SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 796-825, July 1992

A HYBRID GMRES ALGORITHM FOR NONSYMMETRIC LINEAR SYSTEMS*

NOËL M. NACHTIGAL†, LOTHAR REICHEL‡, AND LLOYD N. TREFETHEN§

Abstract. A new hybrid iterative algorithm is proposed for solving large nonsymmetric systems of linear equations. Unlike other hybrid algorithms, which first estimate eigenvalues and then apply this knowledge in further iterations, this algorithm avoids eigenvalue estimates. Instead, it runs GMRES until the residual norm drops by a certain factor, then re-applies the polynomial implicitly constructed by GMRES via a Richardson iteration with Leja ordering. Preliminary experiments suggest that the new algorithm frequently outperforms the restarted GMRES algorithm.

Key words. iterative method, Krylov subspace, CGNR, GMRES, CGS, hybrid, pseudospectrum

AMS(MOS) subject classification. 65F10

1. Introduction.

In this paper we present a new point of view regarding nonsymmetric matrices, and as a natural outgrowth, a new hybrid iterative algorithm. The point of view is that if a matrix is nonsymmetric (more precisely, nonnormal), any attempt to make use of its eigenvalues should be viewed with caution. The new algorithm is a hybrid scheme in which a few steps of GMRES [29] are followed by a Richardson iteration based on the polynomial implicitly constructed by GMRES, with the factors ordered in a Leja sequence for stability [25]. Unlike other hybrid algorithms, this one never estimates any eigenvalues. It is simpler than other hybrid iterations, but more robust, and appears to outperform other methods in many cases.

We begin with a brief explanation and survey of hybrid methods, assuming that the reader is already familiar with GMRES, the Arnoldi process, and polynomial iterations. Suppose we are given a large nonsymmetric system of linear equations

$$Ax = b, \quad A \in \mathbb{C}^{N\times N}, \quad x, b \in \mathbb{C}^N, \tag{1.1}$$

where A may be the matrix that results after preconditioning. The many nonhybrid iterative methods that have been proposed for solving such systems can be divided into two categories: (i) those that require no a priori information about A, of which three of the most important are CGN, CGS, and GMRES,[1] and (ii) those that do require a priori information about A, such as the Richardson and Chebyshev iterations. The idea of a hybrid iteration is to combine these approaches in a two-phase algorithm:

Phase I: acquire information about A via an iteration of type (i);
Phase II: apply that information in further iterative steps of type (ii).

In practice, of course, things need not be quite so simple; a robust code may loop back to Phase I one or more times to ensure an adequate convergence rate.

* Received by the editors April 5, 1990; accepted for publication (in revised form) June 5, 1991.

† Research Institute for Advanced Computer Science, Mail Stop T041-5, NASA Ames Research Center, Moffett Field, California 94035 (santa@riacs.edu) This research was supported by Air Force Office of Scientific Research grant 87-0102 and by National Science Foundation grant 8814314-DMS.

‡ Department of Mathematics and Computer Science, Kent State University, Kent, Ohio 44242 (reichel@mcs.kent.edu). This research was supported in part by IBM Bergen Scientific Centre and National Science Foundation grant DMS-8704196.

§ Department of Computer Science, Cornell University, Ithaca, New York 14853 (lnt@cs.cornell.edu).

[1] CGN is a name for the conjugate gradient iteration applied to the normal equations [14], CGS is a biorthogonalization algorithm adapted from the biconjugate gradient iteration [36], and GMRES is an algorithm based on residual minimization in a Krylov subspace [29].

The assumption underlying the hybrid idea is that algorithms of type (i) cost more per step than those of type (ii), so that a switch from the one to the other is potentially advantageous. This assumption frequently holds for GMRES and for the many other Krylov subspace iterations such as ORTHORES, ORTHOMIN, and ORTHODIR, because these algorithms have the unfortunate property that the work and storage required to carry out the nth step grow in proportion to n. The goal of a hybrid algorithm is to recover some of this factor $O(n)$. On the other hand the assumption does not hold for CGS nor for CGN in problems where A^* is as easy to apply to a vector as A. Thus the natural realm of applicability of hybrid methods is to problems where Krylov subspace methods take fewer steps than the alternatives. For a discussion of when this is likely to be the case, see [20].

The recent literature on hybrid methods begins with a paper of Manteuffel [18]. In Manteuffel's algorithm, a number of extreme eigenvalues of A are first estimated by a modified power iteration (Phase I). These eigenvalue estimates are then surrounded by an ellipse, and a Chebyshev iteration is carried out with parameters corresponding to that ellipse (Phase II). Schematically,

Manteuffel '78:
modified power iteration $\rightarrow$ eigenvalue estimates $\rightarrow$ ellipse $\rightarrow$ Chebyshev iteration.

Ashby has implemented this algorithm in a Fortran code package called ChebyCode [2], which incorporates many safeguards and extra features omitted in this outline.

Let x_0 denote the approximation to the solution $A^{-1}b$ at the beginning of an iterative process, which may correspond to Phase I or Phase II depending on context. We use the following (standard) notation:

$$n\text{th iterate: } x_n,$$

$$n\text{th error: } e_n = A^{-1}b - x_n,$$

$$n\text{th residual: } r_n = Ae_n = b - Ax_n.$$

Manteuffel's algorithm delivers Phase II iterates x_n satisfying

$$x_n = x_0 + q_{n-1}(A)r_0, \qquad q_{n-1} \in P_{n-1}$$

and

$$e_n = p_n(A)e_0, \quad r_n = p_n(A)r_0, \quad p_n \in P_n, \quad p_n(0) = 1, \tag{1.2}$$

where $p_n(z) = 1 - zq_{n-1}(z)$ is a Chebyshev polynomial shifted to the ellipse of eigenvalue estimates. (P_n denotes the set of polynomials of degree less than or equal to n.) The same equations (1.2) hold for other hybrid Krylov subspace iterations, including our own. The various algorithms differ only in the choice of the sequence of polynomials $p_n(z)$, known as *residual polynomials*, and in the mechanics of how they are applied. Our goal is to make $\|p_n(A)r_0\|$ small, and the obvious way to achieve this is to try to make $\|p_n(A)\|$ small.

One modification of Manteuffel's algorithm is to replace the Chebyshev iteration of Phase II by a more general iteration, an idea first proposed by Smolarski and Saylor [34], [35]. Phase I of their algorithm constructs a polygon containing eigenvalue estimates, then solves a discrete least-squares approximation problem on that polygon to obtain an effective residual polynomial $p_\nu(z)$ for some integer ν. Phase II applies this polynomial one or more times by means of a cyclic Richardson iteration

$$p_{k\nu}(z) = [p_\nu(z)]^k, \qquad k = 1, 2, \cdots. \tag{1.3}$$

In outline:

Smolarski and Saylor '81:
modified power iteration $\rightarrow$ eigenvalue estimates $\rightarrow$ polygon $\rightarrow$
L^2-optimal $p_\nu(z) \rightarrow$ Richardson iteration.

The advantage of such an algorithm is that since the approximation problem is posed on an arbitrary domain of estimated eigenvalues, there is no restriction to matrices whose spectra are well approximated by an ellipse. Throughout this paper, $p_\nu(z)$ denotes a residual polynomial of fixed degree which forms the basis of a cyclical Phase II iteration defined by (1.3).

Another modification of Manteuffel's algorithm is to replace the power iteration of Phase I by an Arnoldi iteration, which is now the standard method for estimating eigenvalues of nonsymmetric matrices iteratively. In fact this difference is not as great as the names suggest, for the modified power iteration is essentially the same as the Arnoldi iteration. Together with the Arnoldi point of view, however, comes the important additional advantage that an approximate solution in Phase I can be conveniently constructed by the closely related GMRES algorithm. This kind of hybrid was first proposed by Elman, Saad, and Saylor [7]:

Elman, Saad, and Saylor '86:
Arnoldi/GMRES $\rightarrow$ eigenvalue estimates $\rightarrow$ ellipse $\rightarrow$ Chebyshev iteration.

To be more precise about what we mean by an "Arnoldi/GMRES" calculation, the Arnoldi and GMRES iterations both make use of a Hessenberg matrix obtained by the orthogonalization of a sequence of Krylov vectors. The Arnoldi iteration estimates eigenvalues of A by computing the eigenvalues of the square part of this matrix, while GMRES finds approximate solutions to $Ax = b$ by solving a least-squares problem involving the same matrix made rectangular by the addition of an extra row. In a Phase I calculation of Arnoldi/GMRES type, both of these computations are carried out simultaneously, so that Phase II begins with both eigenvalue information and a good initial guess.

The most general hybrid algorithms combine both of these modifications, and thus differ from Manteuffel's algorithm in both Phases I and II. One of the first of these to be proposed was the PSUP algorithm of Elman and Streit [8], in which an Arnoldi/GMRES iteration to obtain eigenvalue estimates is followed by a solution of an L^∞ approximation problem on a polygonal domain, with the resulting polynomial iteration implemented by a matrix version of Horner's rule:

Elman and Streit '86:
Arnoldi/GMRES $\rightarrow$ eigenvalue estimates $\rightarrow$ polygon $\rightarrow$
L^∞-optimal $p_\nu(z) \rightarrow$ Horner iteration.

Another algorithm of this type, developed at about the same time by Saad [28], solves an L^2 approximation problem on a polygon after first constructing a well-conditioned basis of shifted Chebyshev polynomials, then applies the resulting polynomial in a second-order Richardson iteration:

Saad '87:
Arnoldi/GMRES $\rightarrow$ eigenvalue estimates $\rightarrow$ polygon $\rightarrow$ Chebyshev basis $\rightarrow$
L^2-optimal $p_\nu(z) \rightarrow$ second-order Richardson iteration.

More recently, Saylor and Smolarski have also modified their method to take advantage of GMRES [32], [33]:

Saylor and Smolarski '91:
Arnoldi/GMRES $\rightarrow$ eigenvalue estimates $\rightarrow$ polygon $\rightarrow$
L^2-optimal $p_\nu(z) \rightarrow$ Richardson iteration.

Finally, two recent algorithms developed since this article was first drafted make use of numerically computed Schwarz–Christoffel conformal maps [17], [43]:

Li '91:
Arnoldi/GMRES $\rightarrow$ eigenvalue estimates $\rightarrow$ polygon $\rightarrow$ conformal map $\rightarrow$
rational approximation $\rightarrow (k, l)$-step iteration.

Starke and Varga '91:
Arnoldi/GMRES $\rightarrow$ eigenvalue estimates $\rightarrow$ polygon $\rightarrow$
conformal map $\rightarrow$ Faber polynomials $\rightarrow$ Richardson iteration.

The above algorithms are summarized in Table 1.1. We hasten to add that these algorithms differ in many important ways that we have not mentioned and indeed, all of the one- or two-line summaries of this section represent only the barest of skeletons.

This completes our survey of existing hybrid algorithms of the fully specified sort, where procedures for both Phases I and II are given, so that the algorithm is applicable in principle to an arbitrary matrix. In addition, however, there is a large literature of "polynomial iterations" or "semi-iterative methods" devoted to Phase II by itself, on the assumption that eigenvalue estimates are already available, and each of these becomes a full-fledged hybrid algorithm as soon as it is coupled with an Arnoldi/GMRES iteration for Phase I. For example, Opfer and Schober construct a first-order Richardson iteration from an L^∞-optimal polynomial [22]; Eiermann [5] and Gutknecht [13] investigate Faber and Faber-CF approximations, respectively; Fischer and Reichel [9], [24] and Tal-Ezer [37] derive $p_\nu(z)$ by polynomial interpolation in Féjer points (conformal images of roots of unity); and Gragg and Reichel recommend the use of polynomials orthogonal on the boundary of the eigenvalue domain [11]. There are also a number of important papers by Eiermann, Niethammer, Varga, and others on further aspects of these iterations and their connections with approximation theory and complex analysis; an example is [6]. We will not attempt a survey, but merely note in conclusion that whereas the idea of estimating eigenvalues by the Arnoldi process clearly predominates for Phase I of hybrid algorithms in the current literature, the possibilities for Phase II are numerous.

TABLE 1.1
Hybrid algorithms.

		Phase II	
		Chebyshev	Other
Phase I	Power Method	Wrigley [42] Manteuffel [18] Saylor [30]	Smolarski and Saylor [34], [35]
	Arnoldi/GMRES	Elman, Saad, and Saylor [7]	Elman and Streit [8] Saad [28] Saylor and Smolarski [32], [33] Li [17] Starke and Varga [43]

Hybrid iterative algorithms are closely related to the idea of *polynomial preconditioners* [15], [27]. The polynomial $p_\nu(z)$ constructed by the hybrid algorithm proposed in this paper might be applied as a preconditioner, and a recursive version of this idea has recently been considered by Joubert [16, § 3.5].[2] In the symmetric case, there is a long history of hybrid algorithms and polynomial preconditioners motivated mainly by the search for parameters for Chebyshev iterations. In particular, a recursive conjugate-gradient preconditioner analogous to Joubert's has recently been proposed for the symmetric case by O'Leary [21]. Various further combinations of hybrid and preconditioning ideas will undoubtedly be investigated in the years to come.

2. The trouble with eigenvalues.

What the existing hybrid methods have in common is that they all estimate eigenvalues, construct a domain enclosing them in the complex plane, and then calculate a polynomial $p_\nu(z)$ that is small in some sense on that domain.

$$\begin{array}{l} \textit{Existing algorithms:} \\ \text{Arnoldi/GMRES} \rightarrow \text{eigenvalue estimates} \rightarrow \\ \qquad\qquad \text{enclosing domain} \rightarrow p_\nu(z) \rightarrow \text{iteration.} \end{array} \tag{2.1}$$

In this section we explain why we consider the introduction and subsequent manipulation of eigenvalue estimates inappropriate. The principal problem is that eigenvalues do not generally contain enough information to capture the behavior of a matrix efficiently in the nonnormal case and, in particular, even though the scalar $p_\nu(\lambda)$ may be small whenever λ is an eigenvalue of A, it does not follow that the matrix $p_\nu(A)$ is small in norm. A secondary problem, relevant even for normal matrices, is that the smallness of $p_\nu(z)$ on a set of estimated eigenvalues does not imply that it is small at the exact eigenvalues.

To begin the discussion with the first of these problems, let us suppose first that exact eigenvalues rather than mere estimates happen to be available at the end of Phase I. On the face of it this should be the ideal situation for the standard iterations in Phase II. Following [38], however, we can show by an example that eigenvalue information may be utterly misleading as to the actual behavior of A. Let A be a large upper-triangular Toeplitz matrix of the form

$$A = \begin{pmatrix} 1 & 1 & \frac{1}{2} & & & \\ & 1 & 1 & \frac{1}{2} & & \\ & & 1 & 1 & \frac{1}{2} & \\ & & & 1 & 1 & \frac{1}{2} \\ & & & & 1 & 1 \\ & & & & & 1 \end{pmatrix} \qquad (N \times N). \tag{2.2}$$

This matrix has just the single eigenvalue $\{1\}$.[3] Thanks to this simple eigenvalue distribution, one might naturally expect a Phase II iteration to achieve rapid convergence with the sequence of residual polynomials $p_n(z) = (1 - z)^n$. In actuality, however, this choice will lead to geometric divergence at a rate approximating $(\frac{3}{2})^n$ for large N and $n \ll N$. The reason is that for practical purposes A behaves much more nearly as if its spectrum were

$$\Lambda_{\text{practical}} = f(\bar{D}), \tag{2.3}$$

[2] Joubert's work, though formulated in terms of preconditioning rather than hybrid iterations, is the closest we have seen in the literature to the idea proposed here. His implementation does not take advantage of the $O(\nu)$ speedup in Phase II, however, so little improvement over existing methods is achieved.

[3] In fact it is defective, but the reader should not make too much of that, for the defectiveness is an inessential property that could be removed without changing the matrix behavior significantly by adding small perturbations to the diagonal elements. In any case, similar examples are readily constructed that are nondefective to begin with, such as the matrix (3.6) in § 3.

where $\bar{D}$ is the closed unit disk and $f(z)$ is the *symbol* of this Toeplitz matrix [26],

$$f(z) = 1 + z + \tfrac{1}{2}z^2. \tag{2.4}$$

This domain is illustrated in Fig. 2.1. Although it contains the exact spectrum $\Lambda = \{1\}$, it is much larger than that and, in particular, the fact that it extends to the right as far as the point $z = 2.5$ is what causes divergence at the rate $(\frac{3}{2})^n$ of a Richardson iteration based on $p_n(z) = (1 - z)^n$.

On the other hand, if we take $p_n(z)$ to be a sequence of polynomials that are small on $\Lambda_{\text{practical}}$ instead of just Λ, the convergence of the Richardson iteration for the same example becomes rapid.

This example is contrived, but similar phenomena occur frequently in scientific computing. Convection-diffusion equations, for example (the favorite test problems in papers on iterative methods), sometimes lead to matrices with misleading spectra closely related to (2.2). The matrices that arise in spectral methods also have misleading eigenvalues [23]. We are convinced that this pattern is a common one throughout applications involving nonnormal matrices [39].

If eigenvalues are not the right information on which to base a Phase II iteration, might some different information perform better? It will not do to look at Jordan structure, for aside from the impracticability of estimating Jordan blocks in Phase I, we have already noted that small perturbations would make the Jordan canonical form of this and any other matrix diagonal without changing the eigenvalues very much, in which case we are back where we started. Another unsuitable idea is to consider the spectrum of A in the operator limit $N \to \infty$. That would be satisfactory for the example above, but not for many other problems in which the limit process is less sharp, as occur, for example, in spectral methods. A third idea is to replace the spectrum of A by the the *field of values* $W(A)$, i.e., the set of Rayleigh quotients x^*Ax/x^*x, $x \in \mathbb{C}^N$. However, fields of values are too big to be appropriate for eigenvalue-style applications, besides being always convex. For example, although the matrices

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 3 \\ 0 & 1 \end{pmatrix} \tag{2.5}$$

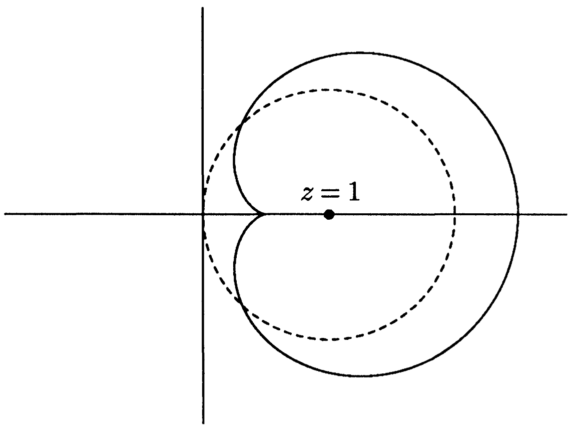


FIG. 2.1. *Spectrum (dot) and pseudospectrum (bounded by solid curve) of the matrix A of (2.2) for large dimension N. Since the pseudospectrum extends outside the dashed circle, a Richardson iteration based on the residual polynomials $p_n(z) = (1 - z)^n$ will diverge as n increases with $n \ll N$.*

are far from singular, their fields of values both contain the origin, and therefore no residual polynomial satisfying the normalization condition $p_n(0) = 1$ can ever be smaller than 1 on $W(A)$. For a further discussion of this point, see [20, § 6].

We believe that if the idea of working with a domain in the complex plane is to be retained, a better approach might be to replace Λ by a fourth candidate, the *ε-pseudospectrum* [23], [26], [38], [39], defined by

$$\Lambda_\varepsilon = \{\lambda \in \mathbb{C}: \|(\lambda I - A)^{-1}\| \geqq \varepsilon^{-1}\}, \tag{2.6}$$

where $\varepsilon > 0$ is a small parameter which for hybrid methods should be taken to be on the order of the residual reduction that has been achieved by Phase I (i.e., $\varepsilon \approx \tau$, with τ defined by (3.3), below). Equivalently, Λ_ε can be defined in terms of perturbations of eigenvalues

$$\Lambda_\varepsilon = \{\lambda \in \mathbb{C}: \lambda \text{ is an eigenvalue of } A + E \text{ for some } \|E\| \leqq \varepsilon\}. \tag{2.7}$$

We have discussed pseudospectra elsewhere, however, and will not pursue the idea further here. Instead, our more fundamental proposal (in § 3) is that domains in the complex plane need not be manipulated at all.

Before leaving the subject of eigenvalue-related quantities, however, we must mention a remarkable phenomenon that may partially explain why existing hybrid algorithms work as well as they do. Eigenvalues *estimates* are sometimes more reliable than exact eigenvalues! We have noticed this effect in our experiments and Manteuffel informs us that he has noticed it too [19]. One way to explain it is to note that eigenvalue estimates tend to come closer to a pseudospectrum than to the exact spectrum [39], and it is usually the pseudospectrum that provides the better iteration parameters. A related phenomenon in another context has been mentioned in [40, § 7].

However, this eigenvalue-estimate effect is not robust enough to provide a foundation for an algorithm to be applied to arbitrary matrices, and to illustrate this we will now turn to the second problem with eigenvalue estimates mentioned in the opening paragraph of this section. For a trivial example, take

$$A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}. \tag{2.8}$$

After one step of the Arnoldi iteration, the estimated eigenvalue is

$$\lambda_{\text{Arnoldi}} = \frac{r_0^T A r_0}{r_0^T r_0} = 0, \tag{2.9}$$

assuming r_0 is real. Considering the symmetry about the origin of the actual eigenvalues $\pm i$, this may seem a natural enough choice, but it is fatal for any polynomial iteration that attempts to construct polynomials $p_n(z)$ that are small on the spectrum but normalized by $p_n(0) = 1$. For example, if $p_n(z)$ is constructed as a product of terms $(1 - z/\lambda_i)$ corresponding to various eigenvalue estimates λ_i (see (5.3)), an eigenvalue estimate $\lambda = 0$ will lead to a factor $(1 - z/0)$ and a consequent division by zero. We shall return to this example at the end of § 3.

For a richer example along the same lines, consider the matrix

$$A = \begin{pmatrix} I + R_1 & 0 \\ 0 & -I + R_2 \end{pmatrix} \qquad (N \times N), \tag{2.10}$$

where R_1 and R_2 are dense matrices of dimension $N/2$ with independent normally distributed random elements of standard deviation $\frac{1}{4}\sqrt{N/2}$. For large N, the eigenvalues of

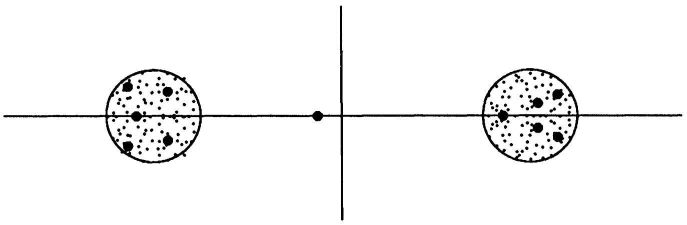


FIG. 2.2. *Arnoldi eigenvalue estimates at step $n = 11$ for the matrix A of* (2.10), $N = 200$. *The small dots are the exact eigenvalues of A. One of the estimates appears near the origin, far from the exact spectrum, which will cause a residual polynomial based on these estimates to perform poorly.*

A are approximately uniformly distributed in the disks $|z \pm 1| \leqq \frac{1}{4}$, well away from the origin, and the condition number is $\kappa(A) \approx 2.06$ [4]. Fig. 2.2 shows that ten of the Arnoldi estimates match this spectrum reasonably well at step $n = 11$, but the eleventh, thanks to the symmetry, appears near the origin. Since this spurious eigenvalue is not exactly at the origin, it does not cause a division by zero, but it certainly leads a polynomial iteration astray.

To summarize, it is not entirely safe to base a matrix iteration on exact eigenvalues, if they happen to be available, nor, so far as we are aware, on eigenvalues estimated by any existing methods. Of course, a new eigenvalue estimator might be found with better properties—and in fact the algorithm we are about to propose might be described in those terms. Since the successful operation of such an algorithm depends on its estimating eigenvalues *incorrectly*, however, we see little to be gained by interpreting it as an eigenvalue estimator.

3. The hybrid GMRES algorithm. We propose that in (2.1), the middle steps should be eliminated:

$$\text{(3.1)} \qquad \begin{array}{l} \textit{New algorithm:} \\ \text{GMRES} \rightarrow p_\nu(z) \rightarrow \text{iteration.} \end{array}$$

The GMRES iteration of Phase I constructs a sequence of residual polynomials that minimize the norm of the residual

$$\text{(3.2)} \qquad \text{GMRES: } \|r_n\| = \|p_n(A)r_0\| = \min_{\substack{p \in P_n \\ p(0)=1}} \|p(A)r_0\|, \qquad n = 1, 2, \cdots .$$

Correspondingly, what Phase II requires is another sequence of residual polynomials. Why translate from polynomials to eigenvalue estimates and back again? We propose to take precisely the GMRES polynomial $p_\nu(z)$ obtained at some step ν of Phase I and continue applying it over and over again in Phase II cyclically as in (1.3):

HYBRID GMRES ALGORITHM

Start with a *random* initial guess x_0.

Phase I: Run GMRES until $\|r_n\|$ drops by a suitable amount. Set $\nu := n$.

Phase II: Re-apply the GMRES polynomial $p_\nu(z)$ cyclically until convergence.

This algorithm is purely a GMRES hybrid, for no Arnoldi eigenvalue estimates are calculated. No domain of estimated eigenvalues is constructed and no approximation problem is solved in the complex plane. The omission of these steps makes our hybrid algorithm simpler than most of those previously proposed.

Of course, many issues have been left out of this description, such as:

1. What is a "suitable" reduction in $\|r_n\|$ for terminating Phase I?
2. How shall $p_\nu(z)$ be constructed from the GMRES iteration?

3. How shall $p_\nu(z)$ be applied in Phase II?
4. What safeguards should be added to ensure convergence?
5. What are the properties of this algorithm in theory?
6. How does it perform in practice?

Most of these questions will be addressed in the next few sections, but not definitively. We have little doubt that improvements can be effected in many of the details of our implementation.

Before we turn to these issues, however, a few remarks will clarify the idea of our algorithm; further details are given in § 6. Suppose that at the νth GMRES step we have

$$\frac{\|r_\nu\|}{\|r_0\|}=\frac{\|p_\nu(A)r_0\|}{\|r_0\|}=\tau \tag{3.3}$$

for some $\tau < 1$. Our hope is that we then have

$$\|p_\nu(A)\| \approx \tau, \tag{3.4}$$

so that further iterations with the same polynomial $p_\nu(z)$ will continue to reduce the residual. Of course, such a conclusion can never be guaranteed, for just as adaptive integrators can always be fooled by integrands with spikes in places that fail to get sampled, adaptive matrix iterators can always be fooled by initial residuals r_0 with small components in key directions. Nevertheless it is a reasonable hope that (3.4) may hold, provided that x_0 (or more precisely r_0) is chosen at random, and provided also that τ lies well enough below 1 so that $p_\nu(z)$ is forced to contain some genuine information about A. Probabilistic theorems to this effect could be proved.

Thus what GMRES "knows" about the matrix A at the end of Phase I, with a little luck, is no more and not much less than (3.4). It does not know anything very precise about the eigenvalues of A, and in particular, there is no reason to expect that the roots of $p_\nu(z)$ must always be good approximations to eigenvalues (though in some cases they will be). More generally, consider the family of lemniscates defined by

$$L_\alpha=\{z\in\mathbb{C}: |p_\nu(z)|=\alpha\}, \qquad \alpha\geqq 0. \tag{3.5}$$

The set of roots of $p_\nu(z)$ is the same as the lemniscate L_0, which we have just claimed to be of little significance. But there is a choice of α of greater interest:

$$L_\tau = \text{"the GMRES lemniscate."}$$

Roughly speaking, the domain enclosed by L_τ is GMRES's best concept at step ν of the effective spectrum of A. (We hope to make this statement more precise in later work.) In running our hybrid algorithm, we have found it informative to plot L_τ at the end of Phase I (by sampling $\log|p_\nu(z)|$ on a grid and calling a contour plotter). On the same plot we generally display the zeros of $p_\nu(z)$ and also the lemniscate L_1 that passes through the origin. These plots of lemniscates give a graphic indication of the manner in which A may be causing difficulty, and in the practical world this translates into guidance in the design of preconditioners.

For example, consider the following banded Toeplitz matrix investigated by Grcar [12]:

$$A=\begin{pmatrix} 1 & 1 & 1 & 1 & & & & \\ -1 & 1 & 1 & 1 & 1 & & & \\ & -1 & 1 & 1 & 1 & 1 & & \\ & & -1 & 1 & 1 & 1 & 1 & \\ & & & \ddots & & \ddots & & \ddots \end{pmatrix} \qquad (N\times N). \tag{3.6}$$

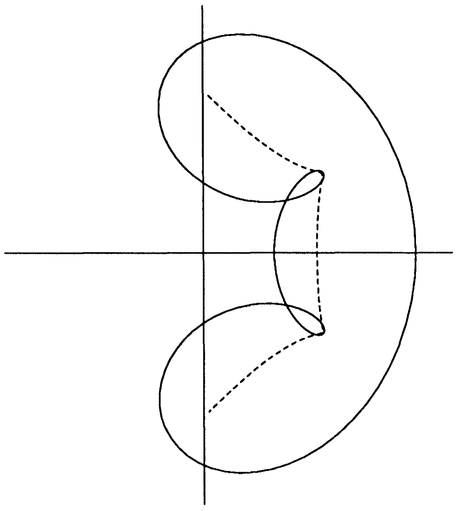


FIG. 3.1. *Spectrum* (*along the dashed curve*) *and* ε-*pseudospectrum* (*enclosed by the solid curve*) *of the matrix* A *of* (3.6), *for large* N *and small* ε. (*The figure looks approximately the same for any* N *and* ε *satisfying, say,* $e^{-N/50} < \varepsilon < 1/50$. *See* [26].)

Like (2.2), this is a matrix whose effective spectrum is quite different from its spectrum. In this case $\Lambda_{\text{practical}}$ is the region of the complex plane enclosed by $f(S)$, where S is the unit circle and

$$f(z) = -z^{-1} + 1 + z + z^2 + z^3 \tag{3.7}$$

for large N. Figure 3.1 shows Λ and $\Lambda_{\text{practical}}$ for this matrix, assuming N is close to ∞, and Fig. 3.2 shows the lemniscates L_τ computed by GMRES at steps $n = 2, 3, 6$, and 20

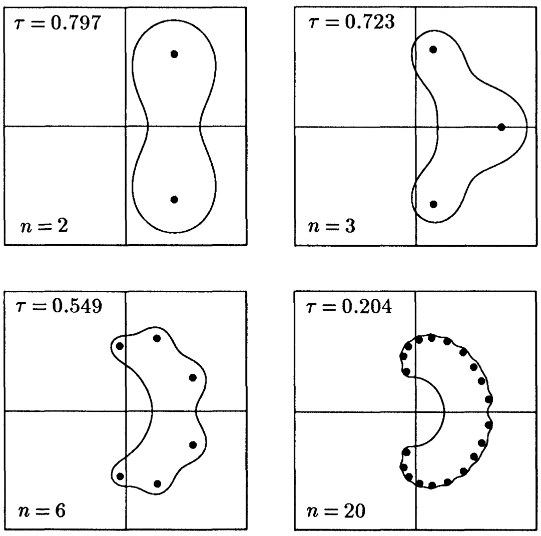


FIG. 3.2. GMRES *zeros and lemniscate for the same matrix at steps* $n = 2, 3, 6, 20$, *with* $N = 200$. *Each square represents the domain* $[-5, 5] \times [-5, 5]$. *The lemniscates approximate the pseudospectra, not the spectrum.*

for the case $N = 200$. Clearly, GMRES has done quite a good job of locating $\Lambda_{\text{practical}}$. It neither can nor should locate the exact spectrum Λ. Note that for this example the convex hull of $\Lambda_{\text{practical}}$ encloses the origin, which means that ChebyCode and some of the other existing hybrid algorithms would fail.

To close this section, let us return to the 2×2 example (2.8) that breaks an Arnoldi hybrid algorithm. In the first step GMRES makes no progress whatever with this matrix, and the corresponding residual polynomial is $p_1(z) = 1$. Thus after one step we have

$$\text{Arnoldi: } p_1(z) = 1 - z/0,$$

$$\text{GMRES: } p_1(z) = 1 - z/\infty.$$

To speak conventionally in terms of eigenvalues, we might say that GMRES has chosen the only other possible eigenvalue estimate besides 0 that is symmetric with respect to the spectrum $\pm i$, namely, ∞; the equation analogous to (2.9) is

$$\lambda_{\text{GMRES}} = \frac{r_0^T A^T A r_0}{r_0^T A r_0} = \infty. \tag{3.8}$$

This choice has made all the difference, however, since it has led to a polynomial $p_1(z)$ that is finite rather than infinite. Brown has pointed out that this phenomenon is general: when the Arnoldi iteration divides by 0, GMRES stagnates harmlessly, and vice versa [3]. Thus, although the performance of our hybrid algorithm can certainly be disappointing, if GMRES converges slowly or if (3.4) fails to hold and some kind of restart is necessary, the finiteness of $\|p_\nu(A)\|$ implies that at least it can never break down.

We have now presented a number of arguments in support of the view that the residual polynomial $p_\nu(z)$ in a hybrid algorithm should be derived from the GMRES method rather than from Arnoldi eigenvalue estimates. This idea also appears to be supported by numerical experiments. Throughout our computations for this project we have subjected each example matrix to two versions of our program, one based on GMRES and the other on Arnoldi. Each of the two methods sometimes outperforms the other, but the Arnoldi variant usually converges more slowly, and it fails considerably more often. (Of course, a failure is not absolute; with a robust implementation it will mean a return to Phase I as described in § 7.) A few comparisons of this sort are reported in Fig. 8.8, below.

4. Construction of $p_\nu(z)$.

Our implementation of the hybrid GMRES algorithm calculates the coefficients of $p_\nu(z)$ explicitly. We have not investigated the stability of this procedure, and it may be that there are better ways to find the roots of $p_\nu(z)$, for example, by solving an eigenvalue or generalized eigenvalue problem (see the additional remarks on stability in the next section). However, computational experience indicates that the explicit approach works well in practice.

Here is how the coefficients are determined. Let K_n denote the $N \times n$ matrix of Krylov vectors

$$K_n = (r_0 \quad Ar_0 \quad \cdots \quad A^{n-1}r_0). \tag{4.1}$$

The Arnoldi/GMRES process constructs an $N \times n$ matrix of orthonormal vectors spanning the same space

$$V_n = (v_1 \quad v_2 \quad \cdots \quad v_n), \tag{4.2}$$

by applying the iterative formula

$$v_{n+1} = h_{n+1,n}^{-1}(Av_n - V_n h_n), \qquad h_n = (h_{1n}, \cdots, h_{nn})^T, \tag{4.3}$$

where the numbers h_{ij} are the elements of a Hessenberg matrix of inner products. (We use the same notation as in [29].) Since the columns of V_n and K_n span the same space for each n, we have that

$$V_n = K_n C_n$$

for some upper-triangular matrix

$$C_n = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ & \ddots & \vdots \\ & & c_{nn} \end{pmatrix}. \tag{4.4}$$

This matrix is not formed during the GMRES iteration as presented in [29], but to find $p_\nu(z)$ explicitly we will need it. The appropriate formula comes from (4.3):

$$\begin{pmatrix} c_{1,n+1} \\ \vdots \\ c_{n+1,n+1} \end{pmatrix} = h_{n+1,n}^{-1} \begin{pmatrix} 0 \\ c_{1,n} \\ \vdots \\ c_{n,n} \end{pmatrix} - h_{n+1,n}^{-1} \begin{pmatrix} C_n h_n \\ 0 \end{pmatrix}. \tag{4.5}$$

By inserting the calculation (4.5) in the GMRES iteration, we generate the elements of C_n column by column as the iteration proceeds.

Having solved a Hessenberg least-squares problem at step $n = \nu$, GMRES produces an iterate x_ν of the form

$$x_\nu = x_0 + V_\nu y \tag{4.6}$$

for some vector y of dimension ν. Since $V_\nu y = K_\nu C_\nu y$, it follows that the vector $C_\nu y$ contains the coefficients of the polynomial $q_{\nu-1}(z)$ of (1.2):

$$C_\nu y = (\alpha_0, \cdots, \alpha_{\nu-1})^T, \qquad q_{\nu-1}(z) = \alpha_0 + \alpha_1 z + \cdots + \alpha_{\nu-1} z^{\nu-1}. \tag{4.7}$$

Since $p_\nu(z) = 1 - z q_{\nu-1}(z)$, this gives us the coefficients of $p_\nu(z)$ as well.

5. Richardson iteration for Phase II.

5. Richardson iteration for Phase II. Phase I is complete and we have determined the polynomials $q_{\nu-1}(z)$ and $p_\nu(z)$ implicit in the GMRES iteration. We now face the question of how best to re-apply these polynomials for the further iterations of Phase II.

As mentioned in the Introduction, many ideas have been advanced for this phase of a hybrid algorithm, of which one of the simplest is the Horner iteration of Elman and Streit [8]. From (1.2) and (4.7) we have that

$$x_n - x_0 = q_{n-1}(A) r_0, \qquad q_{n-1}(z) = \alpha_0 + \alpha_1 z + \cdots + \alpha_{n-1} z^{n-1}, \tag{5.1}$$

and therefore $x_n - x_0$ is the final result w of the loop

(5.2)
```
$w := \alpha_{n-1} r_0$
For $i := 1$ to $n-1$
    $w := Aw + \alpha_{n-i-1} r_0$.
```

In our experiments this method has worked quite well. So has a related and even simpler method in which one forms $q_{\nu-1}(A) r_0$ as a student would do who had never heard of Horner's rule—for the familiar factor-of-2 advantage of the Horner formula vanishes when we are dealing with matrices rather than scalars. The disadvantage of such approaches is that the intermediate steps may correspond to residuals so large that information may be lost due to rounding errors, though this has not troubled us in practice.

The alternative we have preferred is to factor $p_\nu(z)$ numerically,

$$p_\nu(z)=\prod_{i=1}^{\nu}(1-z/\zeta_i), \tag{5.3}$$

and then carry out a first-order Richardson iteration[4] along the lines of Smolarski and Saylor [34], [35]:

(5.4)
```
For $j := 1$ to $\nu$
    $x_j := x_{j-1} + r_{j-1}/\zeta_j$.
```

(One could also formulate the calculation in terms of $q_{\nu-1}(z)$ rather than $p_\nu(z)$ by means of the "grand leap" iteration described in [31], a method that can be slightly more efficient than (5.4).) The reader may object that finding the roots of a polynomial is an ill-conditioned problem, so that incorporating a root-finding step in a hybrid algorithm is likely to make the algorithm unstable. Though we have not yet analyzed the matter fully, we believe that this concern is about half justified. On the one hand, ill-conditioning in the rootfinding problem per se is probably not important, for the success of our algorithm ultimately depends on the size of $p_\nu(z)$ in the complex plane, not the locations of its roots. On the other hand, the size of $p_\nu(z)$ is itself an ill-conditioned function of its coefficients in general. Thus there is a stability issue, but it lies not in the rootfinding but in the representation of $p_\nu(z)$ by its coefficients in the basis of monomials, as alluded to at the beginning of § 4. The ideal hybrid algorithm might begin by constructing a more stable basis in which to represent $p_\nu(z)$. We do not know how worthwhile this extra complication would be in practice.

There is still another question of stability to be addressed. The factorization (5.3) offers a choice of the order in which to label the roots ζ_j, and as discussed first by Young and more recently by Anderssen and Golub [1], Fischer and Reichel [9], [25], and Tal-Ezer [37], this ordering is important for stability. The reason is that although the final result $p_\nu(A)$ will be small in exact arithmetic, floating-point errors may destroy this property unless (approximately speaking) the intermediate products $p_j(A)$ also are reasonably small. This issue is not academic; the factors at stake are potentially enormous.

How can an ordering of the roots $\{\zeta_j\}$ be efficiently selected to ensure that the intermediate products $p_j(A)$ are small? Our choice has been the *weighted Leja ordering* described in [25], which is defined by the condition

$$|\zeta_j|\prod_{i=1}^{j-1}|\zeta_j-\zeta_i|=\max_{j\le l\le\nu}|\zeta_l|\prod_{i=1}^{j-1}|\zeta_l-\zeta_i|,\qquad j=1,2,\cdots,\nu-1, \tag{5.5}$$

assuming the points ζ_j are distinct. (At the first step (5.5) reduces to the condition $|\zeta_1| = \max_{1\le l\le\nu}|\zeta_l|$.) The idea behind this ordering is that it tends to approximately *equidistribute* the points ζ_j in the sense of potential theory. The Leja ordering is easy to calculate, and in the examples we have looked at, it performs dramatically better than more elementary alternatives.

The Richardson iteration with Leja ordering also has the appealing property that since the polynomials $p_n(A)$ tend to decrease steadily in norm, the Phase II iteration can be meaningfully stopped at any point rather than just at the end of a cycle of ν steps.

[4] When A is real, the introduction of complex arithmetic by a complete factorization of $p_\nu(z)$ is unnecessary. One can factor it instead into linear and quadratic terms with real coefficients and obtain a Richardson iteration with steps of both first and second order. See [22], [31], or [34] for details.

It is not hard to argue that (5.5) cannot be exactly the right ordering condition to impose in all cases. For example, this algorithm has a sensitivity to multiple points ζ_j that is unnatural and that in contrived examples may cause instability. A more perfect, if more complicated, ordering algorithm might involve the minimization of an appropriate Leja product not just over the points $\zeta_j, \cdots, \zeta_\nu$ but over some approximation to the lemniscate L_τ. Nevertheless, our experience indicates that the Leja ordering is a reliable solution to the instability problem in the great majority of cases.

6. Switching criterion; behavior of the idealized hybrid GMRES algorithm. The principal feature of our algorithm that we have not yet specified is when Phase I should be terminated—in other words, the choice of ν. Optimizing this decision is a complicated matter, for it depends on both the problem and the computing environment. For example, if matrix-vector products are far more time consuming than other operations and plenty of storage is available, then one might as well stay in Phase I forever with a "GMRES(∞)" iteration. On the other hand, if storage is so limited that only a few vectors can be retained, then GMRES(∞) is out of the question and one must switch quickly to Phase II. Considerations such as these suggest that to a certain extent users of a hybrid GMRES algorithm will inevitably have to make some of the decisions themselves if the aim is optimal performance.

More can be said, however, if we are willing to make some simplifying assumptions. In particular, let us assume that storage is unlimited, so that the only goal is to minimize the computing time. Assume further that only operations on N-vectors are significant, and define a *vector operation*, our fundamental work unit, to be the cost of an "axpy" operation $ax + y$ involving a scalar a and N-vectors x and y. Finally, assume also that

$$\text{(6.1)} \qquad \text{one matrix-vector multiplication costs } \delta \text{ vector operations}$$

for some $\delta \geqq 0$. For a sparse matrix on a serial computer, δ is approximately the average number of nonzero elements per row.

These assumptions are mechanical ones, whose degree of validity depends on straightforward factors readily checked. To motivate our choice of ν, we are now going to make two further highly idealized assumptions that are in another category entirely—approximately true in some cases, perhaps, but sometimes far from true, almost never true exactly, and in any case unverifiable. First, we assume that the GMRES iteration of Phase I accelerates as it proceeds, or at worst, converges steadily:

$$\text{(6.2)} \qquad \text{Phase I:} \quad \frac{\|r_{n+m}\|}{\|r_0\|} \leqq \frac{\|r_n\|}{\|r_0\|} \frac{\|r_m\|}{\|r_0\|} \quad \text{for all } m, n \geqq 0.$$

(If $\|r_n\| / \|r_0\| = \|p_n(A)\|$ at each step n, as in (3.4), then (6.2) follows as a corollary, but in general we only have $\|r_n\| / \|r_0\| \leqq \|p_n(A)\|$.) Second, we assume that the Richardson iteration of Phase II converges steadily at exactly the same rate as in Phase I:

$$\text{(6.3)} \qquad \text{Phase II:} \quad \frac{\|r_{k\nu}\|}{\|r_0\|} = \left(\frac{\|r_\nu\|}{\|r_0\|}\right)^k \quad \text{for all } k \geqq 0.$$

Our strategy for choosing ν is motivated by the following idea:

$$\text{(6.4)} \qquad \text{Goal: equal amounts of work in Phase I and Phase II.}$$

Figure 6.1 explains the thinking behind (6.4) by illustrating the kind of convergence behavior we are hoping for under idealized circumstances. The hybrid algorithm cannot take fewer iterations than GMRES(∞), but with luck it will take nearly as few. If ν is large this will correspond to a large reduction in the total computing time.

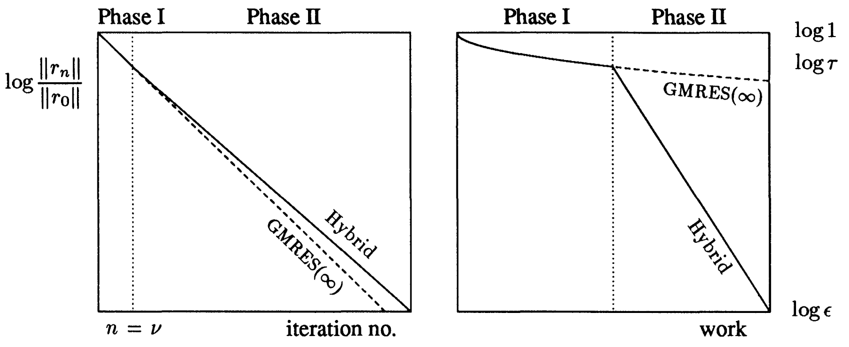


FIG. 6.1. *Convergence of the hybrid* GMRES *algorithm under idealized circumstances as a function of iteration number and computing time. The switchover step ν is determined by the condition that equal amounts of time are spent in Phase* I *and Phase* II. *The resulting factor of improvement over* GMRES(∞) *is $O(\nu^2)$.*

Now we work out the algebra required to implement (6.4). Suppose we have just completed step ν of Phase I, the residual has been reduced by the factor of

$$\frac{\|r_\nu\|}{\|r_0\|} = \tau, \tag{6.5}$$

and our desired accuracy is:

$$\text{convergence tolerance:} \quad \frac{\|r_{\text{final}}\|}{\|r_0\|} = \varepsilon \qquad (\varepsilon < \tau). \tag{6.6}$$

According to estimates in [29], the work performed so far is

$$\text{Phase I work:} \quad \nu(\nu + 3 + \delta) \text{ vector operations.} \tag{6.7}$$

On the other hand in the Richardson iteration of Phase II the work per step will be $1 + \delta$ vector operations, and by (6.3), the total number of steps to convergence will be $\nu \log \varepsilon / \log \tau$—hence in Phase II, $\nu(\log \varepsilon / \log \tau - 1)$. This implies that

$$\text{Phase II work:} \quad \nu(1 + \delta)\left(\frac{\log \varepsilon}{\log \tau} - 1\right) \text{ vector operations.} \tag{6.8}$$

The condition (6.4) can be realized by equating the right-hand sides of (6.7) and (6.8):

$$\text{switching condition:} \quad \nu + 3 + \delta = (1 + \delta)\left(\frac{\log \varepsilon}{\log \tau} - 1\right). \tag{6.9}$$

To summarize, here is how we decide when to terminate Phase I. During Phase I the left-hand side of (6.9) increases monotonically and the right-hand side decreases monotonically (because τ is decreasing). We switch to Phase II as soon as the left-hand side exceeds the right-hand side.

Besides its aesthetic appeal, the condition (6.4) has some more solid justification. In particular, we have the following theorem.

THEOREM 1. *Suppose the assumptions above hold, including* (6.2) *and* (6.3), *and let the transition from Phase* I *to Phase* II *be determined by* (6.9). *Then the hybrid algorithm converges, and no other choice of ν could have reduced the computing time by more than a factor of two.*

Proof. Increasing ν can shorten the computation only by reducing the length of Phase II, and since Phase II consumes only half of the computing time, the improvement can be at most a factor of two. On the other hand decreasing ν would mean entering Phase II with an inferior polynomial $p_\nu(z)$ and further to travel with it, by (6.2) and (6.3), so that the work in Phase II would have to increase. Since that work is already half of the total, the maximum possible improvement is again a factor of two. □

So long as the assumptions (6.2) and (6.3) hold, the reasoning above shows that our strategy for choosing ν is actually optimal in the sense that any other choice might lead to a penalty of a factor greater than two.

7. Practical safeguards.

In practice, of course, our idealized assumptions do not always hold. They may fail in several ways, and one of these is particularly important: equation (3.4) may fail, leaving us with

$$\tau = \frac{\|r_\nu\|}{\|r_0\|} \ll \|p_\nu(A)\|. \tag{7.1}$$

In such circumstances (6.3) will be far from satisfied, and the Richardson iteration of Phase II may converge much more slowly than expected or may not converge at all. The reason why (7.1) may occur is that the GMRES algorithm depends upon the particular initial residual r_0 and, consequently, the coefficients of $p_\nu(z)$ are affected by which components happen to be well represented in that vector.

There are several ways in which one might modify the hybrid algorithm to try to minimize the risk of occurrence of (7.1). For example, one might impose a threshold value of τ—insist that switchover to Phase II not take place until $\|r_n\|$ has been reduced by a factor of at least, say, 2. Or one could monitor the details of convergence in Phase I more carefully than we have proposed, forbidding switchover until some evidence has accumulated that the rate of convergence is steady. Another, more expensive, idea would be to apply Phase I to two or more independent vectors r_0 in parallel—"block GMRES." This would lead to a more reliable polynomial $p_\nu(z)$, though the extra work would be partly wasted since the residual r_n of actual interest would not be reduced. For problems with multiple right-hand sides, however, such an idea would be natural.

But there is a more fundamental implication of (7.1), and that is that any robust hybrid iterative code must include safeguards for coping with failure. If the convergence of the Phase II iteration proves unsatisfactory, there are various actions that may be taken. The simplest might be to restart the hybrid algorithm entirely from scratch from the current best available solution x_n. This would mean throwing away the information obtained in the GMRES steps already carried out, but if $p_\nu(z)$ has performed disappointingly, one might argue that that information is unreliable anyway.

The approach we have used instead is to return to the original GMRES iteration of Phase I and resume that iteration where it was interrupted. Returning to Phase I in this way is an easy matter if one has retained the necessary vectors in storage. Once a new polynomial $p_{\nu'}(z)$ is obtained that is deemed to be substantially better than the old one, we cycle back again to Phase II. To be precise, here is our current scheme, whose effects in one example can be seen in Fig. 8.6 below:

1. If any cycle of ν steps of Phase II reduces $\|r_n\|$ by a factor less than $\sqrt{\tau}$—that is, if the convergence is more than twice as slow as expected—return to Phase I.
2. Carry out additional GMRES steps $\nu + 1, \nu + 2, \cdots, \nu'$ of Phase I until the total computing time in Phase I has doubled, and calculate a new polynomial $p_{\nu'}(z)$.
3. Begin a new Phase II iteration with the new polynomial $p_{\nu'}(z)$, starting from the previous best value x_n, which will come either from the previous Phase II if the convergence there was slow but positive, or from the new Phase I if there was actual divergence in the previous Phase II.

Since this algorithm reverts to Phase I whenever the convergence of Phase II is going badly, it can never do much worse than GMRES(∞), as stated in the following theorem.

THEOREM 2. *The hybrid algorithm with the safeguards described above always converges and never requires more than three times as much computer time as* GMRES(∞).

Sketch of proof. The factor of 3 is attained if the first Phase II computation proceeds twice as slowly as expected in a case where GMRES(∞) would have converged to the desired precision at step $\nu + 1$. Careful consideration of the details of the algorithm, which we shall omit, shows that further cycling between Phase II and Phase I never leads to a factor greater than 3. □

Of course we generally expect convergence much faster than for GMRES. We remind the reader that Theorem 2 depends on our assumption that storage is not limited, so that the hybrid algorithm can be implemented as described. It also ignores rounding errors.

The details of the safeguarding procedure proposed above are arbitrary. There are many other ways to make a hybrid scheme robust, and we hope to have more to say on the subject in the future.

8. Numerical experiments.

Three sorts of problems are chosen most often for testing numerical algorithms: realistic, artificial, and random. Realistic test problems have the advantage that they are tied directly to applications and thus, in a sense, are most reliable. Artificial problems have the advantage that they can be made cleaner and more extreme in their behavior, so that they provide more insight into fundamentals. As for random problems, they also have advantages in some contexts, but not here, for none of the known nonsymmetric matrix iterations beat the $O(N^3)$ (serial) performance of direct methods for random matrices [20]. In other words, iterative methods are useful only for matrices with special properties, which they typically acquire through preconditioning.

In this section we apply our hybrid algorithm to some test problems of the artificial kind and illustrate some of its good and bad properties in the process. We hope to investigate more realistic problems in the future.

Each of our experiments compares four algorithms:

1. Hybrid GMRES (solid curves),
2. Restarted GMRES(ν) (solid curves),
3. CGN (dashed curves),
4. CGS (dots).[5]

So far as we know, these are the best matrix iterations available[6] [20]. To keep the comparison simple, our restarted algorithm is GMRES(ν), where ν is the same switchover step number determined adaptively by the hybrid algorithm. Thus our restarted and hybrid GMRES iterations are identical for the first ν steps, and from that point on the hybrid algorithm re-applies the same residual polynomial $p_\nu(z)$ cyclically, while the restarted algorithm finds a succession of new optimal polynomials of degree ν. Except in Fig. 8.6, all of the hybrid results shown come from the idealized algorithm described in § 6, with none of the safeguards mentioned in § 7.

In each experiment the dimension is $N = 1000$ (except as noted), the convergence tolerance is $\varepsilon = 10^{-5}$, and the right-hand side b and the initial guess x_0 are random real vectors with independent normally distributed elements.

[5] CGS convergence curves are often so erratic that they obscure the rest of the plot. To avoid this clutter without suppressing convergence rates that are often very impressive, we have settled for a single dot representing the residual at the end of each CGS iteration.

[6] See the further remarks about CGS in the Conclusions.

For each example we present three plots. The first plot shows $\log_{10} \|r_n\|$ as a function of the iteration number n. By this measure the hybrid algorithm can do no better than GMRES(ν), whereas CGS and CGN may do better or worse depending on the matrix. The second plot shows $\log_{10} \|r_n\|$ as a function of work measured by vector operations, defined in § 6. By this measure the hybrid algorithm may outperform restarted GMRES by a factor as high as $O(\nu)$. The third plot shows the roots of $p_\nu(z)$ in the complex plane together with the associated GMRES lemniscate L_τ and the lemniscate L_1 passing through the origin. As mentioned in § 3, L_τ gives an indication of the spectrum or τ-pseudospectrum of A.

Example 1. Our first and simplest example, shown in Fig. 8.1, is the triangular Toeplitz matrix (2.2). This is an example where the hybrid algorithm outperforms GMRES(ν) very cleanly. Plotted against the step number, the GMRES(ν) residual converges smoothly and linearly and the hybrid algorithm lags a little behind. Plotted against work, however, that linear convergence curve becomes scalloped, a common phenomenon for restarted GMRES which reflects the fact that later cycles tend to waste time redetermining information that was already obtained in earlier cycles. The hybrid algorithm now does much better, achieving rapid and steady convergence after the point of switchover. In fact, the figure matches the idealized curves of Fig. 6.1 remarkably well.

A comparison of Figs. 2.1 and 8.1 reveals that GMRES has done a good job of locating the τ-pseudospectrum of A.

In this example the hybrid algorithm is the fastest of the four algorithms asymptotically and is roughly tied with CGN for the specified tolerance $\varepsilon = 10^{-5}$. CGS converges erratically and somewhat more slowly. GMRES(ν) converges much more slowly.

Example 2. A similar but somewhat more complicated Toeplitz example is the Grcar matrix (3.6) (Fig. 8.2). As mentioned above, this is a case where ChebyCode and some of the other hybrid algorithms would fail since the pseudospectrum does not lie in a half-plane. Again the hybrid GMRES algorithm substantially outperforms GMRES(ν). CGS does about equally well. CGN does much better, however, because this is a matrix whose singular values (smoothly distributed in the interval [0.89, 3.24]) are much better behaved than its eigenvalues and pseudo-eigenvalues (encircling the origin).

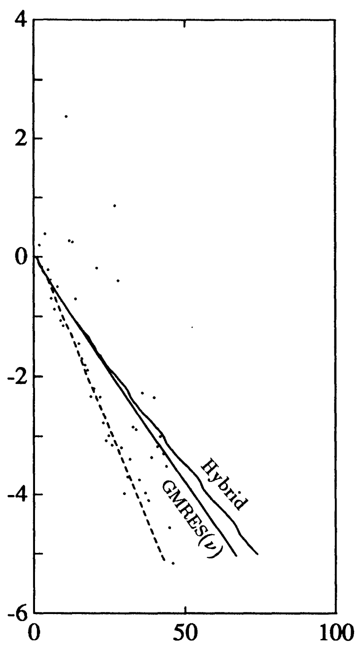
Example 3. For an example in which CGN does poorly, consider the tridiagonal Toeplitz matrix (Fig. 8.3)

$$A = \begin{pmatrix} 5.1 & 3 & & & \\ 2 & 5.1 & 3 & & \\ & 2 & 5.1 & 3 & \\ & & 2 & 5.1 & 3 \\ & & & 2 & 5.1 \end{pmatrix} \quad (1000 \times 1000). \tag{8.1}$$

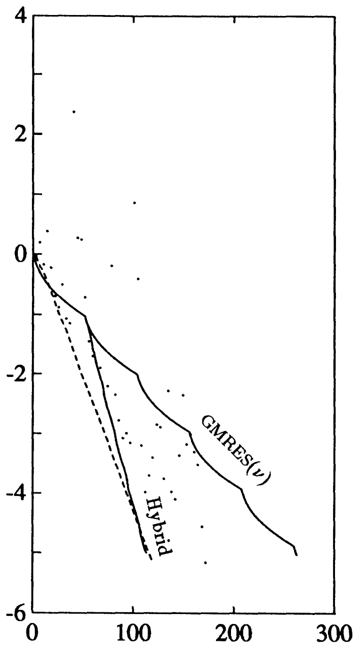
The symbol of this matrix is $f(z) = 2z^{-1} + 5.1 + 3z$, which maps the unit circle into an ellipse whose intersection with the real axis is [0.1, 10.1]. Consequently the condition number is $\kappa \approx 101$ for large N, and since the spectrum and pseudospectra do not wrap around the origin, the Krylov subspace iterations do relatively well.

In this example the convergence of the Richardson iteration of Phase II is disappointing; (6.3) does not hold very closely. Nevertheless, the plot of $\|r_n\|$ against work reveals that the hybrid iteration is the fastest. The GMRES lemniscate closely matches the elliptical pseudospectrum.

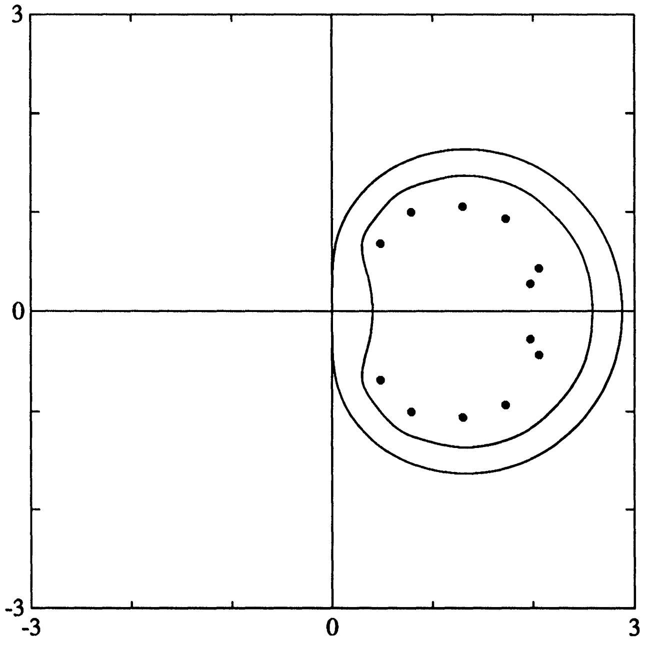
Example 4. Our fourth example is bidiagonal but not Toeplitz (Fig. 8.4). On the diagonal this matrix has the elements $.5 - \omega^1, \cdots, .5 - \omega^{1000}$, where $\omega = -.5 + i\sqrt{3}/2$ is a cube root of unity. The superdiagonal contains uniformly distributed random



(a) $\log_{10} \|r_n\|$ *vs. n*

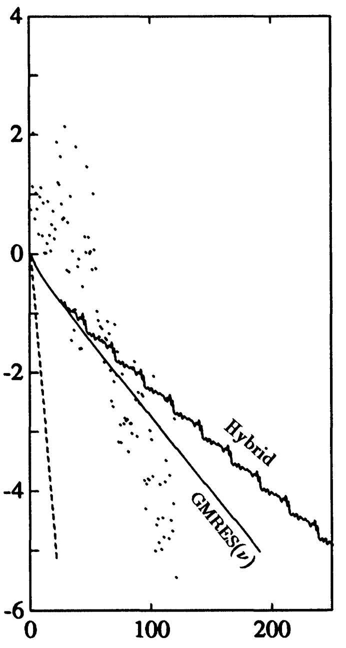


(b) $\log_{10} \|r_n\|$ *vs. work*

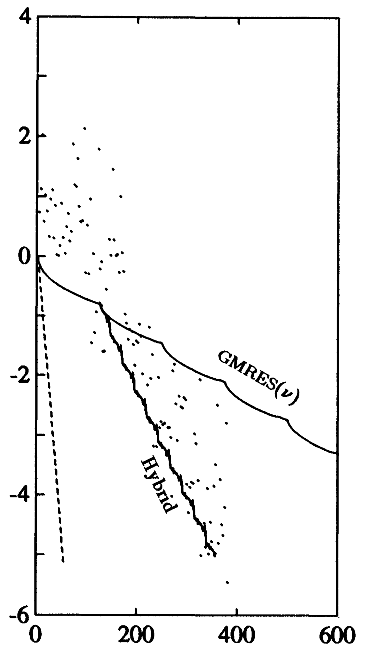


(c) GMRES *lemniscate at step* $\nu = 12$

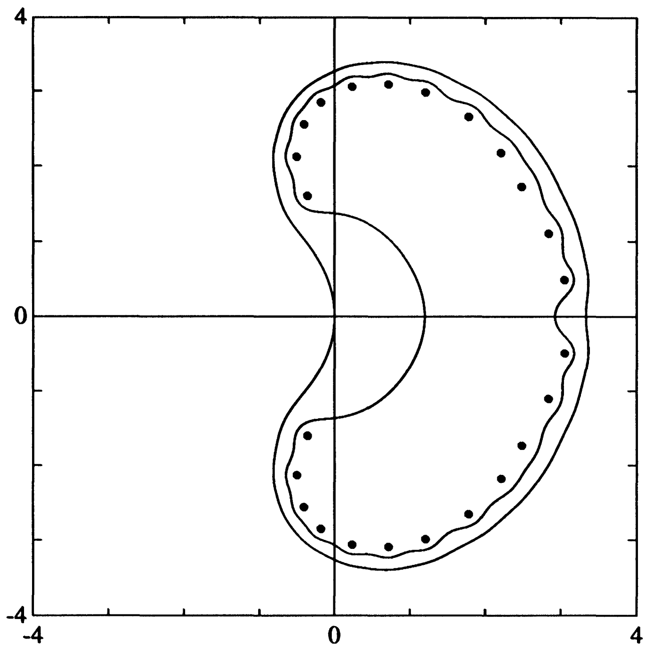
FIG. 8.1. *Example* 1: *the Toeplitz matrix* (2.2). *The* CGN *and hybrid* GMRES *algorithms are the winners. For this and the subsequent examples, the dimension is* $N = 1000$, *except as noted.*



(a) $\log_{10} \|r_n\|$ *vs. n*

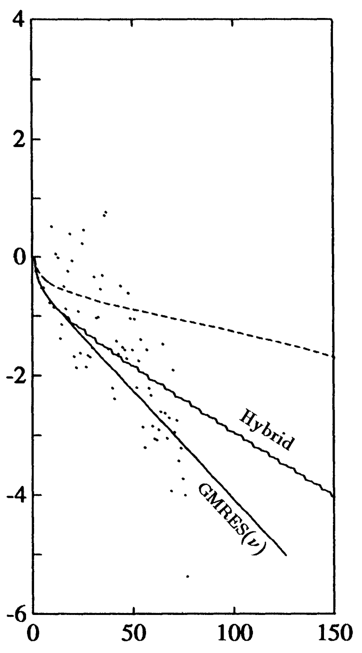


(b) $\log_{10} \|r_n\|$ *vs. work*

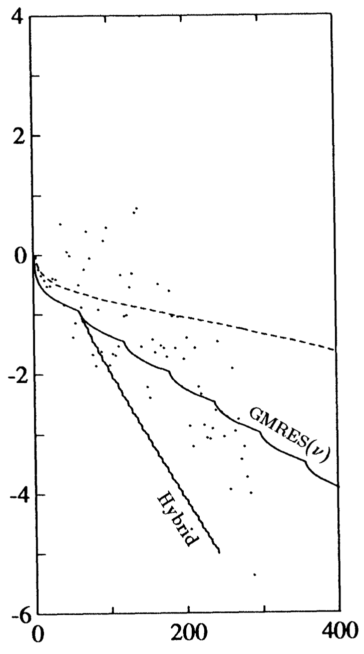


(c) GMRES *lemniscate at step* ν = 24

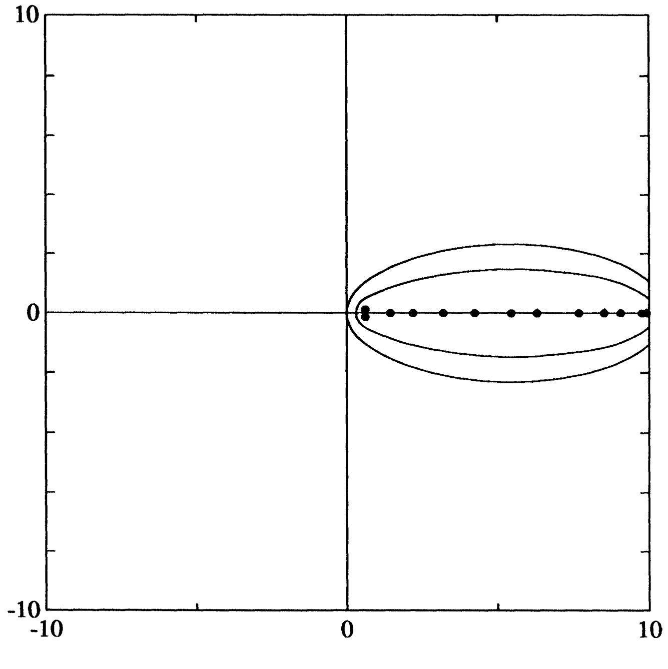
FIG. 8.2. *Example* 2: *Grcar's Toeplitz matrix* (3.6). *The hybrid* GMRES *algorithm again beats* GMRES(ν), *but* CGN *does much better.*



(a) $\log_{10} \|r_n\|$ *vs. n*

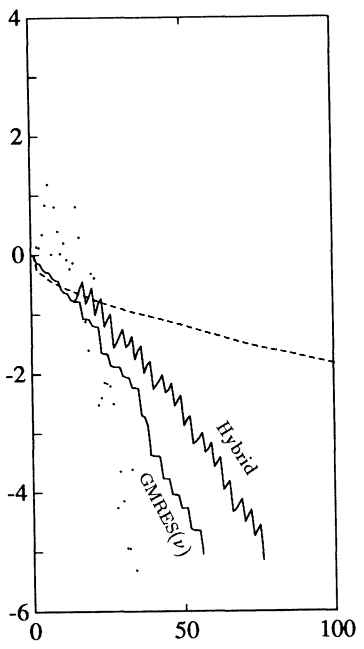


(b) $\log_{10} \|r_n\|$ *vs. work*

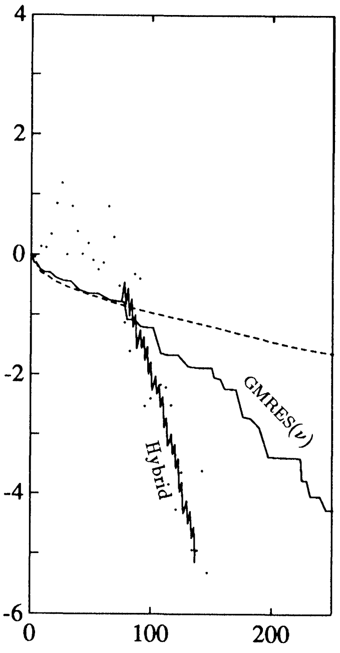


(c) GMRES *lemniscate at step* $\nu = 13$

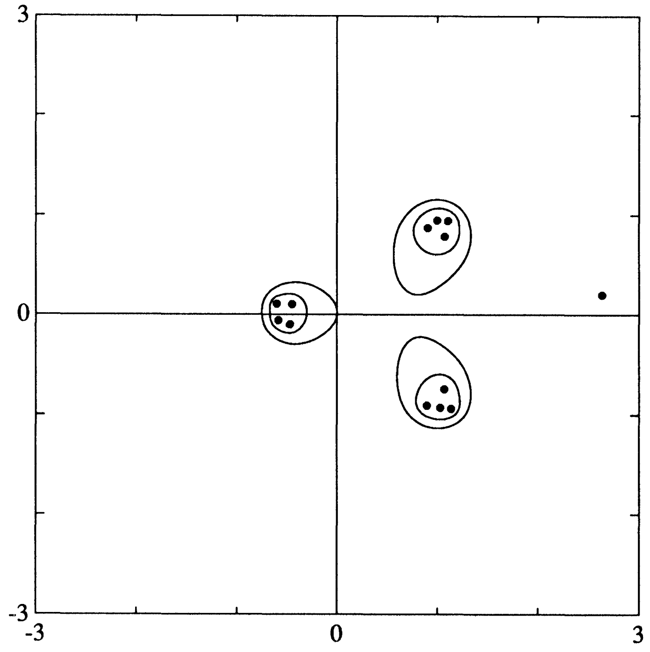
FIG. 8.3. *Example* 3: *the tridiagonal Toeplitz matrix* (8.1). *This matrix has condition number* $\kappa \approx 101$, *and* CGN *converges much more slowly than the other iterations. The hybrid* GMRES *algorithm is the winner.*



(a) $\log_{10} \|r_n\|$ *vs. n*



(b) $\log_{10} \|r_n\|$ *vs. work*



(c) GMRES *lemniscate at step* $\nu = 13$

FIG. 8.4. *Example* 4: *a bidiagonal matrix with three distinct eigenvalues. The* CGS *and hybrid* GMRES *algorithms are the most efficient.*

numbers in the interval [0, 1.5]. This is a matrix whose spectrum consists of just three points but whose pseudospectra are larger domains surrounding those points. The singular values are not very tightly clustered, and we find that CGN converges more slowly than any of the other algorithms. Among the three Krylov subspace iterations the hybrid GMRES iteration does best.

Figure 8.4(c) reveals an outlying root of $p_\nu(z)$ that is worth a comment. Since the pseudospectra have approximate three-fold symmetry but ν $(=13)$ is not divisible by three, it is not surprising that one of the linear factors of the residual polynomial should be nearly useless (compare Fig. 2.2). The GMRES lemniscate L_τ contains a very small lobe near the outlying root (too small to be apparent in the picture), and thus this example illustrates that the connection of L_τ with a pseudospectrum of A is not perfect. The outlying root does no harm to the hybrid iteration, however.

Example 5. Finally, we give an example in which the hybrid GMRES algorithm performs poorly, at least in its idealized form described in § 6. Let A be a diagonal matrix of dimension $N = 1001$ whose diagonal entries are complex numbers lying on the unit semicircle in the right half-plane. Rather than a uniform distribution of points along the semicircle with respect to arc length, we take a uniform distribution with respect to the imaginary coordinate,

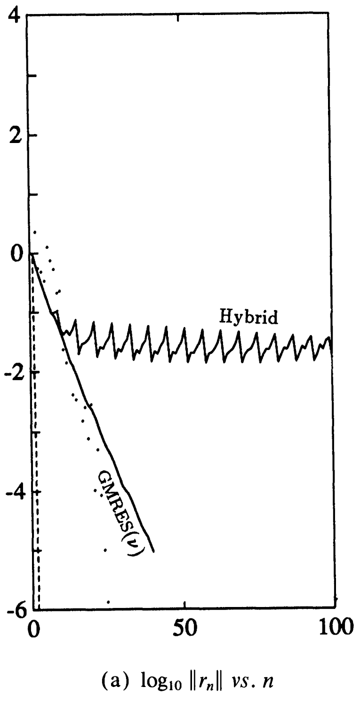
$$a_{jj} = e^{i\theta_j}, \quad \theta_j = \sin^{-1}((j-501)/500), \quad 1 \leqq j \leqq 1001.$$

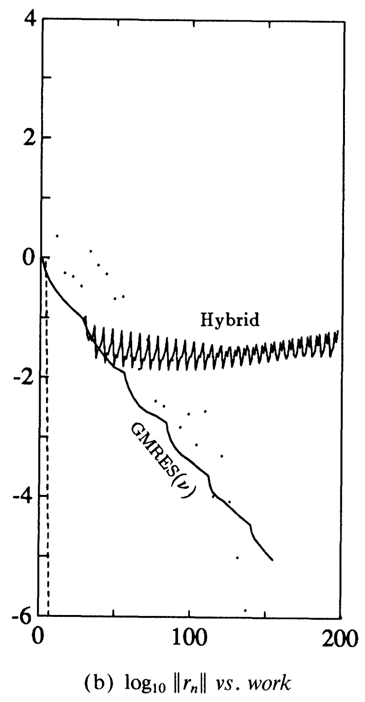
These points are sparsely located near $\pm i$, and as a result, for most initial residuals r_0, GMRES can reduce the residual significantly without going to the considerable trouble of making $|p_\nu(z)|$ substantially smaller than 1 near $z = \pm i$. This is exactly what is revealed in Fig. 8.5. Assumption (6.3) does not hold closely, and we end up with a Phase II iteration that makes little progress. GMRES(ν) beats the hybrid algorithm by a large factor, and CGS does even better. Since the singular values are all equal to 1, CGN converges in one step.

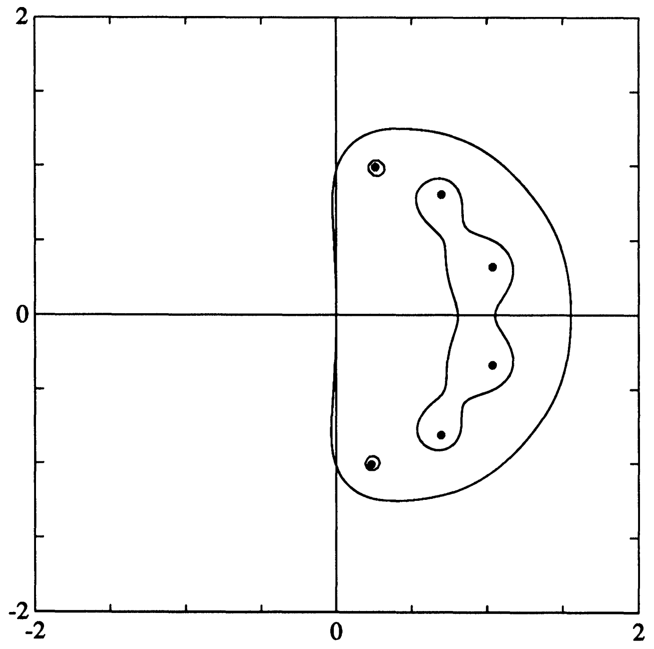
These observations, and Fig. 8.5, pertain to the idealized hybrid algorithm with none of the safeguards mentioned in § 7. In practice, of course, one would never permit so many iterations to be wasted in Phase II before returning to the GMRES iteration to get better information about A. In Fig. 8.6, we do this. The same example is run with the safeguarded hybrid algorithm described in § 7 and the convergence becomes acceptable. Note the plateau in Fig. 8.6(b), revealing a return to Phase I that generates an improved residual polynomial $p_{\nu'}(z)$ without reducing the best available residual.

This example is worth dwelling on because it reveals how important the quality of the information in r_0 is to achieving rapid convergence in Phase II. To put it succinctly, for hybrid iterative algorithms, *multiplicities matter*—even if the matrix is normal. Eigenvalues of higher multiplicities correspond to larger eigenspaces, so they tend to be better represented in a random initial vector, which increases their influence on $p_\nu(z)$. To demonstrate this, Fig. 8.7 repeats the computation of Fig. 8.5 for a new matrix, which is exactly the same as before, except that the multiplicities of the end eigenvalues $\pm i$ have been increased from 1 to 101. Thus the dimension of A is now 1201. The convergence of the hybrid algorithm without safeguards becomes quite rapid, and the explanation can be seen in the difference of the lemniscates L_1 in Figs. 8.5 and 8.7.

We close this section with four final examples to illustrate the difference between our hybrid GMRES algorithm, based on the residual polynomial $p_\nu(z)$ derived from GMRES, and a "hybrid Arnoldi" algorithm in which $p_\nu(z)$ is taken to be the normalized polynomial whose roots are the Arnoldi eigenvalue estimates at step ν. Figure 8.8 compares the convergence of these two algorithms for Examples 2, 3, 5, and 7 above. For Example 3 the Arnoldi variant is faster in Phase II by about 50 percent, but in our experience this

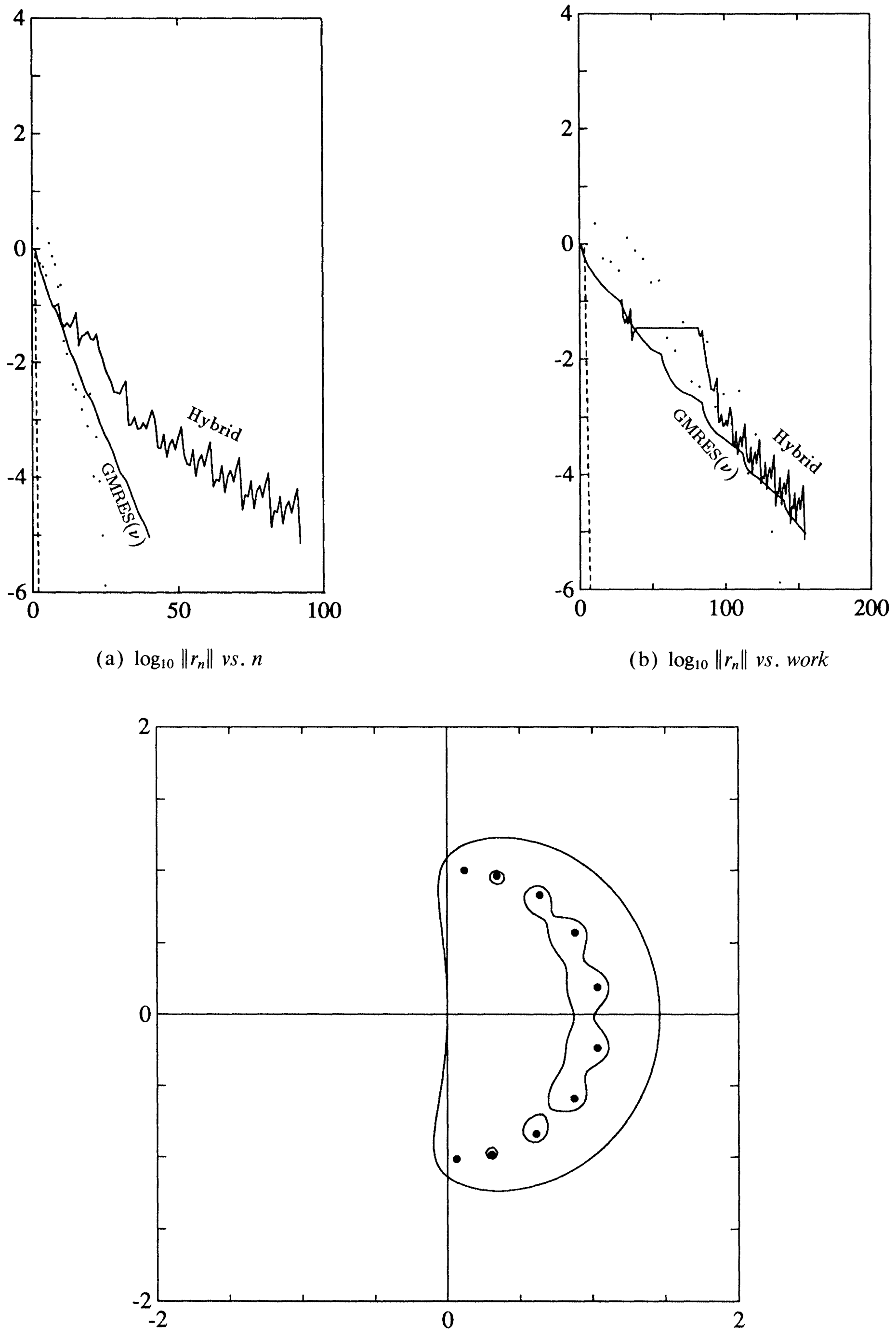






(c) GMRES *lemniscate at step* $\nu = 6$

FIG. 8.5. *Example* 5: *a diagonal matrix of dimension* $N = 1001$ *with eigenvalues on the unit semicircle in the right half-plane. The hybrid algorithm without safeguards constructs a residual polynomial* $p_\nu(z)$ *that is not much smaller than* 1 *near* $z = \pm i$, *and the convergence is very slow.*

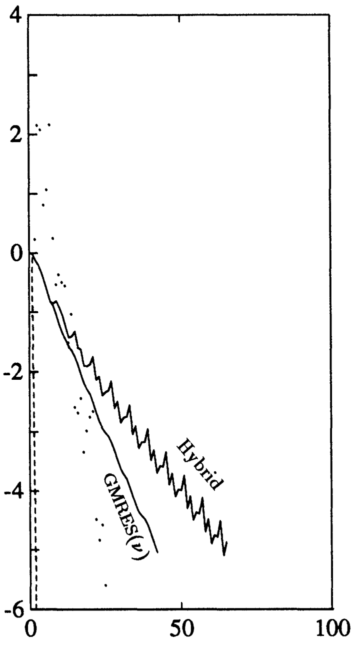


(a) $\log_{10} \|r_n\|$ *vs. n*

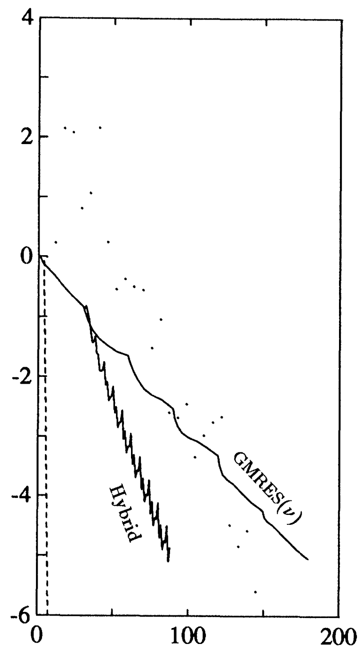
(b) $\log_{10} \|r_n\|$ *vs. work*

(c) GMRES *lemniscate at step* $\nu' = 10$

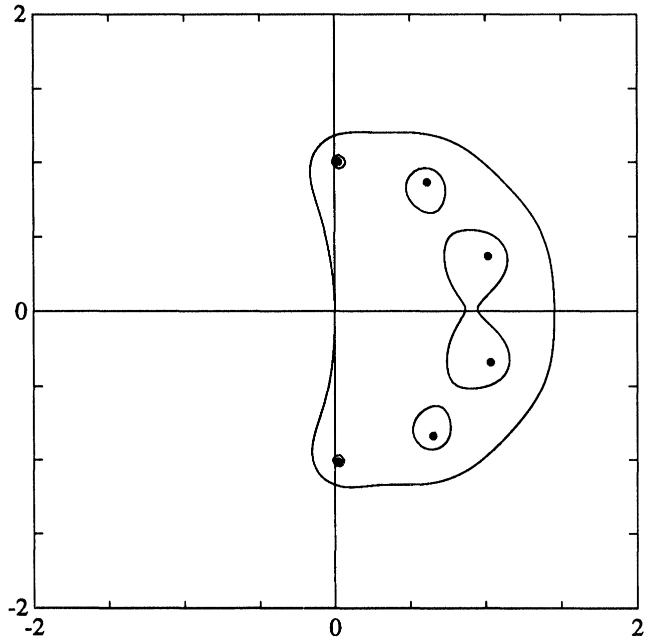
FIG. 8.6. *Example* 5 *again, but solved now with the safeguarded hybrid* GMRES *algorithm described in* § 7. *The algorithm returns to Phase* I *to get better information, and ends up solving the problem with reasonable efficiency.*



(a) $\log_{10}\|r_n\|$ *vs. n*



(b) $\log_{10}\|r_n\|$ *vs. work*



(c) GMRES *lemniscate at step* $\nu = 6$

FIG. 8.7. *Example* 5 *a third time, except that now, the matrix has been changed by increasing the multiplicities of the eigenvalues* $\pm i$ *from* 1 *to* 101, *so that the dimension becomes* 1201. *Now* $p_\nu(z)$ *contains better information, and the convergence of the hybrid algorithm, even without safeguards, is rapid.*

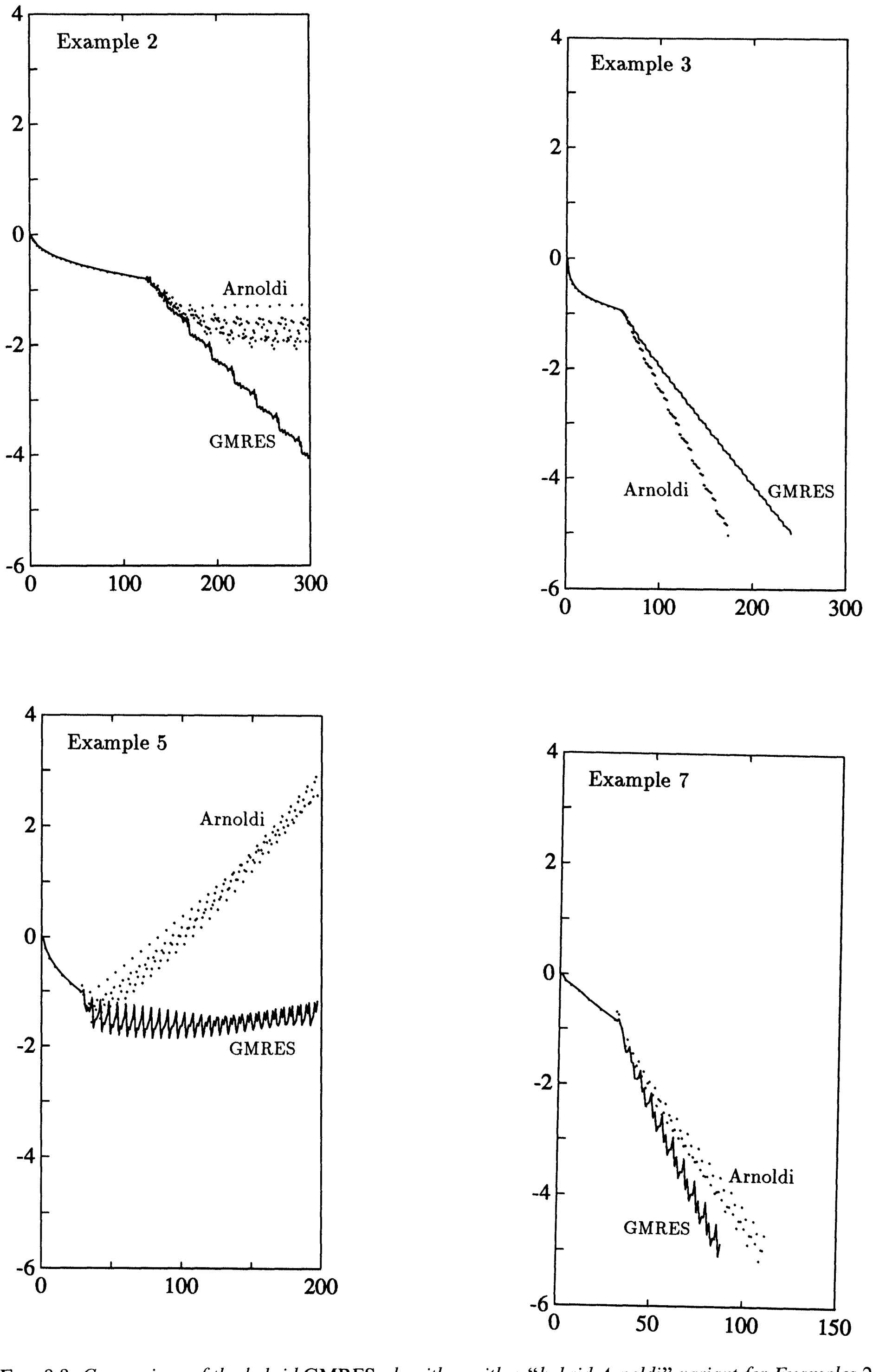


FIG. 8.8. *Comparison of the hybrid* GMRES *algorithm with a "hybrid Arnoldi" variant for Examples* 2, 3, 5, *and* 7 *above. Usually, though not always, the Arnoldi variant performs less well, for the reasons discussed in* § 2.

is not typical. More often it is slower, as in Example 7. In addition, as in Examples 2 and 5, it is not uncommon for the Arnoldi variant to stall, necessitating a return to Phase I that may not be required by the hybrid GMRES algorithm. In principle the hybrid Arnoldi algorithm can break down completely with a division by zero, as mentioned in § 2, but of course the probability of such an event is zero.

9. Conclusions. In conclusion, we would like to summarize the relationships as we see them between our hybrid GMRES algorithm and the four principal classes of competing algorithms for the iterative solution of nonsymmetric linear systems: the restarted and truncated Krylov space iterations such as GMRES(k) and ORTHOMIN(k); previous hybrid algorithms; the normal equations–conjugate gradients combination known as CGN; and the Lanczos-type algorithms such as CGS. We assume as usual that the cost of vector operations is significant enough that a "pure" Krylov space iteration such as GMRES(∞) is not competitive.

The comparison with the first two groups of alternatives turns on the question: how good is the information contained in the initial steps of an Arnoldi/GMRES iteration? The first group, the restarted and truncated algorithms such as GMRES(k), are motivated by the assumption that this information is not reliable and should be replaced regularly as the iteration proceeds even if this increases the work per step substantially. The second group, the existing hybrid algorithms summarized in our Introduction, are motivated by an opposite assumption: that initial Arnoldi/GMRES steps may produce information solid enough that it makes sense to perform further manipulations and "data compression" upon it, in particular, the solution of an approximation problem in the complex plane that typically leads to an iteration polynomial of lower degree.

Our hybrid GMRES algorithm entails an assumption intermediate between these two. It assumes that the information coming from Arnoldi/GMRES steps is too valuable to be discarded, but not so solid that further data compression is appropriate. Of course the validity of this assumption depends upon various factors, notably, the initial vector for the GMRES iteration and the choice of the switchover step ν. We believe that it is a reasonable assumption in many cases, however, and this view of the matter, combined with our numerical experiments, leads us to believe that for most problems our algorithm is faster than GMRES(k) and more robust than other hybrids.

The third comparison, with CGN, is relatively straightforward, at least in principle. The hybrid GMRES algorithm should be the winner when A is ill-conditioned, loosely speaking, or more precisely, when its squared singular values are less favorably distributed than its (pseudo-) eigenvalues in the sense described in [18].

In our opinion, the most serious competitors are the Lanczos-type algorithms such as CGS [4], whose work and storage requirements, unlike those of GMRES and ORTHOMIN, do not grow with the iteration number. These algorithms do not minimize anything, and their convergence is often quite erratic, but it is usually very fast. Most recently (since the time when this manuscript was first submitted for publication), algorithms in this class with less erratic convergence curves have been developed by Freund [10] and van der Vorst [41]. Examples can be devised for which either CGS or hybrid GMRES is superior. We hope that a fuller understanding of the comparison between these two classes of iterative methods will come with further analysis, experiments, and algorithmic development.

Acknowledgments. For advice on many topics we are grateful to Howard Elman, Martin Gutknecht, Wayne Joubert, Tom Manteuffel, Youcef Saad, Paul Saylor, and Richard Varga. For making the computations easy we are grateful to Matlab.

REFERENCES

[1] R. S. ANDERSSEN AND G. H. GOLUB, *Richardson's non-stationary matrix iterative procedure*, Report STAN-CS-72-304, Department of Computer Science, Stanford University, Stanford, CA 1972.

[2] S. F. ASHBY, *CHEBYCODE: A FORTRAN implementation of Manteuffel's adaptive Chebyshev algorithm*, Tech. Report 1203, Department of Computer Science, University of Illinois, Urbana, IL, 1985.

[3] P. BROWN, *A theoretical comparison of the Arnoldi and* GMRES *algorithms*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 58–78.

[4] A. EDELMAN, Personal communication, 1990.

[5] M. EIERMANN, *On semiiterative methods generated by Faber polynomials*, Numer. Math., 56 (1989), pp. 139–156.

[6] M. EIERMANN, W. NIETHAMMER, AND R. S. VARGA, *A study of semiiterative methods for nonsymmetric systems of linear equations*, Numer. Math., 47 (1985), pp. 505–533.

[7] H. C. ELMAN, Y. SAAD, AND P. E. SAYLOR, *A hybrid Chebyshev Krylov subspace algorithm for solving nonsymmetric systems of linear equations*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 840–855.

[8] H. ELMAN AND R. STREIT, *Polynomial iteration for nonsymmetric indefinite linear systems*, in Numerical Analysis, Lecture Notes in Mathematics No. 1230, J. P. Hennert, ed., Springer-Verlag, Berlin, New York, 1986.

[9] B. FISCHER AND L. REICHEL, *A stable Richardson iteration method for complex linear systems*, Numer. Math., 54 (1988), pp. 225–242.

[10] R. FREUND, *Conjugate gradient type methods for linear systems with complex symmetric coefficient matrices*, SIAM J. Sci. Statist. Comp., 13 (1992), pp. 425–448.

[11] W. B. GRAGG AND L. REICHEL, *On the application of orthogonal polynomials to the iterative solution of linear systems of equations with indefinite or non-hermitian matrices*, Linear Algebra Appl., 88 (1987), pp. 349–371.

[12] J. F. GRCAR, *Operator coefficient methods for linear equations*, Report SAND89-8691, Sandia National Laboratories, 1989.

[13] M. H. GUTKNECHT, *An iterative method for solving linear equations based on minimum norm Pick-Nevanlinna interpolation*, in Approximation Theory V, C. K. Chui, L. L. Schumaker, and J. D. Ward, eds., Academic Press, New York, 1986.

[14] M. R. HESTENES AND E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 409–436.

[15] O. G. JOHNSON, C. A. MICCHELLI, AND G. PAUL, *Polynomial preconditioners for conjugate gradient calculations*, SIAM J. Numer. Anal., 20 (1983), pp. 362–376.

[16] W. JOUBERT, *Iterative methods for the solution of nonsymmetric systems of linear equations*, Report CNA-242, Center for Numerical Analysis, University of Texas, Austin, Texas, 1990.

[17] X. LI, *An adaptive method for solving nonsymmetric linear systems involving application of* SCPACK, J. Comp. Appl. Math., to appear.

[18] T. A. MANTEUFFEL, *Adaptive procedure for estimating parameters for the nonsymmetric Tchebychev iteration*, Numer. Math., 31 (1978), pp. 183–208.

[19] ———, Personal communication, 1989.

[20] N. M. NACHTIGAL, S. C. REDDY, AND L. N. TREFETHEN, *How fast are nonsymmetric matrix iterations?*, SIAM J. Matrix Anal. Appl., this issue, pp. 778–795.

[21] D. P. O'LEARY, *Yet another polynomial preconditioner for the conjugate gradient algorithm*, Linear Algebra Appl., 154–158 (1991), pp. 377–388.

[22] G. OPFER AND G. SCHOBER, *Richardson's iteration for nonsymmetric matrices*, Linear Algebra Appl., 58 (1984), pp. 343–367.

[23] S. C. REDDY AND L. N. TREFETHEN, *Lax-stability of fully discrete spectral methods via stability regions and pseudo-eigenvalues*, Comp. Meth. Appl. Mech. Engrg., 80 (1990), pp. 147–164.

[24] L. REICHEL, *Polynomials by conformal mapping for the Richardson iteration method for complex linear systems*, SIAM J. Numer. Anal., 25 (1988), pp. 1359–1368.

[25] L. REICHEL, *The application of Leja points to Richardson iteration and polynomial preconditioning*, Linear Algebra Appl., 154–156 (1991), pp. 389–414.

[26] L. REICHEL AND L. N. TREFETHEN, *Eigenvalues and pseudo-eigenvalues of Toeplitz matrices*, Linear Algebra Appl., 162–164 (1992), pp. 153–185.

[27] Y. SAAD, *Krylov subspace methods on supercomputers*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 1200–1232.

[28] Y. SAAD, *Least-squares polynomials in the complex plane and their use for solving nonsymmetric linear systems*, SIAM J. Numer. Anal., 24 (1987), pp. 155–169.

[29] Y. SAAD AND M. H. SCHULTZ, GMRES: *A generalized minimum residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[30] P. E. SAYLOR, *Use of the singular value decomposition with the Manteuffel algorithm for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 1 (1980), pp. 210–222.

[31] P. E. SAYLOR, *Leapfrog variants of iterative methods for linear algebraic equations*, J. Comp. Appl. Math., 24 (1988), pp. 169–193.

[32] P. E. SAYLOR, *An adaptive algorithm for Richardson's iteration*, in Iterative Methods for Large Linear Systems, D. R. Kincaid and L. J. Hayes, eds., Academic Press, New York, 1990.

[33] P. E. SAYLOR AND D. C. SMOLARSKI, *Implementation of an adaptive algorithm for Richardson's method*, Linear Algebra Appl., 154–156 (1991), pp. 615–646.

[34] D. C. SMOLARSKI, *Optimum semi-iterative methods for the solution of any linear algebraic system with a square matrix*, Tech. Report 1077, Department of Computer Science, University of Illinois, Urbana, IL, 1981.

[35] D. C. SMOLARSKI AND P. E. SAYLOR, *An optimum iterative method for solving any linear system with a square matrix*, BIT, 28 (1988), pp. 163–178.

[36] P. SONNEVELD, CGS, *a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.

[37] H. TAL-EZER, *Polynomial approximation of functions of matrices and applications*, J. Sci. Comput., 4 (1989), pp. 25–60.

[38] L. N. TREFETHEN, *Approximation theory and numerical linear algebra*, in Algorithms for Approximation II, J. C. Mason and M. G. Cox, eds., Chapman and Hall, London, 1990.

[39] L. N. TREFETHEN, *Pseudospectra of matrices*, in Proc. 14th Dundee Biennial Conf. on Numer. Anal., D. F. Griffiths and G. A. Watson, eds., to appear.

[40] L. N. TREFETHEN AND M. R. TRUMMER, *An instability phenomenon in spectral methods*, SIAM J. Numer. Anal., 24 (1987), pp. 1008–1023.

[41] H. A. VAN DER VORST, *Bi*-CGSTAB: *A fast and smoothly converging variant of Bi*-CG *for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644.

[42] H. E. WRIGLEY, *Accelerating the Jacobi method for solving simultaneous equations by Chebyshev extrapolation when the eigenvalues of the iteration matrix are complex*, Comput. J., 6 (1963), pp. 169–176.

[43] G. STARKE AND R. S. VARGA, *A hybrid Arnoldi-Faber iterative method for nonsymmetric systems of linear equations*, Numer. Math., to appear.