AUGMENTED IMPLICITLY RESTARTED LANCZOS BIDIAGONALIZATION METHODS

JAMES BAGLAMA* AND LOTHAR REICHEL†

Abstract. New restarted Lanczos bidiagonalization methods for the computation of a few of the largest or smallest singular values of a large matrix are presented. Restarting is carried out by augmentation of Krylov subspaces that arise naturally in the standard Lanczos bidiagonalization method. The augmenting vectors are associated with certain Ritz or harmonic Ritz vectors. Computed examples show the new methods to be competitive with available schemes.

Key words. Singular value computation, partial singular value decomposition, iterative method, large-scale computation.

AMS subject classifications. 65F15, 65F50, 15A18

1. Introduction. Many problems in scientific computation require knowledge of a few of the largest or smallest singular values of a matrix and associated left and right singular vectors. These problems include the approximation of a large matrix by a matrix of low rank, the computation of the null space of a matrix, total least-squares problems, see, e.g., Björck [5, Sections 4.6 and 7.6.5], as well as the tracking of signals; see, e.g., Comon and Golub [9] for a discussion of the latter.

Let $A \in \mathbb{R}^{\ell\times n}$ be a large sparse matrix. We may assume that $\ell \geq n$, because otherwise we replace the matrix by its transpose. Let

$$\sigma_1^{(A)} \geq \sigma_2^{(A)} \geq \ldots \geq \sigma_n^{(A)} \geq 0 \tag{1.1}$$

denote the singular values of A, and let $u_j^{(A)} \in \mathbb{R}^\ell$ and $v_j^{(A)} \in \mathbb{R}^n$, $1 \leq j \leq n$, be associated left and right singular vectors, respectively. Hence,

$$Av_j^{(A)} = \sigma_j^{(A)} u_j^{(A)}, \qquad A^T u_j^{(A)} = \sigma_j^{(A)} v_j^{(A)}, \qquad 1 \leq j \leq n, \tag{1.2}$$

and

$$A = \sum_{j=1}^{n} \sigma_j^{(A)} u_j^{(A)} (v_j^{(A)})^T.$$

The matrices $U_n^{(A)} = [u_1^{(A)}, u_2^{(A)}, \ldots, u_n^{(A)}]$ and $V_n^{(A)} = [v_1^{(A)}, v_2^{(A)}, \ldots, v_n^{(A)}]$ have orthonormal columns. We refer to $\{\sigma_j^{(A)}, u_j^{(A)}, v_j^{(A)}\}$ as a singular triplet of A. Singular triplets associated with large (small) singular values are referred to as large (small) singular triplets.

This paper presents new restarted Lanczos bidiagonalization methods for computing a few of the largest or smallest singular triplets. The methods compute sequences of projections of A onto judiciously chosen low-dimensional subspaces. Restarting is implemented by augmentation of Krylov subspaces that are determined similarly as in the standard Lanczos bidiagonalization method.

*Department of Mathematics, University of Rhode Island, Kingston, RI 02881. E-mail: jbaglama@math.uri.edu. Home page: http://hypatia.math.uri.edu/~jbaglama. Research supported in part by NSF grant DMS-0311786.

†Department of Mathematical Sciences, Kent State University, Kent, OH 44242. E-mail: reichel@math.kent.edu. Home page: http://www.math.kent.edu/~reichel. Research supported in part by NSF grant DMS-0107858.

Application of m steps of partial Lanczos bidiagonalization to the matrix A with initial unit vector $p_1 \in \mathbb{R}^n$ yields the decompositions

$$AP_m = Q_m B_m, \tag{1.3}$$

$$A^T Q_m = P_m B_m^T + r_m e_m^T, \tag{1.4}$$

where $P_m \in \mathbb{R}^{n\times m}$, $Q_m \in \mathbb{R}^{\ell\times m}$, $P_m^T P_m = I_m$, $P_m e_1 = p_1$, $Q_m^T Q_m = I_m$, $r_m \in \mathbb{R}^n$, and

$$P_m^T r_m = 0. \tag{1.5}$$

Further, the matrix

$$B_m := \begin{bmatrix} \alpha_1 & \beta_1 & & & & 0 \\ & \alpha_2 & \beta_2 & & & \\ & & \alpha_3 & \beta_3 & & \\ & & & & \ddots & \\ & & & & \ddots & \beta_{m-1} \\ 0 & & & & & \alpha_m \end{bmatrix} \in \mathbb{R}^{m\times m} \tag{1.6}$$

is upper bidiagonal, $I_m \in \mathbb{R}^{m\times m}$ denotes the identity matrix, and e_j the jth axis vector. We refer to the decompositions (1.3)-(1.4) as a partial Lanczos bidiagonalization of A and to the vector r_m in (1.4) as the residual vector; see Björck [5, Section 7.6] for a recent discussion on partial Lanczos bidiagonalization. The number of bidiagonalization steps, m, is assumed to be small enough so that the decompositions (1.3)-(1.4) with the stated properties exist.

In applications of interest to us, m is not very large, and the singular triplets $\{\sigma_j^{(B_m)}, u_j^{(B_m)}, v_j^{(B_m)}\}_{j=1}^m$ of B_m can be computed inexpensively by the Golub-Kahan algorithm [11]. Approximate singular triplets of A, denoted by $\{\tilde{\sigma}_j^{(A)}, \tilde{u}_j^{(A)}, \tilde{v}_j^{(A)}\}$, can be determined from the singular triplets of B_m and the matrices P_m and Q_m in the decompositions (1.3)-(1.4); see Section 2 for details.

When the matrix A is large, i.e., when ℓ and possibly n are large, the storage requirement of the partial Lanczos bidiagonalization (1.3)-(1.4) is large, unless the number of Lanczos bidiagonalization steps, m, is small. However, for a small value of m, the desired singular triplets of A may be approximated poorly by computed approximate singular triplets $\{\tilde{\sigma}_j^{(A)}, \tilde{u}_j^{(A)}, \tilde{v}_j^{(A)}\}$. In order to circumvent this difficulty, several methods have been proposed that are based on the computation of partial Lanczos bidiagonalizations (1.3)-(1.4) with m small for a sequence of initial vectors p_1; see, e.g., [6, 14, 15, 16, 17, 18, 29]. These methods are commonly referred to as restarted partial Lanczos bidiagonalization methods. We will comment on some of these methods below. They differ in their choice of initial vector p_1 used for the restarts and in their implementation of the restarting procedure. Ideally, we would like p_1 to be a linear combination of the right singular vectors of A associated with the desired singular values.

Sorensen [30] proposed efficient approaches for restarting the Arnoldi and the Lanczos tridiagonalization procedures. These approaches can be thought of as curtailed QR-algorithms, and, similarly to the QR-algorithms, their performance depends critically on the selection of shifts; see also [1, 8, 22, 31] for discussions and extensions. Björck et al. [6] derived analogous recursion formulas for a restarted Lanczos

bidiagonalization method, and presented an application to the solution of ill-posed problems. Recently, Kokiopoulou et al. [18] applied these recursion formulas to compute a few desired singular triplets of a large sparse matrix. The shifts are applied by "chasing the bulge" in a curtailed QR-algorithm. Other implementations of restarted Lanczos bidiagonalization are discussed in [14, 15, 16, 17, 29]. The present paper describes mathematically equivalent, but numerically more robust, implementations of the methods discussed by Kokiopoulou et al. [18].

This paper is organized as follows. Section 2 reviews Lanczos bidiagonalization and introduces notation used in the remainder of the paper. Section 3 describes our implementations of restarted Lanczos bidiagonalization. Restarting is carried out by augmenting the Krylov subspaces that arise naturally in the standard Lanczos bidiagonalization method by Ritz vectors or harmonic Ritz vectors associated with desired singular triplets. A few computed examples, which compare the performance of several methods and implementations, are presented in Section 4.

2. Lanczos bidiagonalization. The following algorithm determines the partial Lanczos bidiagonalization (1.3)-(1.4) of the large sparse matrix $A \in \mathbb{R}^{\ell \times n}$. The number of Lanczos bidiagonalization steps, m, typically is much smaller than either one of the matrix dimensions ℓ and n. Throughout this paper $\|\cdot\|$ denotes the Euclidean vector norm or the associated induced matrix norm. We remark that the methods described easily can be modified to apply to matrices A with complex-valued entries; the main modification required is that the matrices P_m, Q_m, and B_m in (1.3)-(1.4) may have complex-valued entries, and transposition has to be replaced by transposition and complex conjugation. The MATLAB code used for the computed examples of Section 4 can be applied to matrices A with complex-valued entries.

ALGORITHM 2.1. LANCZOS BIDIAGONALIZATION

Input: $A \in \mathbb{R}^{\ell \times n}$ *or functions for evaluating matrix-vector products with the matrices* A *and* A^T,
$p_1 \in \mathbb{R}^n$: *initial vector of unit length,*
m : *number of bidiagonalization steps.*

Output: $P_m = [p_1, p_2, \ldots, p_m] \in \mathbb{R}^{n \times m}$: *matrix with orthonormal columns,*
$Q_m = [q_1, q_2, \ldots, q_m] \in \mathbb{R}^{\ell \times m}$: *matrix with orthonormal columns,*
$B_m \in \mathbb{R}^{m \times m}$: *upper bidiagonal matrix (1.6) with entries* α_j *and* β_j,
$r_m \in \mathbb{R}^n$: *residual vector.*

1. $P_1 := p_1$; $q_1 := Ap_1$;
2. $\alpha_1 := \|q_1\|$; $q_1 := q_1/\alpha_1$; $Q_1 := q_1$;
3. for $j = 1 : m$
  4. $r_j := A^T q_j - \alpha_j p_j$;
  5. Reorthogonalization: $r_j := r_j - P_j(P_j^T r_j)$;
  6. if $j < m$ *then*
    7. $\beta_j := \|r_j\|$; $p_{j+1} := r_j/\beta_j$; $P_{j+1} := [P_j, p_{j+1}]$;
    8. $q_{j+1} := Ap_{j+1} - \beta_j q_j$;
    9. Reorthogonalization: $q_{j+1} := q_{j+1} - Q_j(Q_j^T q_{j+1})$;
    10. $\alpha_{j+1} := \|q_{j+1}\|$; $q_{j+1} := q_{j+1}/\alpha_{j+1}$; $Q_{j+1} := [Q_j, q_{j+1}]$;
  11. endif
12. endfor

When the computations with Algorithm 2.1 are carried out in finite precision arithmetic and the columns of P_m and Q_m are not reorthogonalized, the computed columns might be far from orthogonal. We therefore reorthogonalize the columns of these matrices in lines 5 and 9 the algorithm.

Several reorthogonalization strategies for the columns of the matrices P_m and Q_m are discussed in the literature. Larsen [19] found that when m is fairly large and one is interested in computing a few of the largest singular triplets of A, partial reorthogonalization gives comparable accuracy and requires less computational work than full reorthogonalization, but when a few of the smallest singular triplets are desired, often (essentially) full reorthogonalization is required to achieve high accuracy. Wu and Simon [34] report that when m is not large, full reorthogonalization should be carried out, because due to the overhead associated with partial reorthogonalization, the latter is not competitive. Moreover, Simon and Zha [29] show that when the matrix A is not very ill-conditioned, only the columns of one of the matrices P_m or Q_m need to be reorthogonalized. Reorthogonalization of the columns of P_m only can reduce the computational effort required to compute the partial Lanczos bidiagonalization (1.3)-(1.4) considerably when $\ell \gg n$. Algorithm 2.1 easily can be modified to implement the latter approach.

The Lanczos bidiagonalization algorithm (Algorithm 2.1) requires the diagonal entries $\alpha_j > 0$, $1 \leq j \leq m$, as well as the superdiagonal entries β_j, $1 \leq j < m$, of B_m to be nonvanishing. Assume for the moment that $\alpha_j > 0$ for $1 \leq j < m$, and $\alpha_m = 0$. Then the vector q_m determined in line 8 of the algorithm vanishes, and the decompositions (1.3)-(1.4) can be expressed as

$$AP_m = Q_{m-1}B_{m-1,m}, \qquad A^T Q_{m-1} = P_m B_{m-1,m}^T,$$

where $B_{m-1,m}$ denotes the the leading $(m-1) \times m$ submatrix of B_m. It follows that A is singular and that the singular values of $B_{m-1,m}$ are singular values of A. The associated singular triplets of A can be determined from P_m, Q_{m-1}, and the singular triplets of $B_{m-1,m}$. The singular values of $B_{m-1,m}$ are nonvanishing. Moreover, the right singular vector of A associated with the zero singular value can be expressed as a linear combination of the columns of P_m, e.g., by using the singular value decomposition of B_m. However, the corresponding left singular vector of A is not readily available. For simplicity, we will in the remainder of this paper assume that all α_j are positive.

It follows from Algorithm 2.1 that β_m, the last superdiagonal element of the upper bidiagonal matrix $B_{m+1} \in \mathbb{R}^{(m+1)\times(m+1)}$ obtained by $m+1$ steps of partial Lanczos bidiagonalization, is given by

$$\beta_m := \|r_m\| \tag{2.1}$$

and therefore can be computed when the decompositions (1.3)-(1.4) are available. We may assume that $\beta_m > 0$, because otherwise the singular values of B_m are singular values of A and the associated singular triplets of A can be determined from the singular value decomposition of B_m and the matrices P_m and Q_m. If $\beta_m = 0$ and the determined singular triplets of A include all the desired ones, then we are done; otherwise we proceed to compute additional singular triplets by the methods described in this paper.

When $\beta_m > 0$, the last column of $P_{m+1} := [P_m, p_{m+1}]$ is given by

$$p_{m+1} := r_m/\beta_m \tag{2.2}$$

and the relation (1.4) can be expressed as

$$A^T Q_m = P_{m+1} B_{m,m+1}^T, \tag{2.3}$$

where the matrix $B_{m,m+1} \in \mathbb{R}^{m\times(m+1)}$ is obtained by appending the column $\beta_m e_m$ to B_m. In particular, the matrices in the decomposition (2.3) are available after m Lanczos bidiagonalization steps. We also note that $B_{m,m+1}$ is the leading $m \times (m+1)$ submatrix of the matrix B_{m+1} obtained after $m+1$ steps of Lanczos bidiagonalization.

We will use the connection between partial Lanczos bidiagonalization (1.3)-(1.4) of A and partial Lanczos tridiagonalization of the matrix $A^T A$. Multiplying equation (1.3) by A^T from the left-hand side yields

$$A^T A P_m = P_m B_m^T B_m + r_m e_m^T B_m = P_m B_m^T B_m + \alpha_m r_m e_m^T, \tag{2.4}$$

where the last equality follows from the fact that B_m is upper bidiagonal. The matrix

$$T_m := B_m^T B_m \in \mathbb{R}^{m\times m} \tag{2.5}$$

is symmetric and tridiagonal, and the expression (2.4) is a partial Lanczos tridiagonalization of $A^T A$ with initial vector $p_1 = P_m e_1$; see, e.g., [12, Section 9.1.2] for a discussion on Lanczos tridiagonalization. Since T_m is tridiagonal, equation (2.4) shows that the columns of P_m satisfy a three-term recurrence relation; the columns form an orthonormal basis of the Krylov subspace

$$\mathbb{K}_m(A^T A, p_1) = \mathrm{span}\{p_1, A^T A p_1, (A^T A)^2 p_1, \ldots, (A^T A)^{m-1} p_1\}. \tag{2.6}$$

Similarly, multiplying (1.4) by A from the left-hand side yields

$$A A^T Q_m = Q_m B_m B_m^T + A r_m e_m^T. \tag{2.7}$$

The columns of Q_m form an orthonormal basis of the Krylov subspace

$$\mathbb{K}_m(A A^T, q_1) = \mathrm{span}\{q_1, A A^T q_1, (A A^T)^2 q_1, \ldots, (A A^T)^{m-1} q_1\} \tag{2.8}$$

with $q_1 := A p_1$. We remark that since the columns of Q_m generally are not orthogonal to $A r_m$, the decomposition (2.7) typically is not a Lanczos tridiagonalization of $A A^T$.

Let $\{\sigma_j^{(B_m)}, u_j^{(B_m)}, v_j^{(B_m)}\}$, $1 \le j \le m$, be the singular triplets of B_m enumerated so that

$$\sigma_1^{(B_m)} \ge \sigma_2^{(B_m)} \ge \ldots \ge \sigma_m^{(B_m)} \ge 0. \tag{2.9}$$

Then, analogously to (1.2),

$$B_m v_j^{(B_m)} = \sigma_j^{(B_m)} u_j^{(B_m)}, \qquad B_m^T u_j^{(B_m)} = \sigma_j^{(B_m)} v_j^{(B_m)}, \qquad 1 \le j \le m, \tag{2.10}$$

and the $m \times m$ matrices of left and right singular vectors

$$U_m^{(B_m)} := [u_1^{(B_m)}, u_2^{(B_m)}, \ldots, u_m^{(B_m)}], \qquad V_m^{(B_m)} := [v_1^{(B_m)}, v_2^{(B_m)}, \ldots, v_m^{(B_m)}],$$

are orthogonal.

We determine approximate singular triplets $\{\tilde\sigma_j^{(A)}, \tilde u_j^{(A)}, \tilde v_j^{(A)}\}$, $1 \le j \le m$, of A from the singular triplets of B_m by

$$\tilde\sigma_j^{(A)} := \sigma_j^{(B_m)}, \qquad \tilde u_j^{(A)} := Q_m u_j^{(B_m)}, \qquad \tilde v_j^{(A)} := P_m v_j^{(B_m)}. \tag{2.11}$$

Combining (2.10) with (1.3)-(1.4) shows that

$$(2.12)\quad A\tilde{v}_j^{(A)} = \tilde{\sigma}_j^{(A)}\tilde{u}_j^{(A)}, \quad A^T\tilde{u}_j^{(A)} = \tilde{\sigma}_j^{(A)}\tilde{v}_j^{(A)} + r_m e_m^T u_j^{(B_m)}, \quad 1 \le j \le m.$$

The equations (2.12) suggest that an approximate singular triplet $\{\tilde{\sigma}_j^{(A)}, \tilde{u}_j^{(A)}, \tilde{v}_j^{(A)}\}$ be accepted as a singular triplet of A if $r_m e_m^T u_j^{(B_m)}$ is sufficiently small. Specifically, our numerical method accepts $\{\tilde{\sigma}_j^{(A)}, \tilde{u}_j^{(A)}, \tilde{v}_j^{(A)}\}$ as a singular triplet of A if

$$(2.13)\qquad \beta_m |e_m^T u_j^{(B_m)}| \le \delta \|A\|$$

for a user-specified value of δ, where we have used (2.1). The quantity $\|A\|$ in (2.13) is easily approximated by the singular value $\sigma_1^{(B_m)}$ of largest magnitude of the bidiagonal matrix B_m. The computation of $\sigma_1^{(B_m)}$ is inexpensive because the matrix B_m is small. During the computations of the desired singular triplets of A, typically, several matrices B_m and their singular value decompositions are computed. We approximate $\|A\|$ by the largest of the singular values of all the matrices B_m generated. This generally gives a good estimate of $\|A\|$.

3. Augmented Lanczos bidiagonalization methods. It is well known that the implicitly restarted Arnoldi and Lanczos tridiagonalization methods described by Sorensen [30] can suffer from numerical instability due to propagated round-off errors. The instability can delay or prevent convergence of desired eigenvalues and eigenvectors; see Lehoucq and Sorensen [22] for a discussion and remedies. Morgan [26] showed that the implicitly restarted Arnoldi method by Sorensen [30] can be implemented by augmenting the available Krylov subspace basis by certain Ritz vectors. Such an implementation can be less sensitive to propagated round-off errors than the implementation in [30]. Recently, Wu and Simon [34] described a so-called thick-restarted Lanczos tridiagonalization method for the symmetric eigenvalue problem. The method is based on augmenting Krylov subspaces by certain Ritz vectors; it is simple to implement and is mathematically equivalent to the implicitly restarted Lanczos tridiagonalization method of Sorensen [30]. This paper presents thick-restarted Lanczos bidiagonalization methods. We remark that thick restarting techniques also have been used in the context of the Jacobi-Davidson and Arnoldi methods for eigenvalue computations; see Stathopoulos et al. [32, 33] for discussions and analyses.

3.1. Augmentation by Ritz vectors. Let the partial Lanczos bidiagonalization (1.3)-(1.4) be available, and assume that we are interested in determining the k largest singular triplets of A, where $k < m$. Note that the approximate right singular vector $\tilde{v}_j^{(A)}$ of A, defined by (2.11), is a Ritz vector of A^TA associated with the Ritz value $(\tilde{\sigma}_j^{(A)})^2$. We have

$$(3.1)\qquad A^TA\tilde{v}_j^{(A)} - (\tilde{\sigma}_j^{(A)})^2\tilde{v}_j^{(A)} = \alpha_m r_m e_m^T v_j^{(B_m)}, \qquad 1 \le j \le m.$$

The observation that the right-hand sides are parallel to r_m for all j forms the basis of the restarted Lanczos tridiagonalization method by Wu and Simon [34] as well as of the restarted Lanczos bidiagonalization method of this subsection.

We derive decompositions of the form (1.3)-(1.4) which allow us to choose the first k columns of the matrix P_m as Ritz vectors. The accuracy of these approximate left singular vectors, as well as of available approximate right singular vectors, is then improved by restarting the computations. It follows from the orthonormality of the

columns of the matrices P_m and $V_m^{(B_m)}$ that the Ritz vectors $\tilde{v}_j^{(A)}$ defined by (2.11) are orthonormal. Moreover, equation (1.5) shows that the $\tilde{v}_j^{(A)}$ are orthogonal to r_m.

Let the Ritz vectors $\tilde{v}_j^{(A)}$, $1 \leq j \leq k$, associated with the k largest Ritz values be available, assume that $r_m \neq 0$, and introduce the matrix

$$\tilde{P}_{k+1} := [\tilde{v}_1^{(A)}, \tilde{v}_2^{(A)}, \ldots, \tilde{v}_k^{(A)}, p_{m+1}]. \tag{3.2}$$

In view of (2.2), the last column of $\tilde{P}_{k+1}$ is parallel to the residual error (3.1). It follows from (2.11) that

$$A\tilde{P}_{k+1} = [\tilde{\sigma}_1^{(A)}\tilde{u}_1^{(A)}, \tilde{\sigma}_2^{(A)}\tilde{u}_2^{(A)}, \ldots, \tilde{\sigma}_k^{(A)}\tilde{u}_k^{(A)}, Ap_{m+1}]. \tag{3.3}$$

Orthogonalization of Ap_{m+1} against the vectors $\tilde{u}_j^{(A)}$ yields

$$Ap_{m+1} = \sum_{j=1}^{k} \tilde{\rho}_j \tilde{u}_j^{(A)} + \tilde{r}_k, \tag{3.4}$$

where the remainder $\tilde{r}_k$ is orthogonal to the vectors $\tilde{u}_j^{(A)}$, $1 \leq j \leq k$. The coefficients $\tilde{\rho}_j := (\tilde{u}_j^{(A)})^T Ap_{m+1}$ can be evaluated inexpensively by using the right-hand side of

$$(\tilde{u}_j^{(A)})^T Ap_{m+1} = p_{m+1}^T A^T \tilde{u}_j^{(A)} = p_{m+1}^T(\tilde{\sigma}_j^{(A)}\tilde{v}_j^{(A)} + r_m e_m^T u_j^{(B_m)}) = \beta_m e_m^T \tilde{u}_j^{(B_m)}.$$

We may assume that the vector $\tilde{r}_k$ is nonvanishing, because otherwise we can terminate the iterations; see below. Introduce the matrices

$$\tilde{Q}_{k+1} := [\tilde{u}_1^{(A)}, \tilde{u}_2^{(A)}, \ldots, \tilde{u}_k^{(A)}, \frac{\tilde{r}_k}{\|\tilde{r}_k\|}] \tag{3.5}$$

and

$$\tilde{B}_{k+1} := \begin{bmatrix} \tilde{\sigma}_1^{(A)} & & 0 & \tilde{\rho}_1 \\ & \ddots & & \vdots \\ & & \tilde{\sigma}_k^{(A)} & \tilde{\rho}_k \\ 0 & & & \tilde{\alpha}_{k+1} \end{bmatrix} \in \mathbb{R}^{(k+1)\times(k+1)}. \tag{3.6}$$

Thus, $\tilde{B}_{k+1}$ may have nonvanishing entries only on the diagonal and in the last column. Substituting (3.4) into (3.3) now yields the decomposition

$$A\tilde{P}_{k+1} = \tilde{Q}_{k+1}\tilde{B}_{k+1}, \tag{3.7}$$

which is analogous to (1.3).

We turn to the matrix

$$A^T\tilde{Q}_{k+1} = [A^T\tilde{u}_1^{(A)}, A^T\tilde{u}_1^{(A)}, \ldots, A^T\tilde{u}_k^{(A)}, A^T\frac{\tilde{r}_k}{\|\tilde{r}_k\|}], \tag{3.8}$$

which we would like to express in terms of $\tilde{P}_{k+1}$ and $\tilde{B}_{k+1}^T$. This will give an analogue of the decomposition (1.4). The first k columns of (3.8) are linear combinations of the vectors $\tilde{v}_j^{(A)}$ and p_{m+1}; specifically,

$$A^T\tilde{u}_j^{(A)} = \tilde{\sigma}_j^{(A)}\tilde{v}_j^{(A)} + r_m e_m^T u_j^{(B_m)} = \tilde{\sigma}_j^{(A)}\tilde{v}_j^{(A)} + p_{m+1}\tilde{\rho}_j, \qquad 1 \leq j \leq k, \tag{3.9}$$

where we have used (2.2). The last column of (3.8) is orthogonal to the Ritz vectors $\tilde{v}_j^{(A)}$,

$$(\tilde{v}_j^{(A)})^T A^T \frac{\tilde{r}_k}{\|\tilde{r}_k\|} = \tilde{\sigma}_j^{(A)} \frac{(\tilde{u}_j^{(A)})^T \tilde{r}_k}{\|\tilde{r}_k\|} = 0, \qquad 1 \leq j \leq k,$$

and therefore it can be expressed as

$$A^T \frac{\tilde{r}_k}{\|\tilde{r}_k\|} = \tilde{\gamma}_1 p_{m+1} + \tilde{f}_{k+1}, \tag{3.10}$$

where $\tilde{f}_{k+1} \in \mathbb{R}^n$ is orthogonal to the vectors $\tilde{v}_j^{(A)}$, $1 \leq j \leq k$, as well as to p_{m+1}. Since

$$p_{m+1}^T A^T \frac{\tilde{r}_k}{\|\tilde{r}_k\|} = \frac{\tilde{r}_k^T}{\|\tilde{r}_k\|} A p_{m+1} = \frac{\tilde{r}_k^T}{\|\tilde{r}_k\|} \left(\sum_{j=1}^{k} \tilde{\rho}_j \tilde{u}_j^{(A)} + \tilde{r}_k \right) = \|\tilde{r}_k\|,$$

it follows from (3.10) that $\tilde{\gamma}_1 = \|\tilde{r}_k\|$. This observation, together with (3.9) and (3.10), gives the expression

$$A^T \tilde{Q}_{k+1} = \tilde{P}_{k+1} \tilde{B}_{k+1}^T + \tilde{f}_{k+1} e_{k+1}^T, \tag{3.11}$$

which is the desired analogue of the decomposition (1.4). We remark that $\tilde{f}_{k+1}$ can be computed from equation (3.10).

The similarity of the decompositions (3.7) and (3.11), and (1.3)-(1.4), suggests that it may be possible to append new columns to the matrices $\tilde{P}_{k+1}$ and $\tilde{Q}_{k+1}$ in a way similar to Lanczos bidiagonalization. We will show that this is indeed the case. For notational simplicity, denote the columns of $\tilde{P}_{k+1}$ and $\tilde{Q}_{k+1}$ by $\tilde{p}_j$ and $\tilde{q}_j$, respectively, i.e.,

$$\tilde{P}_{k+1} = [\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_{k+1}] \in \mathbb{R}^{n \times (k+1)}, \qquad \tilde{Q}_{k+1} = [\tilde{q}_1, \tilde{q}_2, \ldots, \tilde{q}_{k+1}] \in \mathbb{R}^{\ell \times (k+1)}.$$

We may assume that $\tilde{f}_{k+1} \neq 0$, because otherwise it follows from (3.7) and (3.11) that the singular values of $\tilde{B}_{k+1}$ are singular values of A, and we are done. Thus, let $\tilde{\beta}_{k+1} := \|\tilde{f}_{k+1}\|$ and $\tilde{p}_{k+2} := \tilde{f}_{k+1}/\tilde{\beta}_{k+1}$. Then the matrix $\tilde{P}_{k+2} := [\tilde{P}_{k+1}, \tilde{p}_{k+2}]$ has orthonormal columns.

Let

$$\tilde{\alpha}_{k+2} \tilde{q}_{k+2} := (I - \tilde{Q}_{k+1} \tilde{Q}_{k+1}^T) A \tilde{p}_{k+2}, \tag{3.12}$$

where $\tilde{\alpha}_{k+2} > 0$ is a scaling factor, such that $\tilde{q}_{k+2}$ is of unit length. We comment on the possibility of $\tilde{\alpha}_{k+2}$ vanishing below. Equation (3.11) yields

$$A^T \tilde{Q}_{k+1} = \tilde{P}_{k+1} \tilde{B}_{k+1}^T + \tilde{\beta}_{k+1} \tilde{p}_{k+2} e_{k+1}^T, \tag{3.13}$$

and substituting (3.13) into (3.12) shows that

$$\begin{aligned} \tilde{\alpha}_{k+2} \tilde{q}_{k+2} &= A \tilde{p}_{k+2} - \tilde{Q}_{k+1} (A^T \tilde{Q}_{k+1})^T \tilde{p}_{k+2} \\ &= A \tilde{p}_{k+2} - \tilde{Q}_{k+1} (\tilde{B}_{k+1} \tilde{P}_{k+1}^T + \tilde{\beta}_{k+1} e_{k+1} \tilde{p}_{k+2}^T) \tilde{p}_{k+2} \\ &= A \tilde{p}_{k+2} - \tilde{\beta}_{k+1} \tilde{Q}_{k+1} e_{k+1} = A \tilde{p}_{k+2} - \tilde{\beta}_{k+1} \tilde{q}_{k+1}. \end{aligned} \tag{3.14}$$

Let $\tilde{Q}_{k+2} := [\tilde{Q}_{k+1}, \tilde{q}_{k+2}] \in \mathbb{R}^{\ell\times(k+2)}$ and define $\tilde{B}_{k+2} \in \mathbb{R}^{(k+2)\times(k+2)}$ by first appending the column $\tilde{\beta}_{k+1}e_{k+1}$ and then the row $\tilde{\alpha}_{k+2}e_{k+2}^T$ to $\tilde{B}_{k+1}$, i.e.,

$$\tilde{B}_{k+2} := \begin{bmatrix} \tilde{\sigma}_1^{(A)} & & 0 & \tilde{\rho}_1 & 0 \\ & \ddots & & \vdots & \vdots \\ & & \tilde{\sigma}_k^{(A)} & \tilde{\rho}_k & 0 \\ & & & \tilde{\alpha}_{k+1} & \tilde{\beta}_{k+1} \\ 0 & & & & \tilde{\alpha}_{k+2} \end{bmatrix}.$$

It now follows from (3.7) and (3.14) that

$$A\tilde{P}_{k+2} = \tilde{Q}_{k+2}\tilde{B}_{k+2}. \tag{3.15}$$

We turn to the derivation of a decomposition of the form (3.13) with $k+1$ replaced by $k+2$. Let

$$\tilde{\beta}_{k+2}\tilde{p}_{k+3} := (I - \tilde{P}_{k+2}\tilde{P}_{k+2}^T)A^T\tilde{q}_{k+2}, \tag{3.16}$$

where $\tilde{\beta}_{k+2} > 0$ is a scaling factor, such that $\tilde{p}_{k+3}$ is of unit length. We may assume that a positive coefficient $\tilde{\beta}_{k+2}$ exists, otherwise we are done; see below. Substituting (3.15) into (3.16) yields

$$\tilde{\beta}_{k+2}\tilde{p}_{k+3} = A^T\tilde{q}_{k+2} - \tilde{P}_{k+2}\tilde{B}_{k+2}^T e_{k+2} = A^T\tilde{q}_{k+2} - \tilde{\alpha}_{k+2}\tilde{p}_{k+2},$$

which, together with (3.13), shows that

$$A^T\tilde{Q}_{k+2} = \tilde{P}_{k+2}\tilde{B}_{k+2}^T + \tilde{\beta}_{k+2}\tilde{p}_{k+3}e_{k+2}^T. \tag{3.17}$$

The decompositions (3.15) and (3.17) are analogous to (3.7) and (3.13). We therefore can continue in the same fashion by appending new columns to the matrices $\tilde{P}_j$ and $\tilde{Q}_j$, and new rows and columns to the matrices $\tilde{B}_j$, for $j = k+2, k+3, \ldots$. After a total of $m-k$ steps, we obtain the decompositions

$$A\tilde{P}_m = \tilde{Q}_m\tilde{B}_m, \qquad A^T\tilde{Q}_m = \tilde{P}_m\tilde{B}_m^T + \tilde{\beta}_m\tilde{p}_{m+1}e_m^T, \tag{3.18}$$

where the matrices $\tilde{P}_m$ and $\tilde{Q}_m$ have orthonormal columns and

$$\tilde{B}_m = \begin{bmatrix} \tilde{\sigma}_1^{(A)} & & 0 & \tilde{\rho}_1 & & 0 \\ & \ddots & & \vdots & & \\ & & \tilde{\sigma}_k^{(A)} & \tilde{\rho}_k & & \\ & & & \tilde{\alpha}_{k+1} & \tilde{\beta}_{k+1} & \\ & & & & \ddots & \\ & & & & \ddots & \tilde{\beta}_{m-1} \\ 0 & & & & & \tilde{\alpha}_m \end{bmatrix}.$$

The method now proceeds by first computing the singular value decomposition of $\tilde{B}_m$ and then determining the k largest approximate singular triplets of A from the k largest singular triplets of $\tilde{B}_m$, cf. (2.11). These triplets define new decompositions

of the form (3.7) and (3.11), from which we compute new decompositions of the form (3.18). The computations proceed in this manner until sufficiently accurate approximations of the k largest singular triplets of A have been determined. An algorithm is presented in Subsection 3.3 below.

We remark that the computations are analogous for determining the k smallest singular triplets of A. The vectors $\tilde{v}_j^{(A)}$, $1 \leq j \leq k$, in (3.2) then should be replaced by the right approximate singular vectors of the k smallest available approximate singular triplets of A. These approximate singular triplets are used to define decompositions (3.7) and (3.11), from which we compute decompositions (3.18) in the same manner as described above. We are interested in the k smallest singular triplets of the matrix $\tilde{B}_m$ in the decompositions (3.18). These triplets yield approximations of the k smallest triplets of A. The computations are continued until sufficiently accurate approximations of the k smallest singular triplets of A have been found. However, we note that when the k smallest singular triplets of A are desired, it can be advantageous to augment by harmonic Ritz vectors instead. This is discussed in Subsection 3.2.

We finally comment on the cases when $\tilde{r}_k$ vanishes in (3.4), and when the left-hand sides of (3.12) and (3.16) vanish. In all these cases, one can show that the singular values of the matrix $\tilde{B}_j$ also are singular values of A and the singular triplets of $\tilde{B}_j$ yield singular triplets of A.

3.2. Augmentation by harmonic Ritz vectors. Augmenting by Ritz vectors as described in the previous subsection, or equivalently, shifting by Ritz values, often gives good approximations to the largest singular triplets of A. However, Kokiopoulou et al. [18] observed that when seeking to compute the smallest singular triplets of A, shifting by harmonic Ritz values can give faster convergence than shifting by Ritz values. This section describes how shifting by harmonic Ritz values can be implemented via augmentation by harmonic Ritz vectors. Harmonic Ritz vectors are approximate eigenvectors of A^TA associated with harmonic Ritz values of A^TA.

Let the partial Lanczos bidiagonalization (1.3)-(1.4) of A be available and assume that all the diagonal and superdiagonal entries of B_m, as well as β_m given by (2.1), are nonvanishing. Then, in particular, B_m is nonsingular. The harmonic Ritz values $\hat{\theta}_j$ of A^TA associated with the partial Lanczos tridiagonalization (2.4) are the eigenvalues of the generalized eigenvalue problem

$$((B_m^TB_m)^2 + \alpha_m^2\beta_m^2 e_m e_m^T)\hat{w}_j = \hat{\theta}_j B_m^TB_m\hat{w}_j, \qquad 1 \leq j \leq m, \tag{3.19}$$

with $\hat{w}_j \in \mathbb{R}^m \backslash \{0\}$; see, e.g., Morgan [25] or Paige et al. [28] for properties of harmonic Ritz values.

The eigenpairs $\{\hat{\theta}_j, \hat{w}_j\}$ of (3.19) can be computed without forming the matrix $B_m^TB_m$ as follows. Let

$$w_j := B_m\hat{w}_j. \tag{3.20}$$

Then (3.19) can be expressed as

$$(B_mB_m^T + \beta_m^2 e_m e_m^T)w_j = \hat{\theta}_j w_j, \tag{3.21}$$

where we may choose the eigenvectors w_j to be orthonormal. Let $B_{m,m+1}$ be the matrix in (2.3) and note that

$$B_{m,m+1}B_{m,m+1}^T = B_mB_m^T + \beta_m^2 e_m e_m^T. \tag{3.22}$$

Introduce the singular triplets $\{\sigma_j^{(B_{m,m+1})}, u_j^{(B_{m,m+1})}, v_j^{(B_{m,m+1})}\}$, $1 \leq j \leq m$, of the matrix $B_{m,m+1}$ and let them be enumerated so that

$$0 < \sigma_1^{(B_{m,m+1})} \leq \sigma_2^{(B_{m,m+1})} \leq \ldots \leq \sigma_m^{(B_{m,m+1})}. \tag{3.23}$$

This enumeration differs from the one for the singular values of B_m, cf. (2.9), because in the present subsection we are concerned with the computation of the $k < m$ smallest singular triplets of A. Throughout this subsection, we use the following simplified notation

$$\sigma_j' = \sigma_j^{(B_{m,m+1})}, \qquad u_j' = u_j^{(B_{m,m+1})}, \qquad v_j' = v_j^{(B_{m,m+1})}.$$

The k smallest singular triplets of $B_{m,m+1}$ determine the matrices

$$\begin{aligned} U_k' &:= [u_1', u_2', \ldots, u_k'] \in \mathbb{R}^{m\times k}, \\ V_k' &:= [v_1', v_2', \ldots, v_k'] \in \mathbb{R}^{(m+1)\times k}, \\ \Sigma_k' &:= \mathrm{diag}[\sigma_1', \sigma_2', \ldots, \sigma_k'] \in \mathbb{R}^{k\times k}, \end{aligned} \tag{3.24}$$

where U_k' and V_k' have orthonormal columns, and

$$B_{m,m+1}V_k' = U_k'\Sigma_k', \qquad B_{m,m+1}^T U_k' = V_k'\Sigma_k'. \tag{3.25}$$

We refer to (3.25) as a partial singular value decomposition of $B_{m,m+1}$. It follows from (3.22) and (3.25) that $\{(\sigma_j')^2, u_j'\}$ is an eigenpair of (3.21), and (3.20) shows that $\{(\sigma_j')^2, B_m^{-1}u_j'\}$ is an eigenpair of (3.19). Thus, the eigenpairs of (3.19) and (3.21) associated with the k smallest eigenvalues can be determined from the partial singular value decomposition (3.25). Gu and Eisenstat [13] describe how the singular value decomposition of $B_{m,m+1}$ can be computed by updating the singular value decomposition of B_m; see also Bunch and Nielsen [7]. However, when A is large and m is small, the computational effort required for determining the singular value decompositions of B_m and $B_{m,m+1}$ is negligible. We will therefore not dwell on the computation of (3.25).

The harmonic Ritz vector of A^TA associated with the harmonic Ritz value $\hat{\theta}_j$ is given by

$$\hat{v}_j := P_m\hat{w}_j, \tag{3.26}$$

see, e.g., [25, 28]. Morgan [27] recently pointed out that the residual errors associated with different harmonic Ritz pairs $\{\hat{\theta}_j, \hat{v}_j\}$ are parallel. We show this result for the problem at hand, because this property is central for our augmentation method. Thus, using (2.4), (3.20), (3.21), and (3.26), we obtain

$$\begin{aligned} A^TA\hat{v}_j - \hat{\theta}_j\hat{v}_j &= A^TAP_m\hat{w}_j - \hat{\theta}_jP_m\hat{w}_j \\ &= (P_mB_m^TB_m + \alpha_m r_m e_m^T)\hat{w}_j - \hat{\theta}_jP_m\hat{w}_j \\ &= P_m(B_m^TB_m - \hat{\theta}_jI_m)\hat{w}_j + \alpha_m r_m e_m^T\hat{w}_j \\ &= P_mB_m^{-1}(B_mB_m^T - \hat{\theta}_jI_m)w_j + r_m e_m^Tw_j \\ &= P_mB_m^{-1}(-\beta_m^2 e_m e_m^Tw_j) + r_m e_m^Tw_j \\ &= e_m^Tw_j(r_m - \beta_m^2P_mB_m^{-1}e_m). \end{aligned}$$

It is convenient to define the scaled residual vector

$$\hat{r}_m := p_{m+1} - \beta_mP_mB_m^{-1}e_m, \tag{3.27}$$

where we have used (2.2).

We are in a position to derive relations analogous to (3.7) and (3.11) for harmonic Ritz vectors. Equations (3.20), (3.26), and (3.27) yield

$$[\hat{v}_1\sigma'_1, \hat{v}_2\sigma'_2, \ldots, \hat{v}_k\sigma'_k, \hat{r}_m] = P_{m+1}\begin{bmatrix} B_m^{-1}U'_k\Sigma'_k & -\beta_m B_m^{-1}e_m \\ 0 & 1 \end{bmatrix},$$

where the matrix P_{m+1} is the same as in (2.3). Introduce the QR-factorization

$$\begin{bmatrix} B_m^{-1}U'_k\Sigma'_k & -\beta_m B_m^{-1}e_m \\ 0 & 1 \end{bmatrix} = Q'_{k+1}R'_{k+1}, \tag{3.28}$$

where $Q'_{k+1} \in \mathbb{R}^{(m+1)\times(k+1)}$ has orthonormal columns and $R'_{k+1} \in \mathbb{R}^{(k+1)\times(k+1)}$ is upper triangular.

Consider the matrix

$$\hat{P}_{k+1} = [\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_{k+1}] := P_{m+1}Q'_{k+1}. \tag{3.29}$$

Its columns $\hat{p}_j$ are orthonormal since both P_{m+1} and Q'_{k+1} have orthonormal columns. The equations (1.3) and (3.28) yield

$$\begin{aligned} A\hat{P}_{k+1} &= [AP_m, Ap_{m+1}]Q'_{k+1} \\ &= [Q_m B_m, Ap_{m+1}]\begin{bmatrix} B_m^{-1}U'_k\Sigma'_k & -\beta_m B_m^{-1}e_m \\ 0 & 1 \end{bmatrix}(R'_{k+1})^{-1} \\ &= [Q_m U'_k\Sigma'_k, -\beta_m q_m + Ap_{m+1}](R'_{k+1})^{-1}. \end{aligned}$$

In the derivation of our method, we only assume that the matrix B_m is upper triangular, because B_m has this property after restarts. Below we comment on possible simplifications when B_m is upper bidiagonal.

Let

$$\hat{Q}_k := Q_m U'_k. \tag{3.30}$$

The columns of this matrix are orthonormal, and we define the Fourier coefficients

$$\hat{c}_k = [\hat{\gamma}_1, \hat{\gamma}_2, \ldots, \hat{\gamma}_k]^T := \hat{Q}_k^T(-\beta_m q_m + Ap_{m+1}).$$

The vector

$$\hat{\alpha}_{k+1}\hat{q}_{k+1} := -\beta_m q_m + Ap_{m+1} - \hat{Q}_k\hat{c}_k$$

is orthogonal to the columns of $\hat{Q}_k$ and the scaling factor $\hat{\alpha}_{k+1} > 0$ is chosen so that $\hat{q}_{k+1}$ is of unit length. It follows that

$$A\hat{P}_{k+1} = \hat{Q}_{k+1}\hat{B}_{k+1}, \tag{3.31}$$

where

$$\hat{Q}_{k+1} := [\hat{Q}_k, \hat{q}_{k+1}] \in \mathbb{R}^{\ell\times(k+1)}, \tag{3.32}$$

$$\hat{B}_{k+1} := \begin{bmatrix} \sigma'_1 & & & 0 & \hat{\gamma}_1 \\ & \sigma'_2 & & & \hat{\gamma}_2 \\ & & \ddots & & \vdots \\ & & & \sigma'_k & \hat{\gamma}_k \\ 0 & & & & \hat{\alpha}_{k+1} \end{bmatrix}(R'_{k+1})^{-1} \in \mathbb{R}^{(k+1)\times(k+1)}. \tag{3.33}$$

Thus, $\hat{Q}_{k+1}$ has orthonormal columns and $\hat{B}_{k+1}$ is the product of two upper triangular matrices, one of which has nonzero entries only on the diagonal and in the last column. In particular, $\hat{B}_{k+1}$ is upper triangular. The decomposition (3.31) is the desired analogue of (3.7). When B_m is given by (1.6), one can show that $\hat{\alpha}_{k+1} = \alpha_{m+1}$, $\hat{q}_{k+1} = q_{m+1}$, and $\hat{c}_k = 0$.

We now derive an analogue of the decomposition (3.11). Let $\hat{Q}_k$ be given by (3.30). Then equation (3.25) yields

$$A^T \hat{Q}_k = A^T Q_m U_k' = P_{m+1} B_{m,m+1}^T U_k' = P_{m+1} V_k' \Sigma_k'. \tag{3.34}$$

It follows from the decomposition on the left-hand side of (3.25) that

$$[I_m, \beta_m B_m^{-1} e_m] V_k' = B_m^{-1} U_k' \Sigma_k',$$

and therefore

$$V_k' = \begin{bmatrix} B_m^{-1} U_k' \Sigma_k' & -\beta_m B_m^{-1} e_m \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I_k \\ e_{m+1}^T V_k' \end{bmatrix}. \tag{3.35}$$

Substituting (3.35) into (3.34), using (3.28) and (3.29), gives

$$A^T \hat{Q}_k = P_{m+1} Q_{k+1}' R_{k+1}' \begin{bmatrix} I_k \\ e_{m+1}^T V_k' \end{bmatrix} \Sigma_k' = \hat{P}_{k+1} R_{k+1}' \begin{bmatrix} I_k \\ e_{m+1}^T V_k' \end{bmatrix} \Sigma_k'. \tag{3.36}$$

Let $\hat{B}_{k,k+1}$ be the leading $k \times (k+1)$ submatrix of the upper triangular matrix $\hat{B}_{k+1}$ in (3.31). Then (3.31) yields

$$\hat{Q}_k^T A \hat{P}_{k+1} = \hat{B}_{k,k+1}. \tag{3.37}$$

It follows from (3.36) that

$$\hat{P}_{k+1}^T A^T \hat{Q}_k = R_{k+1}' \begin{bmatrix} I_k \\ e_{m+1}^T V_k' \end{bmatrix} \Sigma_k',$$

and comparison with (3.37) shows that

$$R_{k+1}' \begin{bmatrix} I_k \\ e_{m+1}^T V_k' \end{bmatrix} \Sigma_k' = \hat{B}_{k,k+1}^T.$$

Hence, equation (3.36) can be expressed as

$$A^T \hat{Q}_k = \hat{P}_{k+1} \hat{B}_{k,k+1}^T. \tag{3.38}$$

We turn to the last column, $A^T \hat{q}_{k+1}$, of $A^T \hat{Q}_{k+1}$. Equation (3.31) yields

$$\hat{P}_{k+1}^T A^T \hat{q}_{k+1} = \hat{B}_{k+1}^T \hat{Q}_{k+1}^T \hat{q}_{k+1} = \hat{B}_{k+1}^T e_{k+1} = \hat{\alpha}_{k+1} e_{k+1},$$

where $\hat{\alpha}_{k+1}$ denotes the last diagonal entry of $\hat{B}_{k+1}$. Thus,

$$A^T \hat{q}_{k+1} = \hat{\alpha}_{k+1} \hat{p}_{k+1} + \breve{r}_{k+1}, \tag{3.39}$$

where $\hat{P}_{k+1}^T \breve{r}_{k+1} = 0$. Combining (3.38) and (3.39) yields

$$A^T \hat{Q}_{k+1} = \hat{P}_{k+1} \hat{B}_{k+1}^T + \breve{r}_{k+1} e_{k+1}^T, \tag{3.40}$$

which is the desired analogue of (3.11). Note that the residual vector $\breve{r}_{k+1}$ can be computed from equation (3.39), because the other terms are explicitly known.

Let $\hat{\beta}_{k+1} := \|\breve{r}_{k+1}\|$ and $\hat{p}_{k+2} := \breve{r}_{k+1}/\hat{\beta}_{k+1}$, and define

$$\hat{P}_{k+2} := [\hat{P}_{k+1}, \hat{p}_{k+2}], \qquad \breve{B}_{k+1,k+2} := [\hat{B}_{k+1}, \hat{\beta}_{k+1} e_{k+1}].$$

Then (3.40) can be written in the form

$$A^T \hat{Q}_{k+1} = \hat{P}_{k+2} \breve{B}_{k+1,k+2}^T,$$

which is an analogue of (2.3).

Given the decompositions (3.31) and (3.40), we can proceed analogously as in Subsection 3.1 to compute the decompositions

$$A\hat{P}_m = \hat{Q}_m \hat{B}_m, \qquad A^T \hat{Q}_m = \hat{P}_m \hat{B}_m^T + \breve{r}_m e_m^T, \tag{3.41}$$

where $\hat{P}_m \in \mathbb{R}^{n\times m}$ has orthonormal columns with leading $n \times (k+2)$ submatrix $\hat{P}_{k+2}$, $\hat{Q}_m \in \mathbb{R}^{\ell\times m}$ has orthonormal columns with leading $\ell \times (k+1)$ submatrix $\hat{Q}_{k+1}$, and

$$\hat{B}_m = \begin{bmatrix} \hat{B}_{k+1} & \hat{\beta}_{k+1} & & & & 0 \\ & \hat{\alpha}_{k+2} & \hat{\beta}_{k+2} & & & \\ & & \hat{\alpha}_{k+3} & \hat{\beta}_{k+3} & & \\ & & & & \ddots & \\ & & & & \ddots & \hat{\beta}_{m-1} \\ 0 & & & & & \hat{\alpha}_m \end{bmatrix} \in \mathbb{R}^{m\times m}$$

has the $(k+1)\times(k+1)$ leading principal submatrix $\hat{B}_{k+1}$ and an $(m-k-1)\times(m-k-1)$ upper bidiagonal trailing principal submatrix. The residual vector $\breve{r}_m$ is orthogonal to the columns of $\hat{P}_m$. The method of this subsection can be restarted by letting $B_m := \hat{B}_m$, $P_m := \hat{P}_m$, $Q_m := \hat{Q}_m$, $r_m := \breve{r}_m$, and $\beta_m := \|\breve{r}_m\|$.

We remark that the accurate computation of the vector $B_m^{-1} e_m$, used in (3.28), can be difficult when the matrix B_m has a large condition number $\kappa(B_m) := \sigma_m^{(B_m)}/\sigma_1^{(B_m)}$, where the singular values are enumerated as in (3.23). In this case, we switch from augmentation by harmonic Ritz vectors to augmentation by Ritz vectors. Details are provided in the following subsection.

3.3. An augmented Lanczos bidiagonalization algorithm. We describe an algorithm for the computation of a few of the largest or smallest singular triplets of a large matrix A. The algorithm is based on augmentation by Ritz vectors or harmonic Ritz vectors as described in the previous subsections. The Boolean variable *harmonic* suggests the type of augmentation used. If *harmonic* is true and B_m is not too ill-conditioned, then the augmentation scheme of Subsection 3.2 is applied; otherwise augmentation is carried out according to Subsection 3.1.

ALGORITHM 3.1. AUGMENTED LANCZOS BIDIAGONALIZATION

Input: $A \in \mathbb{R}^{\ell\times n}$ *or functions for evaluating matrix-vector products with the matrices* A *and* A^T,
$p_1 \in \mathbb{R}^n$: *initial vector of unit length,*
m : *number of bidiagonalization steps,*

k : number of desired singular triplets,
δ : tolerance for accepting computed approximate singular triplet, cf. (2.13),
ϵ : machine epsilon,
harmonic : Boolean variable that suggests type of augmentation; see above.

Output: Computed set of approximate singular triplets $\{\sigma_j, u_j, v_j\}_{j=1}^{k}$ of A.

1. *Compute the partial Lanczos bidiagonalization (1.3)-(1.4) using Algorithm 2.1.*
2. *Compute the singular value decomposition (2.10) of B_m.*
3. *Check convergence: If all k desired singular triplets satisfy (2.13) then exit.*
4. *Compute the augmenting vectors:*
   4a. if not harmonic or $\kappa(B_m) > \epsilon^{-1/2}$ then
   Determine the matrices $P := \tilde{P}_{k+1}$; $Q := \tilde{Q}_{k+1}$, $B := \tilde{B}_{k+1}$, and vector $r := \tilde{f}_{k+1}$ by (3.2), (3.5), (3.6), and (3.10), respectively.
   4b. if harmonic and $\kappa(B_m) \leq \epsilon^{-1/2}$ then
   Compute the partial singular value decomposition (3.24) of $B_{m,m+1}$ and the QR-factorization (3.28).
   Determine the matrices $P := \hat{P}_{k+1}$, $Q := \hat{Q}_{k+1}$, $B := \hat{B}_{k+1}$, and vector $r := \check{r}_{k+1}$ by (3.29), (3.32), (3.33), and (3.39), respectively.
5. *The available matrices P, Q, B, and the vector r satisfy*

$$AP = QB, \qquad A^T Q = PB^T + re_k^T.$$

   Append $m-k$ columns to the matrices P and Q, and $m-k$ rows and columns to the matrix B. Denote the matrices so obtained by P_m, Q_m, and B_m, respectively. Determine a new residual vector and denote it by r_m.
6. *Goto 2.*

The above algorithm is a simplification of the actual computations carried out. For instance, the algorithm exits when a matrix B_m that is numerically singular has been detected, but the available decompositions of A can be used to determine singular triplets of A; see the discussion in Section 1. Moreover, the number of augmented vectors used at each restart typically is larger than the number of desired singular triplets. Assume that k' of the desired k singular triplets have been found. We then augment by $k + k''$ (instead of k) singular triplets, where k'' is chosen as large as possible, such that $k'' \leq k'$ and $k + k'' \leq m - 3$. The term -3 secures that at least 3 orthogonalization steps can be carried out between restarts. This approach has been advocated by Lehoucq [21] in the context of the implicitly restarted Arnoldi method. It often yields faster convergence without increasing the memory requirement. Finally, our implementation enforces two-sided reorthogonalization when $\kappa(B_m) > \epsilon^{-1/2}$. MATLAB code is available at the authors' home pages.

4. Numerical examples. All computations were carried out using MATLAB version 6.5 R13 on a Dell 530 workstation with two 2.4 GHz (512k cache) Xeon processors and 2 GB (400 MHz) of memory running under the Windows XP operating system. Machine epsilon is $\epsilon = 2.2 \cdot 10^{-16}$.

We compare our methods, outlined by Algorithm 3.1, with methods recently proposed by Hochstenbach [15, 14], Kokiopoulou et al. [18], as well as with MATLAB's internal function `svds`, and the scheme proposed in [2]. Hochstenbach [15, 14] presents a Jacobi-Davidson method. This is a powerful scheme when a good preconditioner for the linear system of equations that has to be solved is available. In our computed

examples, we assume that no good preconditioner is known, and apply the method without preconditioner. The linear system of equations is solved by the GMRES iterative method. The MATLAB implementation[1] `jdsvd` offers several extraction choices, such as standard, u-harmonic, v-harmonic, double-harmonic, and refined. In our numerical examples, refined extraction often gave best accuracy. This is consistent with results reported by Hochstenbach [15]. We refer to the Jacobi-Davidson method with refined extraction as `jdsvd(Ref)`. The code `jdsvd` is still under development and we used the version available to us at the time of the numerical experiments.

Our methods are mathematically, but not numerically, equivalent to the methods proposed by Kokiopoulou et al. [18]. We used the MATLAB implementation `irlanb`[2] by Kokiopoulou et al. [18] in our comparison. The code `irlanb` calls Larsen's MATLAB code `lanbpro` [19] to compute partial Lanczos bidiagonalizations with partial reorthogonalization. `irlanb` is designed for computing a few of the smallest singular triplets, but not for computing a few of the largest ones. The code therefore is not used in Examples 4 and 5 below. Ritz values and harmonic Ritz values can be used as shifts. We refer to `irlanb` with these shift selections as `irlanb(R)` and `irlanb(H)`, respectively. The code `irlanb` is still under development and we report results for the version available to us at the time of the numerical experiments.

The internal MATLAB function `svds` uses FORTRAN codes of ARPACK [23]. It calls an eigenvalue routine to compute eigenvalue-eigenvector pairs associated with positive eigenvalues of the symmetric matrix

$$Z := \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \in \mathbb{R}^{(\ell+n)\times(\ell+n)}. \tag{4.1}$$

The matrix Z has the eigenvalues

$$\pm\sigma_1^{(A)}, \pm\sigma_2^{(A)}, \ldots, \pm\sigma_n^{(A)},$$

as well as $\ell-n$ zero eigenvalues, where we as usual assume that $\ell \geq n$. The eigenvectors of Z yield both the right and left singular vectors of A. The method requires the computation of eigenpairs associated with eigenvalues near zero when determining the smallest singular triplets of A. This can be difficult when Z has many positive and negative eigenvalues of large magnitude. For this reason, the `svds` function uses a shift-and-invert approach for computing the smallest singular triplets. This requires factorization of the matrix Z, and therefore the function `svds` typically demands much more storage than the other methods in our comparison. Since `svds` does not yield the number of linear systems of equations that have to be solved, we only report the CPU-time required.

The MATLAB code `irblsvds` implements an implicitly restarted block-Lanczos method applied to the matrix (4.1) for the computation of a few singular triplets of A. The method and code are described in [2, 3].[3] The matrix Z is only used for evaluation of matrix-vector products; in particular, the matrix Z is not factored. The code `irblsvds` generally is not well suited for computing the smallest singular triplets of A when $\ell - n$ is large, because then Z has $\ell - n$ zero eigenvalues and the code may determine these eigenvalues and associated eigenvectors instead of eigenpairs associated with tiny singular triplets of A. Unless indicated otherwise, we use the

[1] Code is available at http://www.case.edu/artsci/math/hochstenbach/software/jdsvd.html
[2] Code is available at http://www.hpclab.ceid.upatras.gr/scgroup/software/software.html
[3] Code is available at the authors' home pages.

default value 3 for the block-size in our experiments with `irblsvds`. The largest number of consecutive block-Lanczos steps is chosen so that `irblsvds` has about the same storage requirement as the other codes. We note that of the methods used in the examples of this section, only the ones of the present paper are based on augmented matrix formulations.

TABLE 4.1
Parameters for `irlba`.

$adjust$	Initial number of vectors added to the k restart vectors to speed up convergence. Default value: $adjust = 3$.
aug	A 4-letter string. The value RITZ yields the augmentation described in Subsection 3.1; the value HARM gives augmentation according to Subsection 3.2. Default value: aug = HARM if $sigma$ = SS, and aug = RITZ if $sigma$ = LS.
$disps$	When $disps > 0$, available approximations of the k desired singular values and norms of associated residual errors are displayed each iteration; $disps = 0$ inhibits display of these quantities. Default value: $disps = 0$.
k	Number of desired singular triplets. Default value: $k = 6$.
$maxit$	Maximum number of restarts. Default value: $maxit = 100$.
$steps$	Maximum number of Lanczos bidiagonalization steps. The parameter specifies the largest value of steps m in (1.3)-(1.4) and determines the storage requirement of the method. Default value: $steps = 20$.
$reorth$	A 3-letter string. The value ONE yields one-sided full reorthogonalization on the "shorter" vectors; the value TWO gives two-sided full reorthogonalization. When our available estimate of $\kappa(A)$, see the discussion following (2.13), is larger than $\epsilon^{-1/2}$, two-sided full reorthogonalization is used. Default value: $reorth$ = ONE.
$sigma$	A 2-letter string (SS for smallest and LS for largest) which specifies which extreme singular triplets are to be computed. Default value: $sigma$ = LS.
δ	Tolerance used for convergence check; see (2.13). Default value: $\delta = 10^{-6}$.
v_0	Initial vector for Lanczos bidiagonalization. When $\ell \geq n$, $p_1 := v_0$; cf. Algorithm 2.1. Default value: v_0 is a random vector with normally distributed entries.

The schemes of Subsections 3.1 and 3.2 are implemented by the MATLAB code `irlba`.[3] The execution of `irlba` is determined by certain user-specified parameters; see Table 4.1. We refer to the augmentation method of Subsection 3.1, based on Ritz vectors, as `irlba(R)`, and to the augmentation method of Subsection 3.2, based on harmonic Ritz vectors, as `irlba(H)`. The scheme `irlba(R)` with one- and two-sided full reorthogonalization is referred to as `irlba(R1)` and `irlba(R2)`, respectively. The analogous implementations of `irlba(H)` are denoted by `irlba(H1)` and `irlba(H2)`. One-sided full reorthogonalization reorthogonalizes the columns of the smaller one of the matrices P_m and Q_m. Thus, when $\ell \geq n$, the columns of P_m are reorthogonalized.

The codes `irblsvds`, `irlanb`, `jdsvd`, and `svds` allow a user to choose numerous parameters that affect their performance. Unless stated otherwise, we use the default

values for the parameters. Except for the function `svds`, we choose the parameters that affect storage so that all codes require about the same maximum computer storage.

It is impossible to use the same starting vector for all the methods, since some routines work with A and A^T, others with the matrix Z. To make our comparison less dependent on the choice of starting vector, we record the best results for each method over 5 runs using default random starting vector(s) generated by each code. We use the same starting vector when the same method is applied with different parameter values. The reported number of matrix-vector products is the total number of matrix-vector product evaluations with A and A^T for `irlba`, `irlanb`, and `jdsvd`, and with Z for `irblsvds`.

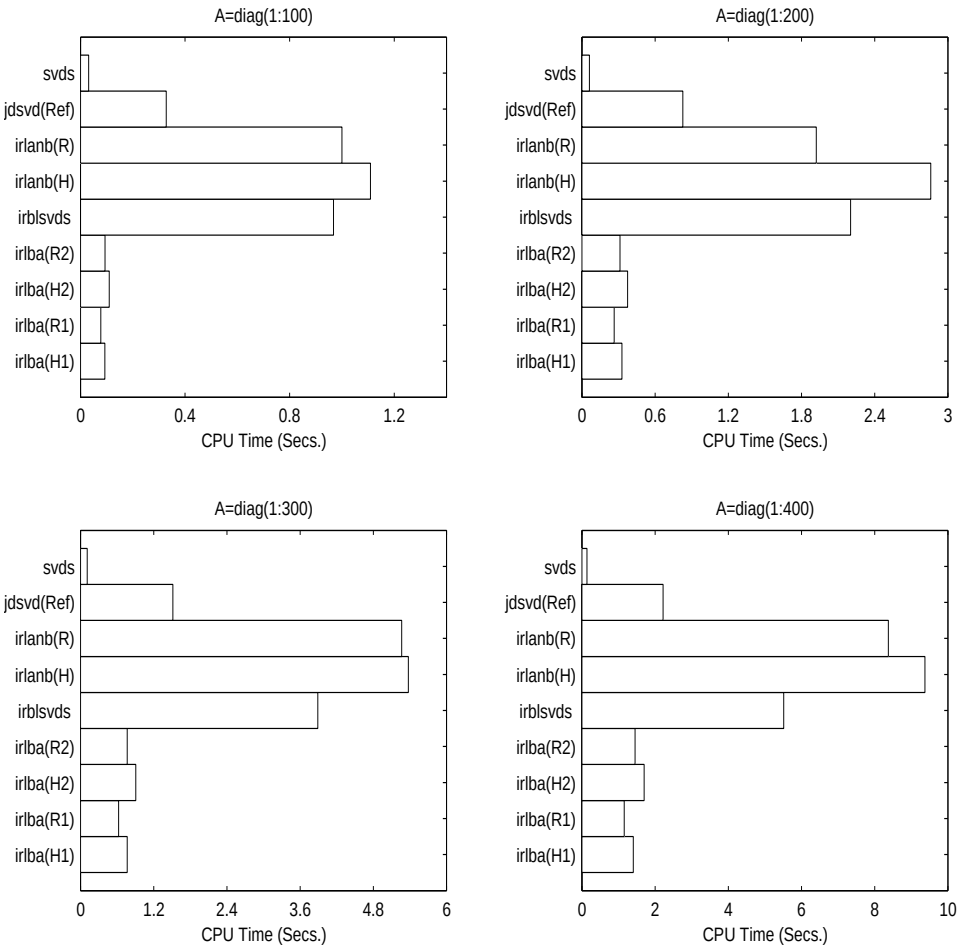

FIG. 4.1. *Example 1: CPU times for computing the smallest singular triplet.*

Example 1. (Smallest singular value). We would like to compute the smallest singular triplet of the diagonal matrices

$$A := \mathrm{diag}[1, 2, \ldots, n], \qquad n = 100, 200, 300, 400.$$

Each code was instructed to determine only one singular triplet, the smallest one. This corresponds to the parameter values $sigma$ = SS and $k = 1$ for `irlba`. We allowed each one of the codes `irlba`, `irlanb`, `jdsvd`, and `irblsvds` about the amount of storage required for carrying out 20 Lanczos bidiagonalization steps. In particular, the largest number of consecutive block-Lanczos steps allowed by `irblsvds` was limited to 7 (with block-size 3). This corresponds to about the same storage requirement as 21 Lanczos bidiagonalization steps. We let $adjust := 4$ for `irbla`, see Table 4.1. This forces both the `irlba` and `irlanb` codes to apply the same number of bidiagonalization steps in the first restart. We chose the tolerance 10^{-6} for all codes, i.e., $\delta := 10^{-6}$ in (2.13). For the `jdsvd` code we only report refined extraction, because this extraction method was the fastest. Figure 4.1 displays the CPU times required for the different methods. MATLAB's `svds` function is seen to be fastest. This may depend on that `svds` is not coded in MATLAB, and that it is based on a shift-and-invert approach. For large problems shift-and-invert may require unacceptable amounts of memory and execution time; however, when computation and storage of factors of matrices of the form $Z - \tau I$, where τ is a scalar, is feasible, then this is approach is attractive. Among the methods that only use the matrices A and A^T for evaluating matrix-vector products, `irlba` is seen to be competitive. □

Example 2. (Smallest singular values). We are interested in determining the smallest singular triplets of four matrices generated in MATLAB with the commands

$$A = randn(n), \qquad A(:,1) = A(:,10),$$

for $n = 200, 400, 600, 800$. Here $A = randn(n)$ determines an $n \times n$ matrix with normally distributed entries with mean zero and variance one. The command $A(:,1) = A(:,10)$ overwrites column 1 by column 10 and secures that the matrix obtained has a zero singular value. This example illustrates that augmentation by harmonic Ritz vectors can give substantially faster convergence than augmentation by Ritz vectors.

The code `irlba` was used with tolerance $\delta := 10^{-16}$, two-sided reorthogonalization, and carried out up to 30 consecutive Lanczos bidiagonalization vectors between restart ($start := 30$). Figure 4.2 shows the convergence to zero of the smallest computed singular value. The vertical axis displays the absolute error in the smallest computed singular value, i.e., the smallest computed singular value, and the horizontal axis shows the number of restarts. The cross-over label indicates when `irlba` switched from augmenting with harmonic Ritz vectors to augmenting with Ritz vectors. As mentioned in Subsection 3.2, augmentation by harmonic Ritz vectors requires the solution of a linear system of equations with the matrix B_m (while augmentation by Ritz vectors does not). We switch from augmentation by harmonic Ritz vectors to augmentation by Ritz vectors when the condition number of the matrix B_m is larger than the square-root of the reciprocal of machine epsilon; cf. Algorithm 3.1.

We remark that computing singular triplets with numerically vanishing singular values is not always possible since the approximated left singular vectors $\tilde{u}_j^{(A)}$ are obtained from the Krylov subspace (2.8), which is restricted to the range of A. However, in presence of round-off errors, `irlba` often is able to successfully compute singular triplets associated with zero singular values. □

Example 3. (Smallest singular values). We consider the 1033×320 matrix WELL1033 and the 1850×712 matrix WELL1850 from the set LSQ in the Harwell-Boeing Sparse Matrix Collection [10]. These matrices arise from surveying problems. We would like to determine the 6 smallest singular triplets of these matrices, and select the parameters for the different codes accordingly. For instance, for `irlba` we

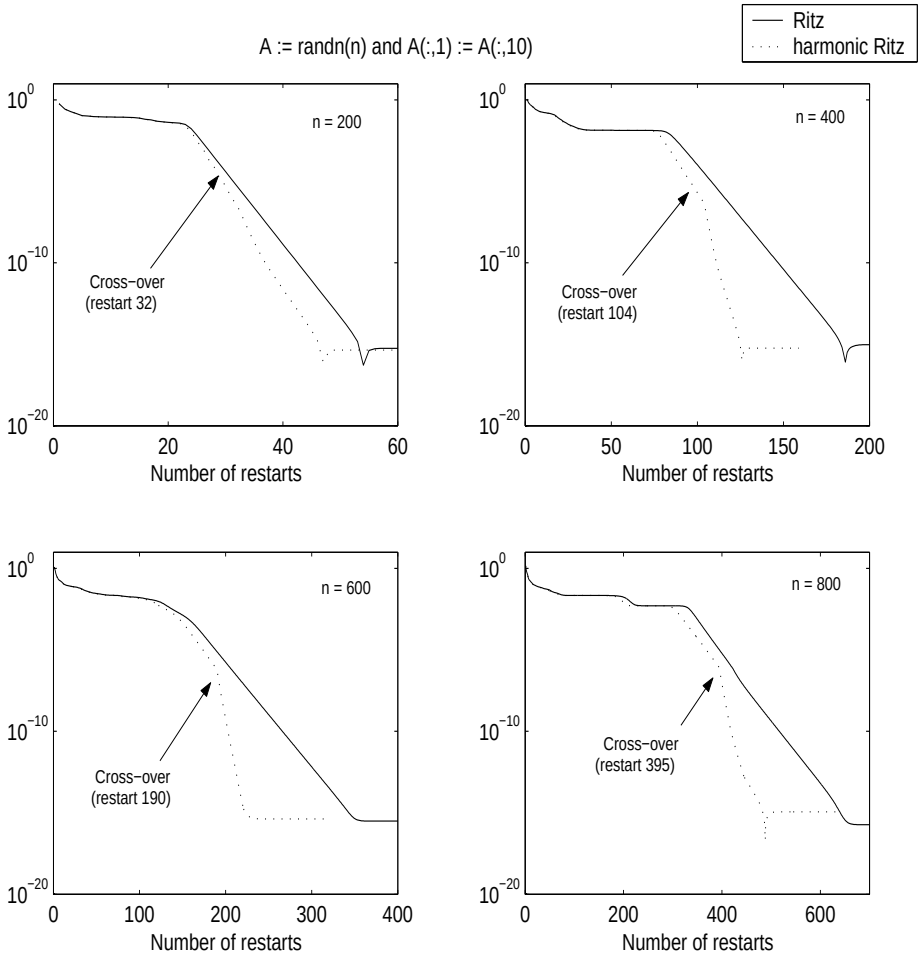

FIG. 4.2. *Example 2: Absolute error in the smallest singular value computed by* `irlba` *versus number of restarts using augmentation with Ritz vectors (solid curve) and augmentation with harmonic Ritz vectors (dotted curve). After the cross-over point augmentation is with Ritz vectors.*

let $sigma$:= SS and $k := 6$. All codes are allowed the amount of storage required for 40 Lanczos bidiagonalization steps. The number of implicit QR-steps in `irlanb` is chosen to be 31; this choice is consistent with the default value of the parameter *adjust* of `irlba`. It forces `irbla` and `irlanb` to apply the same number of Lanczos bidiagonalization steps in the first restart. The tolerance for all methods is set to $\delta = 10^{-6}$. `irlanb` with augmentation by harmonic Ritz vectors gave better results than augmentation by Ritz vectors. We therefore only report results for the former. The restarted block-Lanczos method `irblsvds` used block-size 4 and was allowed to carry out at most 10 consecutive block-Lanczos steps between restarts. `jdsvd` gave the best results for refined extraction. Therefore we do not report results for other extraction methods. MATLAB's `svds` function was unable to find any of the desired singular triplets to requested accuracy for either one of the matrices. Table 4.2 summarizes the computed results. The table shows that `irlba` was able to compute

TABLE 4.2

Example 3: Computation of the 6 smallest singular triplets of the matrices WELL1033 and WELL1850.

`irlba(H1)`

matrix	# matrix-vector products	CPU time	magnitude of largest error
WELL1033	638	0.36s	$2.41 \cdot 10^{-15}$
WELL1850	1442	1.39s	$1.72 \cdot 10^{-13}$

`irlba(H2)`

matrix	# matrix-vector products	CPU time	magnitude of largest error
WELL1033	638	0.59s	$2.14 \cdot 10^{-15}$
WELL1850	1442	2.41s	$1.72 \cdot 10^{-13}$

`irblsvds`

matrix	# matrix-vector products	CPU time	magnitude of largest error
WELL1033	7520	18.67s	$8.53 \cdot 10^{-16}$
WELL1850	16080	77.53s	$1.11 \cdot 10^{-15}$

`irlanb(H)`

matrix	# matrix-vector products	CPU time	magnitude of largest error
WELL1033*	–	–	–
WELL1850	1578	8.31s	$3.03 \cdot 10^{-10}$

* Method failed to convergence.

`jdsvd(Ref)`

matrix	# matrix-vector products	CPU time	magnitude of largest error
WELL1033	1978	2.66s	$4.15 \cdot 10^{-10}$
WELL1850	4244	7.69s	$6.84 \cdot 10^{-12}$

all 6 singular values with the fewest matrix-vector product evaluations and the least CPU time for both matrices. □

Example 4 (Largest singular values). The matrices MEDLINE, CRANFIELD, and CISI are standard term-by-document test matrices and can be obtained from the Cornell SMART FTP server[4] or the TMG web page[5]. They are of size 5735×1033, 4563×1398, and 5544×1460, respectively. We also consider the term-by-document

[4] `ftp://ftp.cs.cornell.edu/pub/smart.`

[5] `http://scgroup.hpclab.ceid.upatras.gr/scgroup/Projects/TMG/.`

matrix HYPATIA[6] of size 11390×1265 with 109056 non-zero terms from the web server at the Department of Mathematics, University of Rhode Island. HYPATIA was created in the same manner as standard term-by-document test matrices. All test matrices use the local term frequency (i.e., number of times a word occurs on a website) and have no global weighting or normalization. The parameter values chosen for the different methods are consistent with our desire to determine the 10 largest singular triplets for each one of these matrices; for `irlba` we let *sigma* :=LS and $k := 10$. The available storage is assumed to be large enough to simultaneously store all vector generated during 20 consecutive steps of Lanczos bidiagonalization. The tolerance δ in the stopping criterion is 10^{-6}. Since we seek to determine the largest singular triplets, `irbla` uses the augmentation method of Section 3.1. We report the results for one-sided and two-sided full reorthogonalization. The code `irblsvds` was used with block-size 4 and was allowed to carry out at most 5 consecutive block-Lanczos steps between restarts. The fastest extraction method for `jdsvd` was refined. Table 4.3 displays the performance of the methods and illustrates the competitiveness of `irlba`. The code `irlanb` is not part of our comparison, since it is designed for the computation a few of the smallest singular triplets, only. □

Example 5 (Condition number). We would like to determine the condition number $\kappa(A) := \sigma_1^{(A)}/\sigma_n^{(A)}$, where the singular values are enumerated according to (1.1), of the Läuchli matrix $A := L(n, \mu) \in \mathbb{R}^{(n+1)\times n}$ for $n = 20000$ and the default value of μ.[7] Thus, A has ones across the top row and μ on the subdiagonal; the remaining matrix entries are zero. This matrix often is used to illustrate the drawback of forming A^TA in least-squares computations; see [19]. The Läuchli matrix A is nonsingular; it has the simple singular value $\sqrt{n+\mu^2}$ and the singular value μ of multiplicity $n-1$ giving the condition number $\kappa(A) = 9.490724975767860 \cdot 10^9$, and therefore A^TA is numerically singular. `irlba` only implicitly works with the matrix A^TA and is able to compute $\kappa(A)$. Specifically, we let *sigma* :=LS and SS, $k := 1$, and allow the storage required for all vectors generated by 20 consecutive steps of Lanczos bidiagonalization. The tolerance δ in the stopping criterion is set to machine epsilon ϵ. Two-sided full reorthogonalization was employed. We augmented by Ritz vectors when computing the largest singular value and, until the matrices B_m became ill-conditioned, by harmonic Ritz vectors when determining the smallest singular value. `irlba` required 0.703 and 0.796 seconds of CPU time to compute the largest and smallest singular values of A, respectively, and gave the condition number $9.490724975767925 \cdot 10^9$ with a relative error of $6.83 \cdot 10^{-15}$. Thus, the fact that A^TA is numerically singular does not cause `irlba` problems. □

5. Acknowledgments. We would like thank Stratis Gallopoulos for a copy of the paper [18], Michiel Hochstenbach and Efi Kokiopoulou for providing copies of their codes, and Costas Bekas, Michiel Hochstenbach, and Efi Kokiopoulou for valuable comments and discussions.

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.

[6] The matrix is available at http://math.uri.edu/∼jbaglama.

[7] The Läuchli matrix can be obtained from the Matrix Market website, http://math.nist.gov/MatrixMarket/deli/Lauchli/. We used $\mu = 1.4901006677403 \cdot 10^{-8}$.

[2] J. Baglama, D. Calvetti, and L. Reichel, *IRBL: An implicitly restarted block Lanczos method for large-scale Hermitian eigenproblems*, SIAM J. Sci. Comput., 24 (2003), pp. 1650–1677.
[3] J. Baglama, D. Calvetti, and L. Reichel, *Algorithm 827: irbleigs: A MATLAB program for computing a few eigenpairs of a large sparse Hermitian matrix*, ACM Trans. Math. Software, 29 (2003), pp. 337–348.
[4] M. W. Berry, SVDPACK, http://www.netlib.org/svdpack/ (1991).
[5] Å. Björck, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.
[6] Å. Björck, E. Grimme, and P. Van Dooren, *An implicit shift bidiagonalization algorithm for ill-posed systems*, BIT, 34 (1994), pp. 510–534.
[7] J. R. Bunch and C. P. Nielsen, *Updating the singular decomposition*, Numer. Math., 31 (1978), pp. 111–129.
[8] 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.
[9] P. Comon and G. H. Golub, *Tracking a few extreme singular values and vectors in signal processing*, Proc. IEEE, 78 (1990), pp. 1327–1343.
[10] I. S. Duff, R. G. Grimes, and J. G. Lewis, *User's guide for the Harwell-Boeing sparse matrix collection (Release I)*, Technical Report TR/PA/92/86, CERFACS, Toulouse, France, 1992. The matrices are available at http://math.nist.bov/MatrixMarket/
[11] G. Golub and W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, J. SIAM Numer. Anal., Ser. B, 2 (1965), pp. 205–224.
[12] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, 1996.
[13] M. Gu and S. C. Eisenstat, *A stable and fast algorithm for updating the singular value decomposition*, Technical Report YALEU/DCS/RR-966, Department of Computer Science, Yale University, CT.
[14] M. E. Hochstenbach, *A Jacobi-Davidson type SVD method*, SIAM J. Sci. Comput., 23 (2001), pp. 606–628.
[15] M. E. Hochstenbach, *Harmonic and refined extraction methods for the singular value problem, with applications to least-squares problems*, BIT, to appear. Available at http://www.case.edu/artsci/math/hochstenbach/
[16] M. E. Hochstenbach, *Subspace Methods for Eigenvalue Problems*, Ph.D. thesis, Department of Mathematics, University of Utrecht, Utrecht, The Netherlands, 2003. Available at http://www.case.edu/artsci/math/hochstenbach/
[17] Z. Jia and D. Niu, *An implicitly restarted refined bidiagonalization Lanczos method for computing a partial singular value decomposition*, SIAM J. Matrix Anal. Appl., 25 (2003), pp. 246–265.
[18] E. Kokiopoulou, C. Bekas, and E. Gallopoulos, *Computing smallest singular triplets with implicitly restarted Lanczos bidiagonalization*, Appl. Numer. Math., 49 (2004), pp. 39–61.
[19] R. M. Larsen, *Lanczos bidiagonalization with partial reorthogonalization*, Chapter of Ph.D. thesis, Department of Computer Science, University of Aarhus, Aarhus, Denmark, 1998. Available at http://soi.stanford.edu/∼rmunk/
[20] P. Läuchli, *Jordan-Elimination und Ausgleichung nach kleinsten Quadraten*, Numer. Math, 3 (1961), pp. 226–240.
[21] R. B. Lehoucq, *Implicitly restarted Arnoldi methods and subspace iteration*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 551–562.
[22] 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.
[23] R. B. Lehoucq, D. C. Sorensen, and C. Yang, ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods, SIAM, Philadelphia, 1998. Code available at the web site http://www.caam.rice.edu/software/ARPACK
[24] The MathWorks, Inc., MATLAB, Version 6.5 R13, Natick, MA, 2002.
[25] R. B. Morgan, *Computing interior eigenvalues of large matrices*, Linear Algebra Appl., 154-156 (1991), pp. 289–309.
[26] R. B. Morgan, *On restarting the Arnoldi method for large nonsymmetric eigenvalue problems*, Math. Comp., 65 (1996), pp. 1213–1230.
[27] R. B. Morgan and M. Zeng, *Harmonic restarted Arnoldi algorithm for calculating eigenvalues and determining multiplicity*, Preprint, 2003.
[28] C. C. Paige, B. N. Parlett, and H. A. van der Vorst, *Approximate solutions and eigenvalue bounds from Krylov subspaces*, Numer. Lin. Alg. Appl., 2 (1995), pp. 115–134.
[29] H. D. Simon and H. Zha, *Low rank matrix approximation using the Lanczos bidiagonalization process with applications*, SIAM J. Sci. Comput., 21 (2000), pp. 2257–2274.
[30] D. C. Sorensen, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J.

Matrix Anal. Appl., 13 (1992), pp. 357–385.
[31] D. C. Sorensen, *Numerical methods for large eigenvalue problems*, Acta Numerica, 11 (2002), pp. 519–584.
[32] A. Stathopoulos and Y. Saad, *Restarting techniques for the (Jacobi-)Davidson symmetric eigenvalue methods*, Elec. Trans. Numer. Anal., 7 (1998), pp. 163–181.
[33] A. Stathopoulos, Y. Saad, and K. Wu, *Dynamic thick restarting of the Davidson and the implicitly restarted Arnoldi methods*, SIAM J. Sci. Comput., 19 (1998), pp. 227–245.
[34] K. Wu and H. Simon, *Thick-restarted Lanczos method for large symmetric eigenvalue problems*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 602–616.

TABLE 4.3

Example 4: Computation of the 10 largest singular values of the matrices MEDLINE, CRANFIELD, CISI, and HYPATIA.

`irlba(R1)`

matrix	# matrix-vector products	CPU time			magnitude of largest error
		matrix-vector products	one-sided reorthogonalization	Total	
MEDLINE	74	0.24s	0.061s	0.34s	$1.08 \cdot 10^{-10}$
CRANFIELD	78	0.38s	0.078s	0.53s	$9.46 \cdot 10^{-12}$
CISI	76	0.33s	0.030s	0.45s	$5.80 \cdot 10^{-11}$
HYPATIA	78	0.39s	0.174s	0.75s	$9.75 \cdot 10^{-10}$

`irlba(R2)`

matrix	# matrix-vector products	CPU time			magnitude of largest error
		matrix-vector products	two-sided reorthogonalization	Total	
MEDLINE	74	0.17s	0.111s	0.41s	$1.08 \cdot 10^{-10}$
CRANFIELD	78	0.35s	0.124s	0.56s	$2.64 \cdot 10^{-12}$
CISI	76	0.30s	0.125s	0.52s	$7.80 \cdot 10^{-11}$
HYPATIA	78	0.47s	0.266s	0.88s	$9.75 \cdot 10^{-10}$

`irblsvds`

matrix	# matrix-vector products	CPU time	magnitude of largest error
MEDLINE	436	4.55s	$5.09 \cdot 10^{-10}$
CRANFIELD	508	5.72s	$3.28 \cdot 10^{-10}$
CISI	564	6.39s	$2.41 \cdot 10^{-10}$
HYPATIA	432	9.36s	$2.15 \cdot 10^{-09}$

`jdsvd(Ref)`

matrix	# matrix-vector products	CPU time	magnitude of largest error
MEDLINE	218	2.25s	$7.32 \cdot 10^{-10}$
CRANFIELD	240	2.58s	$7.16 \cdot 10^{-11}$
CISI	218	2.39s	$2.93 \cdot 10^{-10}$
HYPATIA	240	4.70s	$4.43 \cdot 10^{-10}$

`svds`

matrix	CPU time	magnitude of largest error
MEDLINE	1.06s	$4.94 \cdot 10^{-12}$
CRANFIELD	1.48s	$2.64 \cdot 10^{-11}$
CISI	1.58s	$2.73 \cdot 10^{-12}$
HYPATIA	2.53s	$1.02 \cdot 10^{-11}$