SIAM J. MATRIX ANAL. APPL.
Vol. 26, No. 4, pp. 962–984

A TECHNIQUE FOR ACCELERATING THE CONVERGENCE OF RESTARTED GMRES*

A. H. BAKER†, E. R. JESSUP‡, AND T. MANTEUFFEL§

Abstract. We have observed that the residual vectors at the end of each restart cycle of restarted GMRES often alternate direction in a cyclic fashion, thereby slowing convergence. We present a new technique for accelerating the convergence of restarted GMRES by disrupting this alternating pattern. The new algorithm resembles a full conjugate gradient method with polynomial preconditioning, and its implementation requires minimal changes to the standard restarted GMRES algorithm.

Key words. GMRES, iterative methods, Krylov subspace, restart, nonsymmetric linear systems

AMS subject classification. 65F10

DOI. 10.1137/S0895479803422014

1. Introduction.

Iterative methods are a common choice for solving the large sparse system of linear equations

$$Ax = b, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ is nonsingular and $x, b \in \mathbb{R}^n$. A popular class of iterative methods are Krylov subspace methods. Krylov subspace methods find an approximate solution

$$x_i \in x_0 + K_i(A, r_0), \tag{2}$$

where $K_i(A, r_0) \equiv span\{r_0, Ar_0, \ldots, A^{i-1}r_0\}$ denotes an i-dimensional Krylov subspace, x_0 is the initial guess, and r_0 is the initial residual ($r_0 \equiv b - Ax_0$). Krylov subspace methods are also known as polynomial methods since (2) implies that the residual r_i can be written in terms of a polynomial in A: $r_i = p(A)r_0$.

At present, a large variety of Krylov subspace methods exist. When A is nonsymmetric, choosing the most appropriate method can be difficult (see, e.g., [22]), though the generalized minimum residual algorithm (GMRES) [27] is arguably the

*Received by the editors January 24, 2003; accepted for publication (in revised form) by Z. Strakoš July 30, 2004; published electronically May 6, 2005.

http://www.siam.org/journals/simax/26-4/42201.html

†Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Box 808 L-551, Livermore, CA 94551 (abaker@llnl.gov). The work of this author was primarily supported by the Department of Energy Computational Science Graduate Fellowship Program of the Office of Scientific Computing and Office of Defense Programs in the Department of Energy under contract DE-FG02-97ER25308. Portions of this work were performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract W-7405-Eng-48.

‡Department of Computer Science, University of Colorado, Boulder, CO 80309-0430 (jessup@cs.colorado.edu). The work of this author was supported by the National Science Foundation under grant ACI-0072119.

§Department of Applied Mathematics, University of Colorado, Boulder, CO 80309-0526 (tmanteuf@colorado.edu). The work of this author was supported by the National Institute of Health under grant 1-R01-EY12292-01, the National Science Foundation under grant DMS–0084438, the Department of Energy under grant DE-FG03-94ER25217 and DE-FC02-01ER25479, and the National Science Foundation under VIGRE grant DMS-9810751.

most popular choice. GMRES is often referred to as an "optimal" method because it finds the approximate solution in the Krylov subspace that minimizes the 2-norm of the residual [27].

At each iteration of GMRES, the amount of storage and computational work required increases. Therefore, when the required resources make standard GMRES impractical, the restarted version of the algorithm is used as suggested in [27]. In restarted GMRES (GMRES(m)), the method is "restarted" once the Krylov subspace reaches dimension m, and the current approximate solution becomes the new initial guess for the next m iterations. The restart parameter m is generally chosen small relative to n to keep storage and computation requirements reasonable. However, choosing an appropriate restart parameter can be difficult as the choice can significantly affect the convergence rate (see, e.g., [17, 13]).

In general, restarting slows the convergence of GMRES. When an iterative approach is restarted, the current approximation space is discarded at each restart. Therefore, a well-known drawback of GMRES(m) is that orthogonality to previously generated subspaces is not preserved at each restart. In fact, GMRES(m) can stall as a result. Stalling means that there is no decrease in the residual norm at the end of a restart cycle. Restarting also negates the potential for superlinear convergence behavior [29].

This paper is organized as follows. In section 2, we describe some existing modifications to GMRES(m) aimed at accelerating convergence or overcoming stalling. We introduce our new acceleration technique in section 3. We present numerical results and discuss the convergence behavior of the new algorithm in section 4. We close with concluding remarks in section 5.

2. Background. In this section, we briefly describe some existing modifications to the standard GMRES algorithm. These modifications all have the common goal of enhancing the robustness of restarted GMRES. Two primary categories of modification include hybrid iterative methods and acceleration techniques. Hybrid iterative methods combine standard iterative methods in a variety of ways to reduce the number of required vector operations. Many of these methods are essentially modifications to GMRES(m) aimed at improving its performance. Nachtigal, Reichel, and Trefethen provide a general overview of this class of iterative methods in [21]. Our work falls into the category of acceleration techniques. These techniques attempt to mimic the convergence of full GMRES more closely or to accelerate the convergence of GMRES(m) by retaining some of the information that is typically discarded at the time of restart. In [11], Eiermann, Ernst, and Schneider present a thorough overview and analysis of the most common acceleration techniques.

Augmented methods are a class of acceleration techniques. In particular, these methods seek to avoid stalling by improving information in GMRES at the time of the restart. Typically a (nearly) A-invariant subspace is appended to the Krylov approximation space, resulting in an "augmented Krylov subspace" [5]. The invariant subspace of A associated with the smallest eigenvalues is commonly used, as those eigenvalues are thought to hinder convergence the most. Algorithms that include spectral information at the restart to overcome stalling are presented by Morgan in [18], [19], and [20] (GMRES-E, GMRES-IR, and GMRES-DR, respectively) and are further discussed in [5] and [26]. These augmentation techniques are more suitable for some types of problems than others. They can be very effective when convergence is being hampered by a few eigenvalues [18]. However, they may have little effect on highly nonnormal problems [5], or solving the eigenvalue problem may be too costly for the

technique to be beneficial [18]. Of interest to us is the simple framework provided for appending (non-Krylov) vectors to the approximation space.

Another class of acceleration techniques is based on the fact that ideally the approximation space should contain the correction c such that $x = x_0 + c$ is the exact solution to the problem [11]. The nested Krylov subspace method GMRESR (GMRES Recursive) [30] is one such technique. In GMRESR, the outer generalized conjugate residual method (GCR) [12] invokes an inner iterative method (like GMRES) at each step i to approximate the solution to $Ac = r_i$, where r_i is the current residual at step i. The approximate solution to $Ac = r_i$ then becomes the next direction for the outer approximation space. The goal of this method is to obtain similar convergence to that of full GMRES with less computational cost under certain conditions. Note that the method FGMRES (Flexible GMRES) [24] can also be viewed as a method that approximates solutions to similar residual equations at each step. In fact, both FGMRES and GCR provide a framework for using a GMRES-like method with *any* approximation space.

Another related acceleration technique is GCRO (GCR with inner orthogonalization) [7]. The aim of this method is twofold: to compensate for the information that is lost due to restarting as well as to overcome some of the stalling problems that GMRESR can experience in the inner iteration. GCRO is a modification to GMRESR such that the inner iterative method maintains $A^T A$-orthogonality to the outer approximation space. Thus, the approximation from the inner iteration at step i takes into account both the inner and outer approximation spaces. See also [9] for more details on preserving orthogonality in the inner iteration of a nested Krylov subspace method. In most cases, both GCRO and GMRESR must be truncated to keep storage costs reasonable. Therefore, a truncated version of GCRO, the method GCROT (GCRO Truncated), is subsequently described in [8]. GCROT attempts to determine which subspace of the outer approximation space should be retained for the best convergence of future iterations as well as if any portion of the inner Krylov subspace should also be kept.

As Fokkema, Sleijpen, and van der Vorst point out in [15], "the distinction between preconditioning and acceleration is not a clear one." These acceleration techniques (GMRESR, GCRO, and FGMRES) can also be viewed as methods with variable preconditioning (allowing the preconditioner to change with each iteration step). We show that our new method can also be viewed in this way.

3. A new algorithm: LGMRES. In this section, we describe a new method for accelerating GMRES(m). We begin with observations about the convergence behavior of GMRES(m) that lead us to the new technique. We then present the new algorithm LGMRES (Loose GMRES), discuss some of its properties, and compare LGMRES to closely related existing acceleration techniques.

3.1. Motivation. Consider restarted GMRES(m) when solving problem (1). In this discussion, we refer to the group of m iterations between successive restarts as a cycle. The restart number is denoted with a subscript: r_i is the residual after i cycles or $m \times i$ iterations. The residual at the end of cycle $i+1$ is a polynomial in A times the residual from the previous cycle, $r_{i+1} = p_{i+1}^m(A) r_i$, where $p_{i+1}^m(A)$ is the degree m residual polynomial. During each restart cycle (i), GMRES(m) finds $x_{i+1} \in x_i + K_m(A, r_i)$ such that $r_{i+1} \perp AK_m(A, r_i)$ (see, e.g., [25]).

As previously mentioned, GMRES(m) does not maintain orthogonality between approximation spaces generated at successive restarts. As a result, slow convergence or even stalling can occur. In the case of slow convergence, we have observed a pattern

TABLE 1

Results for GMRES(30). Problem size, iterations required for $\|r_i\|_2/\|r_0\|_2 \leq 10^{-9}$, median skip angle, and median sequential angle are listed for each problem.

Problem	Size (n)	Iterations	Median seq. angle $\angle(r_i, r_{i-1})$	Median skip angle $\angle(r_{i+1}, r_{i-1})$
add20	2395	1002	51.3	5.4
orsirr_1	1030	6659	23.0	6.9
orsreg_1	2205	888	59.3	8.4
sherman_1	1000	3688	27.5	.2

in GMRES(m) where the residual vectors point in nearly the same direction at the end of every other restart cycle. In other words, the angle between r_{i+1} and r_{i-1} is small and $r_{i+1} \approx \alpha\, r_{i-1}$. We refer to the angles between *every other* residual vector as *skip* angles, e.g., $\angle(r_{i+1}, r_{i-1})$, and the angles between consecutive restart cycles as *sequential* angles.

For many problems, we find that skip angles are relatively small even when the sequential angles are a reasonable size (i.e., stalling is not occurring). For example, Table 1 gives results for GMRES(30) on several problems available from the Matrix Market Collection [23]. The number of iterations required for convergence ($\|r_i\|_2/\|r_0\|_2 \leq 10^{-9}$) as well as the median sequential and median skip angle values are listed. GMRES(30) is not stalling for these four problems. However, the low skip angle values appear to indicate that faster convergence should be possible if some degree of orthogonality to previous approximation spaces were maintained, a goal embraced by several acceleration techniques described in section 2. In our experience, this type of alternating pattern is most pronounced (most "exact") for symmetric matrices, but it is noticeable for many nonsymmetric matrices as well.

There no mechanism in GMRES(m) to prevent this alternating phenomenon because it is simply a symptom of the lack of orthogonality between the approximation space generated during a particular restart cycle of GMRES(m) and the approximation spaces from previous cycles. However, only for the special case when the restart parameter is one less than the matrix order can we show that alternating must occur for both symmetric and skew-symmetric problems. Consider the following lemma.

LEMMA 1 (equivalent constraints). *When $A \in \mathbb{R}^{n\times n}$ is symmetric or skew-symmetric, and w and y are arbitrary real vectors of length n, the requirement that $w \perp AK_m(A, y)$ is equivalent to the requirement that $w \perp A^T K_m(A^T, y)$.*

With this easily proved lemma, the following theorem is straightforward.

THEOREM 2 (alternating residuals). *When $A \in \mathbb{R}^{n\times n}$ is symmetric or skew-symmetric and the restart parameter is one less than the matrix order ($m = n - 1$), GMRES(m) produces a sequence of residual vectors at the end of each restart cycle such that $r_{i+2} = \alpha r_i$, $|\alpha| \leq 1$.*

Proof. During restart cycle i,

$$r_i \perp AK_m(A, r_{i-1}) \Rightarrow r_{i-1} \perp A^T K_m(A^T, r_i).$$

From Lemma 1,

$$r_{i-1} \perp A^T K_m(A^T, r_i) \Rightarrow r_{i-1} \perp AK_m(A, r_i). \tag{3}$$

Let $W_m \equiv [w_1\ w_2\ \ldots\ w_m]$ be an orthonormal basis for $AK_m(A, r_i)$. There exists a w_n such that $W_n = [W_m\ w_n]$ is an orthonormal basis for $\mathbb{R}^n$. From (3), $r_{i-1} = \alpha w_n$,

where α is some scalar. During restart cycle $i+1$,

$$r_{i+1} \perp AK_m(A, r_i) \Rightarrow r_{i+1} = \beta w_n,$$

where β is some scalar. Therefore, $r_{i+1} = \frac{\beta}{\alpha} r_{i-1}$, and $|\frac{\beta}{\alpha}| \leq 1$ because the GMRES(m) residual norm is nonincreasing. □

3.2. Idea and implementation. The motivation for the new algorithm, LGMRES, came from a desire to prevent the alternating behavior observed for GMRES(m) which results in repetitive information in successive restart cycles. In addition, we wanted a method for which the idea and implementation easily lent themselves to a block method for solving a single right-hand side system (see, e.g., [2]). Therefore, the new algorithm is a combination of ideas from several existing acceleration techniques described in section 2: GMRES-E, GMRESR, and GCRO. In short, LGMRES utilizes the simple framework of Morgan's GMRES-E method [18] for appending vectors to the standard Krylov space in a manner that allows for the extension to a block method as in [5], for example. GMRESR [30] and GCRO [7], on the other hand, provide ideas for choosing appropriate vectors to append to the standard Krylov approximation space. The algorithmic components from these existing techniques are combined in a manner that results in a new acceleration technique with both a simple implementation and the ability to prevent the previously described alternating behavior.

To prevent alternating, LGMRES mimics GMRESR's technique of including approximations to the error in the current approximation space. Suppose that $\hat{x}$ is the true solution to problem (1). The error after the ith restart cycle of GMRES(m) is denoted by e_i, where

$$e_i \equiv \hat{x} - x_i. \tag{4}$$

As explicitly pointed out in [11] and noted in section 2, if our approximation space contains the exact correction e_i such that $\hat{x} = x_i + e_i$, then we have solved the problem. We define

$$z_i \equiv x_i - x_{i-1} \tag{5}$$

as the approximation to the error after the ith GMRES(m) restart cycle, and $z_j \equiv 0$ for $j < 1$. This error approximation vector serves as our choice of vector with which to augment our next approximation space $K_m(A, r_i)$. Note that $z_i \in K_m(A, r_{i-1})$. Therefore, this error approximation z_i in some sense represents the space $K_m(A, r_{i-1})$ generated in the previous cycle and subsequently discarded and is a natural choice of vector with which to augment our next approximation space $K_m(A, r_i)$.

We denote our new restarted augmented GMRES algorithm by LGMRES(m, k). LGMRES(m, k) augments the standard Krylov approximation space with k previous approximations to the error. Therefore, at the end of restart cycle $i+1$, LGMRES(m, k) finds an approximate solution to (1) in the following way:

$$x_{i+1} = x_i + q_{i+1}^{m-1}(A) r_i + \sum_{j=i-k+1}^{i} \alpha_{ij} z_j, \tag{6}$$

where polynomial q_{i+1}^{m-1} and α_{ij} are chosen such that $\|r_{i+1}\|_2$ is minimized. Note that $k = 0$ corresponds to standard GMRES(m).

The implementation of LGMRES(m, k) is quite similar to that of Morgan's GMRES with eigenvectors (GMRES-E) method described in [18] and requires minimal changes to the standard GMRES(m) implementation. At each restart cycle (i) we

1. $r_i = b - Ax_i$, $\beta = \|r_i\|_2$, $v_1 = r_i/\beta$, $s = m + k$
2. for $j = 1 : s$
3. $\quad u = \begin{cases} Av_j & \text{if } j \le m, \\ Az_{i-(j-m-1)} & \text{otherwise} \end{cases}$
4. $\quad$ for $l = 1 : j$
5. $\quad\quad h_{l,j} = \langle u, v_l \rangle$
6. $\quad\quad u = u - h_{l,j} v_l$
7. $\quad$ end
8. $\quad h_{j+1,j} = \|u\|_2$, $v_{j+1} = u/h_{j+1,j}$
9. end
10. $V_{s+1} = [v_1, \ldots, v_m, \ldots, v_{m+k+1}]$, $W_s = [v_1, \ldots, v_m, z_i, \ldots, z_{i-k+1}]$, $H_s = \{h_{l,j}\}_{1 \le l \le j+1; 1 \le j \le s}$
11. find y_s s.t. $\|\beta e_1 - H_s y_s\|_2$ is minimized
12. $z_{i+1} = W_s y_s$ (also $Az_{i+1} = V_{s+1} H_s y_s$)
13. $x_{i+1} = x_i + z_{i+1}$

FIG. 1. *LGMRES(m, k) for restart cycle i.*

generate the Krylov subspace $K_m(A, r_i)$ and augment it with the k most recent error approximations z_j, $j = (i - k + 1) : i$. The augmented approximation space $\mathcal{M} = K_m(A, r_i) \cup span\{z_j\}_{j=(i-k+1):i}$ has dimension $s \equiv m + k$. We then find the approximate solution from $\mathcal{M}$ whose corresponding residual is a minimum in the Euclidean norm.

One restart cycle (i) of the LGMRES(m, k) algorithm is given in Figure 1. Note that V_{s+1} is the $n \times (s + 1)$ orthonormal matrix whose first $m + 1$ columns are the Arnoldi vectors and last s columns result from orthogonalizing the k error approximation vectors (z_j, $j = (i - k + 1) : i$) against the previous columns of Arnoldi vectors. W_s is the $n \times s$ matrix whose first m columns are equal to the first m columns of V_{s+1} and whose last k columns of W are the k error approximation vectors (typically normalized so that all columns are of unit length). Then the relationship

$$AW_s = V_{s+1} H_s \tag{7}$$

holds for LGMRES(m, k), where H_s denotes an $(s+1) \times s$ Hessenberg matrix whose elements $h_{l,j}$ are defined in the algorithm in Figure 1. This relationship is analogous to equations (11) in [18] and (3) in [27].

When implementing LGMRES(m, k), only m matrix-vector multiplies are required per restart cycle, irrespective of the value of k, provided that we form both z_i and Az_i at the end of cycle i as is done in the algorithm given in Figure 1. Note that forming Az_i does not require an explicit multiplication by A and that at most k pairs of z_j and Az_j need to be stored. Typically the number of vectors appended, k, is much smaller than the restart parameter m (discussed in section 4). The algorithm requires storage for the following vectors of length n: $m + k + 1$ orthogonal basis vectors ($v_1, v_2, \ldots v_{m+k+1}$), k pairs of z_j and Az_j, the approximate solution, and the right-hand side. Therefore, this implementation of LGMRES(m, k) requires storage for $m + 3k + 3$ vectors of length n and m matrix-vector multiplies per restart cycle. Recall that standard GMRES($m + k$) requires storage for $m + k + 3$ vectors of length n and $m + k$ matrix-vector multiplies per restart cycle (see, e.g., [25]). Also, LGMRES(m, k) and GMRES($m + k$) require equivalent numbers of inner products and vector updates. One could reduce the storage requirement for LGMRES(m, k) by

recomputing Az_i in each cycle. The storage requirement for vectors of length n would then drop to $m+2k+3$, but the number of matrix-vector multiplies required per cycle would increase to $m+k$. We prefer the former method (as given in Figure 1) because it reduces the number of matrix-vector multiplies and is therefore generally faster.

Note that only i error approximations are available at the beginning of restart cycles with $i < k$ because $z_j = 0$ when $j < 1$. Therefore, we recommend using additional Arnoldi vectors instead of z_j when $j < 1$ so that the approximation space is of dimension $m + k$ for each cycle. In other words, the first cycle ($i = 0$) of LGMRES(m, k) is equivalent to the first cycle of GMRES($m + k$).

LGMRES(m, k) can be preconditioned in a straightforward manner. Let M^{-1} denote the preconditioner. For left preconditioning, we simply precondition the initial residual in line 1 of the algorithm in Figure 1 ($r_i = M^{-1}b - M^{-1}Ax_i$). Then we replace A with $M^{-1}A$ everywhere in lines 3 and 12. For right preconditioning, the required modifications are more subtle. To include previous approximations to the error in the approximation space, we must now append $\hat{z}_j \equiv M(x_j - x_{j-1}) = Mz_j$ instead of z_j to the standard Krylov subspace (no matrix-vector products with M are explicitly computed). Therefore, we replace A with AM^{-1} everywhere in lines 3 and 12 and z with $\hat{z}$ everywhere in lines 3, 10, and 12. While no explicit change is required for line 13 as given in Figure 1, note that, with right preconditioning, line 13 is equivalent to $x_{i+1} = x_i + M^{-1}\hat{z}_{i+1}$.

3.3. Properties. In this section, we first address the similarity between LGMRES and a full conjugate gradient (FCG) method with polynomial preconditioning. We then discuss skip angles and sequential angles for both GMRES(m) and LGMRES(m, k).

We consider the "full" (i.e., nontruncated) version of LGMRES, denoted by LGMRES(m), in which *all* previous error approximations are kept (i.e., $k = i$):

$$z_{i+1} = q_{i+1}^{m-1}(A)r_i + \sum_{j=1}^{i} \alpha_{ij} z_j. \tag{8}$$

In this form, the resemblance of LGMRES(m) to a minimal residual FCG method that minimizes $\|e_i\|_{A^TA}$ at each step, such as ORTHOMIN, is readily apparent (see, e.g., [25] or [1]). In (8), the GMRES(m) iteration polynomial ($q_{i+1}^{m-1}(A)$) corresponds to a polynomial preconditioner. Notice, however, that LGMRES effectively changes the preconditioner with each iteration i, whereas preconditioned FCG typically uses a constant preconditioner (not dependent on i). Vectors z_j in (8) correspond to conjugate gradient direction vectors in that they are also A^TA-orthogonal, as is shown below. Therefore, we can categorize the LGMRES(m, k) method as a truncated polynomial-preconditioned FCG method.

THEOREM 3 (orthogonality of the error approximations). *The error approximation vectors $z_j \equiv x_j - x_{j-1}$ with which we augment the Krylov space in full LGMRES* (8) *or truncated LGMRES* (6) *are A^TA-orthogonal.*

Proof. First, we define subspaces $\mathcal{M}_{i+1}$ and $\mathcal{M}_i$ as

$$\mathcal{M}_{i+1} \equiv K_m(A, r_i) \cup span\{z_j\}_{j=(i-k+1):i}$$

and

$$\mathcal{M}_i \equiv K_m(A, r_{i-1}) \cup span\{z_j\}_{j=(i-k):(i-1)},$$

respectively. By construction,

$$r_i \perp A\mathcal{M}_i \qquad \text{and} \qquad r_{i+1} \perp A\mathcal{M}_{i+1}.$$

From (5),

$$r_i - r_{i+1} = Az_{i+1}.$$

Therefore,

$$Az_{i+1} \perp A(\mathcal{M}_i \cap \mathcal{M}_{i+1}).$$

Because $\{z_j\}_{j=(i-k+1):i} \subset \mathcal{M}_i \cap \mathcal{M}_{i+1}$,

$$z_{i+1} \perp_{A^TA} \{z_j\}_{j=(i-k+1):i}. \qquad \square$$

Although full LGMRES is interesting from a theoretical point of view, it is not a practical algorithm. Storing all past values of z_j ($j = 1 : i$) requires an increasing amount of storage at each restart. As with GMRESR and GCRO, truncating is necessary. Therefore, in practice, we use truncated LGMRES(m, k) as given in (6) with some $k < i$. In section 4, we show that optimal values for k are typically very small: $k \leq 3$. Furthermore, note that the A^TA-orthogonality of the error approximation vectors shown in Theorem 3 is not exploited in the implementation of LGMRES described in the previous section. In fact, a total of k vector products and updates per restart cycle in the algorithm given in Figure 1 are extraneous due to a zero vector product in line 5. However, for small k, the benefit of modifying the LGMRES(m, k) implementation to exploit this orthogonality is negligible.

Now we compare the skip and sequential angles for GMRES(m) and LGMRES(m, k). For standard restarted GMRES, the angle between two residuals from consecutive restart cycles (i.e., the sequential angle) can be expressed in terms of a ratio of their residual norms. The following result is mathematically equivalent to a result first given by Simoncini as Proposition 4.1 in [28], but here we present it in a simplified form with a more straightforward and concise proof.

THEOREM 4 (GMRES(m) sequential angles). *Let r_{i+1} and r_i be the residuals from GMRES restart cycles $i + 1$ and i, respectively. Then the angle between these residuals is given by*

$$\cos\angle(r_{i+1}, r_i) = \frac{\|r_{i+1}\|_2}{\|r_i\|_2}. \tag{9}$$

Proof. In restart cycle $i + 1$ of GMRES(m), $x_{i+1} = x_i + \delta_{i+1}$, where $\delta_{i+1} \in K_m(A, r_i)$. Therefore, the corresponding residual is

$$r_{i+1} = r_i - A\delta_{i+1}.$$

By construction,

$$\langle r_{i+1},\, A\delta_{i+1}\rangle = 0 \quad \Rightarrow \quad \langle r_{i+1},\, r_i\rangle = \langle r_{i+1},\, r_{i+1}\rangle = \|r_{i+1}\|_2^2. \tag{10}$$

The above, combined with the definition of cosine, completes the proof. $\square$

The above indicates that, for GMRES(m), the convergence rate correlates to the size of the angles between consecutive residual vectors. If consecutive residual vectors are nearly orthogonal to each other, then convergence is fast. (If we find an r_{i+1} such that $r_{i+1} \perp r_i$, then we have found the exact solution.) Note that this result also holds for LGMRES. We refer to the related work in [10] for a more general discussion on how the angles between approximation and residual spaces define convergence for

Krylov methods. Note that Theorem 4 above is also a special case of the more general result in (3.9) in [10]. Now we consider the angle between every other residual (i.e., the skip angle).

THEOREM 5 (GMRES(m) skip angles). *Let r_{i+1} and r_{i-1} be the residuals from GMRES restart cycles $i+1$ and $i-1$, respectively. Then the angle between these residuals is given by*

$$\cos\angle(r_{i+1}, r_{i-1}) = \frac{\|r_{i+1}\|_2}{\|r_{i-1}\|_2} - \frac{\langle A\delta_{i+1},\ A\delta_i\rangle}{\|r_{i+1}\|_2\|r_{i-1}\|_2},$$

where $r_{i+1} = r_i - A\delta_{i+1}$ and $r_i = r_{i-1} - A\delta_i$.

Proof. As in the previous proof, it is easily shown that

$$\langle r_{i+1},\ r_{i-1}\rangle = \langle r_{i+1},\ r_{i+1}\rangle - \langle A\delta_{i+1},\ A\delta_i\rangle. \tag{11}$$

The proof follows directly from (11). ∎

In terms of describing convergence, the above result is not immediately helpful. However, we will discuss a few of its implications after giving a corresponding result for LGMRES. Recall from section 3.2 that LGMRES(m, k) appends k previous approximations to the error to the current Krylov approximation space. Therefore, if $k \geq 1$, then $r_{i+1} \perp AK_m(A, r_i)$ and $r_{i+1} \perp Az_i$ at the end of restart cycle $i+1$. Since $Az_i = r_{i-1} - r_i$,

$$\langle r_{i+1},\ r_{i-1} - r_i\rangle = 0 \tag{12}$$

after $i+1$ LGMRES cycles, and we can prove the following theorem.

THEOREM 6 (LGMRES: Every other residual vector). *Let r_{i+1} and r_{i-1} be the residuals from LGMRES restart cycles $i+1$ and $i-1$, respectively. Then the angle between these residuals is given by*

$$\cos\angle(r_{i+1}, r_{i-1}) = \frac{\|r_{i+1}\|_2}{\|r_{i-1}\|_2}.$$

Proof. This theorem directly follows from Theorems 5 and 3 (noting the correlation between δ_i in GMRES(m) and z_i in LGMRES). Alternatively, from (12) and (10),

$$\langle r_{i+1},\ r_{i-1}\rangle = \langle r_{i+1},\ r_i\rangle = \langle r_{i+1},\ r_{i+1}\rangle.$$

The proof follows directly from the above relation. ∎

This result indicates that, for LGMRES, the progress of the iteration also correlates with the skip angles. Therefore, fast convergence implies large skip angles. More generally, for any $0 \leq j \leq k$ and $i \geq k$, we can show for LGMRES(m, k) that

$$\cos\angle(r_{i+1}, r_{i-j}) = \frac{\|r_{i+1}\|_2}{\|r_{i-j}\|_2}.$$

When a problem exhibits signs of alternating residuals with GMRES(m), then the angle between r_{i-1} and r_{i+1} is small. In this case, since $A\delta_{i+1} = r_i - r_{i+1}$ and $A\delta_i = r_{i-1} - r_i$, then the term $\langle A\delta_{i+1},\ A\delta_i\rangle$ in Theorem 5 is negative. We have observed this result in our experiments, and it can be seen pictorially in Figure 2. Since LGMRES appends a previous error approximation to the approximation space

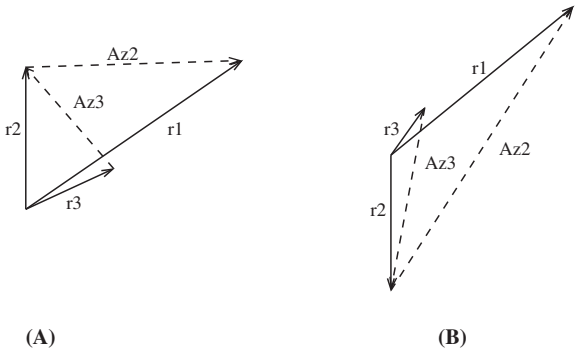

FIG. 2. *Two cases with alternating residual vectors: r_1 and r_3 point in nearly the same direction.* $\langle Az_3,\ Az_2\rangle < 0$ *in both* (A) *and* (B).

during cycle $i+1$, the term $\langle A\delta_i,\ A\delta_{i-1}\rangle$ is equal to zero by construction. We show in section 4.1 that this LGMRES augmenting scheme tends to increase the skip angle over that of GMRES(m) and prevents the alternating behavior often observed in restarted GMRES.

We also investigated adaptive versions of LGMRES that determine whether or not to augment during each restart cycle. One often effective adaptive version is based on the above observation that the term $\langle A\delta_{i+1},\ A\delta_i\rangle$ in Theorem 5 is generally negative when alternating occurs. In particular, after m standard Arnoldi iterations in restart cycle $i+1$, we form the current residual $\hat{r}_{i+1}$. In the $k=1$ case, the decision is made to augment during cycle $i+1$ if $\langle \hat{r}_{i+1},\ A\delta_i\rangle > 0$. Referring back to Theorem 5, note that $\langle r_{i+1},\ A\delta_i\rangle = -\langle A\delta_{i+1},\ A\delta_i\rangle$. Results for this adaptive version of LGMRES are discussed in section 4.1.

3.4. Comparison to existing methods. As previously stated, LGMRES(m, k) acts as an accelerator for GMRES(m). The algorithm is not designed to overcome stalling as the error approximation vectors, z_j, are zero when the residual norm does not decrease within a cycle. Thus, while the LGMRES implementation mimics that of Morgan's GMRES-E [18], we do not compare the two algorithms, as GMRES-E is most effective for problems that stall due to the effects of a few eigenvalues. However, as noted at the beginning of this section, the general idea of LGMRES is very similar to that of GCRO [7]; both methods look for a minimum residual solution in the approximation space consisting of previous approximations to the error as well as a Krylov space built on the current residual. The algorithms are not mathematically equivalent, and we briefly explain their similarities and differences in this section. First, we discuss the GMRESR [30] method, of which GCRO is a modification. Then the theoretical differences between (nontruncated) GCRO and full LGMRES are briefly described, followed by a comparison of the two truncated algorithms GCROT and LGMRES(m, k).

The nested Krylov subspace methods GMRESR and GCRO consist of an outer GCR method that invokes an inner GMRES method at each iteration to find an approximation to the error. Generally a fixed number of GMRES steps are taken at each inner iteration, say, m. GCR is a minimum residual method that maintains two bases, U_i and $C_i = AU_i$, where $C_i^T C_i = I_i$. Typically U_i is an $A^T A$-orthogonal basis for the Krylov space $K_i(A, r_0)$. However, the implementation of GCR is such that U_i can actually contain any vectors (i.e., range$(U_i) \neq K_i(A, r_0)$) [7]. In particular,

in both the GMRESR and GCRO methods, range(U_i) contains all of the previous approximations to the error from the inner GMRES method.

GMRESR is essentially performing two separate minimizations: one over the inner GMRES approximation space to find a new error approximation and one over the outer approximation space (consisting of the new error approximation and all previous error approximations) to update the current global approximate solution. The clever improvement of GCRO over GMRESR is that the GCRO minimization in the inner iteration takes into account the outer approximation space. In other words, the two methods are not mathematically equivalent, and GCRO solves the following minimization problem at each inner iteration:

$$\min\|b - Ax_{i+1}\|_2 \quad \text{s.t.} \quad x_{i+1} \in \text{range}(U_i) \oplus \text{range}(W_m), \tag{13}$$

where W_m is an orthogonal basis for $K_m(A_C, r_i)$ generated by the inner GMRES method and $A_C \equiv (I - C_i C_i^T)A$. The Krylov space $K_m(A_C, r_i)$ is a result of GCRO maintaining orthogonality against C_i from the beginning of the Arnoldi iteration, and W_{m+1} satisfies $W_{m+1} \perp \text{range}(C_i)$. Thus, when r_i is projected onto AW_m resulting in new residual r_{i+1}, that new residual is also orthogonal to range(C_i) as desired. The solution to the global minimization problem of (13) is then found.

Similarly to GCRO, full LGMRES finds a minimum residual solution in an approximation space consisting of all previous error approximations (z_j) together with a Krylov space built off the current residual:

$$\min\|b - Ax_{i+1}\|_2 \quad \text{s.t.} \quad x_{i+1} \in \text{range}(Z_i) \oplus \text{range}(V_m),$$

where V_m is an orthogonal basis for $K_m(A, r_i)$ and $Z_i \equiv [z_1 \dots z_i]$. In the case of LGMRES, the Arnoldi iteration does not maintain orthogonality against the previous error approximations. Instead, the error approximations are simply appended onto the generated Krylov subspace, which leads to a greater number of orthogonalizations than for GCRO if k is large.

The difference in generation of the Krylov subspaces is a subtle difference between GCRO and LGMRES. Matrices A_C and A do not generate equivalent residual spaces ($A_C K_m(A_C, r_i)$ and $AK_m(A, r_i)$, respectively). See [16] for more on matrices that generate equivalent Krylov residual spaces. Therefore, the residual projected onto these spaces is not the same unless the unlikely situation occurs where range(V_m) $\perp$ range(C_i). Finally we remark that as with GCRO, LGMRES is also not equivalent to GMRESR since the error approximation vectors are determined by a single minimization over the global space consisting of previous approximations to the error as well as a Krylov space built on the current residual.

GCROT [8] is a more practical truncated version of GCRO. GCROT truncates the outer approximation space by examining angles between subspaces and determining which subspaces (not vectors) are important for convergence. It is assumed that if a subspace was important for past convergence, then it will be important for future convergence and should be retained. Similarly, vectors from the inner GMRES iteration may also be kept. The implementation of GCROT(m, k_{max}, k_{new}, s, p_1, p_2) requires specification of six different parameters that affect the truncation.

LGMRES(m, k), on the other hand, is truncated in a more obvious manner, retaining only the most recent k error approximation vectors. For ORTHOMIN, it has been observed that truncating the recursion such that only one or two previous direction vectors are retained is quite effective when A is nearly symmetric [31]. Therefore, we attribute the effectiveness of the LGMRES method's naive truncation

strategy, particularly when A is nearly symmetric in some sense, to the relation of LGMRES to the ORTHOMIN algorithm, which was mentioned in section 3.3. In fact, in our experiments we find that LGMRES performs best when k is much smaller than m (typically $k \leq 3$), whereas GCROT often prefers $k > m$. Additionally, as previously mentioned, the LGMRES(m, k) truncation strategy results in a more straightforward implementation that lends itself to a block method.

4. Experimental results. We demonstrate the potential of LGMRES by presenting experimental results from a variety of problems using implementations of LGMRES in both MATLAB and a locally modified version of PETSc (Argonne National Laboratory's Portable, Extensible Toolkit for Scientific Computation) [3, 4]. We tested problems from various sources, including the Matrix Market Collection [23] and the University of Florida Sparse Matrix Collection [6]. In sections 4.1 and 4.2, we compare MATLAB implementations of LGMRES(m, k), GMRES(m), GCRO [7], and GCROT [8] for problems without preconditioning. In section 4.3, we demonstrate the usefulness of LGMRES for larger problems with preconditioning with a PETSc implementation of LGMRES.

4.1. Comparison to GMRES(m). In this section, we demonstrate that LGMRES can significantly accelerate the convergence of restarted GMRES. To compare the performance of LGMRES(m, k) and GMRES(m), we implemented each in MATLAB. Our purpose with these implementations is to gauge the acceleration potential of LGMRES as well as its range of applicability on a variety of problems. Therefore, in this section and section 4.2, we do not use preconditioning for the MATLAB tests, allowing iteration counts to be large. A zero initial guess is used for all problems.

We look at a test set of 18 problems, 15 from the Matrix Market Collection and 3 convection-diffusion (CD) problems. The Matrix Market problems include the following: add20, orsreg_1, orsirr_1, cdde1, pde900, sherman1, sherman4, rdbl1250, cavity05, nos3, watt_2, fs_760_1, e05r0000, steam2, and cavity10. If a right-hand side is not provided, we generate a random right-hand side. The three CD problems are taken from [18] and are variations of the PDE $u_{xx} + u_{yy} + Du_x = -(41)^2$ with increasing degree of nonsymmetry: $D = 1$, $D = 41$, and $D = 41^2$, which we refer to as morgan_1, morgan_41, and morgan_1681, respectively. These PDEs are discretized by central finite differences on the unit square with zero boundary conditions and step-size $h = 1/41$. We stop the iteration when the relative residual norm is less than the convergence tolerance ζ, i.e., when $\|r_i\|_2/\|r_0\|_2 \leq \zeta$. We use $\zeta = 10^{-5}$ for all problems in this comparison. Several restart parameters are chosen for each problem, resulting in a total of 53 test cases. In particular, for the first 11 Matrix Market problems (in the preceding list) and the three CD problems, we use $m = 10$, 20, and 30. We use $m = 10$, 20 for problem fs_760_1 and $m = 20$, 30, and 40 for the last three Matrix Market problems.

For each of these 53 test cases, we compare the performances of GMRES(m) and LGMRES with equal-sized approximation spaces. Figure 3 shows the number of matrix-vector multiplies required for convergence for GMRES(m) and LGMRES($m - k$, k) with $k = 1 : 5$. In both the top and bottom plots, the y-axis is the number of matrix-vector multiplies required for convergence by GMRES(m) divided by the number required by LGMRES($m - k$, k). Note that the log of this ratio is plotted on the y-axis of Figure 3. The x-axis corresponds to the 15 Matrix Market problems followed by the three CD problems in the order given in the previous paragraph, and results for the same problem with increasing m are adjacent. For example, test case 1 corresponds to problem add20 with $m = 10$, for which GMRES(m) re-

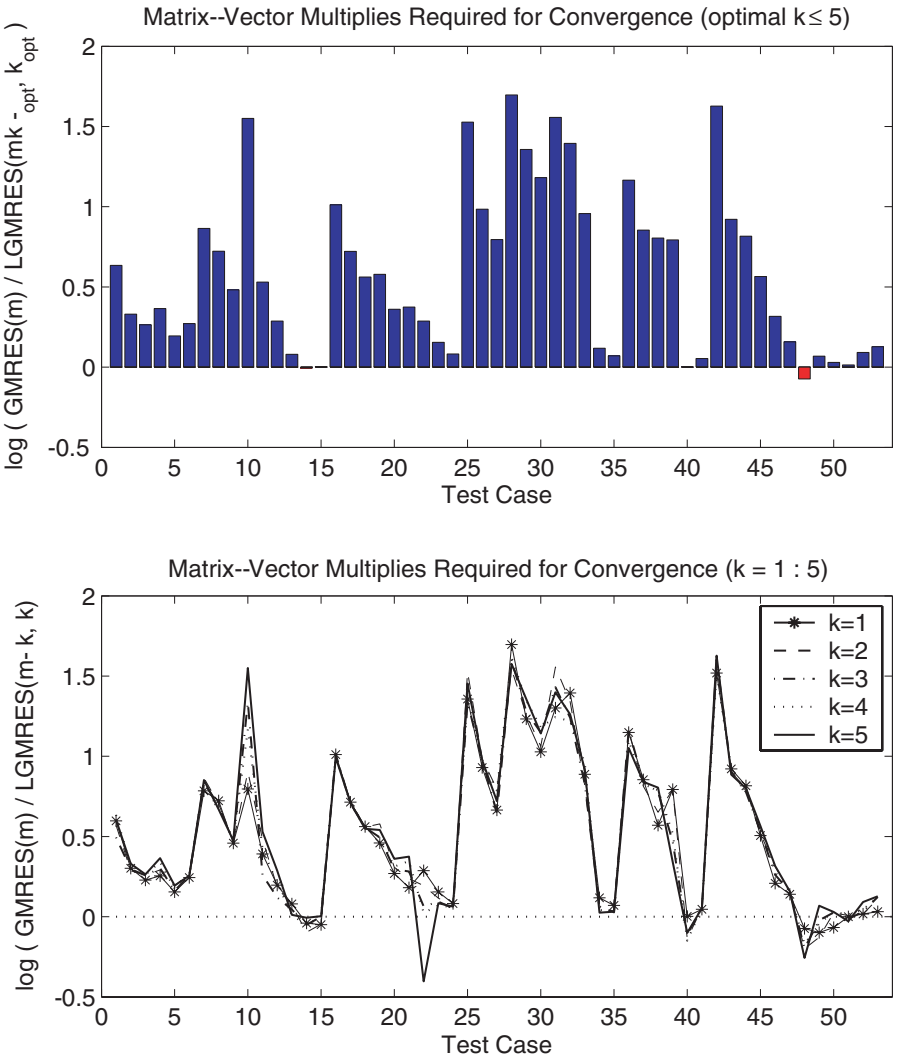

FIG. 3. *A comparison of the number of matrix-vector multiplies required for convergence by GMRES(m) and LGMRES(m − k, k) for* 53 *test cases. The top panel compares GMRES(m) to the "best" LGMRES(m, k). The bottom panel displays results for LGMRES(m − k, k) versus GMRES(m) for five different values of* k $(k = 1 : 5)$.

quires approximately four times as many matrix-vector multiplies as does LGMRES $(m - k, k)$.

In the top panel of Figure 3, the result of the "best" LGMRES$(m - k, k)$ for $k = 1 : 5$ is compared to GMRES(m). The bars extending above the x-axis favor LGMRES$(m - k, k)$ (51 cases)—in these cases GMRES(m) requires more matrix-vector multiplies than does LGMRES$(m - k, k)$. The bars below the x-axis favor GMRES(m) (two cases: pde900 with $m = 20$ and morgan_41 with $m = 10$). Ratios of improvement (as opposed to iteration counts) are given to demonstrate the potential improvement with LGMRES, though we note that the number of iterations required by LGMRES$(m - k, k)$ is less than n (where n is the matrix order) in 46 of the 53

TABLE 2

Results for LGMRES(29, 1). Problem size, iterations required for $\|r_i\|_2/\|r_0\|_2 \leq 10^{-9}$, median skip angle, and median sequential angle are listed for each problem.

Problem	Size (n)	Iterations (matrix-vector multiplies)	Median seq. angle $\angle(r_i, r_{i-1})$	Median skip angle $\angle(r_{i+1}, r_{i-1})$
add20	2395	606 (587)	63.0	79.0
orsirr_1	1030	2190 (2118)	41.0	55.4
orsreg_1	2205	515 (499)	72.2	84.6
sherman_1	1000	757 (733)	61.7	76.4

test cases. In the remaining seven cases (steam2 with $m = 20$ and both e05r0000 and orsirr_1 with $m = 10$, 20, and 30), the number of iterations is less than $2.25n$. The number of iterations required by GMRES(m), on the other hand, is much greater than n for a number of these test cases, as reflected by several large ratios in the top panel of Figure 3.

The plot in the bottom panel of Figure 3 shows the variance in results for LGMRES($m-k$, k) with $k = 1 : 5$. Generally $k \leq 3$ is best for LGMRES($m-k$, k), and in our experiments, returns are diminishing for larger k, especially when m is small.

Furthermore, as with the majority of these test problems in Figure 3, we typically observe that the percentage improvement of LGMRES over GMRES decreases with increasing m. This trend is likely related to smaller values of m resulting in larger iteration counts and more noticeable alternating behavior.

Experimentally, we observe that LGMRES nearly always has a larger median skip angle than does GMRES(m). For example, in Table 2 we list the LGMRES(29, 1) results for the same four problems for which GMRES(30) results were provided in Table 1 in section 3.1. Again, the number of iterations required for convergence ($\|r_i\|_2/\|r_0\|_2 \leq 10^{-9}$) as well as the median sequential and median skip angle values are listed.

Consider two consecutive approximation spaces $\mathcal{S}_i$ and $\mathcal{S}_{i+1}$. As compared to standard GMRES(m), LGMRES(m, k) does not necessarily affect how much of $\mathcal{S}_{i+1}$ can be found in $\mathcal{S}_i$. However, it does typically "improve orthogonality" quite significantly between the current approximation space and the space generated two restart cycles ago: $\mathcal{S}_{i+1}$ and $\mathcal{S}_{i-1}$. This action accelerates the convergence over that of GMRES(m) in many cases. Recall from Theorem 4 that the size of the sequential angles is directly related to the reduction in residual at each cycle. Therefore, if increasing the skip angles occurs at the expense of reducing the average sequential angle, then LGMRES augmenting slows convergence. Intuitively, the method that "wins" generally has large skip angles *and* large sequential angles.

Our experiments seem to indicate that the LGMRES augmenting scheme significantly improves GMRES(m) convergence under the following conditions: GMRES(m) skip angles are small and continue to decrease as the iteration progresses; GMRES(m) sequential angles are relatively small and converging to the same angle as the iteration progresses; or the average skip angle increases significantly after LGMRES augmenting. All of these conditions are typically met for problems that display alternating behavior, although some or all are evident in other problems as well. On the other hand, LGMRES is not as helpful when one of the following occurs: GMRES(m) skip angles are not small; GMRES(m) sequential angles vary greatly from cycle to cycle; GMRES(m) converges in a small number of iterations; or GMRES(m) skip angles and sequential angles are near zero, indicating stalling. We believe that the LGM-

TABLE 3

A comparison of matrix-vector multiplies required for convergence ($\|r_i\|_2/\|r_0\|_2 \leq 10^{-9}$) for $u_{xx} + u_{yy} + Du_x = -(41)^2$, discretized by centered finite differences on the unit square with zero boundary conditions and step-size $h = 1/41$.

Matrix	D	$\frac{\|A-A^T\|_2}{\|A\|_2}$	m	GMRES(m)	LGMRES(m,1)	Adaptive
morgan_1	1	.005	10	735	245	245
			20	415	260	260
			30	272	199	199
morgan_41	41	.22	10	168	252	169
			20	200	301	301
			30	236	296	236
morgan_1681	41^2	.99	10	496	475	464
			20	486	453	469
			30	488	482	477

RES augmenting scheme most benefits problems that are close to symmetric in some sense as these are the problems for which alternating is most easily explained, but we have seen the algorithm perform well for a variety of problems.

Though we have found that scalar measurements of symmetry generally do not correlate with LGMRES performance, a close look at the three aforementioned CD problems with increasing degree of nonsymmetry does provide some insight into LGMRES convergence behavior. In Table 3, results similar to those presented in Figure 3 are listed. However, now we compare GMRES(m) with LGMRES(m, 1) to better examine the effect of appending one error approximation to the Krylov subspace. Whereas previously presented results compared methods with equal-sized approximation spaces or equal storage requirements, here the methods have equal-sized Krylov subspaces at each cycle.

For morgan_1, the coefficient matrix A is nearly symmetric. LGMRES(m, 1) is effective for this problem, particularly for the $m = 10$ case where the GMRES(m) residual vectors alternate noticeably. (The median skip angles in degrees for GMRES(m) are .6, 2.4, and 23.4 for $m = 10$, 20, and 30, respectively.) On the other hand, morgan_41 with $D = 41$ is an excellent example of the type of problem for which LGMRES performs very poorly. Because this problem converges fairly quickly with GMRES(m) and is far from symmetric, we did not expect LGMRES(m, 1) to be very helpful. But the fact that LGMRES(m, 1) actually slows convergence considerably was unanticipated. However, we have since observed that LGMRES generally performs poorly on problems for which the GMRES(m) iteration count increases with increasing m, such as morgan_41. Finally, morgan_1681 is nearly skew-symmetric and benefits only slightly from the augmenting scheme of LGMRES(m, 1). In general, we find in our experiments that nearly skew-symmetric problems do not benefit as much from LGMRES as do nearly symmetric problems.

The morgan_41 problem highlights the need for a potential improvement to the LGMRES algorithm; in particular, an adaptive version that determines whether or not to augment would be beneficial. Designing a simple adaptive LGMRES algorithm effective for all test cases and for all values of m has proved difficult. Our most promising effort to date is described at the end of section 3.3. Results for this algorithm are given in the right column of Table 3 and are decidedly mixed. While this adaptive method usually mitigates the extent to which LGMRES fails on tricky problems, it can be less effective than standard LGMRES on others. Deciding whether or not to

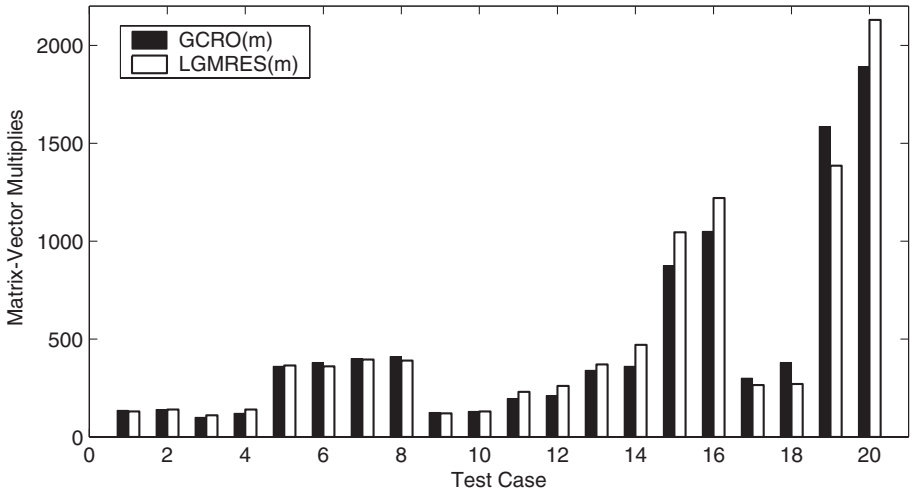

FIG. 4. *A comparison of the numbers of matrix-vector multiplies required for convergence by nontruncated GCRO and full LGMRES. Test cases* 1–20 *correspond to results for* $m = 5$ *followed by* $m = 10$ *for morgan_*1, *morgan_*41, *morgan_*1681, *sherman*1, *sherman*4, *add*20, *cavity*05, *orsirr_*1, *orsreg_*1, *and pores_*3.

augment within a given restart cycle is difficult. We find that analysis within a single restart cycle is not sufficient as augmenting has a cumulative effect. An analysis of convergence across cycles (for both GMRES and LGMRES) would provide a better understanding of the behavior of LGMRES and enable us to design a more effective adaptive strategy.

The effectiveness of LGMRES depends upon the matrix problem and the restart parameter m, but the savings in matrix-vector multiplies are quite substantial in many cases. Though many of the test problems presented in this section would benefit from preconditioning, results for problems such as cavity10 that are difficult to precondition [23] are encouraging. And, in our experience, we find that LGMRES typically does not require more iterations than does restarted GMRES.

4.2. Comparison to GCRO and GCROT. In section 3.4, we discuss the similarities (and differences) between LGMRES(m, k) and GCROT. Here we compare the performance of the two truncated methods, first briefly examining their less practical nontruncated counterparts, full GCRO and full LGMRES (LGMRES(m)).

We evaluate MATLAB implementations of LGMRES(m) and GCRO in the same manner as in section 4.1. That is, we compare the number of matrix-vector multiplies required for the relative residual norm to be less than the convergence tolerance ζ. For these nontruncated methods, we use small values of m, $m = 5$ and $m = 10$, since storage increases with each iteration. We again test the three related CD problems (morgan_1, morgan_41, and morgan_1681) with $\zeta = 10^{-9}$, as these problems were also used in [8]. In addition, we compare results for a subset of the Matrix Market problems from the previous section with $\zeta = 10^{-5}$ as well as one new Matrix Market problem, pores_3, that stalls for both GMRES(10) and GMRES(5).

Figure 4 compares the two methods and indicates that the performance of the two methods, in terms of matrix-vector multiplies, is often similar. In terms of convergence, our experiments seem to indicate that appending vectors to the end of the standard Krylov subspace (as LGMRES does) is not necessarily inferior to orthogo-

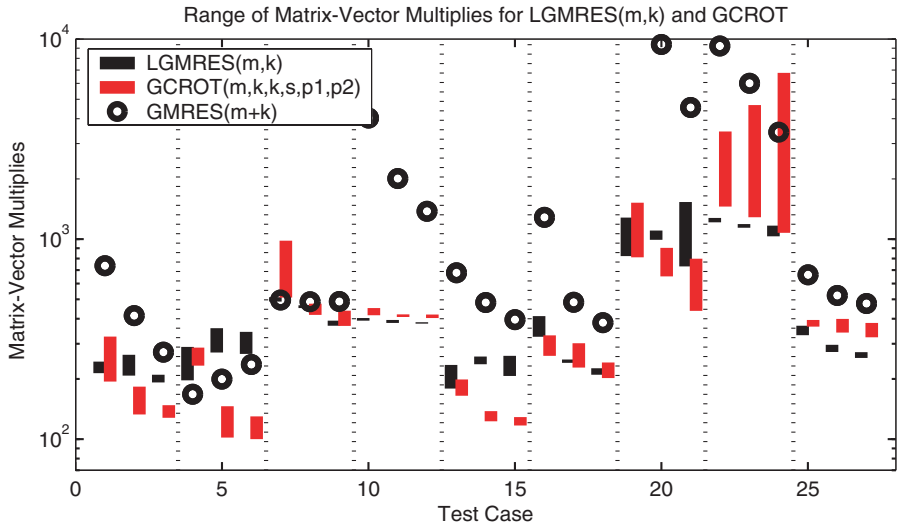

FIG. 5. *A comparison of the minimum to maximum number of matrix-vector multiplies required for convergence by LGMRES(m, k) and GCROT(m, k, k, s, p1, p2) for constant-sized approximation spaces. GMRES(m+k) is also indicated. Test cases* 1–27 *correspond to* $m+k = 10$, $m+k = 20$, *and* $m + k = 30$, *respectively, for morgan_1, morgan_41, morgan_1681, sherman1, sherman4, add20, cavity05, orsirr_1, and orsreg_1.*

nalizing against them at the start of the cycle, and our experience does not clearly indicate which approach is to be preferred in a given situation. Even in the case where GMRES(m) stalls (and intuitively LGMRES(m) would not be helpful), one can find counterexamples such as pores_3 (cases 19 and 20) where slow initial convergence is eventually overcome.

As mentioned in section 3.3, though interesting from a theoretical point of view, nontruncated methods are often impractical due to increasing storage requirements. For example, Figure 4 indicates that both orsirr_1 and pores_3 require storing more than n vectors. Additionally, for LGMRES(m), an increasing number of orthogonalizations are required in each cycle. Therefore, we do not further investigate the convergence behavior of LGMRES(m) but instead focus on a comparison of the more practical versions of the two algorithms: LGMRES(m, k) and GCROT(m, k, k, s, $p1$, $p2$).

For each of these truncated algorithms, the size of the approximation space is $m + k$. We use a MATLAB implementation of GCROT supplied by Oliver Ernst. The test problems are the same as in Figure 4 with approximation spaces of size 10, 20, and 30, although the pores_3 test cases have been dropped since neither truncated algorithm converges for that problem.

For each of the 20 test cases, we ran LGMRES(m, k) with $k = 1 : 3$, as this range was recommended in the previous section. Additionally, ten permutations of GCROT(m, k, k, s, $p1$, $p2$), where $m + k$ is constant, were chosen to reflect the choices in [8] (e.g., $m \leq k$, $s \leq \lceil \frac{m}{2} \rceil$). Figure 5 compares the two methods for all 27 test cases. The bars indicate the range (minimum to maximum) of matrix-vector multiplies required. The circles represent restarted GMRES for each problem with the corresponding approximation space size. Vertical dotted lines separate test cases corresponding to the same matrix problem. The missing circle for test case 19 in-

dicates that GMRES($m + k$) required more than 10,000 iterations. Some of these iteration counts are unrealistically large, but recall that we are not considering preconditioning and are simply evaluating the relative performance of the two algorithms.

Results for the two algorithms are relatively similar in most of the test cases. We again notice that LGMRES(m, k) has particular difficulty with morgan_41 (GCROT has difficulty only for $m + k = 10$). Problem morgan_1681 is somewhat resistant to improvement by both methods, and problem orsirr_1 is highly sensitive to the choice of input parameters with GCROT. It is not clear in our experience or from results presented in [8] how to choose the optimal parameters for GCROT. For the ten of the many possible permutations we chose for GCROT for each fixed $m + k$, there was no observable trend as to which of the ten permutations were most (or least) effective across this set of test problems. In addition, we have found that occasionally $m > k$ can be more effective than the recommended $k > m$ for GCROT (in test cases for problems add20 and orsirr_1, for example). Ernst also found that choosing the parameters for GCROT can be problematic [14]. However, for LGMRES, $k \leq 3$ is nearly always the best choice and the variation in results for $k = 1 : 3$ is generally reasonable.

4.3. Effectiveness for larger preconditioned problems. In this section, we demonstrate that LGMRES can be a helpful addition to preconditioning. We implemented LGMRES in C using a locally modified version of PETSc 2.1.5 [3, 4] in order to easily access preconditioners, test larger problems, and obtain reliable timing results (instead of counting matrix-vector multiplies). Our PETSc implementation is available in PETSc 2.1.6. First, we look at cumulative results for 15 different matrix problems. Then we more closely examine a few of those problems.

We chose a variety of test problems from the Matrix Market Collection [23], the University of Florida (UF) Sparse Matrix Collection [6], and the PETSc [3, 4] collection of test matrices. We use the ILU(p) preconditioner, where p indicates the level of fill (see, e.g., [25]). If a right-hand side is not provided, we generate a random right-hand side. For reference, the test problems are listed in Table 4.

TABLE 4

List of test problems together with the matrix order (n), number of nonzeros (nnz), preconditioner, source, and description of the application area (if known). Source indicates Matrix Market Collection (MM), UF Sparse Matrix Collection (UF), or PETSc test collection (PC), along with a set or directory name if applicable.

	Problem	n	nnz	ILU(p)	Source	Application area
1	fidapm11	22294	623554	ILU(0)	MM: Sparskit	fluid flow
2	memplus	17758	126150	ILU(0)	MM: Hamm	digital circuit simulation
3	arco3	38194	241066	ILU(0)	PC	multiphase flow: oil reservoir
4	arco5	35388	154166	ILU(0)	PC	multiphase flow: oil reservoir
5	arco6	108009	2204937	ILU(0)	PC	multiphase flow: oil reservoir
6	ex40	7740	458012	ILU(0)	UF: FIDAP	fluid flow
7	garon2	13535	390607	ILU(1)	UF: Garon	fluid flow
8	bcircuit	68902	375558	ILU(1)	UF: Hamm	digital circuit simulation
9	xenon1	48600	1181120	ILU(2)	UF: Ronis	crystalline compound analysis
10	pesa	11738	79566	ILU(0)	UF: Gaertner	
11	aft01	8202	125567	ILU(0)	UF: Okunbor	acoustic radiation
12	venkat50	62424	1717792	ILU(0)	UF: Simon	fluid dynamics
13	epb3	84617	463625	ILU(0)	UF: Averous	heat exchanger simulation
14	big	13209	91465	ILU(1)	UF: Gaertner	
15	zhao2	33861	166453	ILU(0)	UF: Zhao	electromagnetic systems

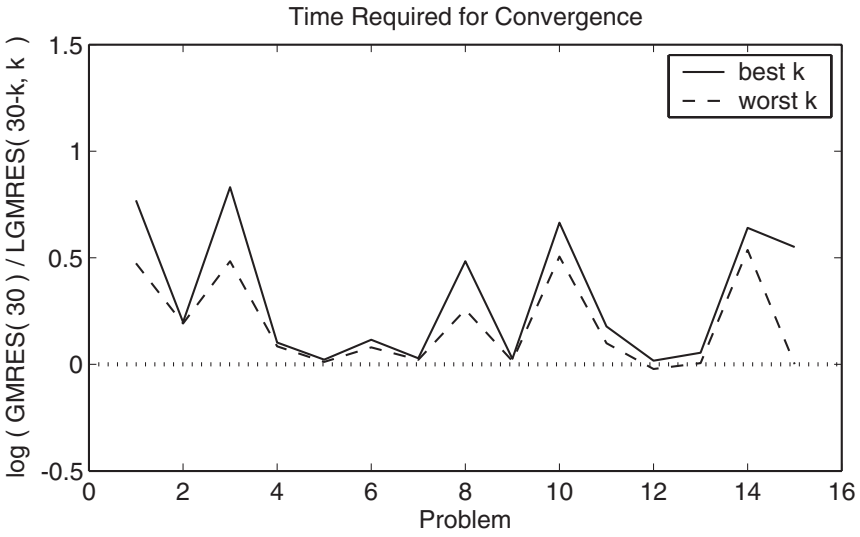

FIG. 6. *A comparison of the time required for convergence for* 15 *different problems with GMRES(*30*) and LGMRES(*$30-k$, k*), $k = 1:3$. All methods use an approximation space of dimension* 30.

We compare the performance of restarted GMRES to that of LGMRES(m, k) with the same approximation space size and then the same storage requirements. For LGMRES, we report results for $k = 1:3$, as we find that choosing k in this range typically results in the most improvement with the least risk of increasing execution time. All tests are run until the relative residual norm is less than the convergence tolerance $\zeta = 10^{-9}$. Recall that GMRES with left preconditioning minimizes the preconditioned residual norm ($\|M^{-1}r\|_2$), and, therefore, the determination of convergence is based on this preconditioned residual norm as usual. The initial guess is a zero vector in all cases. Unless otherwise noted, results provided were run on a Sun UltraSPARC 10 with 256M RAM, a clock-rate of 360 MHz, a 16KB L1 cache, and a 2MB L2 cache. For each problem we report wall clock time for the linear solve. All timings are averages from five runs and have standard deviations of at most two percent, although most are less than one percent. If a method does not converge in 30000 iterations, the execution time reported reflects the time to 30000 iterations, and we say that the method does not converge. Note that iteration counts for problems that converge are well below 30000. We did not compare LGMRES(m, k) to GCROT for these problems because no PETSc implementation of GCROT is available.

In Figure 6, we compare GMRES(30) to LGMRES(29, 1), LGMRES(28, 2), and LGMRES(27, 3). All four of these methods generate an approximation space of dimension 30 during each restart cycle. Similar to the plots seen previously, the y-axis indicates the log of the ratio of the time to converge for GMRES(30) to the time to converge for both the best and worst performing cases of LGMRES($30-k$, k) for $k = 1:3$, and the x-axis corresponds to the 15 test problems in Table 4. Points above the x-axis favor LGMRES and points below favor GMRES. Note that GMRES(30) does not converge (in 30000 iterations) for problems 10, 14, and 15, and LGMRES(27, 3) does not converge for problem 15.

For larger problems in particular, comparing restarted GMRES to an LGMRES method that requires an equal amount of storage is also of interest. Both

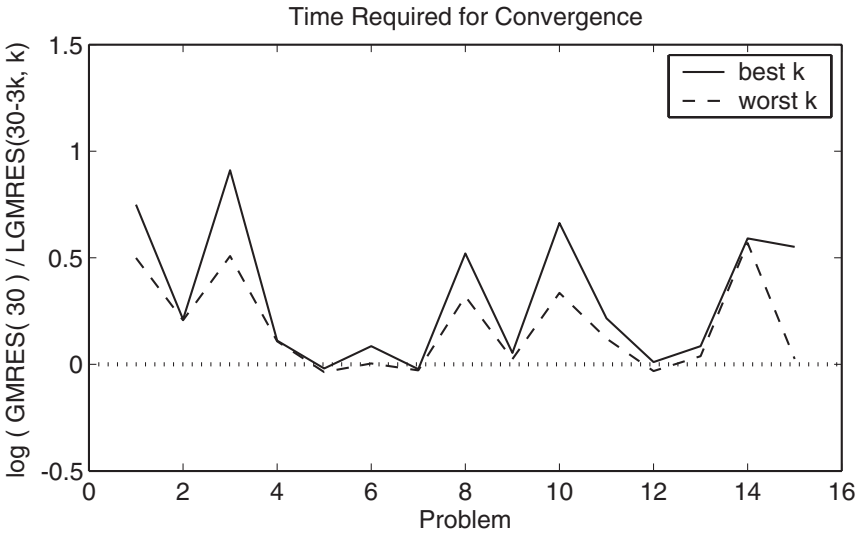

FIG. 7. *A comparison of the time required for convergence for* 15 *different problems with GMRES(*30*) and LGMRES(*$30-3k$*,* k*) with* $k = 1:3$*. All methods require storage for* 33 *vectors of length* n*.*

GMRES(30) and LGMRES($30 - 3k$, k) have the same 33 vector storage requirement (see section 3.2). Similar to the previous figure, Figure 7 compares GMRES(30) to LGMRES(27, 1), LGMRES(24, 2), and LGMRES(21, 3). In this comparison, one augmentation vector must be more helpful than three standard Krylov vectors for LGMRES to win. This requirement is fairly stringent for some of the larger problems given that we allow only 33 vectors of storage. Nevertheless, the majority of the problems still show improvement with LGMRES.

Now we examine problems bcircuit, fidapm11, and big from our test set (in Table 4) in more detail, additionally providing timing results for full GMRES. The results in Tables 5–7 for these three problems demonstrate different possible relations in convergence behavior among LGMRES(m, k), GMRES(m), and full GMRES.

First consider the timing results for problem bcircuit in Table 5. For this problem, full GMRES requires memory resources beyond the physical memory limit of our

TABLE 5

Results for matrix bcircuit and its corresponding right-hand side, with $n = 68902$*,* $nnz = 375558$*, and ILU(1) preconditioning. Times are in seconds and include mean and standard deviations of times for five runs.*

Method	Approx. space dimension	# vectors stored	Matrix-vector multiplies	Time
Full GMRES	1013	1016	1013	2880.364 ± 9.24
GMRES(30)	30	33	5602	1135.38 ± 12.58
LGMRES(29,1)	30	35	2959	615.28 ± 5.61
LGMRES(28,2)	30	37	1730	365.16 ± 2.47
LGMRES(27,3)	30	39	1707	369.70 ± 2.48
LGMRES(27,1)	28	33	2631	533.42 ± 4.54
LGMRES(24,2)	26	33	2467	503.71 ± 3.54
LGMRES(21,3)	24	33	1672	339.42 ± 2.21

TABLE 6

*Results for matrix fidapm*11 *and its corresponding right-hand side, with* $n = 22294$, $nnz = 623554$, *and ILU(*0*) preconditioning. Times are in seconds and include mean and standard deviations of times for five runs.*

Method	Approx. space dimension	# vectors stored	Matrix-vector multiplies	Execution time
Full GMRES	952	955	952	854.02 ± 6.27
GMRES(30)	30	33	16482	2100.23 ± 8.40
LGMRES(29,1)	30	35	5511	704.65 ± 0.18
LGMRES(28,2)	30	37	2915	376.64 ± 0.10
LGMRES(27,3)	30	39	2733	357.19 ± 0.78
LGMRES(27,1)	28	33	5239	664.84 ± 1.89
LGMRES(24,2)	26	33	3399	431.39 ± 1.00
LGMRES(21,3)	24	33	2941	373.96 ± 0.41

TABLE 7

Results for matrix big with a random right-hand side, with $n = 13209$, $nnz = 91465$, *and ILU(*1*) preconditioning. Times are in seconds and include mean and standard deviations of times for five runs.*

Method	Approx. space dimension	# vectors stored	Matrix-vector multiplies	Execution time
Full GMRES	188	191	188	21.70 ± 0.03
GMRES(30)	30	33	> 30000	1231.54 ± 1.08
LGMRES(29,1)	30	35	8500	358.07 ± 0.15
LGMRES(28,2)	30	37	6997	300.18 ± 0.32
LGMRES(27,3)	30	39	6546	281.79 ± 0.70
LGMRES(27,1)	28	33	7654	315.7 ± 0.37
LGMRES(24,2)	26	33	7971	327.96 ± 0.20
LGMRES(21,3)	24	33	8259	332.56 ± 0.09

machine. For this reason, we had to rerun the bcircuit problem on a similar machine with four times as much memory (a Sun UltraSPARC 10 with 1024M RAM, a clock-rate of 440 MHz, a 16KB L1 cache, and a 2MB L2 cache) to obtain timing results for full GMRES. Therefore, for Table 5 only, all results presented for bcircuit were obtained on this second machine. Even with the extra memory provided by the second machine, we see that restarted GMRES(30) is more than twice as fast as full GMRES, and LGMRES is even faster. Conversely, for problem fidapm11 given in Table 6, full GMRES is faster than GMRES(30) on our machine, although LGMRES still has the faster execution time of the three methods on this problem. Finally, results for problem big are given in Table 7. This third problem is interesting because GMRES(30) converges very slowly. In fact, the relative residual norm is still $\approx .002$ after 30000 iterations. Both LGMRES($30 - 3k$, k) and LGMRES($30 - k$, k), on the other hand, improve convergence dramatically over that of GMRES(30). However, for this moderately sized problem, full GMRES requires only 188 iterations and wins by a landslide.

Most of the problems presented here require a restarted method given the resources of the machine chosen for the experiments. On a more powerful machine (more memory and faster processor), full GMRES might be faster for many of these problems. At the same time, because every machine has a limit as to the size problems it can reasonably solve with a full method, restarted methods and acceleration methods provide a great advantage.

Finally, we note that because LGMRES is an accelerator, it is not, in general, a substitute for an effective preconditioner. Although we did encounter a number of test problems for which the ILU preconditioner is not a viable option and LGMRES is a dramatic improvement over GMRES(m), we expect that in those cases an appropriate preconditioner would be even more effective. Nevertheless, LGMRES can be an effective addition to preconditioning for a range of problems. Although LGMRES improvements with preconditioning tend not to be as spectacular as the improvements seen for the nonpreconditioned problems of section 4.1, even moderate acceleration for large problems can translate into significant time savings.

5. Concluding remarks. In this paper, we have described a method that accelerates the convergence of GMRES(m). We have also discussed some interesting observed properties of the convergence of GMRES(m) that motivated the algorithm's development. Experimental results demonstrate that the LGMRES augmentation scheme is an effective accelerator for GMRES(m) with or without preconditioning. Furthermore, the algorithm is straightforward and easy to implement. However, LGMRES is not typically a substitute for preconditioning and does not help when a problem stalls for a given restart parameter. Possible improvements to the algorithm include a robust adaptive variant. In future work, we will describe a more memory-efficient block implementation of the LGMRES algorithm.

Acknowledgments. We thank Oliver Ernst for providing us with his MATLAB implementation of GCROT and the referees for their many helpful suggestions.

REFERENCES

[1] S. F. Ashby, T. A. Manteuffel, and P. E. Saylor, *A taxonomy for conjugate gradient methods*, SIAM J. Numer. Anal., 27 (1990), pp. 1542–1568.

[2] A. Baker, J. Dennis, and E. R. Jessup, *Toward memory-efficient linear solvers*, in VECPAR'2002, Fifth International Conference on High Performance Computing for Computational Science: Selected Papers and Invited Talks, J. Palma, J. Dongarra, V. Hernandez, and A. A. Sousa, eds., Lecture Notes in Comput. Sci. 2565, Springer-Verlag, New York, 2003, pp. 315–327.

[3] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, L. C. McInnes, and B. F. Smith, *PETSc home page*, http://www.mcs.anl.gov/petsc, 2001.

[4] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith, *PETSc Users Manual*, Tech. report ANL-95/11, Revision 2.1.3, Argonne National Laboratory, Argonne, IL, 2002.

[5] A. Chapman and Y. Saad, *Deflated and augmented Krylov subspace techniques*, Numer. Linear Algebra Appl., 4 (1997), pp. 43–66.

[6] T. Davis, *University of Florida sparse matrix collection*, http://www.cise.ufl.edu/research/sparse/matrices, 2002.

[7] E. de Sturler, *Nested Krylov methods based on GCR*, J. Comput. Appl. Math., 67 (1996), pp. 15–41.

[8] E. de Sturler, *Truncation strategies for optimal Krylov subspace methods*, SIAM J. Numer. Anal., 36 (1999), pp. 864–889.

[9] E. de Sturler and D. R. Fokkema, *Nested Krylov methods and preserving orthogonality*, in Sixth Copper Mountain Conference on Multigrid Methods, N. Melson, T. Manteuffel, and S. McCormick, eds., Part 1 of NASA Conference Publication 3324, NASA, 1993, pp. 111–126.

[10] M. Eiermann and O. G. Ernst, *Geometric aspects in the theory of Krylov subspace methods*, in Acta Numerica, Acta Numer. 10, Cambridge University Press, Cambridge, UK, 2001, pp. 251–312.

[11] M. Eiermann, O. G. Ernst, and O. Schneider, *Analysis of acceleration strategies for restarted minimum residual methods*, J. Comput. Appl. Math., 123 (2000), pp. 261–292.

[12] S. C. Eisenstat, H. C. Elman, and M. H. Schultz, *Variational iterative methods for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal., 20 (1983), pp. 345–357.

[13] M. Embree, *The tortoise and the hare restart GMRES*, SIAM Rev., 45 (2003), pp. 259–266.

[14] O. Ernst, *A numerical study of acceleration schemes for restarted minimum residual methods*, Presentation at the Seventh Copper Mountain Conference on Iterative Methods, Copper Mountain, CO, 2002.
[15] D. R. Fokkema, G. L. G. Sleijpen, and H. A. van der Vorst, *Accelerated inexact Newton schemes for large systems of nonlinear equations*, SIAM J. Sci. Comput., 19 (1998), pp. 657–674.
[16] A. Greenbaum and Z. Strakoš, *Matrices that generate the same Krylov residual spaces*, in Recent Advances in Iterative Methods, G. Golub, A. Greenbaum, and M. Luskin, eds., Springer-Verlag, New York, 1994, pp. 95–118.
[17] W. Joubert, *On the convergence behavior of the restarted GMRES algorithm for solving nonsymmetric linear systems*, Numer. Linear Algebra Appl., 1 (1994), pp. 427–447.
[18] R. B. Morgan, *A restarted GMRES method augmented with eigenvectors*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 1154–1171.
[19] R. B. Morgan, *Implicitly restarted GMRES and Arnoldi methods for nonsymmetric systems of equations*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1112–1135.
[20] R. B. Morgan, *GMRES with deflated restarting*, SIAM J. Sci. Comput., 24 (2002), pp. 20–37.
[21] N. M. Nachitgal, L. Reichel, and L. N. Trefethen, *A hybrid GMRES algorithm for nonsymmetric linear systems*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 796–825.
[22] N. M. Nachtigal, S. C. Reddy, and L. N. Trefethen, *How fast are nonsymmetric matrix iterations?*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 778–795.
[23] National Institute of Standards and Technology, Mathematical and Computational Sciences Division, *Matrix Market*, http://math.nist.gov/MatrixMarket, 2002.
[24] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput., 14 (1993), pp. 461–469.
[25] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston, 1996.
[26] Y. Saad, *Analysis of augmented Krylov subspace methods*, SIAM J. Matrix Anal. Appl., 18 (1997), pp. 435–449.
[27] 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.
[28] V. Simoncini, *On the convergence of restarted Krylov subspace methods*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 430–452.
[29] H. A. van der Vorst and C. Vuik, *The superlinear convergence behavior of GMRES*, J. Comput. Appl. Math., 48 (1993), pp. 327–341.
[30] H. A. van der Vorst and C. Vuik, *GMRESR: A family of nested GMRES methods*, Numer. Linear Algebra Appl., 1 (1994), pp. 369–386.
[31] D. M. Young and K. C. Jea, *Generalized conjugate-gradient acceleration of nonsymmetrizable iterative methods*, Linear Algebra Appl., 34 (1980), pp. 154–194.