Electronic Transactions on Numerical Analysis.
Volume 42, pp. 85-105, 2014.
Copyright © 2014, Kent State University.
ISSN 1068-9613.

ETNA
Kent State University
http://etna.math.kent.edu

IMPLICITLY RESTARTING THE LSQR ALGORITHM[*]

JAMES BAGLAMA[†] AND DANIEL J. RICHMOND[†]

Dedicated to Lothar Reichel on the occasion of his 60th birthday

Abstract. The LSQR algorithm is a popular method for solving least-squares problems. For some matrices, LSQR may require a prohibitively large number of iterations to determine an approximate solution within a desired accuracy. This paper develops a strategy that couples the LSQR algorithm with an implicitly restarted Golub-Kahan bidiagonalization method to improve the convergence rate. The restart is carried out by first applying the largest harmonic Ritz values as shifts and then using LSQR to compute the solution to the least-squares problem. Theoretical results show how this method is connected to the augmented LSQR method of Baglama, Reichel, and Richmond [Numer. Algorithms, 64 (2013), pp. 263–293] in which the Krylov subspaces are augmented with the harmonic Ritz vectors corresponding to the smallest harmonic Ritz values. Computed examples show the proposed method to be competitive with other methods.

Key words. Golub-Kahan bidiagonalization, iterative method, implicit restarting, harmonic Ritz values, large-scale computation, least-squares, LSQR, Krylov subspace

AMS subject classifications. 65F15, 15A18

1. Introduction.

In this paper, we investigate large-scale least-squares (LS) problems

$$\min_{x\in\mathbb{R}^n} \|b - Ax\|, \qquad A \in \mathbb{R}^{\ell\times n}, \qquad b \in \mathbb{R}^{\ell} \tag{1.1}$$

where $\|\cdot\|$ denotes the Euclidean vector norm. The matrix A is assumed to be sparse and too large to use direct solvers efficiently. Therefore iterative methods, which can also take advantage of the sparse structure of A, are required in order to solve the LS problem. When $\ell \geq n$ the preferred iterative method for solving LS problems is the LSQR Algorithm of Paige and Saunders [31]. LSQR is a Krylov subspace method that is based on the Golub-Kahan (GK) bidiagonalization, in which orthonormal bases for the m-dimensional Krylov subspaces

$$\begin{aligned} \mathcal{K}_m(AA^T, w_1) &= \text{span}\{w_1, AA^T w_1, \ldots, (AA^T)^{m-1} w_1\}, \\ \mathcal{K}_m(A^TA, p_1) &= \text{span}\{p_1, A^TAp_1, \ldots, (A^TA)^{m-1} p_1\} \end{aligned} \tag{1.2}$$

are formed using the starting vectors $w_1 = r_0/\|r_0\|$ and $p_1 = A^T w_1/\|A^T w_1\|$, respectively, where $r_0 = b - Ax_0$ for an initial guess x_0 of the LS problem. Using the orthonormal bases for the spaces in (1.2), the LSQR Algorithm computes an approximate solution $x_m \in x_0 + \mathcal{K}_m(A^TA, p_1)$ and corresponding residual $r_m = b - Ax_m \in \mathcal{K}_m(AA^T, w_1)$ such that $\|b - Ax_m\|$ is minimized over all possible choices for x_m. The LSQR algorithm is a non-restarted method where the dimension m is increased until an acceptable solution of the LS problem is found. The theoretical foundation of LSQR yields a process that only requires the storage of a few basis vectors for each Krylov subspace. In exact arithmetic, LSQR terminates with the solution of the LS problem when linear dependence is established in (1.2). For LS problems with a well-conditioned matrix A or a small effective condition number, LSQR converges quickly yielding an approximation of the solution of the LS problem of desired accuracy long before linear dependence is encountered in (1.2); see Björck [9] for remarks. However, for LS problems with an ill-conditioned matrix A and a solution vector x

---

[*] Received October 15, 2013. Accepted February 7, 2014. Published online on May 23, 2014. Recommended by Rich LeHoucq.

[†] Department of Mathematics, University of Rhode Island, Kingston, RI 02881 ({jbaglama, dan}@math.uri.edu).

with many components in the direction of the singular vectors associated with the smallest singular values, LSQR may require a prohibitively large number of iterations; see [9]. A contributing reason is that in finite arithmetic, the storage of only a few basis vectors at a time cannot maintain orthogonality among all previously non-stored basis vectors. Hence the generated Krylov subspaces have difficulty obtaining good approximations to the smallest singular triplets. The loss of orthogonality can be overcome by keeping previously computed basis vectors and reorthogonalizing. However, as m becomes large, this can become a computationally expensive, impractical storage requirement. One solution is to use a restarted Krylov subspace method to solve the LS problem. Restarting Krylov subspace methods after m iterations, for $m << n$, can maintain orthogonality with a modest storage requirement. The restarted GMRES method of Saad and Schultz [34] is one of the most popular Krylov subspace methods for solving the LS problem when $\ell = n$. However, using the restarted GMRES method to solve the LS problem introduces another difficulty, stagnation and/or slow convergence, [6, 39]. To overcome stagnation and/or slow convergence, restarted GMRES is often combined with a preconditioner or the minimization is over an augmented Krylov subspace; see [1, 7, 19, 28, 29, 33] and references within.

If we implement a restarted LSQR method, i.e., restarting LSQR after m iterations, we can maintain strong orthogonality among the bases by keeping all the vectors in storage. However, similar to GMRES, the restarted LSQR method can encounter stagnation and even slower convergence than using LSQR without reorthogonalization (cf. [15] for details on restarting the related LSMR algorithm). To overcome stagnation and/or slow convergence of restarting LSQR, we propose to solve the LS problem implicitly over an improved Krylov subspace, a form of preconditioning. We consider implicitly restarting the GK bidiagonalization (and hence LSQR) with a starting vector w_1^+, such that $w_1^+ = \phi(AA^T)w_1$ for some polynomial ϕ that is strong in the direction of the left singular vectors associated with the smallest singular values. The Krylov subspaces $\mathcal{K}_m(AA^T, w_1^+)$ and $\mathcal{K}_m(A^TA, p_1^+)$ will then contain good approximations to the left and right singular vectors corresponding to the smallest singular values, respectively. Also, with judiciously chosen shifts (i.e., zeros of $\phi(AA^T)$) we can ensure that $\mathcal{K}_m(AA^T, w_1^+)$ will contain the LSQR residual vector at each iteration of the restarted method. This is essential so that our restarted LSQR method produces a non-increasing residual curve. Since the singular values of A are not known prior to starting the LSQR method, approximations must be found.

Implicitly restarted GK bidiagonalization methods [2, 3, 5, 21, 22, 24] have been used very successfully in providing good approximations to the smallest and largest singular triplets of a very large matrix A while using a small storage space and not many matrix-vector products. In this paper, we describe an implicitly restarted GK bidiagonalization method which selects a polynomial filter that produces good approximations of the singular vectors associated with the smallest singular values, thus improving the search spaces while simultaneously computing approximate solutions to the LS problem. There are many methods for preconditioning LSQR to improve convergence [8, 9, 10, 23, 33]. However, most methods require constructions prior to approximating solutions to the LS problem adding to the storage and/or computational time.

In [5], we solved the LS problem with an LSQR method over a Krylov subspace that was explicitly augmented by approximate singular vectors of A. Augmenting Krylov subspaces in conjunction with solving the LS problem when $\ell = n$ with the restarted GMRES method was first discussed by Morgan in [28]. Later, Morgan showed the mathematical equivalence between applying harmonic Ritz values as implicit shifts and augmenting the Krylov subspaces by harmonic Ritz vectors to solve the LS problem when $\ell = n$ with restarted GMRES; cf. [29]. Similarly, in Section 5, we show that our proposed method of this paper, applying harmonic Ritz values as implicit shifts to a restarted LSQR method to improve the Krylov subspaces, is mathematically equivalent to the routine in [5] that obtains Krylov subspaces by explicitly augmenting them with the harmonic Ritz vectors to improve convergence. Therefore, the theorems from [5], which show improved convergence for LSQR using augmented spaces, are applicable to this method. Applying the shifts implicitly is simple, and we introduce a new strategy for choosing and applying the shifts, which, based on our heuristics, further improves the convergence rates.

The paper is organized as follows: Section 2 describes, in detail, an implicitly restarted GK bidiagonalization method and the simplifications that can be utilized when using the harmonic Ritz values as shifts. Section 3 describes how LSQR can be successfully restarted by using the implicitly restarted GK bidiagonalization algorithm with harmonic Ritz values as shifts. The numerical issues of implicitly shifting via the bulgechasing method are discussed in Section 4 along with a new method for implicitly applying harmonic Ritz values as a shift. Section 5 gives the theoretical results of how the implicitly restarted LSQR algorithm generates the same updated Krylov subspaces as the augmented LSQR algorithm from [5]. Section 6 gives numerical experiments to show the competitiveness of the proposed method, and Section 7 gives concluding remarks.

Throughout this paper, we will denote $\mathcal{N}(C)$ as the null space and $\mathcal{R}(C)$ as the range of the matrix C.

2. Implicitly restarted Golub-Kahan bidiagonalization.

The GK bidiagonalization forms the basis for the LSQR algorithm discussed in Section 3 and is needed to approximate a set of the smallest singular triplets of A. Define $U_n = [u_1, u_2, \ldots, u_n] \in \mathbb{R}^{\ell \times n}$ and $V_n = [v_1, v_2, \ldots, v_n] \in \mathbb{R}^{n \times n}$ with orthonormal columns, as well as the diagonal matrix $\Sigma_n = \text{diag}[\sigma_1, \sigma_2, \ldots, \sigma_n] \in \mathbb{R}^{n \times n}$. Then

$$AV_n = U_n\Sigma_n \qquad \text{and} \qquad A^TU_n = V_n\Sigma_n \tag{2.1}$$

are singular value decompositions (SVD) of A and A^T, respectively, and

$$AV_k = U_k\Sigma_k \qquad \text{and} \qquad A^TU_k = V_k\Sigma_k$$

for $k << n$ are partial singular value decompositions (PSVD) of A and A^T, respectively. We assume the singular values to be ordered from the smallest to the largest one, i.e.,

$$0 < \sigma_1 \le \sigma_2 \le \ldots \le \sigma_n,$$

since we are interested in the smallest singular values of A.

The GK bidiagonalization was originally proposed in [16] as a method for transforming a matrix A into upper bidiagonal form. However, for its connection to the LSQR algorithm in solving (1.1), we consider the variant that transforms A to lower bidiagonal form (cf. [31, bidiag 1]), described in Algorithm 2.1. The lower bidiagonal algorithm was described by Björk [11] as the more stable version of the GK bidiagonalization method and this form fits nicely into our implicitly restarted method.

ALGORITHM 2.1. GK BIDIAGONALIZATION METHOD

Input: $A \in \mathbb{R}^{\ell \times n}$ *or functions for evaluating matrix-vector products with* A *and* A^T*,*
$w_1 \in \mathbb{R}^{\ell}$: *initial starting vector,*
m : *number of bidiagonalization steps.*

```
Output: $P_m = [p_1, \ldots, p_m] \in \mathbb{R}^{n\times m}$ : matrix with orthonormal columns,
        $W_{m+1} = [w_1, \ldots, w_{m+1}] \in \mathbb{R}^{\ell\times(m+1)}$ : matrix with orthonormal columns,
        $B_{m+1,m} \in \mathbb{R}^{(m+1)\times m}$ : lower bidiagonal matrix,
        $p_{m+1} \in \mathbb{R}^n$ : residual vector,
        $\alpha_{m+1} \in \mathbb{R}$.

1. Compute $\beta_1 := \|w_1\|$;  $w_1 := w_1/\beta_1$;  $W_1 := w_1$.
2. Compute $p_1 := A^T w_1$;  $\alpha_1 := \|p_1\|$;  $p_1 := p_1/\alpha_1$;  $P_1 := p_1$.
3. for $j = 1 : m$
  4. Compute $w_{j+1} := Ap_j - w_j\alpha_j$.
  5. Reorthogonalization step:  $w_{j+1} := w_{j+1} - W_{(1:j)}(W_{(1:j)}^T w_{j+1})$.
  6. Compute $\beta_{j+1} := \|w_{j+1}\|$;  $w_{j+1} := w_{j+1}/\beta_{j+1}$.
  7. Compute $p_{j+1} := A^T w_{j+1} - p_j\beta_{j+1}$.
  8. Reorthogonalization step:  $p_{j+1} := p_{j+1} - P_{(1:j)}(P_{(1:j)}^T p_{j+1})$.
  9. Compute $\alpha_{j+1} := \|p_{j+1}\|$;  $p_{j+1} := p_{j+1}/\alpha_{j+1}$.
  10. if $j < m$
    11. $P_{j+1} := [P_j, p_{j+1}]$.
  12. endif
13. endfor
```

After m steps, Algorithm 2.1 determines matrices W_{m+1} and P_m whose columns form orthonormal bases for the Krylov subspaces $\mathcal{K}_{m+1}(AA^T, w_1)$ and $\mathcal{K}_m(A^TA, p_1)$, respectively, as well as the decompositions

$$\begin{array}{lcl} A^T W_{m+1} & = & P_m B_{m+1,m}^T + \alpha_{m+1} p_{m+1} e_{m+1}^T \\ AP_m & = & W_{m+1} B_{m+1,m} \end{array} \tag{2.2}$$

where $p_{m+1}^T P_m = 0$, and e_{m+1} is the $(m+1)^{\text{st}}$ axis vector. The matrix

$$B_{m+1,m} = \begin{bmatrix} \alpha_1 & & & \mathbf{0} \\ \beta_2 & \alpha_2 & & \\ & \beta_3 & \ddots & \\ & & \ddots & \alpha_m \\ \mathbf{0} & & & \beta_{m+1} \end{bmatrix} \in \mathbb{R}^{(m+1)\times m} \tag{2.3}$$

is lower bidiagonal. We assume that Algorithm 2.1 does not terminate early, that is, $\alpha_j \neq 0$ and $\beta_j \neq 0$ for $1 \leq j \leq m+1$; see [5] for a discussion on how to handle early termination. To avoid loss of orthogonality in finite precision arithmetic in the basis vectors W_{m+1} and P_m, we reorthogonalize in lines 5 and 8 of the algorithm. The reorthogonalization steps do not add significant computational cost when $m << n$. For discussions and schemes on reorthogonalization we refer the reader to [2, 5, 15, 25, 35] and references within. For the numerical examples in Section 6 we follow the same scheme used in [5].

It is well known that using a Krylov subspace to obtain acceptable approximations to the smallest singular triplets of A with equations (2.2) can require a prohibitively large value of m. Therefore, a restarting strategy is required. The most effective restarting strategy is to use an implicit restart technique. By implicitly restarting after $m << n$ steps of the GK bidiagonalization, storage requirements can be kept relatively small and provide good approximations to the desired singular vectors from the generated Krylov subspaces. The following section provides a detailed discussion on how to implicitly restart the GK bidiagonalization method.

2.1. Implicit restart formulas for the GK bidiagonalization. Implicitly restarting a GK bidiagonalization method was first discussed in [11] and used in [2, 3, 5, 21, 22, 24]. Starting with the m-step GK bidiagonalization decomposition (2.2), the implicit restarting is done by selecting a shift μ and applying the shift via the Golub-Kahan SVD step [17, Algorithm 8.6.1]. The algorithm given in [17] assumes an upper bidiagonal matrix is given. We modify the algorithm for a lower bidiagonal matrix, and it is given as the bulgechasing (lower bidiagonal) algorithm (cf. Algorithm 2.2). Algorithm 2.2 uses the shift μ and generates upper Hessenberg orthogonal matrices $Q_L \in \mathbb{R}^{(m+1)\times(m+1)}$ and $Q_R \in \mathbb{R}^{m\times m}$ such that $B^+_{m+1,m} = Q_L^T B_{m+1,m} Q_R$ is lower bidiagonal. Multiplying the first equation of (2.2) by Q_L from the right and the second equation of (2.2) by Q_R also from the right yields

$$\begin{array}{llll} (2.4) & A^T W_{m+1} Q_L & = & P_m B^T_{m+1,m} Q_L + \alpha_{m+1} p_{m+1} e^T_{m+1} Q_L \\ & A P_m Q_R & = & W_{m+1} B_{m+1,m} Q_R. \end{array}$$

Let $W^+_{m+1} = W_{m+1} Q_L$, $P^+_m = P_m Q_R$, and

$$(2.5) \qquad p^+_m = \frac{\alpha^+_m p^+_m + (\alpha_{m+1} q_{L_{m+1,m}}) p_{m+1}}{\|\alpha^+_m p^+_m + (\alpha_{m+1} q_{L_{m+1,m}}) p_{m+1}\|},$$

where α^+_m is the (m, m) diagonal entry of $B^+_{m+1,m}$ and $q_{L_{m+1,m}}$ is the $(m + 1, m)$ entry of Q_L. Now set $\alpha^+_m = \|\alpha^+_m p^+_m + (\alpha_{m+1} q_{L_{m+1,m}}) p_{m+1}\|$. Then we have after removing the last column from both sides of the equations in (2.4) a valid $(m-1)$-step GK bidiagonalization decomposition,

$$\begin{array}{llll} (2.6) & A^T W^+_m & = & P^+_{m-1} B^{+T}_{m,m-1} + \alpha^+_m p^+_m e^T_m, \\ & A P^+_{m-1} & = & W^+_m B^+_{m,m-1}. \end{array}$$

The $(m - 1)$-step GK bidiagonalization decomposition (2.6) is the decomposition that we would have obtained by applying $(m - 1)$ steps of Algorithm 2.1 with the starting vector $w^+_1 = \gamma(AA^T - \mu I)w_1$, i.e., a polynomial filter has been applied to w_1. See [3, 11, 24] for detailed discussions on polynomial filters in the context of implicitly restarting a GK bidiagonalization method. Given a suitable choice of a shift μ, the polynomial filter helps dampen unwanted singular vector components of A from w_1. Multiple shifts ($p = m - k$ shifts $\mu_1, \mu_2, \ldots, \mu_p$) can be applied via this process yielding the following valid k-step GK bidiagonalization decomposition,

$$\begin{array}{llll} (2.7) & A^T W^+_{k+1} & = & P^+_k B^{+T}_{k+1,k} + \alpha^+_{k+1} p^+_{k+1} e^T_{k+1} \\ & A P^+_k & = & W^+_{k+1} B^+_{k+1,k} \end{array}$$

which would have been obtained by applying k-steps of Algorithm 2.1 with the starting vector $w^+_1 = \tilde{\gamma} \prod_{i=1}^p (AA^T - \mu_i I) w_1$. Using the vectors p^+_{k+1}, w^+_{k+1} the $(k + 1)^{\text{st}}$ column vector of W^+_{k+1}, and the scalar α^+_{k+1}, the k-step GK bidiagonalization decomposition (2.7) can be extended to an m-step GK bidiagonalization decomposition (2.2) by starting at step 4 of Algorithm 2.1 and continuing for p more iterations.

ALGORITHM 2.2. BULGECHASING (LOWER BIDIAGONAL)

Input: $B_{m+1,m} \in \mathbb{R}^{(m+1)\times m}$ *lower bidiagonal matrix,*
μ : *implicit shift.*

Output: $Q_L \in \mathbb{R}^{(m+1)\times(m+1)}$: *upper Hessenberg matrix with orthonormal columns,*
$Q_R \in \mathbb{R}^{m\times m}$: *upper Hessenberg matrix with orthonormal columns,*
$B^+_{m+1,m} = Q_L^T B_{m+1,m} Q_R \in \mathbb{R}^{(m+1)\times m}$: *updated lower bidiagonal matrix.*

```
1. Determine the $(m+1)\times(m+1)$ Givens rotation matrix $G(1,2,\theta_1)$ such that
   $\begin{bmatrix} c & s \\ -s & c \end{bmatrix}\begin{bmatrix} b_{1,1}^2-\mu \\ b_{1,1}\cdot b_{2,1} \end{bmatrix} = \begin{bmatrix} \star \\ 0 \end{bmatrix}$.
2. Set $Q_L^T := G(1,2,\theta_1)$;  $Q_R := I_m$;  $B_{m+1,m}^+ := G(1,2,\theta_1)B_{m+1,m}$.
3. for $i = 1 : m-1$
    4. Determine the $m\times m$ Givens rotation matrix $G(i,i+1,\theta_i)$ such that
       $\begin{bmatrix} b_{i,i}^+ & b_{i,i+1}^+ \end{bmatrix}\begin{bmatrix} c & -s \\ s & c \end{bmatrix} = \begin{bmatrix} \star & 0 \end{bmatrix}$.
    5. Update $Q_R := Q_R G(i,i+1,\theta_i)$;  $B_{m+1,m}^+ := B_{m+1,m}^+ G(i,i+1,\theta_i)$.
    6. Determine the $(m+1)\times(m+1)$ Givens rotation matrix $G(i+1,i+2,\theta_{i+1})$
       such that
       $\begin{bmatrix} c & s \\ -s & c \end{bmatrix}\begin{bmatrix} b_{i+1,i}^+ \\ b_{i+2,i}^+ \end{bmatrix} = \begin{bmatrix} \star \\ 0 \end{bmatrix}$.
    7. Update $Q_L^T := G(i+1,i+2,\theta_{i+1})Q_L^T$;  $B_{m+1,m}^+ := G(i+1,i+2,\theta_{i+1})B_{m+1,m}^+$.
8. endfor
```

2.2. Implicit restart with harmonic Ritz values as shifts. The dampening effect of the polynomial filter, $\prod_{i=1}^{p}(AA^T-\mu_i I)$, depends on the choice of shifts μ_i. There are several choices for μ_i that have been investigated in the literature in this context; see, e.g., Ritz and harmonic Ritz values [24], refined Ritz values [21], refined harmonic Ritz values [22], and Leja points [3]. We examine the choice of using harmonic Ritz values as shifts for our implicitly restarted method. Harmonic Ritz values not only provide good approximations to the smallest singular values of A, they have a much needed connection with the LSQR algorithm described in Section 3.

The harmonic Ritz values $\hat{\theta}_j$ of AA^T are defined as the eigenvalues to the generalized eigenvalue problem

$$((B_{m,m}B_{m,m}^T)+\alpha_m^2\beta_{m+1}^2(B_{m,m}B_{m,m}^T)^{-1}e_m e_m^T)g_j = \hat{\theta}_j g_j, \quad 1\le j\le m, \tag{2.8}$$

where $B_{m,m}$ is the $m\times m$ principal submatrix of $B_{m+1,m}$, and $g_j\in\mathbb{R}^m\backslash\{0\}$ is an eigenvector; see e.g., [27, 30] for properties and discussions of harmonic Ritz values. The eigenpairs $\{\hat{\theta}_j, g_j\}_{j=1}^m$ can be computed without forming the matrix $B_{m,m}B_{m,m}^T$ from the SVD of $B_{m+1,m}$,

$$\begin{aligned} B_{m+1,m}\tilde{V}_m &= \left[\tilde{U}_m\ \tilde{u}_{m+1}\right]\begin{bmatrix}\tilde{\Sigma}_m \\ 0\end{bmatrix}, \\ B_{m+1,m}^T\left[\tilde{U}_m\ \tilde{u}_{m+1}\right] &= \tilde{V}_m\left[\tilde{\Sigma}_m \quad 0\right], \end{aligned} \tag{2.9}$$

where the matrices $\tilde{V}_m=[\tilde{v}_1,\tilde{v}_2,\dots,\tilde{v}_m]\in\mathbb{R}^{m\times m}$ and $\tilde{U}_m=[\tilde{u}_1,\tilde{u}_2,\dots,\tilde{u}_m]\in\mathbb{R}^{(m+1)\times m}$ have orthonormal columns, $\tilde{u}_{m+1}\in\mathbb{R}^{m+1}$ (the null vector) is a unit-length vector such that $\tilde{u}_{m+1}^T\tilde{U}_m=0$, and $\tilde{\Sigma}_m=\mathrm{diag}[\tilde{\sigma}_1,\tilde{\sigma}_2,\dots,\tilde{\sigma}_m]\in\mathbb{R}^{m\times m}$. We order the m singular values according to

$$0<\tilde{\sigma}_1<\tilde{\sigma}_2<\ldots<\tilde{\sigma}_m. \tag{2.10}$$

The strict inequalities come from the assumption that the diagonal and sub-diagonal entries of $B_{m+1,m}$ are all nonzero [32, Lemma 7.7.1].

We have $\hat{\theta}_j = \tilde{\sigma}_j^2$ [30]. The eigenvectors g_j are the columns of $[I_m \quad \beta_{m+1} B_{m,m}^{-T} e_m]\tilde{U}_m$; see [5] for details. Furthermore, if $\tilde{\sigma}_j^2$ is used as a shift in Algorithm 2.2, then the return matrix $B_{m+1,m}^+$ has entries $\alpha_m^+ = 0$ and $\beta_{m+1}^+ = \pm\tilde{\sigma}_j$. The following theorem shows this result.

THEOREM 2.3. *Given a lower bidiagonal matrix $B_{m+1,m}$ (2.3) where $\alpha_j \neq 0$ for $1 \leq j \leq m$ and $\beta_j \neq 0$ for $2 \leq j \leq m+1$. In Algorithm 2.2, $\mu = \tilde{\sigma}_j^2$ (cf. (2.10)) if and only if the return matrix $B_{m+1,m}^+$ has $\alpha_m^+ = 0$ and $\beta_{m+1}^+ = \pm\tilde{\sigma}_j$. Furthermore, Algorithm 2.2 returns the matrices Q_L and Q_R such that $Q_L e_{m+1} = \pm\tilde{u}_j$ and $Q_R e_m = \pm\tilde{v}_j$.*

Proof. Compute the QR-factorization of $B_{m+1,m}B_{m+1,m}^T - \mu I_{m+1} = QR$ where $Q \in \mathbb{R}^{(m+1)\times(m+1)}$ is orthogonal and $R \in \mathbb{R}^{(m+1)\times(m+1)}$ is upper triangular. An inspection of steps 1 and 2 in Algorithm 2.2 shows that the first columns of Q_L and Q are equal. Therefore, via the implicit Q Theorem [17, Theorem 7.4.2] we have $Q_L = QD$ where $D = \mathrm{diag}[1, \pm 1, \ldots, \pm 1]$ and

$$B_{m+1,m}^+ B_{m+1,m}^{+T} = Q_L^T B_{m+1,m} B_{m+1,m}^T Q_L = DQ^T B_{m+1,m} B_{m+1,m}^T QD.$$

The matrix $B_{m+1,m}^+ B_{m+1,m}^{+T}$ is a symmetric tridiagonal matrix, and if μ is an eigenvalue of $B_{m+1,m}B_{m+1,m}^T$ then [17, Section 8.3.3]

$$Q_L^T B_{m+1,m} B_{m+1,m}^T Q_L e_{m+1} = DQ^T B_{m+1,m} B_{m+1,m}^T Q D e_{m+1} = \mu e_{m+1}.$$

Therefore,

$$B_{m+1,m}^+ B_{m+1,m}^{+T} e_{m+1} = DQ^T B_{m+1,m} B_{m+1,m}^T Q D e_{m+1} = \mu e_{m+1},$$

$\beta_{m+1}^+ \alpha_m^+ = 0$ and $(\beta_{m+1}^+)^2 = \mu$. Since $\tilde{\sigma}_j^2 \neq 0$ are eigenvalues of $B_{m+1,m}B_{m+1,m}^T$, we have $\alpha_m^+ = 0$ and $\beta_{m+1}^+ = \pm\tilde{\sigma}_j$. One can see that the reverse holds by noticing that $B_{m+1,m}B_{m+1,m}^T$ is unreduced, and if μ is not an eigenvalue, then $B_{m+1,m}^+ B_{m+1,m}^{+T}$ must also be unreduced [32, Lemma 8.13.1]. Algorithm 2.2 returns Q_R, Q_L, and $B_{m+1,m}^+$ satisfying $B_{m+1,m}Q_R = Q_L B_{m+1,m}^+$ and $B_{m+1,m}^T Q_L = Q_R B_{m+1,m}^{+T}$. Using the structure of the last column of $B_{m+1,m}^+$ we have

$$\begin{aligned} B_{m+1,m} Q_R e_m &= Q_L B_{m+1,m}^+ e_m = \pm\tilde{\sigma}_j Q_L e_{m+1} \\ B_{m+1,m}^T Q_L e_{m+1} &= Q_R B_{m+1,m}^{+T} e_{m+1} = \pm\tilde{\sigma}_j Q_R e_m. \end{aligned}$$

The result $Q_L e_{m+1} = \pm\tilde{u}_j$ and $Q_R e_m = \pm\tilde{v}_j$ follows from the SVD of $B_{m+1,m}$ (2.9) and the fact that the singular vectors of non-degenerate singular values are unique up to sign difference [9]. ☐

Calling Algorithm 2.2 with $B_{m+1,m}$ and $\mu = \tilde{\sigma}_m^2$ returns the upper Hessenberg orthogonal matrices

$$\begin{aligned} Q_L &= [Q_{L_m}, \pm\tilde{u}_m] \in \mathbb{R}^{(m+1)\times(m+1)}, & \text{where } Q_{L_m} &= [q_{L_1}, \ldots, q_{L_m}] \in \mathbb{R}^{(m+1)\times m}, \\ Q_R &= [Q_{R_{m-1}}, \pm\tilde{v}_m] \in \mathbb{R}^{m\times m}, & \text{where } Q_{R_{m-1}} &= [q_{R_1}, \ldots, q_{R_{m-1}}] \in \mathbb{R}^{m\times(m-1)}, \end{aligned}$$

and the lower bidiagonal matrix,

$$B_{m+1,m}^+ = \begin{bmatrix} \begin{bmatrix} B_{m,m-1}^+ \end{bmatrix} & 0 \\ 0 & \pm\tilde{\sigma}_m \end{bmatrix} \in \mathbb{R}^{(m+1)\times m}. \tag{2.11}$$

The singular values of $B^{+}_{m,m-1}$ are

$$0 < \tilde{\sigma}_1 < \tilde{\sigma}_2 < \ldots < \tilde{\sigma}_{m-1}$$

and the SVD of $B^{+}_{m,m-1}$ is

$$\begin{aligned} B^{+}_{m,m-1} Q^{T}_{R_{m-1}} \tilde{V}_{m-1} &= Q^{T}_{L_m} \tilde{U}_{m-1} \tilde{\Sigma}_{m-1}, \\ B^{+T}_{m,m-1} Q^{T}_{L_m} \tilde{U}_{m-1} &= Q^{T}_{R_{m-1}} \tilde{V}_{m-1} \tilde{\Sigma}_{m-1}. \end{aligned}$$

Calling Algorithm 2.2 with $B^{+}_{m,m-1}$ and $\mu = \tilde{\sigma}^2_{m-1}$ returns the upper Hessenberg orthogonal matrices

$$\begin{aligned} Q^{+}_{L_m} &= [Q^{+}_{L_{m-1}}, \pm Q^{T}_{L_{m-1}} \tilde{u}_{m-1}] \in \mathbb{R}^{m \times m}, \\ &\qquad \text{where } Q^{+}_{L_{m-1}} = [q^{+}_{L_1}, \ldots, q^{+}_{L_{m-1}}] \in \mathbb{R}^{m \times (m-1)}, \\ Q^{+}_{R_{m-1}} &= [Q^{+}_{R_{m-2}}, \pm Q^{T}_{R_{m-1}} \tilde{v}_{m-1}] \in \mathbb{R}^{(m-1) \times (m-1)}, \\ &\qquad \text{where } Q^{+}_{R_{m-2}} = [q^{+}_{R_1}, \ldots, q^{+}_{R_{m-2}}] \in \mathbb{R}^{(m-1) \times (m-2)}, \end{aligned}$$

and the lower bidiagonal matrix,

$$B^{++}_{m,m-1} = \begin{bmatrix} \begin{bmatrix} B^{++}_{m-1,m-2} \end{bmatrix} & \mathbf{0} \\ \mathbf{0} & \pm\tilde{\sigma}_{m-1} \end{bmatrix} \in \mathbb{R}^{m \times (m-1)}. \tag{2.12}$$

Since the columns of $Q_{R_{m-1}}$ are orthonormal and $\tilde{v}_{m-1} \in \mathcal{R}(Q_{R_{m-1}})$, we have $Q_{R_{m-1}} Q^{T}_{R_{m-1}} \tilde{v}_{m-1} = \tilde{v}_{m-1}$. Likewise $Q_{L_{m-1}} Q^{T}_{L_{m-1}} \tilde{u}_{m-1} = \tilde{u}_{m-1}$. Therefore,

$$\begin{aligned} &\left(Q_L \begin{bmatrix} Q^{+}_{L_m} & 0 \\ 0 & 1 \end{bmatrix} \right)^T B_{m+1,m}\, Q_R \begin{bmatrix} Q^{+}_{R_{m-1}} & 0 \\ 0 & 1 \end{bmatrix} = \\ &\begin{bmatrix} \begin{bmatrix} B^{++}_{m-1,m-2} \end{bmatrix} & \mathbf{0} \\ & \pm\tilde{\sigma}_{m-1} \\ \mathbf{0} & \pm\tilde{\sigma}_m \end{bmatrix}, \end{aligned} \tag{2.13}$$

where

$$Q_R \begin{bmatrix} Q^{+}_{R_{m-1}} & 0 \\ 0 & 1 \end{bmatrix} = [q^{++}_{R_1}, \ldots, q^{++}_{R_{m-2}}, \tilde{v}_{m-1}, \tilde{v}_m] \tag{2.14}$$

and

$$Q_L \begin{bmatrix} Q^{+}_{L_m} & 0 \\ 0 & 1 \end{bmatrix} = [q^{++}_{L_1}, \ldots, q^{++}_{L_{m-1}}, \tilde{u}_{m-1}, \tilde{u}_m]. \tag{2.15}$$

The matrices (2.14) and (2.15) are no longer upper Hessenberg; they have an additional nonzero sub-diagonal below the diagonal, increasing the lower band width to 2.

Repeating the process, we can use Algorithm 2.2 to apply multiple shifts. After applying the largest $p = m - k$ harmonic Ritz values $(\tilde{\sigma}_m^2, \dots, \tilde{\sigma}_{k+1}^2)$ as shifts, we have

$$Q_L^T B_{m+1,m} Q_R = \begin{bmatrix} \begin{bmatrix} B_{k+1,k}^{+} \end{bmatrix} & \mathbf{0} & & \\ & \pm\tilde{\sigma}_{k+1} & & \\ & & \ddots & \\ \mathbf{0} & & & \pm\tilde{\sigma}_m \end{bmatrix}, \tag{2.16}$$

where

$$\begin{aligned} Q_R &= [Q_{R_k}, \tilde{v}_{k+1}, \dots, \tilde{v}_m] \\ Q_L &= [Q_{L_{k+1}}, \tilde{u}_{k+1}, \dots, \tilde{u}_m]. \end{aligned} \tag{2.17}$$

The matrices (2.17) now have a lower band width equal to p. Using the process outlined in Section 2.1 with (2.16) and (2.17), we have analogous to (2.7) a k-step GK bidiagonalization,

$$\begin{aligned} A^T W_{k+1}^{+} &= P_k^{+} B_{k+1,k}^{+T} + \alpha_{k+1}^{+} p_{k+1}^{+} e_{k+1}^T, \\ A P_k^{+} &= W_{k+1}^{+} B_{k+1,k}^{+}, \end{aligned} \tag{2.18}$$

where $W_{k+1}^{+} = W_{m+1} Q_{L_{k+1}}$, $P_k^{+} = P_m Q_{R_k}$, $B_{k+1,k}^{+} = Q_{L_{k+1}}^T B_{m+1,m} Q_{R_k}$,

$$p_{k+1}^{+} = \frac{(\alpha_{m+1} q_{L_{m+1,k+1}})}{|\alpha_{m+1} q_{L_{m+1,k+1}}|} p_{m+1}, \tag{2.19}$$

and $\alpha_{k+1}^{+} = |\alpha_{m+1} q_{L_{m+1,k+1}}|$. Using the vectors p_{k+1}^{+}, w_{k+1}^{+} the $(k+1)^{\text{st}}$ column vector of W_{k+1}^{+}, and the scalar α_{k+1}^{+}, the k-step GK bidiagonalization decomposition (2.18) can be extended to an m-step GK bidiagonalization decomposition (2.2) by starting at step 4 of Algorithm 2.1 and continuing for p more iterations.

We remark again that the importance of using harmonic Ritz values as shifts is the connection with the LSQR method described in Section 3 where zeroing out the diagonal elements of $B_{m+1,m}^{+}$ (cf. (2.11), (2.12), (2.13), (2.16)) is essential for restarting the LSQR method.

2.3. Adaptive shift strategy.

In order to help speed up convergence to the smallest singular triplets and ultimately speed up our implicitly restarted LSQR algorithm, we developed an adaptive shift strategy. It was first observed in [26] that if a shift μ_{k+1} that is numerically close to σ_k^2 is used in the implicitly restarted GK bidiagonalization method, then the component along the k-th left singular vector can be strongly damped in

$$w_1^{+} = \prod_{i=k+1}^{m} (AA^T - \mu_i I) w_1.$$

This can cause the resulting spaces W_{k+1}^{+} and V_k^{+} (2.18) to contain unsatisfactory approximations to the left and right singular vector corresponding to σ_k, respectively. When using an implicitly restarted GK bidiagonalization to solve for a PSVD of A, a heuristic was proposed in [26] to require that the relative gap between the approximating value $\tilde{\sigma}_k^2$ and all shifts μ_i, defined by

$$\text{relgap}_{ki} = \frac{(\tilde{\sigma}_k^2 - e_k) - \mu_i}{\tilde{\sigma}_k^2},$$

where e_k is the error bound on $\tilde{\sigma}_k^2$, be greater than 10^{-3}. In the context of [26], the shifts considered to be too close, i.e., the "bad shifts", were simply replaced by zero shifts. This strategy was adapted and applied in [21, 22] in which harmonic and refined harmonic Ritz values were used as shifts for computing some of the smallest and largest singular triplets. When searching for the smallest singular triplets, the "bad shifts" were replaced with the largest among all shift. In either case, through observation and numerical experiment, this improved the convergence of the smallest singular triplets. When implicitly restarting the GK bidiagonalization in combination with the LSQR algorithm, we cannot replace a "bad shift" by the largest among all shifts, i.e., our combined routine does not allow repeated shifts. This would destroy the required Hessenberg structure of Q_R and Q_L in the equations given in Section 2. We also cannot use a zero shift; this would remove the null vector $\tilde{u}_{m+1}$ of $B_{m+1,m}$ from the space; see Section 3 for details. In our case, we are not just concerned with finding approximations to the k smallest singular triplets of A, but rather to find a solution to the LS problem. Instead of applying p shifts, we therefore opt to dynamically change the number of shifts to apply in order to have the best approximations to a set of singular triplets in our updated spaces W_{k+1}^+ and V_k^+ (2.18). That is, we look for the largest gap between certain $\tilde{\sigma}$s and only apply shifts up to the gap.

Our heuristic is based on two properties: that the harmonic Ritz singular value approximation $\tilde{\sigma}_i$ to σ_i is such that $\sigma_i \leq \tilde{\sigma}_i$ [18] and the interlace property of the harmonic Ritz and Ritz values [30]. Using these properties leads us to examine the gaps between consecutive harmonic Ritz singular value approximations $\tilde{\sigma}$. If $\tilde{\sigma}_i$ is very close to $\tilde{\sigma}_{i+1}$ (and hence σ_i is possibly very close to σ_{i+1}), then components of the updated starting vector in the direction of u_i from (2.1) may be strongly damped by applying $\tilde{\sigma}_{i+1}^2 = \hat{\theta}_{i+1}$ as a shift, which is undesired. To minimize the possibility of this happening, our heuristic method fixes a small value j and searches the interval $[\hat{\theta}_{k+1-j}, \ldots, \hat{\theta}_{k+1}, \ldots, \hat{\theta}_{k+1+j}]$ around $\hat{\theta}_{k+1}$ for the largest gap between any two consecutive harmonic Ritz values. That is, an index k_j is chosen such that

$$\max_{k+1-j \leq k_j \leq k+j} |\hat{\theta}_{k_{j+1}} - \hat{\theta}_{k_j}| \tag{2.20}$$

and k is replaced with k_j where the number of shifts in the implicitly restarted GK bidiagonalization is set to $p = m - k_j$. Through numerical observation, a suitable choice for j is typically between 2 and 6. Choosing j too large can have a dramatic negative effect on the convergence rate. See Table 6.2 in Section 6 for numerical results for different values of j and an improved convergence rate by using this adaptive shifting strategy.

3. Implicitly restarted LSQR.

In this section we describe the proposed implicitly restarted LSQR method, Algorithm 3.2, which is a combination of a restarted LSQR method with the implicitly restarted GK bidiagonalization method described in Section 2. Algorithm 3.1 outlines a single step of a restarted LSQR method that we will need. A first call to Algorithm 3.1 with an initial approximate solution of the LS problem x_0, $r_0 = b - Ax_0$ and $w_1 = r_0$ will produced the same output x_m and r_m as the Paige and Saunders [31] routine. However, in order to call Algorithm 3.1 again after we use the implicitly restarted formulas of Section 2 to reduce the m-step GK bidiagonalization (2.2) to a k-step GK bidiagonalization (2.18), we need to have $r_m \in \mathcal{R}(W_{k+1}^+)$. If $r_m \notin \mathcal{R}(W_{k+1}^+)$, then we are using a Krylov subspace that does not contain the residual vector. This would require an approximation of r_m from the Krylov subspace, which can severely slow down the procedure or lead to non-convergence.

However, it was shown in [5] that if we have an m-step GK bidiagonalization (2.2) and compute x_m from the LSQR equations, i.e., steps 2 and 3 of Algorithm 3.1, then

$r_m = \gamma W_{m+1}\tilde{u}_{m+1}$, where $\tilde{u}_{m+1}$ is the null vector (2.9) of $B_{m+1,m}$, and $\gamma \in \mathbb{R}$. Using the implicitly restarted formulas of Section 2 with an application of the p largest harmonic Ritz values as shifts, we obtain a k-step GK bidiagonalization decomposition (2.18) with $W^+_{k+1} = W_{m+1}Q_{L_{k+1}}$. Equation (2.17) shows that we must have $\tilde{u}_{m+1} \in \mathcal{R}(Q_{L_{k+1}})$ and hence $r_m = W^+_{k+1}f_{k+1}$ for some vector $f_{k+1} \in \mathbb{R}^{k+1}$, i.e., $r_m \in \mathcal{R}(W^+_{k+1})$.

ALGORITHM 3.1. RESTARTED LSQR STEP

```
Input: k-step GK bidiagonalization (2.18) or (4.2) or the k-step factorization (4.1)
       where $r_k \in \mathcal{R}(W^+_{k+1})$,
       $p = m - k$ : number of additional bidiagonalization steps,
       $x_k \in \mathbb{R}^n$ : approximation to LS problem.

Output: m-step GK bidiagonalization (2.2),
        $x_m \in \mathbb{R}^n$ : approximation to LS problem,
        $r_m \in \mathbb{R}^\ell$ : residual vector.

1. Apply $p = m - k$ additional steps of Algorithm 2.1 to obtain an
   m-step GK bidiagonalization (2.2).
2. Solve $\min_{y_m \in \mathbb{R}^m} \left\| \begin{bmatrix} f_{k+1} \\ 0 \end{bmatrix} - B_{m+1,m}y_m \right\|$ for $y_m$,
   where $r_k = W_{m+1}\begin{bmatrix} f_{k+1} \\ 0 \end{bmatrix}$ for some $f_{k+1} \in \mathbb{R}^{k+1}$.
3. Set $x_m = x_k + P_m y_m$.
4. $r_m = r_k - W_{m+1}B_{m+1,m}y_m$.
```

The residual and approximate solution to the LS problem can be updated during step 1 of Algorithm 3.1, i.e., during the GK bidiagonalization Algorithm 2.1. The MATLAB code `irlsqr` used for numerical examples in Section 6 which implements Algorithm 3.2 updates the LSQR approximation and the residual during the GK bidiagonalization steps. Below is our algorithm that outlines the main routine of this paper.

ALGORITHM 3.2. IRLSQR

```
Input: $A \in \mathbb{R}^{\ell\times n}$ or functions for evaluating matrix-vector products with $A$ and $A^T$,
       $x_0 \in \mathbb{R}^n$: Initial approximate solution to LS problem,
       $r_0 = b - Ax_0 \in \mathbb{R}^\ell$ : initial residual vector,
       $m$ : maximum size of GK bidiagonalization decomposition,
       $p$ : number of shifts to apply,
       $j$ : integer used to adjust number of shifts (2.20),
       $\delta$ : tolerance for accepting an approximate solution.

Output: $x_m$ : approximate solution to the LS problem (1.1),
        $r_m = b - Ax_m \in \mathbb{R}^\ell$ : residual vector.

1. Set $w_1 = r_0$ and $k = 0$.
2. Call Algorithm 3.1 to obtain m-step GK bidiagonalization
       $A^TW_{m+1} = P_mB^T_{m+1,m} + \alpha_{m+1}p_{m+1}e^T_{m+1}$
       $AP_m = W_{m+1}B_{m+1,m}$
   and the solution $x_m$ and the residual $r_m$.
3. If $\|A^Tr_m\|/\|A^Tr_0\| < \delta$ then exit.
```

4. Compute the m harmonic Ritz values, (2.10)*.*
5. Adjust the number of shifts p using user input j and (2.20)*.*
6. Apply the largest p harmonic Ritz values as shifts to obtain the k-step GK bidiagonalization (2.18)*,*

$$\begin{array}{rcl} A^T W_{k+1}^+ & = & P_k^+ B_{k+1,k}^{+T} + \alpha_{k+1}^+ p_{k+1}^+ e_{k+1}^T \\ A P_k^+ & = & W_{k+1}^+ B_{k+1,k}^+ \end{array}.$$

7. Set $x_k = x_m$ and $r_k = r_m$ and goto 2.

We remark that the computation of $\|A^T r_m\|/\|A^T r_0\|$ in line 3 can be done efficiently using the formula in [20]. The applications of the implicit shifts of the harmonic Ritz values in step 6 of Algorithm 3.2 with the bulgechasing Algorithm 2.2 does not always yield the required structure of $B_{m+1,m}^+$, i.e., $\alpha_m^+ = 0$. For small values of m we do get $\alpha_m^+ \approx 0$, however, for modest values, $m \approx 100$, we get $\alpha_m^+ \neq 0$. Therefore, we developed an alternate method for applying the shifts that is discussed in the next section.

4. Harmonic bidiagonal method.

The bulgechasing algorithm applies a shift implicitly to the bidiagonal matrix $B_{m+1,m}$ while outputting two orthogonal upper Hessenberg matrices Q_R and Q_L such that $B_{m+1,m}^+ = Q_L^T B_{m+1,m} Q_R$. For the success of our method we need the output matrices Q_R and Q_L to be upper Hessenberg with the last columns as singular vectors and α_m^+ of $B_{m+1,m}^+$ (cf. (2.11), (2.12), (2.13), (2.16)) zero. However, in finite precision arithmetic, Algorithm 2.2 (and the upper bidiagonal form of the algorithm) is prone to round off errors and the diagonal element α_m^+ of $B_{m+1,m}^+$ is not always zero; cf. Table 4 and [38] for a discussion. If the diagonal entry α_m^+ of $B_{m+1,m}^+$ is nonzero, then by Theorem 2.3 we did not shift by a harmonic Ritz value $\tilde{\sigma}_j^2$ and hence, $r_m \notin \mathcal{R}(W_{k+1}^+)$; cf. the discussion in Section 3.

Other implicitly restarted methods [4, 20, 21, 22, 24] that apply shifts implicitly can overcome the issue of a nonzero α_m^+ by incorporating α_m^+ into equation (2.5). This strategy does not work in our method. Alternatively, the bulgechasing Algorithm 2.2 can be called repeatedly with the same shift until α_m^+ becomes small. This process destroys the required upper Hessenberg structure of Q_R and Q_L and often requires many calls for a single shift. To overcome this difficulty and force the desired (i.e., required) structure for this paper, we developed a method, Algorithm 4.1, for implicitly applying the harmonic Ritz values as shifts that utilizes the singular values and vectors of $B_{m+1,m}$.

Algorithm 4.1 takes advantage of the known structure of the orthogonal matrices Q_L and Q_R. That is, in exact arithmetic, the application of $p = m - k$ harmonic Ritz values $(\tilde{\sigma}_m^2, \ldots, \tilde{\sigma}_{k+1}^2)$ with Algorithm 2.2 yields banded upper Hessenberg matrices (2.16)–(2.17),

$$\begin{array}{rcl} Q_L & = & [Q_{L_{k+1}}, \tilde{u}_{k+1}, \ldots, \tilde{u}_m], \\ Q_R & = & [Q_{R_k}, \tilde{v}_{k+1}, \ldots, \tilde{v}_m], \end{array}$$

with p sub-diagonals below the diagonal. The first vector $q_{L_1} \in Q_{L_{k+1}}$ has, at most, the first $p+1$ entries nonzero and is orthogonal to the p vectors $\{\tilde{u}_{k+1}, \ldots, \tilde{u}_m\}$. The vector q_{L_1} can be easily constructed by finding a vector of length $p+1$ that is orthogonal to the first $p+1$ entries of each vector in $\{\tilde{u}_{k+1}, \ldots, \tilde{u}_m\}$ and replacing the first $p+1$ entries of q_{L_1} with that vector. The process can be repeated to find the second vector $q_{L_2} \in Q_{L_{k+1}}$ by finding a vector of length $p+2$ that is orthogonal to first $p+2$ entries of each vector in $\{q_{L_1}, \tilde{u}_{k+1}, \ldots, \tilde{u}_m\}$ and replacing the first $p+2$ entries of q_{L_2} with that vector. The matrix Q_R is constructed in the same manner. The matrices Q_L and Q_R are constructed to be orthogonal banded upper Hessenberg matrices, and $B_{m+1,m}^+ = Q_L^T B_{m+1,m} Q_R$ will have

TABLE 4.1
The numerical value of $|\alpha_m^+|$ from $B_{m+1,m}^+$ after computing an m-step GK bidiagonalization (2.4) for the matrix [14] $ILLC1850 \in \mathbb{R}^{1850\times 712}$ and calling Algorithms 2.2 and 4.1 to apply the largest harmonic Ritz value as a shift. The computation time is not reported since it is considered negligible in the overall method.

m	Method of Implicit Shift	$\lvert\alpha_m^+\rvert$	$\lVert q_{L_{m+1}} - u_m\rVert$	$\lVert q_{R_m} - v_m\rVert$
20	Algorithm 2.2 (Bulgechasing)	2.3e-11	2.2e-11	2.6e-11
20	Algorithm 4.1 (Harmonic Bidiagonal)	1.1e-19	0	0
40	Algorithm 2.2 (Bulgechasing)	7.7e-10	1.1e-4	6.4e-5
40	Algorithm 4.1 (Harmonic Bidiagonal)	3.6e-17	0	0
80	Algorithm 2.2 (Bulgechasing)	1.41	1.52	1.61
80	Algorithm 4.1 (Harmonic Bidiagonal)	2.7e-17	0	0
120	Algorithm 2.2 (Bulgechasing)	1.1e-6	1.4	1.4
120	Algorithm 4.1 (Harmonic Bidiagonal)	1.8e-16	0	0

each $\alpha_i^+ = 0$ for $i = k+1$ to m. However, the lower bidiagonal structure of $B_{m+1,m}^+$ may be compromised. It may happen that for some (or many) values of i, the first $p+i$ entries of the columns of $[\tilde{u}_{k+1},\dots,\tilde{u}_m]$ (and $[\tilde{v}_{k+1},\dots,\tilde{v}_m]$) may form a rank deficient matrix, and hence steps 3 and 6 of Algorithm 4.1 may return multiple vectors that satisfy the above criteria. The matrices Q_L, Q_R, and $B_{m+1,m}^+$, however, will have the required structure for our method; and since Q_L and Q_R are orthogonal transformations, the singular values of the updated $B_{m+1,m}^+$ (not necessarily bidiagonal) matrix obtained from Algorithm 4.1 will be the same as the bidiagonal matrix which would have been obtained from Algorithm 2.2.

After using Algorithm 4.1 to apply the shifts, we have the following k-step factorization

$$\begin{aligned} A^T W_{k+1}^+ &= P_k^+ B_{k+1,k}^{+T} + \alpha_{k+1}^+ p_{k+1}^+ e_{k+1}^T \\ A P_k^+ &= W_{k+1}^+ B_{k+1,k}^+, \end{aligned} \tag{4.1}$$

which is similar to (2.18) except that $B_{k+1,k}^+$ may not be lower bidiagonal. Algorithm 3.2 can be successfully used with equation (4.1) by applying the shifts in step 6 of Algorithm 3.2 via Algorithm 4.1.

The k-step GK bidiagonalization decomposition (2.18) can be recaptured by returning $B_{k+1,k}^+$ to lower bidiagonal form via orthogonal transformations with a row-wise Householder method starting with the last row; see, e.g., [36, 37]. Using row-wise Householder transformations (starting with the last row) creates orthogonal matrices $\breve{Q}_L \in \mathbb{R}^{(k+1)\times(k+1)}$ and $\breve{Q}_R \in \mathbb{R}^{k\times k}$ such that $\breve{B}_{k+1,k} = \breve{Q}_L^T B_{k+1,k}^+ \breve{Q}_R$ is lower bidiagonal where

$$\breve{Q}_L = \begin{bmatrix} \star & 0 \\ 0 & 1 \end{bmatrix}.$$

Letting $\breve{P}_k = P_k^+ \breve{Q}_R$ and $\breve{W}_{k+1} = W_{k+1}^+ \breve{Q}_L$, we can recover a k-step GK bidiagonalization decomposition

$$\begin{aligned} A^T \breve{W}_{k+1} &= \breve{P}_k \breve{B}_{k+1,k}^T + \alpha_{k+1}^+ p_{k+1}^+ e_{k+1}^T, \\ A \breve{P}_k &= \breve{W}_{k+1} \breve{B}_{k+1,k}, \end{aligned} \tag{4.2}$$

where $\breve{B}_{k+1,k}$ is lower bidiagonal. The MATLAB code `irlsqr`, used for numerical examples in Section 6 and which implements Algorithm 3.2, can be used with either structure (4.1) and (4.2). The authors notice no numerical differences.

ALGORITHM 4.1. HARMONIC BIDIAGONAL METHOD

```
Input: $[\tilde{u}_{k+1}, \tilde{u}_{k+2}, \ldots, \tilde{u}_m] \in \mathbb{R}^{(m+1)\times p}$ : left singular vectors of $B_{m+1,m}$ (2.9),
       $[\tilde{v}_{k+1}, \tilde{v}_{k+2}, \ldots, \tilde{v}_m] \in \mathbb{R}^{m\times p}$ : right singular vectors of $B_{m+1,m}$ (2.9).
Output: $Q_L \in \mathbb{R}^{(m+1)\times(m+1)}$ : banded orthogonal upper Hessenberg,
        $Q_R \in \mathbb{R}^{m\times m}$ : banded orthogonal upper Hessenberg,
        $B^+_{m+1,m} = Q_L^T B_{m+1,m} Q_R \in \mathbb{R}^{(m+1)\times m}$ : updated matrix.
1. Set $Q_{L_{k+1}} := [\,]$ and $Q_{R_{k+1}} := [\,]$.
2. for $i = 1 : k+1$
    3. Find a vector $q_{L_i} \in \mathbb{R}^{p+i}$ orthogonal to the first $p+i$ rows of each
       column of $[Q_{L_{k+1}}, \tilde{u}_{k+1}, \tilde{u}_{k+2}, \ldots, \tilde{u}_m]$.
    4. Set $Q_{L_{k+1}} := \left[ Q_{L_{k+1}}, \begin{bmatrix} q_{L_i} \\ 0 \end{bmatrix} \right]$.
    5. if $i \leq k$
        6. Find a vector $q_{R_i} \in \mathbb{R}^{p+i}$ orthogonal to the first $p+i$ rows of each
           column of $[Q_{R_{k+1}}, \tilde{v}_{k+1}, \tilde{v}_{k+2}, \ldots, \tilde{v}_m]$.
        7. Set $Q_{R_{k+1}} := \left[ Q_{R_{k+1}}, \begin{bmatrix} q_{R_i} \\ 0 \end{bmatrix} \right]$.
    8. endif
9. endfor
10. Set $B^+_{m+1,m} = Q_L^T B_{m+1,m} Q_R$.
```

Steps 3 and 6 of Algorithm 4.1 can be done in several ways, e.g., the MATLAB command `null` applied to the transpose of the first $p+i$ rows of $[Q_{L_{k+1}}, \tilde{u}_{k+1}, \tilde{u}_{k+2}, \ldots, \tilde{u}_m]$.

5. Connection to augmented LSQR.

5. Connection to augmented LSQR. This section shows the parallels between the augmented LSQR algorithm described in [5] and the implicitly restarted LSQR algorithm described in this paper. Both algorithms use a restarted GK bidiagonalization in conjunction with LSQR to solve the LS problem. The augmented LSQR algorithm of [5] is carried out by explicitly augmenting the Krylov subspaces (1.2) with the harmonic Ritz vectors associated with the smallest harmonic Ritz values. We briefly describe the spaces that result from the augmenting routine and refer the reader to [5] for the full details.

The harmonic Ritz vector of AA^T associated with the harmonic Ritz value $\hat{\theta}_j$ is defined as

$$\hat{u}_j = W_m g_j,$$

where g_j is the corresponding eigenvector from equation (2.8). Furthermore, it was shown in [5] that the eigenvector g_j can also be expressed as

$$g_j = \left[\begin{array}{cc} I_m & \beta_{m+1} B_{m,m}^{-T} e_m \end{array} \right] \tilde{u}_j,$$

where $\tilde{u}_j$ is the corresponding left singular vector associated with the singular value $\tilde{\sigma}_j$ from (2.9). Similar to our method for the initial iteration, the augmenting method in [5] sets $w_1 = r_0$ and calls Algorithm 2.2 to obtain the m-step GK bidiagonalization (2.2). The augmenting restarting step of [5] creates a variant of the equations (2.18),

$$\begin{aligned} A^T \hat{W}_{k+1} &= \hat{P}_k \hat{B}^T_{k+1,k} + (\alpha_{m+1} \hat{q}_{m+1,k+1}) p_{m+1} e^T_{m+1}, \\ A\hat{P}_k &= \hat{W}_{k+1} \hat{B}_{k+1,k}, \end{aligned} \tag{5.1}$$

where $\hat{W}_{k+1} = W_{m+1}\hat{Q}$, $\hat{P}_k = P_m\tilde{V}_k$, $\hat{B}_{k+1,k} = \hat{Q}^T\tilde{U}_k\tilde{\Sigma}_k$, and $\hat{q}_{m+1,k+1}$ is the $(m+1, k+1)$ element of $\hat{Q}$. The matrices $\tilde{U}_k$ and $\tilde{V}_k$ are the left and right singular vectors of $B_{m+1,m}$, respectively, associated with the k smallest singular values, and $\tilde{\Sigma}_k$ is the diagonal matrix of the k smallest singular values. The matrix $\hat{Q}$ is taken from the QR decomposition of

$$\hat{Q}\hat{R} = \left[\begin{array}{c|c} \left[\tilde{u}_{m+1_{m+1}}I_m \quad -\tilde{u}_{m+1_{1:m}}\right]\tilde{U}_k & \tilde{u}_{m+1} \\ \hline 0 & \end{array}\right], \tag{5.2}$$

where $\tilde{u}_{m+1_{m+1}} \in \mathbb{R}$ is the $(m+1)$st element of the null vector $\tilde{u}_{m+1}$ and $\tilde{u}_{m+1_{1:m}} \in \mathbb{R}^m$ has as entries the first m elements of the null vector $\tilde{u}_{m+1}$. The matrix on the right side of (5.2) is considered to be full rank, and hence $\hat{R}$ is invertible. We will show that $\mathcal{R}(W^+_{k+1}) = \mathcal{R}(\hat{W}_{k+1})$ and $\mathcal{R}(P^+_k) = \mathcal{R}(\hat{P}_k)$.

THEOREM 5.1. *Let $w_1 = r_0$ and call Algorithm 2.2 to obtain the m-step GK bidiagonalization (2.2). Then the matrices W^+_{k+1} and P^+_k of (2.18) that are created by applying the $p = m - k$ largest harmonic Ritz values as shifts span, respectively, the same spaces as the matrices $\hat{W}_{k+1}$ and $\hat{P}_k$ of (5.1), i.e., $\mathcal{R}(W^+_{k+1}) = \mathcal{R}(\hat{W}_{k+1})$ and $\mathcal{R}(P^+_k) = \mathcal{R}(\hat{P}_k)$.*

Proof: Using the formulas of Section 2 to apply the largest $p = m - k$ harmonic Ritz value as shifts generates the orthogonal matrices $Q_R = [Q_{R_k}, \tilde{v}_{k+1}, \ldots, \tilde{v}_m]$, and $Q_L = [Q_{L_{k+1}}, \tilde{u}_{k+1}, \ldots, \tilde{u}_m]$, cf. (2.17). Since $\mathcal{R}(Q_{R_k}) = \mathcal{R}(\tilde{V}_k)$, $P^+_k = P_mQ_{R_k}$, and $\hat{P}_k = P_m\tilde{V}_k$, we have

$$\mathcal{R}(P^+_k) = \mathcal{R}(\hat{P}_k).$$

Define $\tilde{U}_{k+1:m} = [\tilde{u}_{k+1}, \ldots, \tilde{u}_m]$ and notice that $\tilde{U}^T_{k+1:m}Q_{L_{k+1}} = 0$, and

$$\tilde{U}^T_{k+1:m}\left[\begin{array}{c|c} \left[\tilde{u}_{m+1_{m+1}}I_m \quad -\tilde{u}_{m+1_{1:m}}\right]\tilde{U}_k & \tilde{u}_{m+1} \\ \hline 0 & \end{array}\right] = 0.$$

Since the matrices of (5.2) are of full rank we have $\mathcal{N}(\hat{Q}^T) = \mathcal{N}(Q^T_{L_{k+1}})$ and $\mathcal{R}(Q_{L_{k+1}}) = \mathcal{R}(\hat{Q})$. Since $W^+_{k+1} = W_{m+1}Q_{L_{k+1}}$, and $\hat{W}_{k+1} = W_{m+1}\hat{Q}$, we have

$$\mathcal{R}(W^+_{k+1}) = \mathcal{R}(\hat{W}_{k+1}). \qquad \square$$

In Section 3 we showed that the residual r_m of LSQR is in the restarted space W^+_{k+1} when implicit restarting is applied with the largest p harmonic Ritz values as shifts, and it is in $\hat{W}_{k+1}$ by construction (cf. [5, equation 3.9]). Furthermore, the restart vector p^+_{k+1} (2.19) of the implicitly restarted method is a multiple of p_{m+1}, and extending the k-step GK bidiagonalization methods (2.18) and (5.1) back to m-step GK bidiagonalization will produce $\mathcal{R}(W^+_{m+1}) = \mathcal{R}(\hat{W}_{m+1})$ and $\mathcal{R}(P^+_m) = \mathcal{R}(\hat{P}_m)$. The process is then repeated.

6. Numerical examples.

In this section we present some numerical examples to show the performance of Algorithm 3.2, which is implemented by the Matlab code `irlsqr`*. The

*Code is available at `http://www.math.uri.edu/~jbaglama`.

TABLE 6.1
List of matrices A, properties, and b vectors used in numerical examples. The first two matrices are taken from the Matrix Market Collection and the last two are from the University of Florida Sparse Matrix Collection.

Example	A	ℓ	n	nnz	b
6.1	ILLC1850	1850	712	8638	ILLC1850_RHS1
6.2	E30R0000	9661	9661	305794	E30R0000_RHS1
6.3	LANDMARK	71952	2704	1146848	`rand(71952,1)`
6.4	BIG_DUAL	30269	30269	89858	$A\cdot$`rand(30269,1)`

TABLE 6.2
Number of matrix vector products with A and A^T required to get $\|A^Tr\|/\|A^Tr_0\| \leq 10^{-12}$ using different values of j in the gap strategy given in Section 2. The table shows two different choices of p (number of shifts) and sets $m = 100$ for the examples with the matrix ILLC1850 and $m = 200$ for the examples with the matrix E30R0000. Column $j = 0$ corresponds to no adjustments.

A	p	$j=0$	$j=3$	$j=6$	$j=9$
ILLC1850	20	3825	3647	3630	3657
ILLC1850	30	3750	3689	3681	3679
E30R0000	30	31753	30953	30153	42223
E30R0000	40	34731	30723	31037	31223

code uses the following user-specified parameters:

tol	Tolerance for accepting a computed approximate solution x as a solution to the LS problem (1.1), i.e., $\|A^Tr\|/\|A^Tr_0\| \leq$*tol*.
m	Maximum number of GK bidiagonal steps.
p	Number of shifts to apply.
j	Size of interval around $\hat{\theta}_{k+1}$ to find largest gap.
maxit	Maximum number of restarts.
reorth12	String deciding whether one or two sided reorthogonalization is used in Algorithm 2.1.

We compared the IRLSQR algorithm against the LSQR and LSMR algorithms. The latter codes were adapted to output the norm of the residuals after each bidiagonal step. Furthermore, LSQR was adapted to perform either one or two sided reorthogonalization against any specified number of previous GK vectors. In order to make a fair comparison between the three methods and to keep storage requirements the same with the GK vectors, LSQR and LSMR were reorthogonalized against only the previous m vectors. No comparisons were made with ALSQR of [5], since these routines are mathematically equivalent; see Section 5.

All computations were carried out using MATLAB version 7.12.0.0635 R2011a on a Dell XPS workstation with an Intel Core2 Quad processor and 4 GB of memory running under the Windows Vista operating system. Machine precision is $2.2 \cdot 10^{-16}$. We present numerical examples with matrices taken from the University of Florida Sparse Matrix Collection [13] and from the Matrix Market Collection [12]; see Table 6.1. All examples use the initial approximate solution as $x_0 = 0$, and $r_0 = b$.

In Table 6.2, we used two matrices ILLC1850 and E30R0000 from the Matrix Market Collection along with the accompanied b vectors ILLC1850_RHS1 and E30R0000_RHS1, respectively, to show the results for various values j for the gap strategy outlined in Section 2.3. For all of the numerical examples, we set j equal to 5.

EXAMPLE 6.1. We let the matrix A and vector b be ILLC1850 and ILLC1850_RHS1, respectively. This matrix and b vector are from the LSQ group of the Matrix Market which

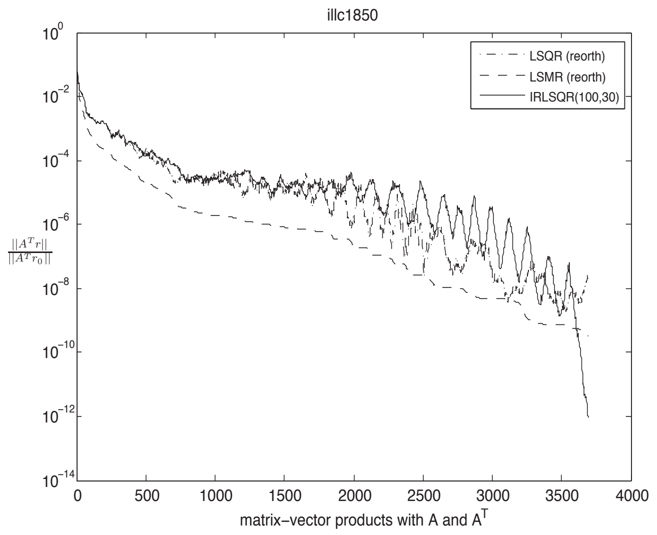


FIG. 6.1. *Example 6.1: A = ILLC1850, b = ILLC1850_RHS1. IRLSQR(100,30) indicates $m = 100$ and $p = 30$. LSMR (reorth) and LSQR (reorth) indicate reorthogonalization is performed against the m most recent vectors. IRLSQR converged at 3,693 matrix-vector products.*

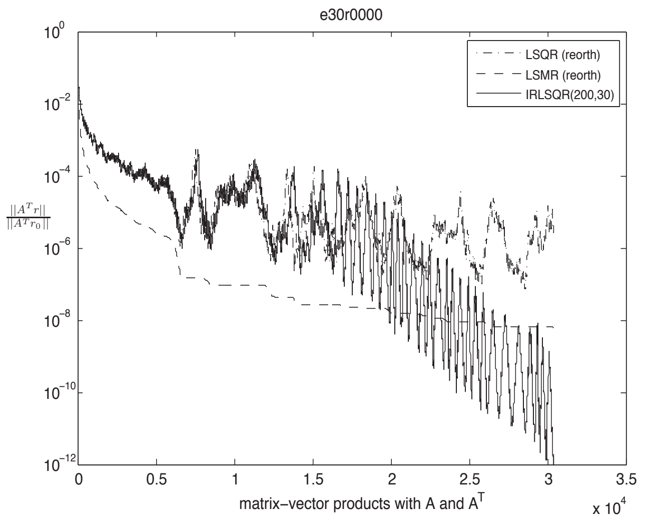
consist of LS problems in surveying. The use of the matrix is to test iterative solvers, and it was one of the matrices used by Paige and Saunders [31] in the testing of the LSQR algorithm. For this example, $b \notin \mathcal{R}(A)$, and therefore we only show convergence of the quotient $\|A^T r\|/\|A^T r_0\|$. The function `irlsqr` was used with parameters $m = 100$, $p = 30$, and $reorth12 = 1$. We set $tol = 1 \cdot 10^{-12}$. In Figure 6.1, we plot $\|A^T r\|/\|A^T r_0\|$ versus the number of matrix-vector products with A and A^T.

EXAMPLE 6.2. The matrix A was chosen to be E30R0000, and the right hand side b was taken as E30R0000_RHS1. The matrix A and vector b are from the DRICAV group of matrices from the Matrix Market Collection. The group consists of matrices used from modeling 2D fluid flow in a driven cavity, and the main purpose of matrices from this collection is for testing iterative solvers. The matrix is nonsymmetric and indefinite. Since the matrix A is square and full rank, $b \in \mathcal{R}(A)$, and therefore we show convergence of the quotients $\|A^T r\|/\|A^T r_0\|$ and $\|r\|/\|r_0\|$; see Figure 6.2. We used `irlsqr` with parameters $m = 200$, $p = 30$, and $reorth12 = 2$. We used two-sided reorthogonalization since the condition number of this matrix is approximately $3.47 \cdot 10^{11}$. We set $tol = 1 \cdot 10^{-12}$ and accept an iterate x as a solution to the LS problem if $\|A^T r\|/\|A^T r_0\| < 1 \cdot 10^{-12}$.

EXAMPLE 6.3. The matrix A was chosen to be LANDMARK of the Pereyra group from the University of Florida Sparse Matrix Collection. It comes from an LS problem. The matrix LANDMARK does not have a corresponding b vector, hence we chose it to be random with the MATLAB command `rand(71952,1)`. The rank of the matrix A is 2671 and we do not assume $b \in \mathcal{R}(A)$, therefore we only show convergence of the quotient $\|A^T r\|/\|A^T r_0\|$; see Figure 6.3. We used `irlsqr` with parameters $m = 250$, $p = 35$, and $reorth12 = 1$. Setting $tol = 1 \cdot 10^{-10}$, an iterate x is accepted as a solution to the LS problem if $\|A^T r\|/\|A^T r_0\| < 1 \cdot 10^{-10}$.

EXAMPLE 6.4. The matrix A was chosen to be BIG_DUAL of the AG-Monien group from the University of Florida Sparse Matrix Collection. The matrix is from a 2D finite element problem. The matrix BIG_DUAL does not have a corresponding b vector. The rank

ETNA
Kent State University
http://etna.math.kent.edu



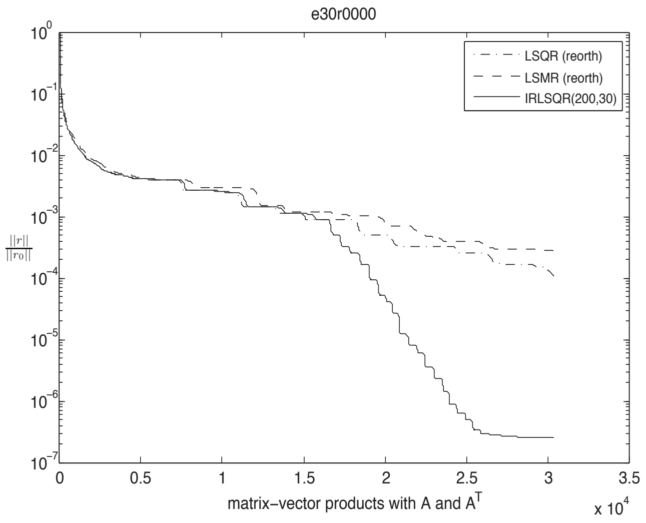


FIG. 6.2. *Example 6.2: A = E30R0000, b = E30R0000_RHS1. IRLSQR(200,30) indicates $m = 200$ and $p = 30$. LSMR (reorth) and LSQR (reorth) indicate reorthogonalization is performed against the 200 most recent vectors. The top graph shows the convergence of $\|A^T r\|/\|A^T r_0\|$ and the bottom graph shows the convergence of $\|r\|/\|r_0\|$.* `irlsqr` *converged at 30,421 matrix-vector products.*

of the matrix is 30,239 (not full rank), we chose the vector b to be A·`rand(30269,1)` so that $b \in \mathcal{R}(A)$. We plot the quotients $\|A^T r\|/\|A^T r_0\|$ and $\|r\|/\|r_0\|$; see Figure 6.4. We used`irlsqr` with parameters $m = 300$, $p = 45$, and $reorth12 = 1$. Setting $tol = 1 \cdot 10^{-14}$, an iterate x is accepted as a solution to the LS problem if $\|A^T r\|/\|A^T r_0\| < 1 \cdot 10^{-14}$.

7. Conclusion.

We have presented a new implicitly restarted LSQR algorithm for solving the LS problem. Theoretical results show the restarting to be equivalent to the augmented LSQR algorithm of [5]. However, this version is much simpler to implement. The gap strategy and ease of implementation of this method make it desirable. Numerical examples show the proposed new method is competitive with existing methods.

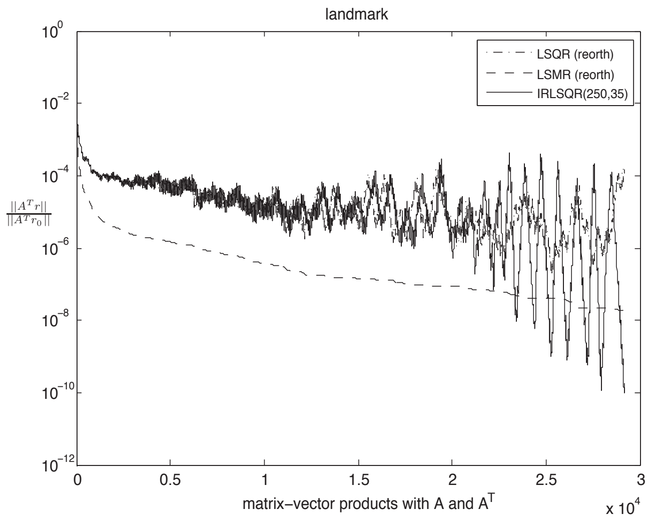


FIG. 6.3. *Example 6.3: A = LANDMARK and $b =$* `rand(71952,1)`*. IRLSQR(250,35) indicates $m = 250$ and $p = 35$. LSMR (reorth) and LSQR (reorth) indicate reorthogonalization is performed against the 250 most recent vectors. The graph shows the convergence of $\|A^T r\|/\|A^T r_0\|$.* `irlsqr` *converged at 29,185 matrix-vector products.*

REFERENCES

[1] J. BAGLAMA, D. CALVETTI, G.H. GOLUB, AND L. REICHEL, *Adaptively preconditioned GMRES algorithms*, SIAM J. Sci. Comput., 20, (1998), pp. 243–269.

[2] J. BAGLAMA AND L. REICHEL, *Augmented implicitly restarted Lanczos bidiagonalization methods*, SIAM J. Sci. Comput., 27 (2005), pp. 19–42.

[3] ———, *Restarted block Lanczos bidiagonalization methods*, Numer. Algorithms, 43 (2006), pp. 251–272.

[4] ———, *An implicitly restarted block Lanczos bidiagonalization method using Leja shifts*, BIT, 53 (2013), pp. 285–310.

[5] J. BAGLAMA, L. REICHEL, AND D. RICHMOND, *An augmented LSQR method*, Numer. Algorithms, 64 (2013), pp. 263–293.

[6] A. H. BAKER, E. R. JESSUP, AND T. MANTEUFFEL, *A technique for accelerating the convergence of restarted GMRES*, SIAM. J. Matrix Anal. Appl., 26 (2005), pp. 962–984.

[7] M. BENZI, *Preconditioning techniques for large linear systems: a survey*, J. Comput. Phys., 182 (2002), pp. 418–477.

[8] M. BENZI AND M. TUMA, *A robust preconditioner with low memory requirements for large sparse least squares problems*, SIAM J. Sci. Comput., 25 (2003), pp. 499–512.

[9] A. BJÖRCK, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.

[10] A. BJÖRCK AND J. Y. YUAN, *Preconditioners for least squares problems by LU factorization*, Electron. Trans. Numer. Anal. 8 (1999), pp. 26–35.
http://etna.math.kent.edu/vol.8.1999/pp26-35.dir

[11] A. BJÖRCK, E. GRIMME, AND P. V. DOOREN, *An implicit shift bidiagonalization algorithm for ill-posed systems*, BIT, 34 (1994), pp. 510–534.

[12] R. BOISVERT, R. POZO, K. REMINGTON, B. MILLER, AND R. LIPMAN, *Matrix Market*, 1996. Available at http://math.nist.gov/MatrixMarket/.

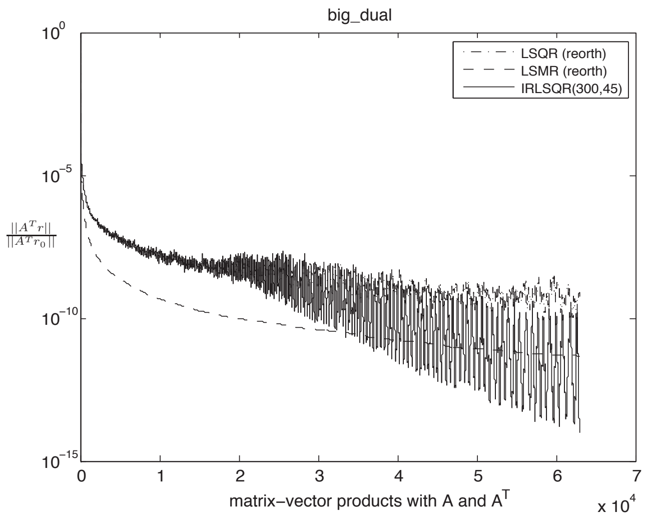
[13] T. A. DAVIS AND Y. HU, *The University of Florida Sparse Matrix Collection*, ACM Trans. Math. Software, 38 (2011), 1 (25 pages).

[14] 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. Matrices available at [12].

[15] D. C.-L. FONG AND M. SAUNDERS, *LSMR: An iterative algorithm for sparse least-squares problems*, SIAM J. Sci. Comp., 33 (2011), pp. 2950–2971.

[16] G. H. GOLUB AND W. KAHAN, *Calculating the singular values and pseudo-inverse of a matrix*, J. Soc. Indust. Appl. Math. Ser. B Numer. Anal., 2 (1965), pp. 205–224.

[17] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, 3rd ed. John Hopkins University Press, Baltimore, 1996.



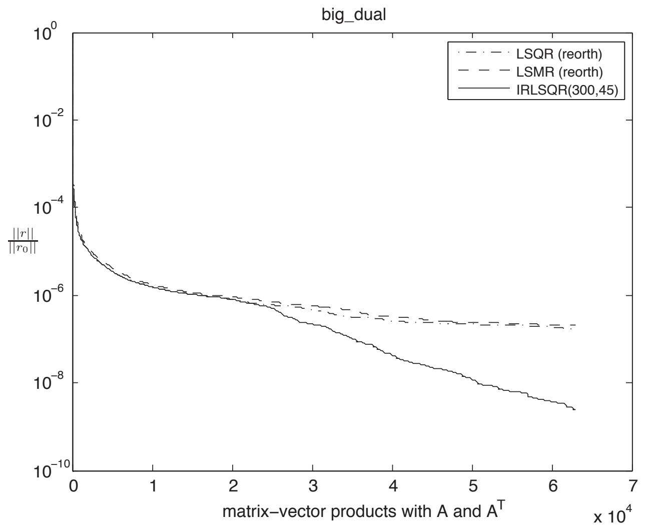


FIG. 6.4. *Example 6.4: A = BIG_DUAL and $b = A\cdot$* `rand(30269,1)`. *IRLSQR(300,45) indicates $m = 300$ and $p = 45$. LSMR (reorth) and LSQR (reorth) indicate reorthogonalization is performed against the 300 most recent vectors. The graph shows the convergence of $\|A^T r\|/\|A^T r_0\|$. IRLSQR converged at 62,961 matrix-vector products.*

[18] M. E. HOCHSTENBACH, *Harmonic and refined extraction methods for the singular value problem, with applications in least squares problems*, BIT, 44 (2004), pp. 721–754.
[19] T. ITO AND K. HAYAMI, *Preconditioned GMRES methods for least squares problems*, Japan J. Indust. Appl. Math., 25 (2008), pp. 185–207.
[20] Z. JIA, *Some properties of LSQR for large sparse linear least squares problems*, J. Syst. Sci. Complex, 23 (2010), pp. 815–821.
[21] 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.
[22] ———, *A refined harmonic Lanczos bidiagonalization method and an implicitly restarted algorithm for computing the smallest singular triplets of large matrices*, SIAM J. Sci. Comput., 32 (2010), pp. 714–744.
[23] S. KARIMI, D. K. SALKUYEH, AND F. TOUTOUNIAN, *A preconditioner for the the LSQR algorithm*, J. Appl. Math. and Inform., 26 (2008), pp. 213–222.

[24] E. KOKIOPOULOU, C. BEKAS, AND E. GALLOPOULOS, *Computing smallest singular triplets with implicitly restarted Lanczos bidiagonalization*, Appl. Numer. Math., 49 (2004), pp. 39–61.
[25] R. M. LARSEN, *Lanczos Bidiagonalization with Partial Reorthogonalization*, Ph.D. Thesis, Dept. Computer Science, University of Aarhus, Denmark, (1998).
[26] R. M. LARSEN, *Combining implicit restarts and partial reorthogonalization in Lanczos bidiagonalization*, Talk at UC Berkeley, Berkeley, 2001
[27] R. B. MORGAN, *Computing interior eigenvalues of large matrices*, Linear Algebra Appl., 154–156 (1991), pp. 289–309.
[28] ———, *A restarted GMRES method augmented with eigenvectors*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 1154–1171.
[29] ———, *Implicitly restarted GMRES and Arnoldi methods for nonsymmetric systems of equations*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1112–1135.
[30] C. C. PAIGE, B. N. PARLETT, AND H. A. VAN DER VORST, *Approximate solutions and eigenvalue bounds from Krylov subspaces*, Numer. Linear Algebra Appl., 2 (1995), pp. 115–134.
[31] C. C. PAIGE AND M. A. SAUNDERS, *LSQR: an algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Sofrware, 8 (1982), pp. 43–71.
[32] B. PARLETT, *The Symmetric Eigenvalue Problem*, SIAM, Philadelphia, 1998.
[33] Y. SAAD, *Iterative Methods for Sparse Linear Systems*, 2nd. ed., SIAM, Philadelphia, 2003.
[34] Y. SAAD AND M. H. SCHULTZ, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.
[35] 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.
[36] G. STEWART, *A Krylov-Schur Algorithm for Large Eigenproblems*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 601–614.
[37] M. STOLL, *A Krylov-Schur Approach to the Truncated SVD*, Linear Algebra Appl., 436 (2012), pp. 2795–2806.
[38] D. S. WATKINS, *The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods*, SIAM, Philadelphia, 2007.
[39] I. ZAVORIN, D. P. O'LEARY, AND H. ELMAN, *Complete stagnation of GMRES*, Linear Algebra Appl., 367 (2003), pp. 165–183.