MATHEMATICS OF COMPUTATION
VOLUME 62, NUMBER 205
JANUARY 1994, PAGES 179-207

A BREAKDOWN-FREE VARIATION OF THE NONSYMMETRIC LANCZOS ALGORITHMS

QIANG YE

ABSTRACT. The nonsymmetric Lanczos tridiagonalization algorithm is essentially the Gram-Schmidt biorthogonalization method for generating biorthogonal bases of a pair of Krylov subspaces. It suffers from breakdown and instability when a pivot at some step is zero or nearly zero, which is often the result of mismatch of the two Krylov subspaces. In this paper, we propose to modify one of the two Krylov subspaces by introducing a "new-start" vector when a pivot is small. This new-start vector generates another Krylov subspace, which we add to the old one in an appropriate way so that the Gram-Schmidt method for the modified subspaces yields a recurrence similar to the Lanczos algorithm. Our method enforces the pivots to be above a certain threshold and can handle both exact breakdown and near-breakdown. In particular, we recover look-ahead Lanczos algorithms and Arnoldi's algorithm as two special cases. We also discuss theoretical and practical issues concerning the new-start procedure and present a convergence analysis as well as some numerical examples.

1. INTRODUCTION

Eigenvalue problems arise in various applications of science and engineering. Numerical discretization of physical problems frequently leads to computations of some eigenvalues of large sparse matrices, both symmetric and nonsymmetric. Over decades, numerous computational methods have been developed, and attention, in particular, has been paid to large sparse problems in recent years. For a list of applications and literature, we refer to [21] for the symmetric problem and [18] for the nonsymmetric problem. For large symmetric matrices, the symmetric Lanczos algorithm has proved to be an effective method for computing a few eigenvalues. As is well known, some substantial difficulties emerge in the nonsymmetric case, and a long-standing problem is to search for an efficient way to overcome these difficulties, which is the problem we are concerned with in this paper.

The symmetric Lanczos algorithm [19] is based on constructing an orthonormal basis for a Krylov subspace by the Gram-Schmidt method. Using a three-term recurrence, it is very economical in both computation time and storage space and, at the same time, offers fast convergence to eigenvalues (cf. [15, 21]). For nonsymmetric matrices, the Gram-Schmidt (biorthogonalization) method

Received by the editor January 13, 1992 and, in revised form, September 5, 1992.
1991 *Mathematics Subject Classification*. Primary 65F15.
Key words and phrases. Lanczos algorithms, breakdown, nonsymmetric eigenvalue problems, new-start procedure, convergence.

can be used to construct biorthogonal bases of two Krylov subspaces and Lanczos [19] developed an algorithm, now called the nonsymmetric Lanczos algorithm, for constructing these bases via a two-sided three-term recurrence. Though having some attractive features of the symmetric case, the nonsymmetric Lanczos algorithm is liable to breakdown and near-breakdown (i.e., appearance of a zero or nearly zero pivot in a division) and is potentially unstable. This is one of several problems arising in the generalization to the nonsymmetric case. Another problem concerns the analysis of convergence, which has recently been done in [30].

Several modifications of the Lanczos algorithm have been introduced to deal with the difficulty of breakdown. In [23, 27], a look-ahead Lanczos algorithm (LAL) was developed from a modified LDL^* decomposition of a moment matrix. It uses a block pivot, if necessary, and constructs slightly changed biorthogonal bases of the same Krylov subspaces. On the other hand, the relation between the Lanczos tridiagonalization algorithms and formal orthogonal polynomials has been studied (see [12], for example) and a nongeneric Lanczos algorithm was developed in [10, 11]. Based on the recurrence formula satisfied by formal orthogonal polynomials, the nongeneric Lanczos algorithm produces block biorthogonal bases of the two Krylov subspaces. Consideration of near-breakdown and an efficient implementation were given in [8, 11]. Unfortunately, both of these two methods may encounter so-called incurable breakdown (see [11, 22, 27]) and may not resolve within certain steps, a curable breakdown. Although theoretically an incurable breakdown yields some eigenvalues, its numerical detection is not easy, and the eigenvalues obtained may not be the ones that are of interest. Another generalization of the symmetric Lanczos algorithm is Arnoldi's algorithm (see [1, 26]), which does not have the difficulty of breakdown. Using a single Krylov subspace, the Arnoldi algorithm produces an orthonormal basis by the Gram-Schmidt method; however, all iterative vectors are present in the recurrence. This significantly increases the computational cost and the demand of storage, and is therefore of limited use in applications to large matrices. The breakdown phenomenon has also been studied in the context of nonsymmetric linear systems (see [4, 7, 9, 11, 16]). For further discussions of the Lanczos algorithms, see [2, 3, 5, 20, 22, 24].

In this paper, we derive a new algorithm, which is a generalization of both the original look-ahead Lanczos algorithm and the Arnoldi algorithm, and provides a connection between them. We observe that the magnitude of the pivots in the nonsymmetric Lanczos algorithm is essentially fixed by the Krylov subspaces. The difficulty with the incurable breakdown suggests that the two Krylov subspaces may not match well, and should therefore be changed. When a near-breakdown occurs (the pivot given by two basis vectors constructed is less than a threshold parameter), we propose to replace one of the two vectors by a "new-start" vector. Then a Krylov subspace can be generated from the new-start vector. We then modify one of the old Krylov subspaces by adding the new one to it in an appropriate way, so that the biorthogonalization step in the Gram-Schmidt method for the modified pair of subspaces is reduced to a recurrence similar to the nonsymmetric Lanczos algorithm with a few more terms added. With the new-start strategy, our method keeps the pivots above the threshold and, at the same time, produces a matrix in condensed form (banded upper

Hessenberg form, see §§3 and 4), which can be used to approximate the original matrix.

We also point out that the Lanczos algorithms may suffer from a bad choice of the Krylov subspaces not only in the nonsymmetric case but also in the classical symmetric case. For instance, if the initial vector in the symmetric Lanczos algorithm has small components in the eigendirections of interest, then convergence could be significantly slow. Of course, a single Krylov subspace cannot contain information on multiplicities of eigenvalues. In practice, it will also be difficult to find clustered eigenvalues from a single Krylov subspace. Under these circumstances, changing the single Krylov subspace to a sum of two will be beneficial and necessary. So a symmetric version of the algorithm presented in this paper will be relevant also in the symmetric case. We leave the details to future work.

The paper is organized as follows. We first briefly describe in §2 the nonsymmetric Lanczos algorithm and its modifications, as well as the Arnoldi algorithm. We then present our main algorithm and related results in §3, and follow this by a section on a projection matrix. We also discuss theoretical and practical issues concerning the new-start procedure in §5. It turns out that the look-ahead Lanczos algorithm and the Arnoldi algorithm can be recovered as two special cases of our algorithm. Also, there is a symmetric version of the algorithm for symmetric pencil problems. These topics, together with some comparisons, are the subjects of §6. Following that, we establish convergence results in §7. Finally, we present some numerical examples in §8 and concluding remarks in §9.

We follow the notational convention used in numerical analysis. In particular, $\|\cdot\|$ represents the 2-norm for both vectors and matrices. The $m \times m$ identity matrix is denoted by I_m, and $e_{i,m}$ denotes the ith coordinate vector in R^m, i.e., $[e_{1,m}, \dots, e_{m,m}] = I_m$. Furthermore, δ_{ij} is the Kronecker symbol and $\langle x, y\rangle$ is the angle between the vectors x and y.

2. Lanczos algorithms

There are several different generalizations of the classical symmetric Lanczos algorithm to the nonsymmetric problem. These include, for example, the (two-sided) nonsymmetric Lanczos algorithm, the look-ahead Lanczos algorithm, the nongeneric Lanczos algorithm and the Arnoldi algorithm. They are based on different interpretations of the symmetric Lanczos algorithm, such as the Gram-Schmidt orthogonalization method, the LDL^* decomposition of a moment matrix and generation of orthogonal polynomials. In this section, we briefly describe three generalizations that are closely related to our method; the details can be found elsewhere.

2.1. **Nonsymmetric Lanczos algorithm.**

In [19], Lanczos presented his algorithm for nonsymmetric matrices in the form of two-sided iteration. Given two initial vectors p_1, q_1, they determine two *Krylov subspaces* (or more precisely, two sequences of Krylov subspaces)

$$K_m(q_1) = \operatorname{span}\{q_1, Aq_1, \dots, A^{m-1}q_1\}$$

and

$$\hat{K}_m(p_1) = \operatorname{span}\{p_1, A^*p_1, \ldots, (A^{m-1})^*p_1\}.$$

It is well known that applying the Gram-Schmidt biorthogonalization method (cf. [23]) to the two given bases of $K_m(q_1)$ and $\hat{K}_m(p_1)$ yields the following three-term recurrence:

$$\beta_{j+1}q_{j+1} = s_j = Aq_j - \alpha_j q_j - \gamma_j q_{j-1}, \tag{1}$$

$$\gamma_{j+1}p_{j+1}^* = r_j^* = p_j^*A - \alpha_j p_j^* - \beta_j p_{j-1}^*, \tag{2}$$

where p_0, $q_0 = 0$, and $\beta_{j+1}\gamma_{j+1} = r_j^* s_j$, $\alpha_j = p_j^* A q_j$.

If $r_j^* s_j = 0$, then p_{j+1} and q_{j+1} cannot be defined and the algorithm is said to *break down* at step j. This, together with the instability that it causes when the algorithm is close to breakdown (i.e., when $\cos\langle r_j, s_j\rangle \approx 0$) has been a major difficulty in the application of the nonsymmetric Lanczos algorithm. Note here that $\omega_j := \cos\langle p_j, q_j\rangle$ is generally referred to as the jth *pivot.*

When no breakdown occurs (i.e., in the generic case), the algorithm produces, at step m, biorthogonal bases $\{p_1, p_2, \ldots, p_m\}$, $\{q_1, q_2, \ldots, q_m\}$, i.e.,

$$p_i^* q_j = \delta_{ij},$$

and a tridiagonal matrix

$$T_m = \begin{pmatrix} \alpha_1 & \gamma_2 & & & & \\ \beta_2 & \alpha_2 & \cdot & & & \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \alpha_{m-1} & \gamma_m \\ & & & & \beta_m & \alpha_m \end{pmatrix},$$

such that

$$T_m = P_m^* A Q_m,$$

where $P_m = [p_1, \ldots, p_m]$ and $Q_m = [q_1, \ldots, q_m]$. In particular, at step n, the matrix T_n is similar to A. An important point about the Lanczos algorithm is that, even when $m << n$, some eigenvalues of T_m give good approximations to those of T_n, and hence those of A (see [30] for an analysis).

2.2. **Look-ahead Lanczos algorithm.** As we have mentioned, the nonsymmetric Lanczos algorithm suffers from breakdown and is, therefore, regarded as unstable in general. In dealing with this difficulty, the look-ahead Lanczos algorithm (LAL) was introduced in [23, 27]. The idea can be described as follows.

Consider the jth step of the Lanczos algorithm and let the vectors r_j, s_j be generated by (1) and (2). The regular Lanczos algorithm generates the next basis vectors p_{j+1}, q_{j+1} by normalizing r_j, s_j, which involves a division by the $(j+1)$st pivot $\omega_{j+1} = \cos\langle r_j, s_j\rangle$. So the difficulty arises when ω_{j+1} is zero or close to zero. One observation leading to LAL is that forming the subsequent r_{j+1} and s_{j+1} for building up bases of the Krylov subspaces uses only the directions of p_{j+1} and q_{j+1}, i.e., those of r_j, s_j, rather than the vectors p_{j+1}, q_{j+1} themselves. Specifically, in LAL, r_{j+1} and s_{j+1} are computed from r_j and s_j before forming p_{j+1}, q_{j+1} by a formula similar to (1) and (2). Then p_{j+1}, $p_{j+2} \in \operatorname{span}\{r_j, r_{j+1}\}$ and q_{j+1}, $q_{j+2} \in \operatorname{span}\{s_j, s_{j+1}\}$ are constructed appropriately such that they are biorthogonal. This construction involves a factorization of a 2×2 matrix and can be understood as using a 2×2 block pivot. In the version of LAL worked out in [23], q_{j+1} and q_{j+2} are chosen as normalizations of s_j and s_{j+1} as usual. The subsequent vectors $p_{j+3}, \dots, p_m$ and $q_{j+3}, \dots, q_m$ depend on r_j, r_{j+1}, s_j and s_{j+1} and can be recovered from the corresponding p's and q's. So LAL involves a local change of the two basis vectors p_{j+1} and p_{j+2}, and the other basis vectors remain unchanged.

A limitation of the algorithm is that it is not always possible to obtain better denominators $\omega_{j+1} = \cos\langle p_{j+1}, q_{j+1}\rangle$ and $\omega_{j+2} = \cos\langle p_{j+2}, q_{j+2}\rangle$. Indeed, the best possible denominators are determined by the angles between span $\{r_j, r_{j+1}\}$ and span $\{s_j, s_{j+1}\}$. For example, if one of the angles is a right angle, i.e., the 2×2 block pivot is singular, then one of the new denominators must be zero (see [23]). In such a case, it was suggested to use a $t \times t$ pivot, or equivalently to include t vectors in the subspaces, say, $r_j, \dots, r_{j+t-1}$ and $s_j, \dots, s_{j+t-1}$, and then choose $p_{j+1}, \dots, p_{j+t}$ and $q_{j+1}, \dots, q_{j+t}$ from the two subspaces respectively. However, the implementation becomes very complicated without there being any guarantee of success. A possible contingency is *incurable breakdown*, i.e., all block pivots are singular (see [23, 27]). In such a case, all Ritz values are eigenvalues, by the mismatch theorem of Taylor, but this is known at the cost of forming all the block pivots.

We also remark that the LAL process is equivalent to applying the Gram-Schmidt method to a slightly modified Krylov basis of $\{p_1, Ap_1, \dots, A^{m-1}p_1\}$ with $A^{j-1}p_1$, $A^j p_1$ replaced by two linear combinations of them.

2.3. **Nongeneric Lanczos algorithm.** The approach in [10] uses the theory of formal orthogonal polynomials. The matrix A and two initial vectors p_1, q_1 define, on the space P^n of polynomials of degree not exceeding $n-1$, a formal inner product by

$$(3) \qquad (h, g) = p_1^* \bar{h}(A) g(A) q_1 \quad \text{for } h, g \in P^n.$$

Then a monic polynomial f_i of degree i is called a *true formal orthogonal polynomial* (true FOP) if $(f_i, x^j) = 0$ for all $0 \le j < i$. Because (3) is not a true inner product, a true FOP may not exist, and even if it exists, it may not be unique. This prompts the classification of true FOPs as *regular* and *irregular* according to their uniqueness. An interesting result is that when no true FOP exists for a certain i, a so-called *deficient* FOP exists.

It is well known that the p_i, q_i generated by the Lanczos algorithm in the generic case (i.e., without any breakdown) satisfy

$$(4) \qquad p_i = f_{i-1}(A^*) p_1 \Pi_i, \qquad q_i = f_{i-1}(A) q_1 \hat{\Pi}_i,$$

where f_{i-1} is the $(i-1)$st regular FOP and Π_i, $\hat{\Pi}_i$ are two scalar constants. The occurrence of breakdown at step j corresponds to nonexistence of the jth regular FOP. Generally, an ith FOP f_i, though it might be irregular or deficient, always exists, and then defines p_i, q_i through (4). Furthermore, a recurrence formula for FOPs has been found, which in turn gives a recurrence among p_i, q_i. This leads to the nongeneric Lanczos algorithm of [10, 11].

Clearly, $p_i \in \hat{K}_i(p_1)$ and $q_i \in K_i(q_1)$. The formal orthogonal property of FOPs ensures that the p's and the q's are block biorthogonal provided f_{n-1} is regular. A transformation of the basis vectors in each block makes the bases biorthogonal and recovers LAL. So, in theory, the nongeneric Lanczos algorithm differs from LAL by a local change of the bases in each block. However, it has been shown that the former is preferable to the latter in implementations, particularly when pivots of size greater than 2 are involved (see [8, 11]). For essentially the same reason as in LAL, the nongeneric Lanczos algorithm may encounter incurable breakdown. In terms of FOPs, this occurs when f_{n-1} is not regular.

2.4. **Arnoldi algorithms.** Another generalization of the symmetric Lanczos algorithm is the Arnoldi algorithm (see [1, 26]). It uses a single Krylov subspace $K_m(q_1)$ and generates an orthonormal basis of $K_m(q_1)$ by the Gram-Schmidt method. Specifically, a basis $\{q_1, \dots, q_m\}$ is constructed by the following recurrence

$$h_{j+1,j}q_{j+1} = s_j = Aq_j - h_{jj}q_j - \cdots - h_{1j}q_1,$$

where h_{ij} is chosen so that $q_{j+1}^* q_i = \delta_{j+1,i}$ for $1 \le i \le j+1$. Because of the nonsymmetry of A, the coefficients h_{ij} are generally nonzero and all j terms will be present in the iteration. This significantly increases the computational cost and use of storage, which are crucial for large-scale problems. On the other hand, using only one sequence of vectors (i.e., a single Krylov subspace), the Arnoldi algorithm yields an orthonormal basis. In this regard, the algorithm is more stable.

At step j, a $j \times j$ upper Hessenberg matrix is obtained and can be used to approximate some eigenvalues of the original matrix. As we will see in §7, however, convergence is expected to be slower. Finally, we refer to [26] for more discussions on the Arnoldi algorithm.

3. Avoiding breakdown by new-start

The various methods presented in §2 construct (block) biorthogonal bases of two Krylov subspaces. Clearly, the bases constructed depend on the initial vectors. In particular, the best (block) pivots are essentially fixed. The difficulty with the incurable breakdown suggests that the Krylov subspaces may be wrong in the sense that they do not match well. In such a case, any modification in the recurrence would not help, and it is the Krylov subspaces that need to be changed.

An old strategy is to abandon the computation and start over with new initial vectors. Unfortunately, there is no known method to make the new choice a better one, not to mention the waste of computation.

The new idea in this paper is to replace one of the two Krylov subspaces by a sum of two Krylov subspaces. When a breakdown occurs at step j, instead of using the normalization of r_{j-1} for p_j, as in the Lanczos algorithm, we choose a new vector for p_j, which we call a new-start vector. With this new p_j, we replace the Krylov subspace $\hat{K}_n(p_1)$ by an appropriate sum of $\hat{K}_n(p_1)$ and $\hat{K}_n(p_j)$. Obviously, the Lanczos three-term recurrence no longer works, but the Gram-Schmidt biorthogonalization method is still applicable for the new pair of subspaces. Moreover, it turns out that the basis of the new subspace can be arranged in such a way that the biorthogonalization step will collapse to a four-term recurrence.

Let $\epsilon \leq 1$ be a threshold parameter for breakdown, and let r_{j-1} and s_{j-1} be generated by (1) and (2). From now on, breakdown is defined by ϵ, i.e., we say that a *breakdown* occurs at step $j-1$, if the pivot $|\omega_j| = |\cos\langle r_{j-1}, s_{j-1}\rangle| \leq \epsilon$. When it occurs, let $q_j = s_{j-1}/\|s_{j-1}\|$ and choose p_j to be any vector satisfying $p_j^* q_i = \delta_{ij}$ for $1 \leq i \leq j-1$ and $|\omega_j| = |\cos\langle p_j, q_j\rangle| > \epsilon$ (assuming the existence of p_j). We call p_j a *new-start vector*.

We now describe how to continue the construction of the subsequent basis vectors after the introduction of the new-start vector p_j. First, p_{j+1} and q_{j+1} can be generated by

$$\gamma_{j+1}^{(2)} p_{j+1}^* = r_j^* = p_{j-1}^* A - \beta_{j-1} p_{j-2}^* - \alpha_{j-1} p_{j-1}^* - \gamma_j p_j^*,$$

$$\beta_{j+1} q_{j+1} = s_j = A q_j - \alpha_j q_j - \gamma_j q_{j-1},$$

where $\alpha_j = p_j^* A q_j$, $\gamma_j = p_{j-1}^* A q_j$ and $\gamma_{j+1}^{(2)}$, β_{j+1} are chosen such that $p_{j+1}^* q_{j+1} = 1$. Clearly, the definition of α_j, γ_j ensures local biorthogonality. Furthermore, it can be checked that q_{j+1} is orthogonal to p_i, and p_{j+1} is orthogonal to q_i, for $1 \leq i \leq j$. Note that at this stage an additional term is added to the formula for the p's but not for the q's. However, from step $j+1$ until the next breakdown, both formulas have an additional term. We have the following four-term recurrence for p_{l+1}, q_{l+1} $(l > j)$:

$$\gamma_{l+1}^{(2)} p_{l+1}^* = r_l^* = p_{l-1}^* A - \beta_{l-1} p_{l-2}^* - \alpha_{l-1} p_{l-1}^* - \gamma_l p_l^*,$$

$$\beta_{l+1} q_{l+1} = s_l = A q_l - \alpha_l q_l - \gamma_l q_{l-1} - \gamma_l^{(2)} q_{l-2},$$

where α_l, γ_l, $\gamma_{l+1}^{(2)}$, and β_{l+1} are chosen similarly as above. Again, it can be verified that the vectors obtained are indeed biorthogonal.

When a further breakdown occurs, a similar recurrence with five terms can be used and so on. In general, $\gamma_j^{(k)}$ will be introduced into the recurrence when there are $k-1$ breakdowns. To be consistent, we rewrite γ_j as $\gamma_j^{(1)}$. Also, to distinguish between the two formulas for p_{j+1} and q_{j+1} immediately after a new-start, it is necessary to introduce two index parameters k and k', which are related to the number of consecutive breakdowns. The following is the main algorithm of this paper.

Algorithm 3.1. Input a breakdown threshold parameter ϵ and two initial vectors p_1 and q_1 with $\|q_1\|_2 = 1$ and $p_1^* q_1 = 1$. Initialize $k = k' = k'' = 1$, $\delta_k = 0$ and $p_0 = q_0 = 0$ and $\beta_1 = 0$.

```
For $l = 1, 2, 3, \dots, m-1$ do
1)  $k' = k''$ (this defines $k' = k'(l)$);
2)  $\alpha_l = p_l^* A q_l$ and $\gamma_l^{(j)} = p_{l-j}^* A q_l$ for $1 \leq j \leq k'$;
3)  $s(l, k') = (A - \alpha_l) q_l - \sum_{j=1}^{k'} \gamma_l^{(j)} q_{l-j}$;
4)  $\beta_{l+1} = \|s(l, k')\|$;
5)  If $\beta_{l+1} = 0$, stop (emergence of a right invariant subspace);
    else, $q_{l+1} = s(l, k')/\beta_{l+1}$;
6)  $k = k'' = k' + \delta_k$ (this defines $k = k(l)$);
7)  $r^*(l, k'') = p_{l-k''+1}^*(A - \alpha_{l-k''+1}) - \beta_{l-k''+1} p_{l-k''}^* - \sum_{j=1}^{k''-1} \gamma_{l-k''+j+1}^{(j)} \cdot$
    $p_{l-k''+j+1}^*$;
```

```
8) If $\|r(l, k'')\| = 0$ and $k'' = 1$, stop (emergence of a left invariant
   subspace);
   If $\|r(l, k'')\| = 0$ and $k'' > 1$, then $k'' \leftarrow k'' - 1$ and goto (7);
9) If $|r^*(l, k'')q_{l+1}|/\|r(l, k'')\| > \epsilon$,
       $p_{l+1} = r(l, k'')/r^*(l, k'')q_{l+1}$ and $\delta_k = 0$;
   else
       choose a nonzero $p_{l+1}$ by newstart and $\delta_k = 1$;
       (i.e., $p^*_{l+1}q_j = \delta_{l+1,j}$ for $1 \le j \le l+1$)
   end if.
```

Two index functions are generated with $k = k(l)$ defined by the value of k at 6) and $k' = k'(l)$ defined by the value of k' at 1).

We add some remarks to help in understanding the algorithm.

Remark 1. The two index functions $k(l)$, $k'(l)$ are related to the number of successive breakdowns that occur at step l and control the number of terms involved in the recurrence with k for the p's and k' for the q's. One has $k(l) = k'(l) + \delta_k$ (see 6)) with $\delta_k = 0$ or 1 depending on whether or not a breakdown occurs at step $l - 1$. This difference is due to that between the recurrences for p_j and q_j immediately after a breakdown. Also note that, at step l, the value of k'' is initially equal to $k(l)$ at 6) but might be decreased at 8). Then the value of k'' at the end of 8) carries to step $l+1$ and defines $k'(l+1)$. Therefore, $k'(l+1) \le k(l)$. In particular, if $k'(l+1) < k(l)$, then $r(l, k'') = 0$ for $k'(l+1) + 1 \le k'' \le k(l)$.

Remark 2. For $1 \le k \le l$, we define

$$s(l, k) = (A - \alpha_l)q_l - \gamma_l^{(1)}q_{l-1} - \cdots - \gamma_l^{(k)}q_{l-k}, \tag{5}$$

$$\begin{aligned} r^*(l, k) =& p^*_{l-k+1}(A - \alpha_{l-k+1}) - \beta_{l-k+1}p^*_{l-k} \\ & - \gamma^{(1)}_{l-k+2}p^*_{l-k+2} - \cdots - \gamma_l^{(k-1)}p^*_l . \end{aligned} \tag{6}$$

At step l of the algorithm, $\gamma_l^{(1)}, \dots, \gamma_l^{(k'(l))}$ are defined at 2). Then the use of $s(l, k'(l))$ at 3) is justified. Note here that $s(l, k'(l))$ depends on l only. Also, $r(l, k'')$ for $k'(l+1) \le k'' \le k(l)$ are used at 7) and they are well defined owing to Lemma 3.4 below.

Remark 3. If $s(l, k') = 0$, then $Aq_l \in \operatorname{span}\{q_{l-k'}, \dots, q_l\}$, which implies that $\operatorname{span}\{q_1, q_2, \dots, q_l\}$ is a right invariant subspace. If $r(l, k) = 0$ with $k > 1$, then $A^*p_{l-k+1} \in \operatorname{span}\{p_{l-k}, \dots, p_l\}$, which is not sufficient to imply A^*-invariance of $\operatorname{span}\{p_1, \dots, p_l\}$, as A^*p_{l-k+2} may not be in this subspace. It is natural then to consider $r(l, k-1)$ that uses A^*p_{l-k+2}, and continue until $r(l, 1) = \cdots = r(l, k) = 0$ when a left invariant subspace is obtained. A rigorous proof will be presented in Corollary 3.6.

Remark 4. If $r^*(l, k)q_{l+1}/\|r(l, k)\|_2 \le \epsilon$, a breakdown occurs. Then we choose a new-start vector p_{l+1} by a procedure *newstart* (see §5). Because of the new-start process, k, k', and k'' will be increased by 1 after the next q's have been formed. This is done by setting an increment $\delta_k = 1$ and add it to k later at 6).

Remark 5. It can be easily checked that $r^*(l+1, k+1) = r^*(l, k) - \gamma_{l+1}^{(k)} p_{l+1}^*$. If a breakdown occurs at step l, then $r(l, k)$ formed there is not used, but it can be used at the next step in forming $r(l+1, k+1)$ to save some computations.

Remark 6. As in the Lanczos process, various normalizations exist, which, however, are theoretically equivalent. We have chosen to normalize the vectors so that $\|q_i\| = 1$ and $p_i^* q_i = 1$ for the sake of theoretical simplicity.

From Remark 1, we have the following

Lemma 3.2. *For $l \geq 2$, there holds*

$$k'(l) \leq k(l) \leq k'(l) + 1 \leq l \quad \textit{and} \quad k'(l+1) \leq k(l).$$

We also note that, at each step, $k(l)$ and $k'(l)$ can increase at most by 1, i.e., $k(l+1) - k(l) \leq 1$ and $k'(l+1) - k'(l) \leq 1$. This immediately leads to the following property.

Lemma 3.3. *If $i \geq l$, then $k(i) - k(l) \leq i - l$ and $k'(i) - k'(l) \leq i - l$.*

The definition of $r(l, i)$ depends on $\gamma_{l-i+j+1}^{(j)}$ for $1 \leq j \leq i-1$. The next lemma shows that $r(l, i)$ is well defined for $1 \leq i \leq k(l)$ at step l.

Lemma 3.4. *At step l of Algorithm 3.1, $\gamma_{l-i+j+1}^{(j)}$ is defined for $1 \leq i \leq k(l)$ and $1 \leq j \leq i-1$.*

Proof. First, $l \geq l - i + j + 1 \geq 2$. By Lemma 3.3, $k(l) - k(l-i+j+1) \leq i - j - 1$. So, $k(l-i+j+1) \geq k(l) - i + j + 1 \geq j + 1$. Hence, $j \leq k'(l-i+j+1)$ by Lemma 3.2, and $\gamma_{l-i+j+1}^{(j)}$ is defined at step $(l-i+j+1)$ of the algorithm. □

Let T_m be the $m \times m$ matrix whose lth column for $1 \leq l \leq m$ is

$$(0, \dots, 0, \gamma_l^{(k'(l))}, \dots, \gamma_l^{(1)}, \alpha_l, \beta_{l+1}, 0, \dots, 0)^T,$$

where α_l is in position l. Then $\gamma_l^{(k'(l))}$ is in position $l - k'(l)$ and

$$(7) \quad T_m = \begin{pmatrix}
\alpha_1 & \gamma_2^{(1)} & 0 & \cdots & 0 & & & & & \\
\beta_2 & \alpha_2 & \ddots & \ddots & & \ddots & & & & \\
 & \ddots & \ddots & \gamma_j^{(1)} & 0 & \cdots & 0 & & & \\
 & & \beta_j & \alpha_j & \gamma_{j+1}^{(1)} & \gamma_{j+2}^{(2)} & \cdots & 0 & & \\
 & & & \beta_{j+1} & \ddots & \ddots & \ddots & & \ddots & \\
 & & & & \ddots & \alpha_i & \gamma_{i+1}^{(1)} & \gamma_{i+2}^{(2)} & \cdots & \gamma_m^{(k')} \\
 & & & & & \beta_{i+1} & \alpha_{i+1} & \gamma_{i+2}^{(1)} & \ddots & \vdots \\
 & & & & & & \beta_{i+2} & \alpha_{i+2} & \ddots & \gamma_m^{(2)} \\
 & & & & & & & \ddots & \ddots & \gamma_m^{(1)} \\
 & & & & & & & & \beta_m & \alpha_m
\end{pmatrix},$$

where we note that by Lemma 3.3

$$l + 1 - k'(l+1) \geq l - k'(l).$$

Because the first nonzero entry of the lth column is in position $l-k'(l)$, it is easily seen that if $1 \le i < m-k'(m)$, the ith row of T_m is

$$(0, \dots, 0, \beta_i, \alpha_i, \gamma_{i+1}^{(1)}, \gamma_{i+2}^{(2)}, \dots, \gamma_l^{(l-i)}, 0, \dots, 0), \tag{8}$$

where l is the integer satisfying $l-k'(l) \le i < l+1-k'(l+1)$, and if $m-k'(m) \le i \le m$, the ith row of T_m is

$$(0, \dots, 0, \beta_i, \alpha_i, \gamma_{i+1}^{(1)}, \gamma_{i+2}^{(2)}, \dots, \gamma_m^{(m-i)}). \tag{9}$$

In both (8) and (9), α_i is in position i.

The next theorem summarizes the results of Algorithm 3.1. Recall that $e_{i,m}$ is the ith column of I_m.

Theorem 3.5. *Algorithm* 3.1 *produces biorthogonal sequences* $p_1, p_2, \dots, p_m$ *and* $q_1, q_2, \dots, q_m$, *i.e.,* $p_i^* q_j = \delta_{ij}$ $(1 \le i, j \le m)$ *and* $\|q_i\| = 1$, *and a matrix* T_m *of* (7), *such that*

$$AQ_m = Q_m T_m + s(m, k')e_{m,m}^* \tag{10}$$

and

$$\begin{aligned} P_m^* A = T_m P_m^* &+ e_{m-k+1,m} r^*(m, k) + e_{m-k+2,m} r^*(m, k-1) \\ &+ \cdots + e_{m,m} r^*(m, 1), \end{aligned} \tag{11}$$

where $k' = k'(m)$, $k = k(m)$ *and* $P_m = [p_1, p_2, \dots, p_m]$, $Q_m = [q_1, q_2, \dots, q_m]$.

Furthermore,

$$P_m^* s(m, k') = Q_m^* r(m, k) = \cdots = Q_m^* r(m, 1) = 0. \tag{12}$$

Proof. For $1 \le l \le m$, by the definition of $s(l, k)$ (see (5)),

$$Aq_l = s(l, k'(l)) + \alpha_l q_l + \sum_{j=1}^{k'} \gamma_l^{(j)} q_{l-j}. \tag{13}$$

If $l \le m-1$, then $s(l, k'(l)) = \beta_{l+1} q_{l+1}$, and hence

$$Aq_l = \beta_{l+1} q_{l+1} + \alpha_l q_l + \sum_{j=1}^{k'} \gamma_l^{(j)} q_{l-j},$$

which together with (13) yields (10).

In defining p_{l+1}, as well as $k'(l+1)$ and $k(l+1)$ at step l, there are three cases.

Case 1. $r(l, j) = 0$ for $i < j \le k(l)$ and $r(l, i) \ne 0$ for some i ($1 \le i \le k(l)$) with

$$r^*(l, i) q_{l+1} / \|r(l, i)\| > \epsilon.$$

In this case, no breakdown is encountered and $p_{l+1} = r(l, i)/r^*(l, i) q_{l+1}$. Then

$$k(l+1) = k'(l+1) = i \le k(l).$$

Let

$$\hat{\gamma}_{l+1}^{(k(l+1))} = r^*(l, k(l+1)) q_{l+1}.$$

Hence, for $(i =)\ k(l+1) \leq j \leq k(l)$,

$$\begin{aligned} p_{l-j+1}^* A &= \beta_{l-j+1} p_{l-j}^* + \alpha_{l-j+1} p_{l-j+1}^* + \cdots + \gamma_l^{(j-1)} p_l^* + r^*(l, j) \\ &= \beta_{l-j+1} p_{l-j}^* + \cdots + \gamma_l^{(j-1)} p_l^* + \begin{cases} \hat{\gamma}_{l+1}^{(k(l+1))} p_{l+1}^* & \text{if } j = k(l+1), \\ 0 & \text{if } j > k(l+1). \end{cases} \end{aligned} \tag{14}$$

Case 2. $r(l, j) = 0$ for $i < j \leq k(l)$ and $r(l, i) \neq 0$ for some i $(1 \leq i < k(l))$ with

$$r^*(l, i) q_{l+1} / \|r(l, i)\| \leq \epsilon.$$

In this case, a breakdown is encountered and p_{l+1} is formed by *newstart*. As it is assumed that $i < k(l)$, we have

$$k(l+1) = k'(l+1) + 1 = i + 1 \leq k(l).$$

So, for $k(l+1) \leq j \leq k(l)$, we have $r(l, j) = 0$, i.e.,

$$p_{l-j+1}^* A = \beta_{l-j+1} p_{l-j}^* + \alpha_{l-j+1} p_{l-j+1}^* + \cdots + \gamma_l^{(j-1)} p_l^*. \tag{15}$$

Case 3. $r(l, k(l)) \neq 0$ with $r^*(l, k(l)) q_{l+1} / \|r(l, k(l))\| \leq \epsilon$. Again, in this case, a breakdown is encountered and p_{l+1} is formed by *newstart*. Then

$$k(l+1) = k'(l+1) + 1 = k(l) + 1.$$

Interestingly, we have

$$l - j + 1 = \begin{cases} (l-1) - k(l) + 1 & \text{if } j = k(l+1), \\ (l+1) - k(l+1) + 1 & \text{if } j = k(l). \end{cases} \tag{16}$$

Now, it can be easily seen that

$$\{1, 2, \ldots, m - k(m)\} \subseteq \bigcup_{1 \leq l \leq m-1} \{l - k(l) + 1, \ldots, l - k(l+1) + 1\}.$$

Because of (16), the union can be restricted to those l in cases 1 and 2, i.e., those satisfying $k(l+1) \leq k(l)$. So, for any i with $1 \leq i \leq m - k(m)$, there are l, j with $1 \leq l \leq m-1$, $k(l+1) \leq j \leq k(l)$ such that $i = l - j + 1$. Then $p_i^* A$ can be written as in (14) or (15). If it is (14), then either $l - k'(l) \leq i < l + 1 - k'(l+1)$ (when $j > k(l+1)$) or $l + 1 - k'(l+1) \leq i < l + 2 - k'(l+2)$ (when $j = k(l+1)$). If it is (15), then $l - k'(l) \leq i < l + 1 - k'(l+1)$. A comparison with (8) shows that we have obtained the ith row of (11) with $\hat{\gamma}_{l+1}^{(k'(l+1))}$ possibly replaced by $\gamma_{l+1}^{(k'(l+1))}$. We will show $\hat{\gamma}_{l+1}^{(k'(l+1))} = \gamma_{l+1}^{(k'(l+1))}$, and therefore the first $m - k(m)$ rows of (11) hold. Furthermore, for $m - k(m) + 1 \leq i \leq m$, by (6),

$$p_i^* A = \alpha_i p_i^* + \cdots + \gamma_m^{(m-i)} p_m^* + r^*(m, m - i + 1).$$

This verifies the last $k(m)$ rows of (11) and thus (11).

Next, we show the biorthogonality and (12). First we prove (12) by assuming $p_i^* q_j = \delta_{ij}$ for $1 \leq i, j \leq m$. For $1 \leq i \leq m - k'(m) - 1 \leq m - k(m)$, one has $i = l - j + 1$ with $1 \leq l \leq m-1$, $k(l+1) \leq j \leq k(l)$. If $l \leq m-2$, then

$$p_i^* s(m, k'(m)) = p_i^* A q_m = p_{l-j+1}^* A q_m = 0,$$

where we use (14) and (15). If $l = m-1$, then $m - j = i \leq m - k'(m) - 1$, i.e., $k'(m) + 1 \leq j \leq k(m-1)$. So $p_{l-j+1}^* A$ is given either by (14), where

$k(m) = k'(m) < j$, or by (15), both of which lead to $p_{l-j+1}^* A q_m = 0$. Hence, $p_i^* s(m, k'(m)) = 0$. For i with $m - k'(m) \leq i \leq m$, it follows from the definition of α_m and $\gamma_m^{(i)}$ that $p_i^* s(m, k'(m)) = 0$. So we have shown that $P_m^* s(m, k'(m)) = 0$.

On the other hand, let $1 \leq i \leq k(m)$. For $1 \leq j \leq m - i - 1$,

$$r^*(m, i) q_j = p_{m-i+1}^* A q_j = 0,$$

and for $j = m - i$,

$$r^*(m, i) q_j = p_{m-i+1}^* A q_{m-i} - \beta_{m-i+1} = \beta_{m-i+1} - \beta_{m-i+1} = 0,$$

where we note that Aq_j is given by (13). For $j = m - i + 1$,

$$r^*(m, i) q_j = p_{m-i+1}^* A q_{m-i+1} - \alpha_{m-i+1} = 0.$$

For $m \geq j \geq m - i + 2$, we have $j - m + i - 1 \geq 1$. Also $k(m) - k(j-1) \leq m - j + 1$. We have $j - k(j-1) \leq m - i + 1$, and hence

$$r^*(m, i) q_j = p_{m-i+1}^* A q_j - \gamma_j^{(j-m+i-1)} = \gamma_j^{(j-m+i-1)} - \gamma_j^{(j-m+i-1)} = 0.$$

In summary, $r^*(m, k) Q_m = 0$ for $1 \leq i \leq k(m)$.

Now, p_{m+1} (q_{m+1}, resp.) is obtained from the normalization of $r(m, i)$ or from *newstart* (from the normalization of $s(m, k')$, resp.). Then $p_{m+1}^* q_j = p_j^* q_{m+1} = 0$ for $1 \leq j \leq m$. An induction argument yields the biorthogonality and hence (12).

Finally, we have for $k' = k'(l+1)$

$$\hat{\gamma}_{l+1}^{(k')} = s^*(l, k') q_{l+1} = p_{l-k'+1}^* A q_{l+1} = \gamma_{l+1}^{(k')}.$$

This completes the proof. □

An immediate consequence of the theorem justifies the claims of Remark 3.

Corollary 3.6. *If* $s(m, k'(m)) = 0$, *then* $\operatorname{span}\{q_1, \ldots, q_m\}$ *is a right invariant subspace. If* $r(m, k(m)) = \cdots = r(m, 1) = 0$, *then* $\operatorname{span}\{p_1, \ldots, p_m\}$ *is a left invariant subspace.*

Proof. If $s(m, k'(m)) = 0$, then $AQ_m = Q_m T_m$ by (10), which shows that span $\{q_1, \ldots, q_m\}$ is a left invariant subspace. The rest is proved similarly. □

4. Projection matrix

The algorithm presented in the previous section constructs biorthogonal bases and, at the same time, a banded upper Hessenberg matrix T_m in the form of (7). We note that the first nonzero entry of the lth column of T_m is in position $l - k'(l)$. By Lemma 3.3, $l - k'(l) \leq i - k'(i)$ for $i \geq l$. Thus, T_m can also be characterized as the Hessenberg matrix having the following structure:

$$(17) \qquad T_m = \begin{array}{c} \begin{array}{cc} l & m-l \end{array} \\ \begin{pmatrix} T_{11} & 0 \\ T_{21} & T_{22} \end{pmatrix} \begin{array}{l} l-k'(l+1) \\ m-l+k'(l+1) \end{array} \end{array}.$$

The matrix T_m is indeed a skew projection of A onto the column subspaces of P_m and Q_m.

Theorem 4.1. *Under the hypotheses and notation of Theorem* 3.5,

$$T_m = P_m^* A Q_m. \tag{18}$$

In particular, $P_n^* = Q_n^{-1}$ *and*

$$T_n = Q_n^{-1} A Q_n. \tag{19}$$

Proof. By the biorthogonality, $P_m^* Q_m = I_m$. Then by (12) of Theorem 3.5,

$$P_m^* A Q_m = P_m^* Q_m T_m + P_m^* s(m, k(m)) = T_m.$$

In particular, when $m = n$, then $P_n^* Q_n = I_n$, so $P_n^* = Q_n^{-1}$ and hence (19) holds. □

If the algorithm is carried out to the full n steps, we obtain a similarity reduction of A to the condensed form T_n, which is somewhere between the tridiagonal form of the nonsymmetric Lanczos algorithm and the upper Hessenberg form of the Arnoldi algorithm. Without restricting ourselves to orthogonal bases, we achieve a more condensed form than the Hessenberg form. This is particularly important for the convergence behavior (see §7). On the other hand, relaxing the tridiagonal form of the Lanczos algorithm, we gain control over the magnitude of the pivots and hence the quality of the bases.

We point out that a method can be developed in a straightforward manner to reduce A to T_n of form (7) through a sequence of elementary similar transformations. Again, by relaxing the condensed form, the magnitude of pivots can be controlled. Some discussions on the similarity reduction can be found in [13, 17].

An eigenvalue θ of T_m is called a *Ritz value.* If u, v are left and right eigenvectors of T_m, respectively, i.e.,

$$u^* T_m = \theta u^*, \qquad T_m v = \theta v,$$

then

$$x = P_m u \quad \text{and} \quad y = Q_m v \tag{20}$$

are called *left and right Ritz vectors,* respectively. In particular, (θ, u^*, v) is called a *Ritz triple.*

Ritz values can be used as an approximation to the eigenvalues of A. An interesting point of the algorithms of this type is that good approximations to eigenvalues can be obtained from Ritz values for $m << n$. This makes it a powerful method for large-scale problems. An analysis of convergence will be presented in §7. Here, we discuss the computations of Ritz values.

An important issue in our method is the efficient computation of eigentriples of the projection matrix. Since T_m is an upper Hessenberg matrix, the QR algorithm can be used to compute the eigenvalues and eigenvectors of T_m. However, one step of QR iteration will destroy the sparseness structure of T_m. Therefore, the QR algorithm may not be the best choice. An alternative to the QR algorithm is the LR algorithm (cf. Chapter 8 of [28, §8.3]). A typical single-shift step of the LR algorithm is to compute the LU decomposition of $T_m - \sigma I$ and form $UL + \sigma I$ as the next iteration matrix (this can usually be performed by a so-called "bulge chasing" process). The next theorem shows that the structure of T_m (i.e., the envelope of T_m) is invariant under the LR algorithm.

Theorem 4.2. *Let T_m be a matrix of the form* (7) *and $T_m - \sigma I = LU$ be an LU decomposition with some shift σ. Then $UL + \sigma I$ has the same structure* (7) *as T_m.*

Proof. The matrix $T_m - \sigma I$ has the same structure as T_m. Then L is a lower bidiagonal matrix, i.e., $l_{ij} = 0$ if $j > i$ or $j < i - 1$. By comparing the rows of $T_m - \sigma I$ and LU from the first to the last, we see that U has the same structure as T_m. Multiplying U on the right by L is to add a multiple of the $(j+1)$st column to the jth column, giving a form of T_m. Clearly, $UL + \sigma I$ yields the same structure as T_m. □

The theorem shows that the LR algorithm preserves the sparseness structure of T_m and is potentially an efficient method for T_m. It is well known, however, that the LR algorithm can break down for certain shifts. Even so, it has been suggested that the LR algorithm is a competitive alternative to the QR algorithm for tridiagonal matrices (see [17], for example), and we expect the same in our context.

5. New-start procedure

One of the key steps in our algorithm is the new-start procedure at 9), i.e., the choice of a vector for p_{l+1} such that $p_{l+1}^* q_i = \delta_{l+1,i}$ $(1 \le i \le l+1)$ as well as $|\omega_l| = |\cos\langle p_{l+1}, q_{l+1}\rangle| > \epsilon$. Clearly, if $\epsilon = 0$, the choice always exists, and we can always conquer the exact breakdown. For $\epsilon > 0$, however, the choice is not always possible. Indeed, the best pivot is given in the next theorem. Here we remark that an extreme case is $\epsilon > 1$, for which the choice never exists.

Theorem 5.1. *Let $q_1, \dots, q_l, q_{l+1}$ be linearly independent vectors of norm* 1, *and let*

$$S_l = \operatorname{span}\{q_1, \dots, q_l\}.$$

If $q_{l+1} = u + v$, where $u \in S_l$ and $v \in S_l^\perp$, then

$$\max_{x \in S_l^\perp} |\cos\langle x, q_{l+1}\rangle| = \|v\|.$$

Proof. For any $x \in S_l^\perp$, we have

$$|\cos\langle x, q_{l+1}\rangle| = \frac{|x^* q_{l+1}|}{\|x\|} = \frac{|x^* v|}{\|x\|},$$

which is maximized when $x = v$. □

The best possible pivot at step l is $\|v\|$ and finding v requires forming and solving an $l \times l$ linear system. Note that $\|v\|$ is indeed the sine of the angle between q_{l+1} and the subspace S_l. Therefore, a small $\|v\|$ suggests that q_{l+1} is nearly linearly dependent of $q_1, \dots, q_l$, and also S_l is close to a right invariant subspace. The precise measure of the invariance of S_l, however, depends on $\|s(l, k)\|$ as well.

Theorem 5.2. *Under the hypotheses and notation of Theorems* 3.5 *and* 5.1, *we have*

$$\min_T \|AQ_l - Q_lT\| = \|s(l, k')\| \, \|v\|,$$

where T *is an* $l \times l$ *matrix.*

Proof. By Theorem 3.5,

$$AQ_l - Q_lT = Q_l(T_l - T) + s(l, k')e_{l,l}^*.$$

Now $s(l, k') = \|s(l, k')\|q_{l+1} = \|s(l, k')\|u + \|s(l, k')\|v$. Therefore, the last column of $AQ_l - Q_lT$ is $\hat{u} + \|s(l, k')\|v$, where $\hat{u} \in S_l$ and $\|s(l, k')\|v \in S_l^\perp$. Hence,

$$\|AQ_l - Q_lT\| \geq \|\hat{u} + \|s(l, k')\|v\| \geq \|s(l, k')\| \, \|v\|.$$

Clearly, equality is achieved by $T = T_l + R_1$ with $\|s(l, k)\|ue_{l,l}^* = Q_lR_1$. The proof is complete. □

The theorem shows that $\|s(l, k')\| \, \|v\|$ measures the invariance of S_l. Yet how it bounds the eigenvalue approximation is not clear in the nonsymmetric case. Also, from the proof, we see that the minimum over T may not be achieved at T_l.

5.1. **Random new-start.** A natural way to choose a new-start vector other than finding the vector v is to pick up a random vector x and then orthogonalize it against $q_1, q_2, \ldots, q_l$. In view of the existence of the dual biorthogonal basis $p_1, p_2, \ldots, p_l$, this can be achieved very conveniently by

$$r = x - \alpha_1 p_1 - \cdots - \alpha_l p_l,$$

where $\alpha_i = q_i^* x$ for $1 \leq i \leq l$. If $|\cos\langle r, q_{l+1}\rangle| > \epsilon$, normalizing r yields p_{l+1}. Otherwise, we try another random vector. As we observed, there may not exist a vector satisfying the given threshold condition. So after a certain number of unsuccessful trials, we decrease ϵ. The following is a basic algorithm of the subroutine *newstart*:

Algorithm 5.3. For $J = 1, 2, \ldots, J_{\max}$, do

```
1)   generate a random vector $x$;
2)   for $i = 1, 2, \ldots, l$ do
         $x = x - (q_i^* x)p_i$;
3)   if $|\cos\langle x, q_{l+1}\rangle| \geq \epsilon$, then $p_{l+1} = x/x^* q_{l+1}$ and exit.
```

Of course, more sophisticated devices can be employed in place of 2) to deal with some difficulties associated with the naive implementation (see [15, §5.2], for example).

The new-start procedure requires the storage of all preceding vectors p_i, q_i. This cost of storage is critical when the size of the matrix is large. Unlike the Arnoldi algorithm, however, this difficulty occurs only during the new-start procedure and can be overcome by storing some admissible new-start vectors in advance. For example, we can compute some admissible new-start vectors simultaneously with the iteration by choosing a set of random vectors at the beginning and orthogonalizing them against q_l whenever a new q_l is formed. Then, when a breakdown occurs, the vectors in the subspace spanned by these vectors are candidates for new-start vectors. In particular, random linear combinations of the vectors can be used as trials.

A method presented in §5.3 will enable us to increase the number of admissible vectors. Then the number of vectors to be stored in advance need not be large.

5.2. **New-start without reorthogonalizaion.** A probably better way to avoid the cost of storage is not to do orthogonalization in the new-start phase.

It is well known that, for all Lanczos-type methods, the orthogonality will be lost after certain iterations in finite arithmetic (see [28, §6.32]). Usually, the methods can be implemented with or without reorthogonalization (cf. [21, Ch. 13] and [6, Ch. 4]). The same situation applies to Algorithm 3.1. When a reorthogonalization is adopted, then the new-start procedure is just a step of rebiorthogonalization and does not incur any extra cost. On the other hand, when no reorthogonalization is used, we propose to do the orthogonalization in the new-start procedure only for local vectors, because the biorthogonality among the vectors are lost anyway. In this implementation, a new-start based on random vectors often leads to a small pivot, since only a few vectors are orthogonalized against. We suggest to start from q_j rather than a random vector and then orthogonalize it against local vectors.

The theoretical basis for the implementation without reorthogonalization is that, even when the orthogonality is lost, equations (11) and (10) remain valid (see [25] or [21, §13–4]). It is easy to see that equations (11) and (10) still hold if the orthogonalization is not adopted in the new-start procedure. This again justifies the implementation of the new-start without orthogonalization. Finally, we point out that the numerical examples in §7 show that this is indeed a practically effective way to implement our algorithm.

5.3. **New-start subspaces.** The methods presented in §5.1 are based on orthogonalization, which leads to the problem of storage space. Although §5.1 suggests that this can be overcome by storing some candidate vectors, it is difficult to decide on the number of such vectors required at the beginning.

When there is a new-start vector on hand, more admissible vectors can be obtained from the Krylov subspace generated by the vector. The choice of the Krylov subspace is shown in the next theorem. Note that this property is also used in LAL [23].

Theorem 5.4. *Let* r_0 *be a vector orthogonal to* $S_l = \mathrm{span}\{q_1, \dots, q_l\}$, *where* $q_1, q_2, \dots, q_l$ *are generated by Algorithm* 3.1. *Let*

$$r_{i+1}^* = r_i^* A - \zeta_i p_l^* \quad \textit{for } i = 0, 1, 2, \dots,$$

where $\zeta_i = r_i^* A q_l$. *Then* r_i *is orthogonal to* S_l *for* $i = 0, 1, 2, \dots$.

Proof. We prove this by induction. Assume $r_i \perp S_l$ for some i. Then, for $1 \le j \le l-1$, we have $Aq_j \in S_l$ by Theorem 3.5. Hence,

$$r_{i+1}^* q_j = r_i^* A q_j = 0.$$

Also, $r_{i+1}^* q_l = r_i^* A q_l - \zeta_i = 0$. Thus, $r_{i+1} \perp S_l$. The proof is complete. □

Note that the computation of r_i does not require storage of q_i. So, when we have one candidate r_0 of the new-start vector, any vectors in $\mathrm{span}\{r_0, r_1, \dots\}$

can be used as candidates. This significantly increases the number of vectors available. For example, in the strategy of §5.1, we only need to store a few vectors, and when a breakdown occurs, more vectors can be constructed in this way.

In particular, an admissible new-start vector at step l is $r(l, k)$ $(\perp S_l)$. Then a sequence of admissible vectors r_i can be generated from $r_0 = r(l, k)$. The new-start vector can be chosen to be a random linear combination of $r_0, r_1, \ldots$ and choosing r_1 leads to LAL (see the next section).

6. Some special cases

There are a few special cases of Algorithm 3.1 that are of particular interest. For example, if the new-start vector is chosen in a special way, Algorithm 3.1 yields the look-ahead Lanczos algorithm of [23]. On the other extreme, if $\epsilon = 1$ and $p_1 = q_1$, then the two sequences of the basis vectors are necessarily the same, and we recover the Arnoldi algorithm. We discuss symmetric matrix pencils as a special nonsymmetric problem and a symmetric version of the algorithm involving two-sided new-start will be described. Also, we will present some comparisons between the new algorithm and LAL in this section.

6.1. **Look-ahead Lanczos algorithm.** Assume the regular Lanczos algorithm breaks down at step j (see §2.1). Let $r_0 = r(j, 1)$ and $r_1^* = r^*(j, 1)A - \zeta_1 p_j^*$ as in §5.3. Now, if $r_1^* q_{j+1} \neq 0$, then r_1 can be chosen as the new-start vector and $p_{j+1} = r_1 / r_1^* q_{j+1}$ as is done in [23].

We now proceed with the construction of the subsequent vectors. The vector q_{j+2} is generated as usual by normalizing $s(j+1, 1)$, but with $k(j+1) = 2$; the vector p_{j+2} is obtained by

$$\gamma_{j+2}^{(2)} p_{j+2}^* = r^*(j+1, 2) = p_j^*(A - \alpha_j) - \beta_j p_{j-1}^* - \gamma_{j+1} p_{j+1}^* = r_0^* - \gamma_{j+1} p_{j+1}^*.$$

Then p_{j+3} is determined by $r(j+2, 2)$ and q_{j+3} by $s(j+2, 2)$. Furthermore, it is easily seen that

$$r(j+3, 2) \in \operatorname{span}\{p_1, \ldots, p_{j+3}\}$$

and, by Theorem 3.5,

$$r(j+3, 2) \perp \operatorname{span}\{q_1, \ldots, q_{j+3}\}.$$

Using biorthogonality, we have $r(j+3, 2) = 0$ and k is therefore decreased to one. Hence, the subsequent p_l for $l \geq j+4$ are formed by $r(l-1, 1)$, i.e., through the regular Lanczos iteration. A direct comparison with [23] shows that what we obtained is exactly the 2×2 block case of the look-ahead Lanczos algorithm.

The more general version of LAL uses a factorization of a $t \times t$ block pivot and is also a special case of Algorithm 3.1. To be more specific, assume that the LU decomposition with pivoting is used in constructing $p_{j+1}, \ldots, p_{j+t}$ and $q_{j+1}, \ldots, q_{j+t}$ in LAL. Then

$$p_{j+1}, \ldots, p_{j+t} \in \hat{K}_{j+t}(p_1) \text{ and } q_i \in K_i(q_1) \quad \text{for } j+1 \leq i \leq j+t.$$

If we apply Algorithm 3.1 with $p_{j+1}, \ldots, p_{j+t}$ as t consecutive new-start vectors, then we obtain the same $q_{j+1}, \ldots, q_{j+t}$, since q_i $(j+1 \leq i \leq j+t)$ is in

$K_i(q_1)$ and orthogonal to $p_1, \ldots, p_{i-1}$. Furthermore,

$$r(j+t+1, t), \ldots, r(j+t+1, 2) \in \hat{K}_{j+t+1}(p_1) = \text{span}\{p_1, \ldots, p_{j+t+1}\}$$

and

$$r(j+t+1, t), \ldots, r(j+t+1, 2) \perp \text{span}\{q_1, \ldots, q_{j+t+1}\}.$$

Hence, $r(j+t+1, t) = \cdots = r(j+t+1, 2) = 0$. Thus, k is decreased to 1 and subsequent p_l, q_l (for $l \geq j+t+2$) are formed by $r(l-1, 1)$ and $s(l-1, 1)$, i.e., by the regular Lanczos recurrence.

6.2. **Arnoldi algorithm.** The Arnoldi algorithm constructs an orthonormal basis and can be regarded as enforcing all the pivots to be one. If we impose that all pivots in Algorithm 3.1 are one, i.e., $\omega_i = 1$, then p_i must be a multiple of q_i, which is not the case in general. So the algorithm will break down at every step and the new-start must be adopted to meet the criterion. The most convenient choice is $p_i = q_i$. Then $\omega_i = 1$ and $k'(l) = l-1$. Since p_i is always replaced by q_i, the iteration for p_i need not be carried out. The iteration for q_i reads

$$\begin{aligned}\beta_{l+1}q_{l+1} = s(l, k') &= (A-\alpha_l)q_l - \gamma_l^{(1)}q_{l-1} - \cdots - \gamma_l^{(k')}q_{l-k'} \\ &= (A-\alpha_l)q_l - \gamma_l^{(1)}q_{l-1} - \cdots - \gamma_l^{(l-1)}q_1,\end{aligned}$$

in agreement with the Arnoldi algorithm. Note that the orthogonality among the q's follows from the biorthogonality, and T_m becomes an upper Hessenberg matrix.

6.3. **Symmetric pencils and two-sided new-start.** In Algorithm 3.1, we adopt a new-start strategy for the left vector p_l. The algorithm can be generalized further to allow new-start on both sides, i.e., for both p_l and q_l. In that case, the formulae for $s(l, k)$ and $r(l, k)$ will be replaced by formulas combining the two. However, there is no obvious reason, in general, for doing this, since the projection matrix will not be in the upper Hessenberg form. On the other hand, the idea of two-sided new-start is natural in an effort to preserve symmetry for symmetric pencil problems, including the classical symmetric problem.

An important class of the eigenvalue problems is the symmetric pencil problem

$$Ax = \lambda Bx, \tag{21}$$

where A and B are Hermitian matrices with B nonsingular and (λ, x) is an eigenpair. The eigenvalue problem (21) is equivalent to the problem for $B^{-1}A$ and is essentially nonsymmetric if neither A nor B is definite (see [29]). When we consider a numerical method for (21), it is important to exploit the symmetric structure.

The Lanczos algorithm has been applied to $B^{-1}A$ in a symmetric fashion. By letting $p_1 = Bq_1$, we have $p_l = Bq_l$ for all l; so the two sequences of vectors are reduced to one, and a symmetric compression pencil is obtained (see [20, 24, 29]). If B is indefinite, this symmetric version of the algorithm inherits the breakdown, i.e., when $q_j^*Bq_j = p_j^*q_j = 0$. To overcome this difficulty, Algorithm 3.1 can be applied to $B^{-1}A$, but the new-start strategy is used only for p_j and not for q_j. Immediately, the symmetric relation $p_l = Bq_l$ is lost.

In order to maintain symmetry, therefore, it is necessary to apply the new-start to both p_j and q_j, and this can be carried out implicitly.

Now assume that the algorithm breaks down at step j, i.e., $\omega_j = q_j^* B q_j \le \epsilon$. We choose a new-start vector for q_j and, simultaneously, $p_j = Bq_j$, which is, of course, not explicitly formed. Then q_{j+1} and q_{j+2} are constructed according to

$$\gamma_j^{(2)} q_{j+1} = B^{-1}Aq_{j-1} - \bar{\gamma}_{j-1}^{(1)} q_{j-2} - \alpha_{j-1} q_{j-1} - \gamma_j^{(1)} q_j$$

and

$$\gamma_{j+1}^{(2)} q_{j+2} = B^{-1}Aq_j - \bar{\gamma}_j^{(1)} q_{j-1} - \alpha_j q_j - \gamma_{j+1}^{(1)} q_{j+1},$$

where

$$\alpha_j = \frac{q_j^* A q_j}{q_j^* B q_j}, \quad \gamma_j^{(1)} = \frac{q_j^* A q_{j-1}}{q_j^* B q_j} \quad \text{and} \quad \gamma_{j+1}^{(1)} = \frac{q_{j+1}^* A q_j}{q_{j+1}^* B q_{j+1}}.$$

It can be verified that q_{j+1} and q_{j+2} satisfy the orthogonality. Then, q_{l+1} for $l \ge j+2$ is generated by

$$\gamma_l^{(2)} q_{l+1} = B^{-1}Aq_{l-1} - \bar{\gamma}_{l-2}^{(2)} q_{l-3} - \bar{\gamma}_{l-1}^{(1)} q_{l-2} - \alpha_{l-1} q_{l-1} - \gamma_l^{(1)} q_l .$$

Note that the last recurrence is a combination of (6) and (5) and involves five terms. The recurrence produces a B-orthogonal basis $\{q_1, \dots, q_m\}$ and a symmetric pencil in a condensed form. When a further breakdown occurs, the same technique can be applied and a general algorithm can be derived. All other discussions are similar to those for Algorithm 3.1 and are therefore omitted here.

If B is positive definite or equal to I (i.e., the classical case), there is no difficulty of breakdown. Still we can apply this algorithm at some step j to introduce a new-start vector. The benefit of doing this is that the Krylov subspace is changed to a sum of two, which may be necessary in some cases.

6.4. **Comparisons.** We have seen in this section that Algorithm 3.1 is closely related to some existing algorithms. In particular, we make some comparisons between Algorithm 3.1 and LAL here. The fundamental difference seems to be that Algorithm 3.1 introduces a new-start vector and hence changes the old Krylov subspace, while LAL does not. This difference appears in the following three aspects.

First, LAL may encounter incurable breakdown, which is due to mismatch of the two Krylov subspaces. Algorithm 3.1 overcomes this problem by correcting an improper Krylov subspace. Although, theoretically, all Ritz values are eigenvalues in the case of an incurable breakdown, no eigenvector is obtained. This makes it difficult to use the information obtained to find the rest of the eigenvalues. Also, there is no easy way to detect an incurable breakdown numerically. Second, if the Lanczos algorithm is to be terminated at some step m ($< n$) as is usually the case, then LAL is confined to $\hat{K}_m(p_1)$ and may not be able to resolve a curable breakdown within step m. In contrast, Algorithm 3.1 can freely choose a new-start vector from C^n (or $\hat{K}_n(p_1)$). Finally, the symmetric version of Algorithm 3.1 is new and of significance also in the classical symmetric case.

7. Convergence bounds

In this section, we present some theoretical analyses concerning the convergence of Ritz triples. In particular, we give generalizations of familiar results of the symmetric case, including residuals of the Ritz triples.

7.1. **Convergence analysis.**

For the nonsymmetric Lanczos algorithm, a convergence analysis was derived in [30] based on the tridiagonal structure, which generalizes the classical results for the symmetric case. The idea also applies to the structure of the projection matrix T_m of (7). We outline this analysis in this section.

Let Algorithm 3.1 be carried out to full n steps, giving rise to the matrix T_n (see Theorem 4.1). We can assume that no breakdown occurs after step m, i.e., $k'(l) \le k'(m)$ for $l \ge m$. Write

$$T_n = \begin{pmatrix} T_m & E \\ \hat{E} & \hat{T}_{n-m} \end{pmatrix} \quad \text{and} \quad E = \begin{pmatrix} 0 \\ E_0 \end{pmatrix} \begin{matrix} {\scriptstyle m-k'(m)} \\ {\scriptstyle k'(m)} \end{matrix}.$$

Let $l_0 = m - k'(m)$ and construct a strictly monotonically decreasing sequence l_i, till $l_{m_1} = 0$ for some m_1, by

$$l_i = l_{i-1} - k'(l_{i-1} + 1).$$

Obviously, m_1 is uniquely determined by the function $k'(l)$, i.e., the sparseness structure of T_m and, if T_m is tridiagonal, $m_1 = m - 1$. The following theorem shows that T_n^i and T_m^i have essentially the same first row (first column, (1,1) element) for some i.

Theorem 7.1. *Assume* $l_0 > l_1 > \cdots > l_{m_1} = 0$. *Then*

$$T_n^i e_{1,n} = \begin{pmatrix} T_m^i e_{1,m} \\ 0 \end{pmatrix} \quad \textit{for } i \le m - 1, \tag{22}$$

$$e_{1,n}^* T_n^i = (e_{1,m}^* T_m^i, 0) \quad \textit{for } i \le m_1, \tag{23}$$

and

$$e_{1,n}^* T_n^i e_{1,n} = e_{1,m}^* T_m^i e_{1,m} \quad \textit{for } i \le m + m_1. \tag{24}$$

Proof. Observe that (22) depends on the lower triangular part of T_n only and can be shown by the same proof as in [30] for the tridiagonal case. A detailed proof is omitted here. For (23), we first show that, for $1 \le i \le m_1$,

$$T_m^i E = \begin{pmatrix} 0 \\ E_i \end{pmatrix} \begin{matrix} {\scriptstyle l_i} \\ {\scriptstyle m-l_i} \end{matrix}.$$

This is obviously true for $i = 0$; assume that it is true for $i \le m_1 - 1$. Then

$$T_m^{i+1} E = T_m \begin{pmatrix} 0 \\ E_i \end{pmatrix} \begin{matrix} {\scriptstyle l_i} \\ {\scriptstyle m-l_i} \end{matrix} = \begin{pmatrix} 0 \\ E_{i+1} \end{pmatrix} \begin{matrix} {\scriptstyle l_{i+1}} \\ {\scriptstyle m-l_{i+1}} \end{matrix},$$

where T_m is partitioned according to (17) with $l = l_i$. Now clearly, $e_{1,n}^* T_n = (e_{1,m}^* T_m, 0)$, and if (23) is true for some $i < m_1$, then

$$e_{1,n}^* T_n^{i+1} = (e_{1,m}^* T_m^{i+1}, e_{1,m} T_m^i E) = (e_{1,m}^* T_m^{i+1}, 0).$$

So (23) is proved. Finally, for $i \le m_1$ and $j \le m-1$

$$e_{1,n}^* T_n^{i+j+1} e_{1,n} = e_{1,n}^* T_n^i T_n T_n^j e_{1,n} = e_{1,m}^* T_m^{i+j+1} e_{1,m},$$

where we use (23) and (22). This completes the proof of (24). □

In the following, we derive an identity concerning the Ritz values, using Theorem 7.1. For the sake of simplicity, we assume that T_m and A are diagonalizable, i.e.,

$$(25) \qquad T_m = S^*\Theta T \quad \text{and} \quad A = Z_r^* \Lambda Z_l,$$

where $\Theta = \operatorname{diag}(\theta_1, \ldots, \theta_m)$, $S^*T = I_m$ and $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$, $Z_r^* Z_l = I_n$. Let $S = (s_{ij})$, $T = (t_{ij})$ and

$$Z_r = [z_1^{(r)}, \ldots, z_n^{(r)}] \quad \text{and} \quad Z_l = [z_1^{(l)}, \ldots, z_n^{(l)}].$$

Then $z_i^{(r)}$ ($z_i^{(l)}$, resp.) is the right (left, resp.) eigenvector of A. Letting $X = Z_r P_n = (x_{ij})$ and $Y = Z_l Q_n = (y_{ij})$, we have

$$(26) \qquad T_n = X^* \Lambda Y \quad \text{and} \quad X^* Y = I_n.$$

Note that x_{i1} is the $z_i^{(l)}$-component of the initial vector p_1, and y_{i1} is the $z_i^{(r)}$-component of the initial vector q_1, i.e.,

$$p_1 = \sum_{i=1}^n x_{i1} z_i^{(l)} \quad \text{and} \quad q_1 = \sum_{i=1}^n y_{i1} z_i^{(r)}.$$

Now substitute (25) and (26) into (24) to obtain

$$e_{1,n}^* X^* \Lambda^i Y e_{1,n} = e_{1,m}^* S^* \Theta^i T e_{1,m}$$

for any $i \le m + m_1$. Hence, for any polynomial f of degree not exceeding $m + m_1$,

$$e_{1,n}^* X^* f(\Lambda) Y e_{1,n} = e_{1,m}^* S^* f(\Theta) T e_{1,m}.$$

Thus,

$$\sum_{i=1}^n f(\lambda_i) \bar{x}_{i1} y_{i1} = \sum_{i=1}^m f(\theta_i) \bar{s}_{i1} t_{i1}.$$

In particular, using $f(x) = (x - \theta_1) h(x)$, we obtain the following theorem.

Theorem 7.2. *Assume* $|\lambda_1 - \theta_1| = \min_j |\lambda_1 - \theta_j|$. *Then for any polynomial* h *of degree not exceeding* $m + m_1 - 1$, *we have*

$$\lambda_1 - \theta_1 = \frac{1}{h(\lambda_1)\bar{x}_{11} y_{11}} \left(-\sum_{i=2}^n (\lambda_i - \theta_1) h(\lambda_i) \bar{x}_{i1} y_{i1} + \sum_{i=2}^m (\theta_i - \theta_1) h(\theta_i) \bar{s}_{i1} t_{i1} \right).$$

Some special polynomials, such as Chebyshev polynomials, can be used to give various bounds on $\lambda_1 - \theta_1$ as in [30]. Besides, bounds on Ritz vectors can also be derived from (23) and (22), as in [30].

A conclusion drawn from Theorem 7.2 is that the rate of convergence is proportional to $m + m_1 - 1$. From the definition of m_1, it is clear that m_1 is determined by the sparseness structure of T_m and hence by the number of breakdowns that occurred. This suggests that the more new-starts we encounter, the smaller is m_1 and thus the slower is convergence. While using the new-start stabilizes computational results, it slows down the convergence. This behavior is also confirmed in our numerical examples (§8). Hence, choosing an appropriate threshold parameter ϵ becomes very important in implementations.

7.2. **Residuals of Ritz triples.** Another important aspect of convergence concerns the residuals of Ritz pairs. For our algorithm, the residuals are given in the following theorem.

Theorem 7.3. *Let u and v be the right and left eigenvector corresponding to an eigenvalue θ of T_m and x and y be the left and right Ritz vectors defined in* (20). *If $u = (u_1, u_2, \dots, u_m)^T$ and $v^* = (v_1, v_2, \dots, v_m)$, then*

$$Ax - \theta x = u_m s(m, k') \tag{27}$$

and

$$\begin{aligned} y^*A - y^*\theta = {} & v_{m-k+1} r^*(m, k) \\ & + v_{m-k+2} r^*(m, k-1) + \cdots + v_m r^*(m, 1), \end{aligned} \tag{28}$$

where $k = k(m)$, $k' = k'(m)$ are defined in Algorithm 3.1.

Proof. Note that $x = P_m u$. Multiplying (11) by u, we immediately obtain (27). Multiplying (10) by v^* leads to (28). □

The theorem shows that the residuals are small if the components u_m and $v_{m-k+1}, \dots, v_m$ are small and $s(m, k')$ and $r(m, i)$ ($1 \le i \le k$) are bounded. Note that all these quantities are computable at step m, so the residuals can be computed without forming the Ritz vectors x and y. This indeed provides an *a posteriori* convergence criterion.

We remark that the proof of the theorem uses only equations (11) and (10), but not biorthogonality. When the algorithm is implemented without orthogonalization (including the new-start process), equations (11) and (10) are valid, and so are (27) and (28).

8. Numerical examples

This section is devoted to numerical examples. The algorithm is tested for various choices of the threshold parameter ϵ. The choice $\epsilon = 0$ yields the nonsymmetric Lanczos algorithm (note that no exact breakdown occurs numerically). We compare the cases $\epsilon > 0$ with the Lanczos algorithm. The interest here is in demonstrating the numerical behavior of Algorithm 3.1 rather than in efficient implementations. Of course, a careful implementation would very likely improve the performance reported here.

It is expected that biorthogonality will be lost in finite arithmetic owing to cancellation. We use a full rebiorthogonalization device in our examples, except in Example 4, where the algorithm and the new-start procedure are run both with and without orthogonalization. The new-start strategy employed is described in §§5.1 and 5.2.

As is pointed out in §4, the LR algorithm can be used to compute eigenvalues and eigenvectors of T_m. For reliability, however, we have adopted the QR algorithm here.

TABLE 1. Results of a serious breakdown in Example 8.1

	$\epsilon = 0.$		$\epsilon = 1.E-3$		$\epsilon = 1.E-1$	
i	ω_i	$\theta_i - \lambda_i$	ω_i	$\theta_i - \lambda_i$	ω_i	$\theta_i - \lambda_i$
1	1.E0	*	1.E0	5.1E-10	1.E0	-1.1E-13
2	1.3E-1	*	1.3E-1	(2.7, -1.8)E-10	1.3E-1	(-1.1, -1.8)E-13
3	-7.2E-3	*	-7.2E-3	(2.7, 1.8)E-10	(E-3)-1.5E-1	(-1.1, 1.8)E-13
4	-1.8E-15	*	(E-15)1.1E-3	(-2.4, 6.0)E-11	(E-3)-3.2E-1	(4.6, -5.0)E-14
5	-1.8E-15	*	1.6E-2	(-2.4, -6.0)E-11	(E-2)-2.5E-1	(4.6, 5.0)E-14
6	-3.1E-3	*	1.2E-3	-2.2E-12	-3.2E-1	-1.3E-13

In the first two examples, the algorithm is run to the end providing all eigenvalues. We compare the accuracy obtained for various thresholds ϵ and list the pivots ω_i and the approximation errors $\lambda_i - \theta_i$ (in Tables 1–3). If a new-start occurs, we list the pivot for the new-start vector with the magnitude of the old pivot in parentheses. The influence of the magnitudes of pivots ω_i on the approximation and the stabilizing effect of the new algorithm are evident.

Example 8.1. Our first example is taken from Example 1 of [23]. The 6×6 matrix is

$$A = \begin{pmatrix} 0 & & & & & 1 \\ 1 & 0 & & & & \\ & 1 & 0 & & & \\ & & 1 & 0 & & \\ & & & 1 & 0 & \\ & & & & 1 & 0 \end{pmatrix},$$

and the initial vectors are $p_1 = q_1 = [1, 2, 3, 4, 5, 6]^T$. The eigenvalues of A are the sixth roots of unity.

Table 1 lists the results for $\epsilon = 0, 1.E-3, 1.E-1$. The Lanczos algorithm ($\epsilon = 0$) yields a serious breakdown (a pivot of the magnitude $E-15$), no approximation being obtained (represented by * in the table). The accuracy is clearly improved as ϵ increases.

Example 8.2. The following matrix comes from Example 5.13 of [14],

$$A = \begin{pmatrix} B & 2B \\ 4B & 3B \end{pmatrix}, \qquad B = \begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & 0 & 1 & \\ & & & 0 & 1 \\ 10^{-5} & & & & 0 \end{pmatrix}.$$

The eigenvalues of A are (see [14])

$$\lambda_k = 0.5\exp(2k\pi i/5), \qquad k = 1, 2, \dots, 5,$$

$$\lambda_k = -0.1\exp(2k\pi i/5), \qquad k = 1, 2, \dots, 5.$$

In this example, the two initial vectors are chosen to be different and random and the algorithm is run to full 10 steps.

The results for two pairs of initial vectors are listed in Tables 2 and 3 for $\epsilon = 0., 5.E-2, 1.E-1$, and $\epsilon = 0., 1.E-2, 1.E-1$, respectively. Again, the accuracy is improved by increasing ϵ.

TABLE 2. Results for the first pair of random initial vectors in Example 8.2

	$\epsilon = 0.$		$\epsilon = 5.0E-2$		$\epsilon = 1.E-1$	
i	ω_i	$\theta_i - \lambda_i$	ω_i	$\theta_i - \lambda_i$	ω_i	$\theta_i - \lambda_i$
1	3.5E-1	1.6E-9	3.5E-1	1.0E-11	3.5E-1	3.4E-12
2	-2.2E-1	(1.5, 2.2)E-9	-2.2E-1	(1.4, 0.4)E-11	-2.2E-1	(3.3, 2.7)E-12
3	-1.1E-2	(1.5, -2.2)E-9	(E-2)-2.1E-1	(1.4, -0.4)E-11	(E-2)-2.1E-1	(3.3, -2.7)E-12
4	-1.0E-2	(1.3, 0.9)E-8	6.2E-2	(0.2, 1.0)E-9	(E-2)1.1E-1	(5.5, 4.3)E-10
5	2.1E-2	(1.3, -0.9)E-8	(E-2)6.9E-2	(0.2, -1.0)E-9	(E-2)-1.9E-1	(5.5, -4.3)E-10
6	-5.0E-2	(0.5, 1.4)E-8	-4.1E-2	(5.7, 4.7)E-10	(E-2)1.5E-1	(2.2, 6.2)E-10
7	-2.3E-2	(0.5, -1.4)E-8	(E-2)-2.0E-1	(5.7, -4.7)E-10	(E-2)1.3E-1	(2.2, -6.2)E-10
8	-5.4E-2	1.5E-8	7.6E-2	5.6E-10	(E-2)2.1E-1	6.5E-10
9	-4.8E-2	(2.4, 2.1)E-9	1.3E-1	(7.1, 5.6)E-12	(E-2)-2.6E-1	(1.5, 2.4)E-12
10	-2.5E-2	(2.4, -2.1)E-9	9.6E-2	(7.1, -5.6)E-12	-2.0E-1	(1.5, -2.4)E-12

TABLE 3. Results for the second pair of random initial vectors in Example 8.2

	$\epsilon = 0.$		$\epsilon = 1.E-2$		$\epsilon = 1.E-1$	
i	ω_i	$\theta_i - \lambda_i$	ω_i	$\theta_i - \lambda_i$	ω_i	$\theta_i - \lambda_i$
1	5.9E-1	4.6E-10	5.9E-1	3.2E-11	5.9E-1	2.8E-11
2	3.9E-1	(2.3, 0.6)E-9	3.9E-1	(2.1, 3.6)E-11	3.9E-1	(2.0, 3.1)E-11
3	-2.6E-1	(2.3, -0.6)E-9	-2.6E-1	(2.1, -3.6)E-11	-2.6E-1	(2.0, -3.1)E-11
4	-2.9E-1	(3.4, 1.6)E-5	-2.9E-1	(1.4, 0.7)E-10	-2.9E-1	(6.5, 3.7)E-11
5	1.0E-1	(3.4, -1.6)E-5	1.0E-1	(1.4, -0.7)E-10	1.0E-1	(6.5, -3.7)E-11
6	-3.9E-2	(0.5, 5.3)E-5	3.9E-2	(0.3, 2.0)E-10	(E-2)1.1E-1	(2.1, 8.4)E-11
7	8.8E-4	(0.5, -5.3)E-5	(E-4)6.1E-2	(0.3, -2.0)E-10	(E-2)2.2E-1	(2.1, -8.4)E-11
8	-8.6E-4	5.9E-5	-1.4E-1	2.2E-10	1.3E-1	8.6E-11
9	3.2E-1	(1.6, 1.2)E-9	3.2E-1	(3.8, 3.8)E-11	(E-2)1.3E-1	(3.4, 3.6)E-11
10	5.2E-1	(1.6, -1.2)E-9	4.6E-1	(3.8, -3.8)E-11	-1.0E-1	(3.4, -3.6)E-11

In the next two examples, the intermediate Ritz values are computed in demonstrating convergence. By §7.1, convergence is expected to be slowed down as ϵ increases, but numerical errors destroy the faster convergence predicted theoretically for smaller ϵ. In the long run, more Ritz values will be obtained by increasing ϵ, owing to the stabilizing effect.

Example 8.3. This example comes from Example 3 of [23], where the matrix is the following 100×100 diagonal matrix

$$A = \operatorname{diag}(1, 2, \dots, 20, 41, 62, \dots, 440, 481, 522, \dots, \\ 1260, \dots, 2561, 2642, \dots, 4019, 4100). \tag{29}$$

The two initial vectors are random and different. We mark a Ritz value converged if it is correct to the fifth significant digit and $\alpha(m)$ is the number of Ritz values converged at step m.

Figure 1 plots the curve of $\alpha(m)$ for a pair of random initial vectors. The threshold parameter is chosen to be $\epsilon = 0.$, $1.E-4$, $1.E-3$. Although, initially, the case $\epsilon = 0$ demonstrates faster convergence, it deteriorates as m increases, with only 19 Ritz values converged at step 100. In contrast, both

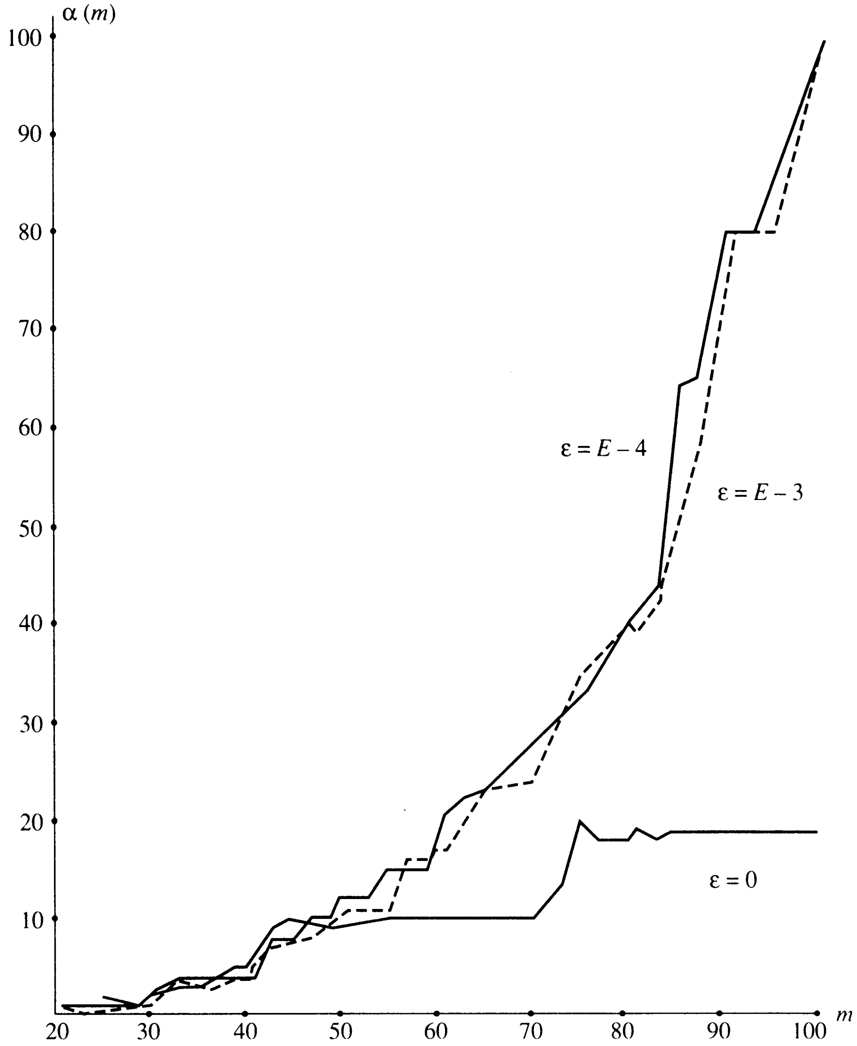

FIGURE 1. Convergence results of 100 steps for the 100×100 diagonal matrix of Example 8.3 with rebiorthogonalization

the cases $\epsilon = 1.E - 4$ and $1.E - 3$ yield all the 100 eigenvalues to working accuracy at step 100.

Example 8.4. Our last example is a 500×500 block upper triangular matrix. The first 250 diagonal entries consist of 1×1 blocks $[a_i]$ and the remaining 250 diagonal entries consist of 2×2 blocks of the from

$$A = \begin{pmatrix} b_i & c_i \\ -c_i & b_i \end{pmatrix}.$$

Here, a_i, b_i, and c_i are pseudorandom numbers of normal distribution with mean 0 and variance 1. The remaining entries above the diagonal blocks are pseudorandom numbers uniformly distributed in $[-0.5, 0.5]$. The two initial vectors are identical random vectors. The eigenvalue distribution is plotted in Figure 4 (for the upper half plane). We mark a Ritz value converged to an eigenvalue if the relative error is less than $1.E-3$ and $\alpha(m)$ again denotes the number of converged Ritz values.

The threshold parameters are $\epsilon = 0.$, $1.E-6$, $1.E-4$, $1.E-3$. We first implement the algorithm with full rebiorthogonalization, and the results of $\alpha(m)$ for $m = 130$ are plotted in Figure 2. We then implement the algorithm without rebiorthogonalization and the results of $\alpha(m)$ for $m = 130$ are plotted in Figure 3. In this case, some repeated Ritz values appear for all three cases, but are not counted in $\alpha(m)$ and not shown in the figure. Again, a behavior similar to that in the last example is observed.

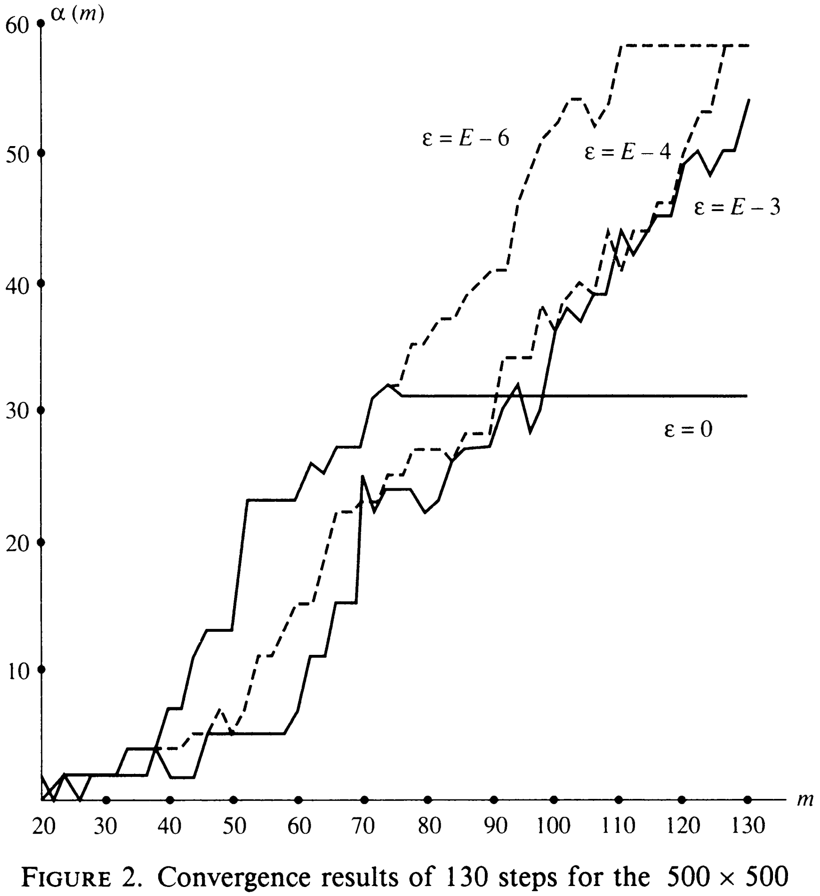

FIGURE 2. Convergence results of 130 steps for the 500×500 random matrix of Example 8.4 with rebiorthogonalization

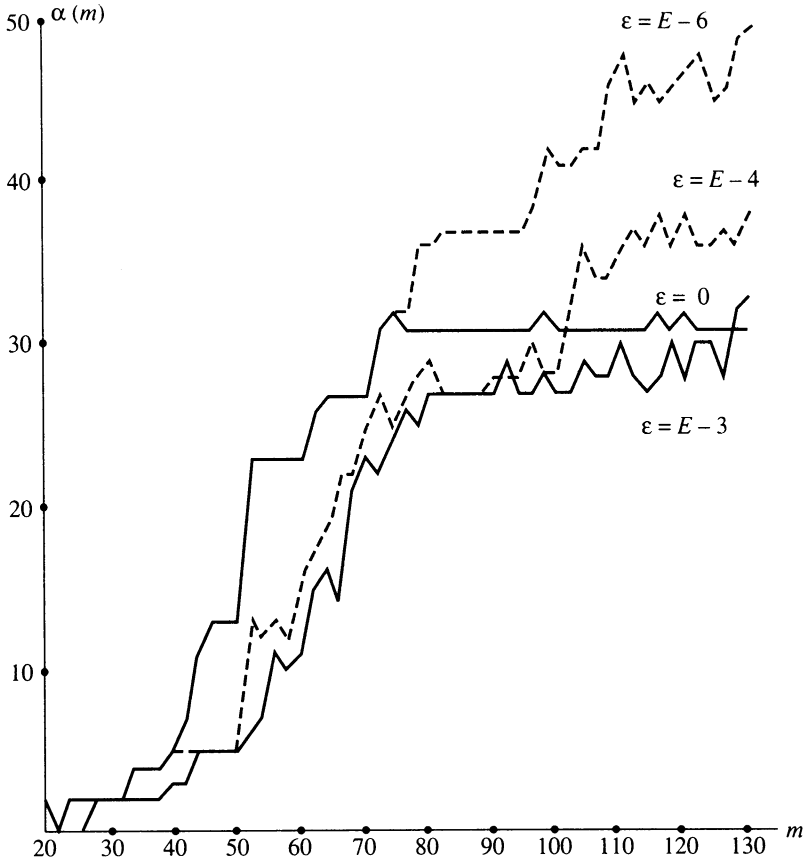

FIGURE 3. Convergence results of 130 steps for the 500 × 500 random matrix of Example 8.4 without rebiorthogonalization

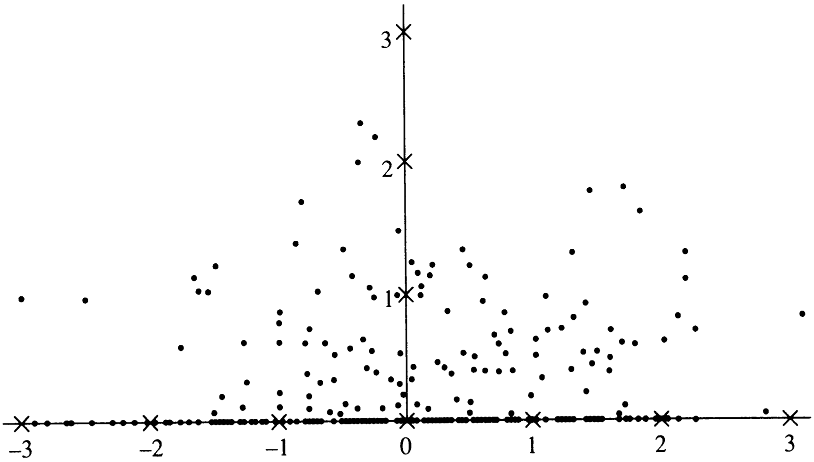

FIGURE 4. The distribution of the eigenvalues in Example 8.4

9. Conclusions

We have developed a new method to reduce a matrix to a condensed form, which is between the tridiagonal form and the upper Hessenberg form. The sparseness structure is controlled by a threshold parameter. By relaxing from the tridiagonal form, we are able to maintain the pivots above the threshold parameter and hence gain in stability. In theory, stability is gained at the cost of reduced convergence rate owing to more breakdowns. In numerical practice, however, the minor slowdown in convergence is paid off by stability. On the other hand, from the stability point of view, it is not necessary to enforce the pivots to be one, which will not only slow down convergence but also significantly increase computational cost. Therefore, our algorithm is optimal in the sense that it balances stability with convergence rate.

We have also sketched a symmetric version of Algorithm 3.1, which is new and of significance also in the classical symmetric case.

The question remains of how to choose the best threshold parameter. Too small an ϵ will result in fewer breakdowns but causes deterioration of the quality of computational results and eventually reduces the number of eigenvalues that can be obtained. Too large an ϵ yields more breakdowns and slows down convergence, but in the long run, more eigenvalues will be obtained. An appropriate threshold parameter ϵ should be as small as possible while sufficiently large to obtain all desired eigenvalues. Inevitably, the best choice will depend on the number of the eigenvalues desired as well as on the condition of the matrix. A successful solution to this problem will lead to a robust general algorithm for large sparse nonsymmetric eigenvalue problems.

Acknowledgment

The author thanks the referee for many valuable comments.

Bibliography

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

2. D. Boley and G. Golub, *The nonsymmetric Lanczos algorithm and controllability*, Systems Control Lett. **16** (1991), 97–105.

3. D. Boley, S. Elhay, G. Golub, and M. Gutknecht, *Nonsymmetric Lanczos algorithms and finding orthogonal polynomials associated with indefinite weights*, Numerical Algorithms **1** (1991), 21–43.

4. C. Brezinski, M. Redivo Zaglia, and H. Sadok, *Avoiding breakdown and near-breakdown in Lanczos type algorithms*, Numerical Algorithms **1** (1991), 261–284.

5. J. Cullum, W. Kerner, and R. Willoughby, *A generalized nonsymmetric Lanczos procedure*, Comput. Phys. Comm. **53** (1989), 19–48.

6. J. Cullum and R. A. Willoughby, *Lanczos algorithms for large symmetric eigenvalue computations*, Birkhäuser, Boston, 1985.

7. R. Freund, *Krylov subspace methods for complex non-hermitian linear systems*, RIACS Report 91.11, NASA Ames Research Center, May 1991.

8. R. Freund, M. Gutknecht, and N. Nachtigal, *An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices*, SIAM J. Sci. Statist. Comput. **14** (1993), 137–158.

9. R. Freund and N. Nachtigal, *QMR: a quasi-minimal residual method for non-Hermitian linear systems*, Numer Math. **60** (1991), 315–339.

10. M. Gutknecht, *A completed theory of the unsymmetric Lanczos process and related algorithms*, part I, SIAM J. Matrix Anal. Appl. **13** (1992), 594–639.

11. ______, *A completed theory of the unsymmetric Lanczos process and related algorithms*, part II, SIAM J. Matrix Anal. Appl. (to appear)

12. ______, *The unsymmetric Lanczos algorithms and their relations to Padé approximation, continued fraction and the qd algorithm*, Preliminary Proceedings of the Copper Mountain Conference on Iterative Methods.

13. G. A. Geist, *Reduction of a general matrix to tridiagonal form*, SIAM J. Matrix Anal. Appl. **12** (1991), 362–373.

14. R. T. Gregory and D. L. Karney, *A collection of matrices for testing computational algorithms*, Robert E. Krieger Publishing Company, Huntington, NY, 1978.

15. G. H. Golub and C. F. Van Loan, *Matrix computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.

16. W. D. Joubert, *Lanczos methods for the solution of nonsymmetric systems of linear equations*, SIAM J. Matrix Anal. Appl. **13** (1992), 926–943.

17. A. Dax and S. Kaniel, *The ELR method for computing the eigenvalues of a general matrix*, SIAM J. Numer. Anal. **18** (1981), 597–605.

18. W. Kerner, *Large-scale complex eigenvalue problems*, J. Comput. Phys. **85** (1989), 1–85.

19. C. Lanczos, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Standards **45** (1950), 255–282.

20. P. Lancaster and Q. Ye, *Rayleigh-Ritz and Lanczos methods for symmetric matrix pencils*, Linear Algebra Appl. **185** (1993), 173–201.

21. B. N. Parlett, *The symmetric eigenvalue problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

22. ______, *Reduction to tridiagonal form and minimal realizations*, SIAM J. Matrix Anal. Appl. **13** (1992), 567–593.

23. B. N. Parlett, D. R. Taylor, and Z. A. Liu, *A look-ahead Lanczos algorithm for unsymmetric matrices*, Math. Comp. **44** (1985), 105–124.

24. B. N. Parlett and H. C. Chen, *Use of an indefinite inner product for computing damped natural modes*, Linear Algebra Appl. **140** (1990), 53–88.

25. C. C. Paige, *The computation of eigenvalues and eigenvectors of very large sparse matrices*, Ph.D. thesis, London University, London, 1971.

26. Y. Saad, *Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices*, Linear Algebra Appl. **34** (1980), 269–295.

27. D. R. Taylor, *Analysis of the look ahead Lanczos algorithm*, Ph.D. thesis, University of California, Berkeley, 1982.

28. J. H. Wilkinson, *The algebraic eigenvalue problem*, Clarendon Press, Oxford, 1965.

29. Q. Ye, *Variational principles and numerical algorithms for symmetric matrix pencils*, Ph.D. thesis, University of Calgary, Calgary, 1989.

30. ______, *A convergence analysis of nonsymmetric Lanczos algorithms*, Math. Comp. **56** (1991), 677–691.

Department of Applied Mathematics, University of Manitoba, Winnipeg, Manitoba, Canada R3T 2N2

E-mail address: `ye@newton.amath.umanitoba.ca`