^T & & & & 0 \\ B_1 & D_2 & B_2^T & & & \\ & B_2 & D_3 & B_3^T & & \\ & & B_3 & D_4 & B_4^T & \\ & & & B_4 & D_5 & B_5^T \\ 0 & & & & B_5 & D_6 \end{pmatrix}, \tag{18}$$

where D_1, D_2, D_3, B_1, B_2 are 3×3 blocks, D_4, D_5, D_6, B_4, B_5 are 2×2 blocks, and B_3 is a 2×3 block. This structure allowed the IRBL method to apply 6 shifts. Without special handling of the breakdown only 5 shifts can be applied.

When applying Option II without replacing starting vector v_3 with a random vector in any iteration, the breakdown occurs in the third off-diagonal block B_3 in the first iteration and in the third off-diagonal block B_3 in at least half of the iterations afterwards. Like the previous experiment with Option I, this breakdown did not reoccur in every iteration due to round-off errors. We were also able to apply 6 shifts when a breakdown occurred. This produced faster convergence, i.e, fewer matrix-vector multiplications were required before the termination criterion was satisfied, but the reduction of block size resulted in a missed multiple eigenvalue. When applying Option II and replacing starting vector v_3 with a random vector in the second iteration, the breakdown occurred only in the first iteration in the third off-diagonal block B_3.

6 Conclusion

This paper illustrates how to handle an unfortunate breakdown in the restarted block Lanczos methods. We showed that when Ritz vectors are used to restart the block Lanczos method, breakdown can be handled the same way as in the non-restarted block Lanczos method with no modification of the starting vectors. However, in the IRBL method a modification of the starting vectors is required.

7 Acknowledgments

The author is grateful to Lothar Reichel and the anonymous referee for insightful comments and suggestions that resulted in an improved presentation.

References

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

[2] J. Baglama, D. Calvetti and L. Reichel, *Fast Leja points*, Elec. Trans. Numer. Anal., 7 (1998), pp. 124–140.

[3] J. Baglama, D. Calvetti, L. Reichel, and A. Ruttan, *Computation of a few close eigenvalues of a large matrix with application to liquid crystal modeling*, J. Comp. Physics 146, (1998), pp. 203-226.

[4] M. W. Berry, *A survey of public domain Lanczos-based software*, Proceedings of the Cornelius Lanczos International Centenary Conference, SIAM (1994), pp. 332-334.

[5] Å. Björck, *Numerics of Gram-Schmidt orthogonalization*, Linear Algebra Appl., 197-198 (1994), pp. 297–316.

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

[7] F. Chatelin, *Eigenvalues of Matrices*, Wiley, Chichester, 1993.

[8] J. Cullum and W. E. Donath, *A block Lanczos algorithm for computing the q algebraically largest eigenvalues and a corresponding eigenspace for large, sparse symmetric matrices*, Proceedings of the 1974 IEEE Conference on Decision and Control, New York, 1974, pp. 505-509.

[9] J. Cullum and R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations*, Vol. 1, Birkhäuser, Boston, 1985.

[10] J. J. Dongarra, J. DuCroz, I. F. Duff, and S. Hammarling, *A set of level 3 basic linear algebra subprograms*, ACM Trans. on Math. Software, 16 (1990), pp. 1–17.

[11] R. W. Freund, *Templates for Band Lanczos Methods and for the Complex Symmetric Lanczos Method*, Numerical Analysis Manuscript No. 99-3-01, Bell Laboratories, January 1999.

[12] G. H. Golub and R. Underwood, *The block Lanczos method for computing eigenvalues*, Mathematical Software III, J. R. Rice, ed., Academic Press, New York, 1977, pp. 361–377.

[13] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2nd ed., Johns Hopkins University Press, Baltimore, 1989.

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

[15] W. Karush, *An iterative method for finding characteristic vectors of a symmetric matrix*, Pacific J. Math., 1 (1951), pp. 233–248.

[16] C. Lanczos, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, National Bureau of Standards, 45, (1950), pp. 225–282.

[17] R. B. Lehoucq, *Analysis and implementation of an implicitly restarted Arnoldi iteration*, Ph.D. Thesis, Rice University, Houston, 1995.

[18] R. B. Lehoucq, S. K. Gray, D. H. Zhang, and J. C. Light, *Vibrational Eigenstates of Four-Atom Molecules: A Parallel Strategy Employing the Implicitly Restarted Lanczos Method*, Comput. Phys. Comm., (109) pp. 15-26 (1998).

[19] R. B. Lehoucq and K. J. Maschhoff, *Block Arnoldi Method*, Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide, Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. Van der Vorst, eds., SIAM, to be published.

[20] R. B. Lehoucq and D. C. Sorensen, *Deflation techniques for an implicitly restarted Arnoldi iteration*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 789–821.

[21] R. B. Lehoucq, D. C. Sorensen, P. A. Vu and C. Wang, ARPACK: An implementation of an implicitly restarted Arnoldi method for computing some of the eigenvalues and eigenvectors of a large sparse matrix, 1996. Code available from Netlib in directory scalapack.

[22] O. A. Marques, BLZPACK: A Fortran 77 implementation of the block Lanczos algorithm. Code available at http://www.nersc.gov/∼osni/marques_software.html.

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

[24] C. C. Paige, *Computational variants of the Lanczos method for the eigenproblem*, J. Inst. Math. Appl., 10 (1972), pp. 373–381.

[25] B. N. Parlett, *The rewards for maintaining semi-orthogonality among Lanczos vectors*, Numer. Linear Alg. Appl., 1 (1992), pp. 243–267.

[26] B. N. Parlett, The Symmetric Eigenvalue Problem, SIAM, Philadelphia, 1998.

[27] B. N. Parlett and B. Nour-Omid, *Towards a black box Lanczos program*, Comput. Phys. Comm., 53 (1989), pp. 169–179.

[28] B. N. Parlett and D. S. Scott, *The Lanczos algorithm with selective orthogonalization*, Math. Comp., 33 (1979), pp. 311–328.

[29] L. Reichel and W. B. Gragg, *Algorithm 686: FORTRAN subroutines for updating the QR decomposition of a matrix*, ACM Trans. Math. Software, 16 (1990), pp. 369–377.

[30] A. Ruhe, *Implementation aspect of band Lanczos algorithms for computation of eigenvalues of large sparse symmetric matrices*, Math. Comp., 33 (1979), pp. 680-687.

[31] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, Halstead Press, New York, 1992.

[32] D. S. Scott, LASO2 - FORTRAN implementation of the Lanczos process with selective orthogonalization. Code and documentation available from Netlib.

[33] 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.

[34] D. C. Sorensen, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385.

[35] K. Wu and H. Simon, *Thick-Restart Lanczos Method for Symmetric Eigenvalue Problems*, LBNL Report 41412, 1998.

[36] Qiang Ye, *An Adaptive Block Lanczos Algorithm*, Num. Algs., 12, (1996), pp 97 -110.