Iterative Solution of Linear Systems

Roland W. Freund *
RIACS, Mail Stop Ellis Street
NASA Ames Research Center
Moffett Field, CA 94035
E-mail: na.freund@na-net.ornl.gov

Gene H. Golub †
Computer Science Department
Stanford University
Stanford, CA 94305
E-mail: golub@sccm.stanford.edu

Noël M. Nachtigal ‡
RIACS, Mail Stop Ellis Street
NASA Ames Research Center
Moffett Field, CA 94035
E-mail: na.nachtigal@na-net.ornl.gov

Recent advances in the field of iterative methods for solving large linear systems are reviewed. The main focus is on developments in the area of conjugate gradient-type algorithms and Krylov subspace methods for non-Hermitian matrices.

CONTENTS

1 Introduction
2 Background
3 Lanczos-based Krylov subspace methods
4 Solving the normal equations is not always bad
5 Estimating spectra for hybrid methods
6 CG-type methods for complex linear systems
7 Large dense linear systems
8 Concluding remarks
References

* The work of this author was supported by DARPA via Cooperative Agreement NCC 2-387 between NASA and the Universities Space Research Association (USRA).
† The work of this author was supported in part by the National Science Foundation under Grant NSF CCR-8821078.
‡ The work of this author was supported by Cooperative Agreement NCC 2-387 between NASA and the Universities Space Research Association (USRA).

1. Introduction

One of the fundamental building blocks of numerical computing is the ability to solve linear systems

$$A\mathbf{x} = \mathbf{b}. \tag{1.1}$$

These systems arise very frequently in scientific computing, for example, from finite difference or finite element approximations to partial differential equations, as intermediate steps in computing the solution of nonlinear problems, or as subproblems in linear and nonlinear programming.

For linear systems of small size, the standard approach is to use direct methods, such as Gaussian elimination. These algorithms obtain the solution of (1.1) based on a factorization of the coefficient matrix A. However, in practice linear systems arise that can be arbitrarily large; this is particularly true when solving partial differential equations. Fortunately, the resulting systems usually have some special structure; sparsity, i.e., matrices with only few nonzero entries, is the most common case. Often, direct methods can be adapted to exploit the special structure of the matrix and then remain useful even for large linear systems. However, in many cases, especially for systems arising from three-dimensional partial differential equations, direct approaches are prohibitive both in terms of storage requirements and computing time, and then the only alternative is to use iterative algorithms.

Especially attractive are iterative methods that involve the coefficient matrix only in the form of matrix-vector products with A or A^H. Such schemes naturally exploit the special structure of large sparse linear systems. They are also well suited for the solution of certain dense large systems for which matrix-vector products can be obtained cheaply. The most powerful iterative scheme of this type is the conjugate gradient method (CG) due to Hestenes and Stiefel (1952), which is an algorithm for solving Hermitian positive definite linear systems. Although CG was introduced as early as 1952, its true potential was not appreciated until the work of Reid (1971) and Concus *et al.* (1976) in the seventies. Since then, a considerable part of the research in numerical linear algebra has been devoted to generalizations of CG to indefinite and non-Hermitian linear systems.

A straightforward extension to general non-Hermitian matrices is to apply CG to either one of the Hermitian positive definite linear systems

$$A^H A\mathbf{x} = A^H\mathbf{b}, \tag{1.2}$$

or

$$AA^H\mathbf{y} = \mathbf{b}, \quad \mathbf{x} = A^H\mathbf{y}. \tag{1.3}$$

Solving (1.2) by CG was mentioned already by Hestenes and Stiefel (1952); we will refer to this approach as CGNR. Applying CG to (1.3) was proposed by Craig (1955); we will refer to this second approach as CGNE.

Although there are special situations where CGNR or CGNE are the optimal extensions of CG, both algorithms generally converge very slowly and hence they usually are not satisfactory generalizations of CG to arbitrary non-Hermitian matrices.

Consequently, CG-type algorithms were sought that are applied to the original system (1.1), rather than (1.2) or (1.3). A number of such methods have been proposed since the mid-seventies, the most widely used of which is the generalized minimum residual algorithm (GMRES) due to Saad and Schultz (1986). While GMRES and related schemes generate at each iteration optimal approximate solutions of (1.1), their work and storage requirements per iteration grow linearly. Therefore, it becomes prohibitive to run the full version of these algorithms and restarts are necessary, which often leads to very slow convergence.

For this reason, since the late eighties, research in non-Hermitian matrix iterations has focused mainly on schemes that can be implemented with low and roughly constant work and storage requirements per iteration. A number of new algorithms with this feature have been proposed, all of which are related to the nonsymmetric Lanczos process. It is these recent developments in CG-type methods for non-Hermitian linear systems that we will emphasize in this survey.

The outline of this paper is as follows. In Section 2, we present some background material on general Krylov subspace methods, of which CG-type algorithms are a special case. We recall the outstanding properties of CG and discuss the issue of optimal extensions of CG to non-Hermitian matrices. We also review GMRES and related methods, as well as CG-like algorithms for the special case of Hermitian indefinite linear systems. Finally, we briefly discuss the basic idea of preconditioning. In Section 3, we turn to Lanczos-based iterative methods for general non-Hermitian linear systems. First, we consider the nonsymmetric Lanczos process, with particular emphasis on the possible breakdowns and potential instabilities in the classical algorithm. Then we describe recent advances in understanding these problems and overcoming them by using look-ahead techniques. Moreover, we describe the quasi-minimal residual algorithm (QMR) proposed by Freund and Nachtigal (1990), which uses the look-ahead Lanczos process to obtain quasi-optimal approximate solutions. Next, a survey of transpose-free Lanczos-based methods is given. We conclude this section with comments on other related work and some historical remarks. In Section 4, we elaborate on CGNR and CGNE and we point out situations where these approaches are optimal. The general class of Krylov subspace methods also contains parameter-dependent algorithms that, unlike CG-type schemes, require explicit information on the spectrum of the coefficient matrix. In Section 5, we discuss recent insights in obtaining appropriate spectral information for parameter-dependent Krylov subspace methods. After that, we turn to special classes of linear systems. First, in Section 6, we consider CG-type algorithms for complex symmetric and shifted Hermitian matrices. In Section 7, we review cases of dense large linear systems for which iterative algorithms are a viable alternative to direct methods. Finally, in Section 8, we make some concluding remarks.

Today, the field of iterative methods is a rich and extremely active research area, and it has become impossible to cover in a survey paper all recent advances. For example, we have not included any recent developments in preconditioning of linear systems, nor any discussion of the efficient use of iterative schemes on advanced architectures. Also, we would like to point the reader to the following earlier survey papers. Stoer (1983) reviews the state of CG-like algorithms up to the early eighties. In the paper by Axelsson (1985), the focus is on preconditioning of iterative methods. More modern iterative schemes, such as GMRES, and issues related to the implementation of Krylov subspace methods on supercomputers are treated in the survey by Saad (1989). An annotated bibliography on CG and CG-like methods covering the period up to 1976 was compiled by Golub and O'Leary (1989). Finally, readers interested in direct methods for sparse linear systems are referred to the book by Duff *et al.* (1986) and, for the efficient use of these techniques on parallel machines, to Heath *et al.* (1991).

Throughout the paper, all vectors and matrices are in general assumed to be complex. As usual, $i = \sqrt{-1}$. For any matrix $M = [\, m_{jk} \,]$, we use the following notation:

$$\begin{aligned}
\overline{M} &= [\, \overline{m_{jk}} \,] \;=\; \text{the complex conjugate of } M, \\
M^T &= [\, m_{kj} \,] \;=\; \text{the transpose of } M, \\
M^H &= \overline{M}^T \;=\; \text{the Hermitian of } M, \\
\operatorname{Re} M &= (M + \overline{M})/2 \;=\; \text{the real part of } M, \\
\operatorname{Im} M &= (M - \overline{M})/(2i) \;=\; \text{the imaginary part of } M, \\
\sigma(M) &= \text{the set of singular values of } M, \\
\sigma_{\max}(M) &= \text{the largest singular value of } M, \\
\sigma_{\min}(M) &= \text{the smallest singular value of } M, \\
\|M\|_2 &= \sigma_{\max}(M) \;=\; \text{the 2-norm of } M, \\
\kappa_2(M) &= \sigma_{\max}(M)/\sigma_{\min}(M) \\
&= \text{the 2-condition number of } M, \text{ if } M \text{ has full rank.}
\end{aligned}$$

For any vector $\mathbf{c} \in \mathbb{C}^m$ and any matrix $B \in \mathbb{C}^{m \times m}$, we denote by

$$\mathcal{K}_n(\mathbf{c}, B) = \operatorname{span}\{\mathbf{c}, B\mathbf{c}, \ldots, B^{n-1}\mathbf{c}\}$$

the nth Krylov subspace of $\mathbb{C}^m$, generated by $\mathbf{c}$ and B. Furthermore, we use the following notation:

$$\begin{aligned}
\|\mathbf{c}\|_2 &= \sqrt{\mathbf{c}^H\mathbf{c}} \;=\; \text{Euclidean norm of } \mathbf{c},\\
\|\mathbf{c}\|_B &= \sqrt{\mathbf{c}^H B\mathbf{c}}\\
&= B\text{-norm of } \mathbf{c}, \text{ if } B \text{ is Hermitian positive definite},\\
\lambda(B) &= \text{the set of eigenvalues of } B,\\
\lambda_{\max}(B) &= \text{the largest eigenvalue of } B, \text{ if } B \text{ is Hermitian},\\
\lambda_{\min}(B) &= \text{the smallest eigenvalue of } B, \text{ if } B \text{ is Hermitian}.
\end{aligned}$$

Moreover, we denote by I_n the $n \times n$ identity matrix; if the dimension n is evident from the context, we will simply write I. The symbol 0 will be used both for the number 0 and for the zero matrix; in the latter case, the dimension will always be apparent. We denote by

$$\mathcal{P}_n = \{\phi(\lambda) \equiv \sigma_0 + \sigma_1\lambda + \cdots + \sigma_n\lambda^n \;\big|\; \sigma_0, \sigma_1, \ldots, \sigma_n \in \mathbb{C}\}$$

the set of complex polynomials of degree at most n.

Throughout this paper, N denotes the dimension of the coefficient matrix A of (1.1) and $A \in \mathbb{C}^{N\times N}$ is in general non-Hermitian. In addition, unless otherwise stated, A is always assumed to be nonsingular. Moreover, we use the following notation:

$$\begin{aligned}
\mathbf{x}_0 &= \text{initial guess for the solution of (1.1)},\\
\mathbf{x}_n &= n\text{th iterate},\\
\mathbf{r}_n &= \mathbf{b} - A\mathbf{x}_n \;=\; n\text{th residual vector}.
\end{aligned}$$

If it is not apparent from the context which iterative method we are considering, quantities from different algorithms will be distinguished by superscripts, e.g., $\mathbf{x}_n^{\mathrm{CG}}$ or $\mathbf{x}_n^{\mathrm{GMRES}}$.

2. Background

In this section, we present some background material on general Krylov subspace methods.

2.1. Krylov subspace methods

Many iterative schemes for solving the linear system (1.1) belong to the class of Krylov subspace methods: they produce approximations $\mathbf{x}_n$ to $A^{-1}\mathbf{b}$ of the form

$$\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, A), \quad n = 1, 2, \ldots . \tag{2.1}$$

Here, $\mathbf{x}_0 \in \mathbb{C}^N$ is any initial guess for the solution of (1.1), $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$ is the corresponding residual vector, and $\mathcal{K}_n(\mathbf{r}_0, A)$ is the nth Krylov subspace generated by $\mathbf{r}_0$ and A. In view of

$$\mathcal{K}_n(\mathbf{r}_0, A) = \{\phi(A)\mathbf{r}_0 \;\big|\; \phi \in \mathcal{P}_{n-1}\}, \tag{2.2}$$

schemes with iterates (2.1) are also referred to as polynomial-based iterative methods. In particular, the residual vector corresponding to the nth iterate $\mathbf{x}_n$ can be expressed in terms of polynomials:

$$\mathbf{r}_n = \mathbf{b} - A\mathbf{x}_n = \psi_n(A)\mathbf{r}_0, \tag{2.3}$$

where

$$\psi_n \in \mathcal{P}_n, \quad \text{with} \quad \psi_n(0) = 1. \tag{2.4}$$

Generally, any polynomial satisfying (2.4) is called an nth residual polynomial.

As (2.3) shows, the goal in designing a Krylov subspace method is to choose at each step the polynomial ψ_n such that $\mathbf{r}_n \approx 0$ in some sense. One option is to actually minimize some norm of the residual $\mathbf{r}_n$:

$$\begin{aligned} \|\mathbf{r}_n\| &= \min_{\mathbf{x}\in\mathbf{x}_0+\mathcal{K}_n(\mathbf{r}_0,A)} \|\mathbf{b} - A\mathbf{x}\| \\ &= \min_{\psi\in\mathcal{P}_n:\psi(0)=1} \|\psi(A)\mathbf{r}_0\|. \end{aligned} \tag{2.5}$$

Here $\|\cdot\|$ is a vector norm on $\mathbb{C}^N$, which may even depend on the iteration number n (see Section 3.3). Another option is to require that the residual satisfies a Galerkin-type condition:

$$\mathbf{s}^H\mathbf{r}_n = 0 \quad \text{for all} \quad \mathbf{s} \in \mathcal{S}_n, \tag{2.6}$$

where $\mathcal{S}_n \subset \mathbb{C}^N$ is a subspace of dimension n. Note that an iterate satisfying (2.6) need not exist for each n; in contrast, the existence of iterates with (2.5) is always guaranteed. The point is that iterates with (2.5) or (2.6) can be obtained from a basis for $\mathcal{K}_n(r_0, A)$ (and a basis for $\mathcal{S}_n$ in the case of (2.6)), without requiring any a priori choice of other iteration parameters.

In contrast to parameter-free schemes based on (2.5) or (2.6), parameter-dependent Krylov subspace methods require some advance information on the spectral properties of A for the construction of ψ_n. Usually, knowledge of some compact set $\mathcal{G}$ with

$$\lambda(A) \subseteq \mathcal{G} \subset \mathbb{C}, \quad 0 \notin \mathcal{G}, \tag{2.7}$$

is needed. For example, assume that A is diagonalizable, and let U be any matrix of eigenvectors of A. For the case of the Euclidean norm, it then follows from (2.3) and (2.7) that

$$\begin{aligned} \frac{\|\mathbf{r}_n\|_2}{\|\mathbf{r}_0\|_2} &\leq \kappa_2(U) \max_{\lambda\in\lambda(A)} |\psi_n(\lambda)| \\ &\leq \kappa_2(U) \max_{\lambda\in\mathcal{G}} |\psi_n(\lambda)|. \end{aligned} \tag{2.8}$$

Ideally, one would like to choose the residual polynomial ψ_n such that the right-hand side in (2.8) is minimal, i.e.,

$$\max_{\lambda \in \mathcal{G}} |\psi_n(\lambda)| = \min_{\psi \in \mathcal{P}_n : \psi(0)=1} \max_{\lambda \in \mathcal{G}} |\psi(\lambda)|. \tag{2.9}$$

Unfortunately, the exact solution of the approximation problem (2.9) is known only for a few special cases. For example, if $\mathcal{G}$ is a real interval, then shifted and scaled Chebyshev polynomials are optimal in (2.9); the resulting algorithm is the well-known Chebyshev semi-iterative method for Hermitian positive definite matrices (see Golub and Varga, 1961). Later, Manteuffel (1977) extended the Chebyshev iteration to the class of non-Hermitian matrices for which $\mathcal{G}$ in (2.7) can be chosen as an ellipse. We remark that, in this case, Chebyshev polynomials are always nearly optimal for (2.9), but—contrary to popular belief—in general they are not the exact solutions of (2.9), as was recently shown by Fischer and Freund (1990, 1991). The solution of (2.9) is also known explicitly for complex line segments $\mathcal{G}$ that are parallel to the imaginary axis and symmetric about the real line (see Freund and Ruscheweyh, 1986); this case corresponds to shifted skew-symmetric matrices A of the form (2.14) below. In the general case however, the exact solution of (2.9) is not available and is expensive to compute numerically. Instead, one chooses polynomials that are only asymptotically optimal for (2.9). An elegant theory for semi-iterative methods of this type was developed by Eiermann *et al.* (1985).

In this survey, we will focus mainly on parameter-free algorithms with iterates characterized by (2.5) or (2.6). Parameter-dependent Krylov subspace methods will be only briefly discussed in Section 5.

2.2. CG and optimal extensions

Classical CG is a Krylov subspace method for Hermitian positive definite matrices A with two outstanding features. First, its iterates $\mathbf{x}_n$ satisfy a minimization property, namely (2.5) in the A^{-1}-norm:

$$\|\mathbf{b} - A\mathbf{x}_n\|_{A^{-1}} = \min_{\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, A)} \|\mathbf{b} - A\mathbf{x}\|_{A^{-1}}. \tag{2.10}$$

Secondly, $\mathbf{x}_n$ can be computed efficiently, based on simple three-term recurrences.

An ideal extension of CG to non-Hermitian matrices A would have similar features. However, since in general $\|\cdot\|_{A^{-1}}$ is no longer a norm, one usually replaces (2.10) with either the minimization property

$$\|\mathbf{b} - A\mathbf{x}_n\|_2 = \min_{\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, A)} \|\mathbf{b} - A\mathbf{x}\|_2, \tag{2.11}$$

or the Galerkin condition

$$\mathbf{s}^H(\mathbf{b} - A\mathbf{x}_n) = 0 \quad \text{for all} \quad \mathbf{s} \in \mathcal{K}_n(\mathbf{r}_0, A). \tag{2.12}$$

In the sequel, a Krylov subspace algorithm with iterates (2.1) defined by

(2.11) or (2.12) will be called a minimal residual (MR) method or an orthogonal residual (OR) method, respectively. We remark that (2.10) and (2.12) are equivalent for Hermitian positive definite A, and hence (2.12) is an immediate extension of (2.10). Unfortunately, for non-Hermitian A and even for Hermitian indefinite A, an iterate $\mathbf{x}_n$ with (2.12) need not exist at each step n. In contrast, there is always a unique iterate $\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, A)$ satisfying (2.11). We note that the conjugate residual algorithm (CR) due to Stiefel (1955) is a variant of CG that generates iterates characterized by (2.11) for the special case of Hermitian positive definite A.

An ideal CG-like scheme for solving non-Hermitian linear systems would then have the following features:

(i) its iterates would be characterized by the MR or OR property, and

(ii) it could be implemented based on short vector recursions, so that work and storage requirements per iteration would be low and roughly constant.

Unfortunately, it turns out that, for general matrices, the conditions (i) and (ii) can not be fulfilled simultaneously. This result is due to Faber and Manteuffel (1984, 1987) who proved the following theorem (see also Voevodin, 1983, and Joubert and Young, 1987).

Theorem 2.1 (Faber and Manteuffel, 1984 and 1987)
Except for a few anomalies, ideal CG-like methods that satisfy both requirements (i) and (ii) exist only for matrices of the special form

$$A = e^{i\theta}(T + \sigma I), \quad \text{where} \quad T = T^H, \quad \theta \in \mathbb{R}, \quad \sigma \in \mathbb{C}. \tag{2.13}$$

The class (2.13) consists of just the shifted and rotated Hermitian matrices. Note that the important subclass of real nonsymmetric matrices

$$A = I - S, \quad \text{where} \quad S = -S^T \quad \text{is real}, \tag{2.14}$$

is contained in (2.13), with $e^{i\theta} = i$, $\sigma = -i$, and $T = iS$. Concus and Golub (1976) and Widlund (1978) were the first to devise an implementation of a OR method for the family (2.14). The first MR algorithm for (2.14) was proposed by Rapoport (1978), and different implementations were given by Eisenstat *et al.* (1983) and Freund (1983). For a brief discussion of actual CG-type algorithms for the general class of complex non-Hermitian matrices (2.14), we refer the reader to Section 6.2.

Finally, we remark that ideal CG-like methods also exist for the more general family of shifted and rotated B-Hermitian matrices

$$A = e^{i\theta}(T + \sigma I), \quad \text{where} \quad TB = (TB)^H, \quad \theta \in \mathbb{R}, \quad \sigma \in \mathbb{C}. \tag{2.15}$$

Here B is a fixed given Hermitian positive definite $N \times N$ matrix (see Ashby *et al.*, 1990). However, since for any matrix A of the form (2.15),

$$A' = B^{1/2} A B^{-1/2}$$

is of the type (2.13), without loss of generality the case (2.15) can always be reduced to (2.13).

2.3. CG-type algorithms for Hermitian indefinite linear systems

The family (2.13) also contains Hermitian indefinite matrices; next we review CG-type methods for this special case.

Luenberger (1969) was the first to propose a modification of standard CG for Hermitian indefinite matrices; however, his algorithm encountered some unresolved computational difficulties. The first numerically stable schemes for Hermitian indefinite linear systems were derived by Paige and Saunders (1975). Their SYMMLQ algorithm is an implementation of the OR approach and hence the immediate generalization of classical CG. As pointed out earlier, an OR iterate $\mathbf{x}_n$ satisfying (2.12) need not exist for each n, and, in fact, SYMMLQ generates $\mathbf{x}_n$ only indirectly. Instead of the OR iterates, a second sequence of well-defined iterates $\mathbf{x}_n^L$ is updated, from which existing $\mathbf{x}_n$ can then be obtained cheaply. Paige and Saunders also proposed the MINRES algorithm, which produces iterates defined by the MR property (2.11) and thus can be viewed as an extension of CR to Hermitian indefinite matrices. SYMMLQ and MINRES both use the Hermitian Lanczos recursion to generate an orthonormal basis for the Krylov subspaces $\mathcal{K}_n(\mathbf{r}_0, A)$, and, like the latter, they can be implemented based on simple three-term recurrences. We would like to stress that the work of Paige and Saunders was truly pioneering, in that they were the first to extend CG beyond the class of Hermitian positive definite matrices in a numerically stable manner.

SYMMLQ is also closely connected with an earlier algorithm due to Fridman (1963), which generates iterates $\mathbf{x}_n^{\mathrm{ME}} \in \mathbf{x}_0 + \mathcal{K}_n(A\mathbf{r}_0, A)$ defined by the minimal error (ME) property

$$\|A^{-1}\mathbf{b} - \mathbf{x}_n^{\mathrm{ME}}\|_2 = \min_{\mathbf{x}\in\mathbf{x}_0+\mathcal{K}_n(A\mathbf{r}_0,A)} \|A^{-1}\mathbf{b} - \mathbf{x}\|_2. \tag{2.16}$$

Unfortunately, Fridman's original implementation of the ME approach is unstable. Fridman's algorithm was later rediscovered by Fletcher (1976) who showed that, in exact arithmetic, the ME iterate $\mathbf{x}_n^{\mathrm{ME}}$ coincides with the auxiliary quantity $\mathbf{x}_n^L$ in SYMMLQ. Hence, as a by-product, SYMMLQ also provides a stable implementation of the ME method. Another direct stabilization of Fridman's algorithm was proposed by Stoer and Freund (1982).

Finally, we remark that Chandra (1978) proposed the SYMMBK algorithm, which is a slightly less expensive variant of SYMMLQ, and derived another stable implementation of the MR method, different from MINRES.

2.4. GMRES and related algorithms

We now return to Krylov subspace methods for general non-Hermitian matrices. Numerous algorithms for computing the iterates characterized by the MR or OR property (2.11) or (2.12), respectively, have been proposed; see Vinsome (1976), Axelsson (1980, 1987), Young and Jea (1980), Saad (1981, 1982, 1984), Elman (1982), Saad and Schultz (1985, 1986). Interestingly, a simple implementation of the MR approach was already described in a paper by Khabaza (1963), which is not referenced at all in the recent literature. In view of Theorem 2.1, all these algorithms generally involve long vector recursions, and typically work and storage requirements per iteration grow linearly with the iteration index n. Consequently, in practice, one cannot afford to run the full algorithms, and it becomes necessary to use restarts or to truncate the vector recursions.

The most elegant and most widely used scheme of this type is GMRES, due to Saad and Schultz (1986), and here we sketch only this particular algorithm. GMRES is modeled after MINRES, where now a generalization of the Hermitian Lanczos process, namely the Arnoldi process (see Arnoldi, 1951, and Saad, 1980), is used to generate orthonormal basis vectors for the Krylov subpaces $\mathcal{K}_n(\mathbf{r}_0, A)$.

Algorithm 2.2 (Arnoldi process)

0) Choose $\mathbf{v}_1 \in \mathbb{C}^N$ with $\|\mathbf{v}_1\|_2 = 1$.

For $n = 1, 2, \ldots,$ do :

1) For $k = 1, 2, \ldots, n$, compute

$$h_{kn} = \mathbf{v}_k^H A \mathbf{v}_n.$$

2) Set

$$\tilde{\mathbf{v}}_{n+1} = A\mathbf{v}_n - \sum_{k=1}^{n} h_{kn}\mathbf{v}_k.$$

3) Compute

$$h_{n+1,n} = \|\tilde{\mathbf{v}}_{n+1}\|_2.$$

4) If $h_{n+1,n} = 0$, stop.
Otherwise, set

$$\mathbf{v}_{n+1} = \tilde{\mathbf{v}}_{n+1}/h_{n+1,n}.$$

The vector recurrences in step 2) of Algorithm 2.2 can be rewritten compactly in matrix form as follows:

$$AV_n = V_{n+1}H_n^{(e)}, \tag{2.17}$$

where

$$V_n = [\, \mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_n \,] \tag{2.18}$$

has orthonormal columns, and

$$H_n^{(e)} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1n} \\ h_{21} & \ddots & & \vdots \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & h_{n,n-1} & h_{nn} \\ 0 & \cdots & 0 & h_{n+1,n} \end{bmatrix} \tag{2.19}$$

is an $(n+1) \times n$ upper Hessenberg matrix of full rank n.

If one chooses the starting vector $\mathbf{v}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$ in Algorithm 2.2, then all possible iterates (2.1) can be parametrized as follows:

$$\mathbf{x}_n = \mathbf{x}_0 + V_n \mathbf{z}_n, \quad \text{where} \quad \mathbf{z}_n \in \mathbb{C}^n. \tag{2.20}$$

Moreover, with (2.20) and (2.17), the minimal residual property (2.11) reduces to the $(n+1) \times n$ least squares problem

$$\|\mathbf{d}_n - H_n^{(e)} \mathbf{z}_n\|_2 = \min_{\mathbf{z} \in \mathbb{C}^n} \|\mathbf{d}_n - H_n^{(e)} \mathbf{z}\|_2, \tag{2.21}$$

where

$$\mathbf{d}_n = [\, \|\mathbf{r}_0\|_2 \quad 0 \quad \cdots \quad 0 \,]^T \in \mathbb{R}^{n+1}. \tag{2.22}$$

GMRES is an implementation of the minimal residual approach (2.11) that obtains the nth MR iterate $\mathbf{x}_n$ by first running n steps of the Arnoldi process and then solving the $(n+1) \times n$ least squares problem (2.21). Note that (2.21) always has a unique solution, since $H_n^{(e)}$ is of full column rank. For a detailed description of the algorithm, we refer the reader to Saad and Schultz (1986).

The Arnoldi Algorithm 2.2 can also be used to compute the nth OR iterate characterized by (2.12). Indeed, as Saad (1981) has shown, $\mathbf{x}_n$ is again of the form (2.20) where $\mathbf{z}_n$ is now the solution of the $n \times n$ linear system

$$H_n \mathbf{z}_n = \mathbf{d}_{n-1}. \tag{2.23}$$

Here

$$H_n = [\, I_n \quad 0 \,]\, H_n^{(e)} \tag{2.24}$$

is the matrix obtained from $H_n^{(e)}$ by deleting the last row in (2.19). The problem with this approach is that H_n can be singular, and then the linear system (2.23) is inconsistent. In fact, H_n is singular if, and only if, no OR iterate satisfying (2.12) exists.

An interesting alternative is to use quasi-Newton techniques, such as Broyden's method (Broyden, 1965). Although designed for general nonlinear equations, these schemes can be applied to non-Hermitian linear systems as a special case. In addition to the iterates $\mathbf{x}_n$, these algorithms also produce approximations to A^{-1}, updated from step to step by a simple rank-1 correction. While these schemes look different at a first glance, they also belong to the class of Krylov subspace methods, as was first observed by Elman (1982). Furthermore, Deuflhard *et al.* (1990) have demonstrated that Broyden's rank-1 update combined with a suitable line search strategy leads to an iterative algorithm that is competitive with GMRES. Eirola and Nevanlinna (1989) have proposed two methods based on a different rank-1 update and shown that one of the resulting schemes is mathematically equivalent to GMRES. These algorithms were studied further by Vuik (1990).

2.5. Preconditioned Krylov subspace methods

For the solution of realistic problems, it is crucial to combine Krylov subspace methods with an efficient preconditioning technique. The basic idea here is as follows. Let M be a given nonsingular $N \times N$ matrix, which approximates in some sense the coefficient matrix A of the original linear system (1.1). Moreover, assume that M is decomposed in the form

$$M = M_1 M_2. \tag{2.25}$$

The Krylov subspace method is then used to solve the preconditioned linear system

$$A'\mathbf{x}' = \mathbf{b}', \tag{2.26}$$

where

$$A' = M_1^{-1} A M_2^{-1}, \ \mathbf{b}' = M_1^{-1}\mathbf{b}, \ \mathbf{x}' = M_2\mathbf{x}.$$

Clearly, (2.26) is equivalent to (1.1). This process generates approximate solutions of (2.26) of the form

$$\mathbf{x}'_n \in \mathbf{x}'_0 + \mathcal{K}_n\left(\mathbf{r}'_0, A'\right). \tag{2.27}$$

Usually, one avoids the explicit calculation of primed quantities, and instead one rewrites the resulting algorithm in terms of the corresponding quantities for the original system. For example, iterates and residual vectors for (1.1) and (2.26) are connected by

$$\mathbf{x}_n = M_2^{-1}\mathbf{x}'_n \quad \text{and} \quad \mathbf{r}_n = M_1\mathbf{r}'_n. \tag{2.28}$$

In particular, note that, by (2.27) and (2.28), the resulting approximations to $A^{-1}\mathbf{b}$ are of the form

$$\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n\left(M^{-1}\mathbf{r}_0, M^{-1}A\right).$$

We remark that the special cases $M_1 = I$ or $M_2 = I$ in (2.25) are referred to as right or left preconditioning, respectively. For right preconditioning, by

(2.28), the preconditioned residual vectors coincide with their counterparts for the original system. For this reason, right preconditioning is usually preferred for MR-type Krylov subspace methods based on (2.11). Moreover, if A has some special structure, the decomposition (2.25) can often be chosen such that the structure is preserved for A'. For example, for Hermitian positive definite A this is the case if one sets $M_2 = M_1^H$ in (2.25).

Obviously, there are two (in general conflicting) requirements for the choice of the preconditioning matrix M for a given Krylov subspace method. First, M^{-1} should approximate A^{-1} well enough so that the algorithm applied to (2.26) will converge faster than for the original system (1.1). On the other hand, preconditioned Krylov subspace methods require at each iteration the solution of one linear system of the type

$$M\mathbf{p} = \mathbf{q}. \tag{2.29}$$

Moreover, for algorithms that involve matrix-vector products with A^T (see Section 3), one has to solve an additional linear system of the form

$$M^T\mathbf{p} = \mathbf{q}. \tag{2.30}$$

Therefore, the preconditioner M needs to be such that linear systems (2.29) respectively (2.30) can be solved cheaply. In this paper, the problem of how to actually construct such preconditioners is not addressed at all. Instead, we refer the reader to the papers by Axelsson (1985) and Saad (1989) for an overview of common preconditioning techniques.

Finally, one more note. In Section 3, explicit descriptions of some Krylov subspace algorithms are given. For simplicity, we have stated these algorithms without preconditioning. It is straightforward to incorporate preconditioning by using the transition rules (2.28).

3. Lanczos-based Krylov subspace methods

In this section, we discuss Krylov subspace methods that are based on the nonsymmetric Lanczos process.

3.1. The classical Lanczos algorithm and BCG

The nonsymmetric Lanczos method was proposed by Lanczos (1950) as a means to reduce an arbitrary matrix $A \in \mathbb{C}^{N\times N}$ to tridiagonal form. One starts the process with two nonzero vectors $\mathbf{v}_1 \in \mathbb{C}^N$ and $\mathbf{w}_1 \in \mathbb{C}^N$ and then generates basis vectors $\{\mathbf{v}_j\}$ for $\mathcal{K}_n(\mathbf{v}_1, A)$ and $\{\mathbf{w}_j\}$ for $\mathcal{K}_n(\mathbf{w}_1, A^T)$ such that the biorthogonality condition

$$\mathbf{w}_j^T\mathbf{v}_k = \begin{cases} 1 & k = j, \\ 0 & \text{otherwise}, \end{cases} \tag{3.1}$$

holds. The point is that the two bases can be built with just three-term recurrences, thus requiring minimal amounts of work and storage per step. The complete algorithm is straightforward and can be stated as follows.

Algorithm 3.1 (Classical Lanczos method)

0) Choose $\tilde{\mathbf{v}}_1,\, \tilde{\mathbf{w}}_1 \in \mathbb{C}^N$ with $\tilde{\mathbf{v}}_1,\ \tilde{\mathbf{w}}_1 \neq 0$.
   Set $\mathbf{v}_0 = \mathbf{w}_0 = 0$.

For $n = 1, 2, \ldots,$ do :

1) Compute $\delta_n = \tilde{\mathbf{w}}_n^T \tilde{\mathbf{v}}_n$.
   If $\delta_n = 0$, set $L = n - 1$ and stop.
2) Otherwise, choose $\beta_n, \gamma_n \in \mathbb{C}$ with $\beta_n \gamma_n = \delta_n$.
   Set $\mathbf{v}_n = \tilde{\mathbf{v}}_n / \gamma_n$ and $\mathbf{w}_n = \tilde{\mathbf{w}}_n / \beta_n$.
3) Compute

$$
\begin{aligned}
\alpha_n &= \mathbf{w}_n^T A \mathbf{v}_n, \\
\tilde{\mathbf{v}}_{n+1} &= A\mathbf{v}_n - \alpha_n \mathbf{v}_n - \beta_n \mathbf{v}_{n-1}, \\
\tilde{\mathbf{w}}_{n+1} &= A^T\mathbf{w}_n - \alpha_n \mathbf{w}_n - \gamma_n \mathbf{w}_{n-1}.
\end{aligned}
$$

   If $\tilde{\mathbf{v}}_{n+1} = 0$ or $\tilde{\mathbf{w}}_{n+1} = 0$, set $L = n$ and stop.

The particular choice of the coefficients α_n, β_n, and γ_n ensures that the biorthogonality relation (3.1) is satisfied.

Similar to (2.18), (2.19), and (2.24), let

$$
V_n = [\, v_1 \quad v_2 \quad \cdots \quad v_n \,], \quad W_n = [\, w_1 \quad w_2 \quad \cdots \quad w_n \,],
$$

$$
H_n^{(e)} = \begin{bmatrix}
\alpha_1 & \beta_2 & 0 & \cdots & 0 \\
\gamma_2 & \alpha_2 & \ddots & \ddots & \vdots \\
0 & \ddots & \ddots & \ddots & 0 \\
\vdots & \ddots & \ddots & \ddots & \beta_n \\
\vdots & & \ddots & \gamma_n & \alpha_n \\
0 & \cdots & \cdots & 0 & \gamma_{n+1}
\end{bmatrix} \in \mathbb{C}^{(n+1)\times n}, \tag{3.2}
$$

and

$$
H_n = [\, I_n \quad 0 \,]\, H_n^{(e)}.
$$

Then the recurrences in the Lanczos process can be written compactly as

$$
\begin{aligned}
AV_n &= V_n H_n + [\, 0 \quad \cdots \quad 0 \quad \tilde{\mathbf{v}}_{n+1} \,], \\
A^T W_n &= W_n H_n^T + [\, 0 \quad \cdots \quad 0 \quad \tilde{\mathbf{w}}_{n+1} \,],
\end{aligned} \tag{3.3}
$$

while the biorthogonality relation (3.1) can be written as

$$
W_n^T V_n = I_n. \tag{3.4}
$$

We note that the Lanczos method is invariant under shifts of the form

$$A \mapsto A + \sigma I, \quad \text{where} \quad \sigma \in \mathbb{C},$$

in that the process generates the same vectors $\{\mathbf{v}_j\}$ and $\{\mathbf{w}_j\}$ and only the tridiagonal matrix (3.2) is shifted:

$$H_n \mapsto H_n + \sigma I_n.$$

In particular, for the Lanczos algorithm it is irrelevant whether the matrix A is singular or not.

Moreover, we would like to stress that the Lanczos process can also be formulated with A^H instead of A^T, by simply conjugating the three-term recurrences for the vectors $\{\mathbf{w}_j\}$. We chose the transpose because one can then avoid complex conjugated recurrence coefficients. Finally, we remark that the Lanczos process reduces to only one recursion in two important special cases, namely $A = A^H$ (with starting vectors $\tilde{\mathbf{w}}_1 = \overline{\tilde{\mathbf{v}}_1}$) and complex symmetric matrices $A = A^T$ (with starting vectors $\tilde{\mathbf{w}}_1 = \tilde{\mathbf{v}}_1$). In both cases, one must also choose $\beta_n = \gamma_n$. In the first case, the resulting algorithm is the well-known Hermitian Lanczos method, which has been studied extensively (see, e.g., Golub and Van Loan, 1989, and the references therein). In the second case, the resulting algorithm is the complex symmetric Lanczos process.

In exact arithmetic, the classical Lanczos method terminates after a finite number of steps. As indicated in Algorithm 3.1, there are two different situations in which the process can stop. The first one, referred to as regular termination, occurs when $\mathbf{v}_{L+1} = 0$ or $\mathbf{w}_{L+1} = 0$. In this case, the Lanczos algorithm has found an invariant subspace of $\mathbb{C}^N$: if $\mathbf{v}_{L+1} = 0$, then the right Lanczos vectors $\mathbf{v}_1, \ldots, \mathbf{v}_L$ span an A-invariant subspace, while if $\mathbf{w}_{L+1} = 0$, then the left Lanczos vectors $\mathbf{w}_1, \ldots, \mathbf{w}_L$ span an A^T-invariant subspace. The second case, referred to as a serious breakdown by Wilkinson (1965), occurs when $\mathbf{w}_L^T \mathbf{v}_L = 0$ with neither $\mathbf{v}_L = 0$ nor $\mathbf{w}_L = 0$. In this case, the Lanczos vectors span neither an A-invariant subspace nor an A^T-invariant subspace of $\mathbb{C}^N$. We will discuss in Section 3.2 techniques for handling the serious breakdowns. We remark that, in the special case of the Hermitian Lanczos process, breakdowns are excluded. In contrast, breakdowns can occur in the complex symmetric Lanczos algorithm (see Cullum and Willoughby, 1985, and Freund, 1989b, 1992).

The Lanczos algorithm was originally introduced to compute eigenvalues, as—in view of (3.3)—the eigenvalues of H_n can be used as approximations for eigenvalues of A. However, Lanczos (1952) also proposed a closely related method, the biconjugate gradient algorithm (BCG), for solving general nonsingular non-Hermitian linear systems (1.1). By and large, BCG was ignored until the mid-seventies, when Fletcher (1976) revived the method.

The BCG algorithm is a Krylov subspace approach that generates iterates defined by a Galerkin condition (2.6) with the special choice of subspaces

$$\mathcal{S}_n = \{\mathbf{s} = \overline{\mathbf{w}} \mid \mathbf{w} \in \mathcal{K}_n(\tilde{\mathbf{r}}_0, A^T)\}.$$

Here, $\tilde{\mathbf{r}}_0$ is a nonzero starting vector discussed below. The standard implementation of the BCG method is as follows.

Algorithm 3.2 (BCG method)

0) Choose $\mathbf{x}_0 \in \mathbb{C}^N$ and set $\mathbf{q}_0 = \mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$.
   Choose $\tilde{\mathbf{r}}_0 \in \mathbb{C}^N$, $\tilde{\mathbf{r}}_0 \neq 0$, and set $\tilde{\mathbf{q}}_0 = \tilde{\mathbf{r}}_0$, $\rho_0 = \tilde{\mathbf{r}}_0^T \mathbf{r}_0$.

For $n = 1, 2, \ldots,$ do :

1) Compute

$$\begin{aligned}
\sigma_{n-1} &= \tilde{\mathbf{q}}_{n-1}^T A \mathbf{q}_{n-1}, \\
\alpha_{n-1} &= \rho_{n-1}/\sigma_{n-1}, \\
\mathbf{x}_n &= \mathbf{x}_{n-1} + \alpha_{n-1}\mathbf{q}_{n-1}, \\
\mathbf{r}_n &= \mathbf{r}_{n-1} - \alpha_{n-1} A \mathbf{q}_{n-1}, \\
\tilde{\mathbf{r}}_n &= \tilde{\mathbf{r}}_{n-1} - \alpha_{n-1} A^T \tilde{\mathbf{q}}_{n-1}.
\end{aligned}$$

2) Compute

$$\begin{aligned}
\rho_n &= \tilde{\mathbf{r}}_n^T \mathbf{r}_n, \\
\beta_n &= \rho_n/\rho_{n-1}, \\
\mathbf{q}_n &= \mathbf{r}_n + \beta_n \mathbf{q}_{n-1}, \\
\tilde{\mathbf{q}}_n &= \tilde{\mathbf{r}}_n + \beta_n \tilde{\mathbf{q}}_{n-1}.
\end{aligned}$$

3) If $\mathbf{r}_n = 0$ or $\tilde{\mathbf{r}}_n = 0$, stop.

Note that BCG requires a second nonzero starting vector $\tilde{\mathbf{r}}_0 \in \mathbb{C}^N$, which can be chosen freely. Usually, one sets $\tilde{\mathbf{r}}_0 = \mathbf{r}_0$ or $\tilde{\mathbf{r}}_0 = \overline{\mathbf{r}_0}$, or one chooses $\tilde{\mathbf{r}}_0$ as a vector with random entries.

The BCG algorithm is the archetype of an entire class of Lanczos-based Krylov subspace methods for non-Hermitian matrices, some of which we will discuss later. Unfortunately, BCG typically exhibits a rather erratic convergence behavior with wild oscillations in the residual norm $\|\mathbf{r}_n\|_2$. Even worse, the BCG process can break down completely. More precisely, the BCG Algorithm 3.2 cannot be continued if

$$\tilde{\mathbf{q}}_{n-1}^T A \mathbf{q}_{n-1} = 0, \quad \tilde{\mathbf{r}}_{n-1} \neq 0, \ \ \mathbf{r}_{n-1} \neq 0, \tag{3.5}$$

or if

$$\tilde{\mathbf{r}}_{n-1}^T \mathbf{r}_{n-1} = 0, \quad \tilde{\mathbf{r}}_{n-1} \neq 0, \ \ \mathbf{r}_{n-1} \neq 0. \tag{3.6}$$

The source of the breakdown (3.5) is the Galerkin condition (2.6) used to define the iterates. As was pointed out in Section 2.2, the existence of an iterate satisfying (2.6) is not guaranteed at every step, and in fact (3.5) occurs if, and only if, no BCG iterate exists. Furthermore, it can be shown that (3.5) is equivalent to the Lanczos matrix H_n being singular. The source of the second breakdown (3.6) is the underlying nonsymmetric Lanczos process, which can have a serious breakdown. It turns out that the vectors $\mathbf{r}_{n-1}$ and $\tilde{\mathbf{r}}_{n-1}$ in the BCG Algorithm 3.2 are scalar multiples of the vectors $\mathbf{v}_n$ and $\mathbf{w}_n$, respectively, that are generated by the classical Lanczos Algorithm 3.1 started with

$$\tilde{\mathbf{v}}_1 = \mathbf{r}_0 \quad \text{and} \quad \tilde{\mathbf{w}}_1 = \tilde{\mathbf{r}}_0.$$

Hence, a breakdown in the Lanczos process will be paralleled by a breakdown (3.6) in the BCG algorithm.

As the discussion above shows, BCG, while requiring little work and storage per step, is susceptible to breakdowns and numerical instabilities. In addition, another possible disadvantage of the classical BCG algorithm is its use of the transpose of A, which may not be readily available in some situations. As a result, variants of BCG were sought which would preserve the low work and storage requirements, while curing the possible breakdowns and avoiding the use of the transpose. In the next section, we will discuss the look-ahead Lanczos algorithm, an extension of the Lanczos method which handles in almost all cases the serious breakdowns in the Lanczos process. In Section 3.4 we present the quasi-minimal residual approach, based on the look-ahead Lanczos algorithm and using a quasi-minimization property to avoid the breakdowns caused by the Galerkin condition. Finally, in Section 3.5 we survey some of the so-called transpose-free algorithms, which typically replace the multiplication by A^T in the BCG algorithm by one or more multiplications by A.

3.2. A look-ahead Lanczos algorithm

One of the possible terminations of the Lanczos algorithm is a serious breakdown, when $\delta_n = 0$ in Algorithm 3.1, with neither $\tilde{\mathbf{v}}_n = 0$ nor $\tilde{\mathbf{w}}_n = 0$. As a result, the vectors $\tilde{\mathbf{v}}_n$ and $\tilde{\mathbf{w}}_n$ cannot be scaled to obtain the Lanczos vectors $\mathbf{v}_n$ and $\mathbf{w}_n$ corresponding to the basis vectors $A^n\mathbf{v}_1$ and $(A^T)^n\mathbf{w}_1$. Furthermore, it turns out that even if $\mathbf{v}_n$ and $\mathbf{w}_n$ were computed using a different scaling method, the next pair of vectors $\tilde{\mathbf{v}}_{n+1}$ and $\tilde{\mathbf{w}}_{n+1}$ could not be computed so as to fulfill (3.1). The problem here is not just one of scaling, but also that the biorthogonality required of the vectors $\tilde{\mathbf{v}}_{n+1}$ and $\tilde{\mathbf{w}}_{n+1}$ cannot be satisfied. However, it could happen that the biorthogonality condition (3.1) can once again be fulfilled for a pair of vectors corresponding to some higher power of A and A^T. A procedure which somehow advances to this next pair of Lanczos vectors will be called a look-ahead Lanczos procedure.

The main idea behind the look-ahead Lanczos algorithms is to relax the biorthogonality relation (3.1) when a breakdown is encountered. For each fixed $n = 1, 2, \ldots$, the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ and $\mathbf{w}_1, \ldots, \mathbf{w}_n$ generated by the look-ahead procedure can be grouped into $l = l(n)$ blocks

$$\begin{aligned} V^{(k)} &= [\, \mathbf{v}_{n_k} \quad \mathbf{v}_{n_k+1} \quad \cdots \quad \mathbf{v}_{n_{k+1}-1} \,], \\ W^{(k)} &= [\, \mathbf{w}_{n_k} \quad \mathbf{w}_{n_k+1} \quad \cdots \quad \mathbf{w}_{n_{k+1}-1} \,], \end{aligned} \qquad k = 1, \ldots, l-1,$$

and

$$\begin{aligned} V^{(l)} &= [\, \mathbf{v}_{n_l} \quad \mathbf{v}_{n_l+1} \quad \cdots \quad \mathbf{v}_{n} \,], \\ W^{(l)} &= [\, \mathbf{w}_{n_l} \quad \mathbf{w}_{n_l+1} \quad \cdots \quad \mathbf{w}_{n} \,], \end{aligned}$$

where

$$1 = n_1 < n_2 < \cdots < n_k < \cdots < n_l \leq n < n_{l+1}.$$

The blocks are constructed so that (3.1) is relaxed to

$$\left(W^{(j)}\right)^T \left(V^{(k)}\right) = \begin{cases} D^{(k)} & \text{if } j = k, \\ 0 & \text{if } j \neq k \end{cases}, \qquad j, k = 1, \ldots, l,$$

where $D^{(k)}$ is nonsingular for $k = 1, \ldots, l-1$, and $D^{(l)}$ is nonsingular if $n = n_{l+1} - 1$. The first vectors $\mathbf{v}_{n_k}$ and $\mathbf{w}_{n_k}$ in each block are called regular, and the remaining vectors are called inner. We note that in practice, for reasons of stability, the computed vectors are usually scaled to have unit length (see Taylor, 1982).

Two such look-ahead procedures have been proposed, one by Parlett *et al.* (1985), and a second one by Freund *et al.* (1991b). The Freund *et al.* implementation requires the same number of inner products per step as the classical Lanczos algorithm, and reduces to the classical Lanczos procedure in the absence of look-ahead steps. In contrast, the Parlett *et al.* implementation always requires more work per step and does not reduce to the classical Lanczos algorithm in the absence of look-ahead steps. It also does not generalize easily to blocks of more than two vectors. Therefore, we will focus on the implementation proposed by Freund *et al.*. The basic structure of this look-ahead Lanczos algorithm is as follows.

Algorithm 3.3 (Sketch of the look-ahead Lanczos process)

0) Choose $\mathbf{v}_1$, $\mathbf{w}_1 \in \mathbb{C}^N$ with $\|\mathbf{v}_1\|_2 = \|\mathbf{w}_1\|_2 = 1$.
   Set $V^{(1)} = \mathbf{v}_1$, $W^{(1)} = \mathbf{w}_1$, $D^{(1)} = (W^{(1)})^T V^{(1)}$.
   Set $n_1 = 1$, $l = 1$, $\mathbf{v}_0 = \mathbf{w}_0 = 0$, $V_0 = W_0 = \emptyset$, $\rho_1 = \xi_1 = 1$.

For $n = 1, 2, \ldots$, do :

1) Decide whether to construct $\mathbf{v}_{n+1}$ and $\mathbf{w}_{n+1}$ as regular or inner vectors and go to 2) or 3), respectively.

2) (Regular step.) Compute

$$
\begin{aligned}
\tilde{\mathbf{v}}_{n+1} &= A\mathbf{v}_n - V^{(l)}(D^{(l)})^{-1}(W^{(l)})^T A\mathbf{v}_n \\
&\quad - V^{(l-1)}(D^{(l-1)})^{-1}(W^{(l-1)})^T A\mathbf{v}_n, \\
\tilde{\mathbf{w}}_{n+1} &= A^T\mathbf{w}_n - W^{(l)}(D^{(l)})^{-T}(V^{(l)})^T A^T\mathbf{w}_n \\
&\quad - W^{(l-1)}(D^{(l-1)})^{-T}(V^{(l-1)})^T A^T\mathbf{w}_n.
\end{aligned} \tag{3.7}
$$

Set $n_{l+1} = n+1$, $l = l+1$, $V^{(l)} = W^{(l)} = \emptyset$, and go to 4).

3) (Inner step.) Compute

$$
\begin{aligned}
\tilde{\mathbf{v}}_{n+1} &= A\mathbf{v}_n - \zeta_n\mathbf{v}_n - (\eta_n/\rho_n)\,\mathbf{v}_{n-1} \\
&\quad - V^{(l-1)}(D^{(l-1)})^{-1}(W^{(l-1)})^T A\mathbf{v}_n, \\
\tilde{\mathbf{w}}_{n+1} &= A^T\mathbf{w}_n - \zeta_n\mathbf{w}_n - (\eta_n/\xi_n)\,\mathbf{w}_{n-1} \\
&\quad - W^{(l-1)}(D^{(l-1)})^{-T}(V^{(l-1)})^T A^T\mathbf{w}_n.
\end{aligned} \tag{3.8}
$$

4) Compute $\rho_{n+1} = \|\tilde{\mathbf{v}}_{n+1}\|_2$ and $\xi_{n+1} = \|\tilde{\mathbf{w}}_{n+1}\|_2$.
If $\rho_{n+1} = 0$ or $\xi_{n+1} = 0$, set $L = n$, and stop.
Otherwise, set

$$
\mathbf{v}_{n+1} = \tilde{\mathbf{v}}_{n+1}/\rho_{n+1}, \quad \mathbf{w}_{n+1} = \tilde{\mathbf{w}}_{n+1}/\xi_{n+1},
$$

$$
V^{(l)} = \begin{bmatrix} V^{(l)} & \mathbf{v}_{n+1} \end{bmatrix}, \quad W^{(l)} = \begin{bmatrix} W^{(l)} & \mathbf{w}_{n+1} \end{bmatrix},
$$

$$
D^{(l)} = (W^{(l)})^T V^{(l)}.
$$

Step 2) of the algorithm is a block version of the classical Lanczos recurrences of step 3) in Algorithm 3.1. Step 3) builds inner vectors spanning the gap between two regular vectors. In this implementation, the inner vectors are generated by a three-term inner recurrence, and then biorthogonalized against the last block. The vectors generated by the look-ahead Lanczos algorithm still obey (3.3), but now H_n, instead of having simply the tridiagonal structure (3.2), is an upper Hessenberg block tridiagonal matrix with small blocks of size $(n_k - n_{k-1}) \times (n_k - n_{k-1})$ on the diagonal. Furthermore, the biorthogonality relation (3.4) now reads

$$
W_n^T V_n = D_n = \operatorname{diag}\left(D^{(1)}, D^{(2)}, \ldots, D^{(l)}\right).
$$

Note that D_n is guaranteed to be nonsingular if $n = n_{l+1} - 1$.

If only regular steps are performed, then the algorithm above reduces to the classical Lanczos process. Thus, the strategy used in step 1) for deciding when to construct inner vectors should perform regular steps whenever possible. In addition, in practice the look-ahead algorithm must also be able to handle near-breakdowns, that is, situations when

$$\tilde{\mathbf{w}}_n^T \tilde{\mathbf{v}}_n \approx 0, \quad \tilde{\mathbf{v}}_n \not\approx 0, \quad \tilde{\mathbf{w}}_n \not\approx 0.$$

Freund *et al.* (1991b) proposed a practical procedure for the decision in step 1) based on three different checks. For a regular step, it is necessary that $D^{(l)}$ be nonsingular. Therefore, one of the checks monitors the size of $\sigma_{\min}(D^{(l)})$. The other two checks attempt to ensure the linear independence of the Lanczos vectors. The algorithm monitors the size of the components along the two previous blocks $V^{(l)}$ and $V^{(l-1)}$, respectively $W^{(l)}$ and $W^{(l-1)}$, in (3.7), and performs a regular step only if these terms do not dominate the components $A\mathbf{v}_n$ and $A^T\mathbf{w}_n$ in the new Krylov spaces. For details, see Freund *et al.* (1991b).

The look-ahead algorithm outlined above will handle serious breakdowns and near-breakdowns in the classical Lanczos algorithm, except for the special event of an incurable breakdown (Taylor, 1982). These are situations where the look-ahead procedure would build an infinite block, without ever finding a nonsingular $D^{(l)}$. Taylor (1982) has shown in his Mismatch Theorem that in case of an incurable breakdown, one can still recover eigenvalue information, as the eigenvalues of the H_{n_l} are also eigenvalues of A. For linear systems, an incurable breakdown would require restarting the procedure with a different choice of starting vectors. Fortunately, in practice round-off errors will make an incurable breakdown highly unlikely.

Finally, we remark that, for the important class of p-cyclic matrices A, serious breakdowns in the Lanczos process occur in a regular pattern. In this case, look-ahead steps are absolutely necessary if one wants to exploit the p-cyclic structure. For details of a look-ahead Lanczos algorithm for p-cyclic matrices, we refer the reader to Freund *et al.* (1991a).

3.3. The QMR algorithm

We now turn to the quasi-minimal residual approach. The procedure was first proposed by Freund (1989b) for the case of complex symmetric linear systems, and then extended by Freund and Nachtigal (1991) for the case of general non-Hermitian matrices.

Recall from (2.2) that the nth iterate of any Krylov subspace method is of the form

$$\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{r}_0, A).$$

If now we choose

$$\mathbf{v}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2 \tag{3.9}$$

in Algorithm 3.3, then the right Lanczos vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ span the Krylov space $\mathcal{K}_n(\mathbf{r}_0, A)$, hence we can write

$$\mathbf{x}_n = \mathbf{x}_0 + V_n \mathbf{z}_n,$$

for some $\mathbf{z}_n \in \mathbb{C}^n$. Together with (3.9) and the first relation in (3.3), this gives for the residual

$$\mathbf{r}_n = \mathbf{r}_0 - AV_n\mathbf{z}_n = V_{n+1}\left(\mathbf{d}_n - H_n^{(e)}\mathbf{z}_n\right), \tag{3.10}$$

where $\mathbf{d}_n$ is defined as in (2.22). As V_{n+1} is not unitary, it is not possible to minimize the Euclidean norm of the residual without expending $\mathcal{O}(Nn^2)$ work and $\mathcal{O}(Nn)$ storage. Instead, one minimizes just the Euclidean norm of the coefficient vector in (3.10), that is, $\mathbf{z}_n \in \mathbb{C}^n$ is chosen as the solution of the least squares problem

$$\|\mathbf{d}_n - H_n^{(e)}\mathbf{z}_n\|_2 = \min_{\mathbf{z}\in\mathbb{C}^n} \|\mathbf{d}_n - H_n^{(e)}\mathbf{z}\|_2. \tag{3.11}$$

As was pointed out by Manteuffel (1991), solving the minimization problem (3.11) is equivalent to minimizing the residual in a norm that changes with the step number:

$$\min_{\mathbf{x}\in\mathbf{x}_0+\mathcal{K}_n(\mathbf{r}_0,A)} \|D_{n+1}^{-1}W_{n+1}^T(\mathbf{b} - A\mathbf{x})\|_2, \quad n = n_k - 2.$$

Thus, the QMR does not contradict the Faber and Manteuffel Theorem 2.1, which excludes only methods that minimize in a *fixed* norm.

To solve the least-squares problem (3.11), one uses a QR factorization of $H_n^{(e)}$. As $H_n^{(e)}$ is upper Hessenberg, its QR factorization can be easily computed and updated using Givens rotations; the approach is a standard one (see, e.g., Golub and Van Loan, 1989). One computes a unitary matrix $Q_n \in \mathbb{C}^{(n+1)\times(n+1)}$ and an upper triangular matrix $R_n \in \mathbb{C}^n$ such that

$$Q_n H_n^{(e)} = \begin{bmatrix} R_n \\ 0 \end{bmatrix}, \tag{3.12}$$

and then obtains $\mathbf{z}_n$ from

$$\mathbf{z}_n = R_n^{-1}\mathbf{t}_n, \quad \mathbf{t}_n = [\, I_n \quad 0\,]\, Q_n\mathbf{d}_n, \tag{3.13}$$

which gives

$$\mathbf{x}_n = \mathbf{x}_0 + V_n R_n^{-1}\mathbf{t}_n. \tag{3.14}$$

This gives the following QMR algorithm.

Algorithm 3.4 (QMR algorithm)

0) Choose $\mathbf{x}_0 \in \mathbb{C}^N$ and set $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$, $\rho_0 = \|\mathbf{r}_0\|_2$, $\mathbf{v}_1 = \mathbf{r}_0/\rho_0$.
Choose $\mathbf{w}_1 \in \mathbb{C}^N$ with $\|\mathbf{w}_1\|_2 = 1$.

For $n = 1, 2, \ldots,$ do :

1) Perform the nth iteration of the look-ahead Lanczos Algorithm 3.3.

This yields matrices V_n, V_{n+1}, $H_n^{(e)}$ which satisfy

$$AV_n = V_{n+1} H_n^{(e)}.$$

2) Update the QR factorization (3.12) of $H_n^{(e)}$ and the vector $\mathbf{t}_n$ in (3.13).
3) Compute $\mathbf{x}_n$ from (3.14).
4) If $\mathbf{x}_n$ has converged, stop.

We note that since Q_n is a product of Givens rotations, the vector $\mathbf{t}_n$ is easily updated in step 2). Also, as H_n is block tridiagonal, R_n also has a block structure that is used in step 3) to update $\mathbf{x}_n$ using only short recurrences. For complete details, see Freund and Nachtigal (1991).

The point of the quasi-minimal residual approach is that the least-squares problem (3.11) always has a unique solution. From step 4) of Algorithm 3.3, ρ_k, the subdiagonal entries of $H_n^{(e)}$, are all nonzero, hence $H_n^{(e)}$ has full column rank, R_n is nonsingular, and so (3.11) always defines a unique iterate $\mathbf{x}_n$. This then avoids the Galerkin breakdown in the BCG algorithm. But more importantly, the quasi-minimization (3.11) is strong enough to enable us to prove a convergence theorem for QMR. This is in contrast to BCG and methods derived from BCG, for which no convergence results are known. Indeed, one can prove two theorems for QMR, both relating the QMR convergence behavior to the convergence of GMRES.

Theorem 3.5 (Freund and Nachtigal, 1991)
Suppose that the $L \times L$ matrix H_L generated by the look-ahead Lanczos algorithm is diagonalizable, and let $X \in \mathbb{C}^{L \times L}$ be a matrix of eigenvectors of H_L. Then for $n = 1, \ldots, L-1$,

$$\frac{\|\mathbf{r}_n^{\mathrm{QMR}}\|_2}{\|\mathbf{r}_0\|_2} \le \|V_{n+1}\|_2 \, \kappa_2(X) \, \epsilon_n,$$

where

$$\epsilon_n = \max_{\psi \in \mathcal{P}_n : \psi(0)=1} \; \min_{\lambda \in \lambda(A)} |\psi(\lambda)|.$$

By comparison, the convergence result for GMRES reads as follows.

Theorem 3.6 (Saad and Schultz, 1986)
Suppose that A is diagonalizable, and let U be a matrix of eigenvectors of A. Then, for $n = 1, 2, \ldots$,

$$\frac{\|\mathbf{r}_n^{\mathrm{GMRES}}\|_2}{\|\mathbf{r}_0\|_2} \le \kappa_2(U) \, \epsilon_n,$$

where ϵ_n is as above.

Thus, Theorem 3.5 shows that GMRES and QMR solve the same approximation problem. The second convergence result gives a quantitative description of the departure from optimality due to the quasi-optimal approach.

Theorem 3.7 (Nachtigal, 1991)

$$\|\mathbf{r}_n^{\mathrm{QMR}}\|_2 \le \kappa_2(V_{n+1})\,\|\mathbf{r}_n^{\mathrm{GMRES}}\|_2.$$

For both Theorem 3.5 and Theorem 3.7, we note that the right Lanczos vectors $\{\mathbf{v}_j\}$ obtained from Algorithm 3.4 are unit vectors, and hence the condition number of V_{n+1} can be bounded by a slowly growing function,

$$\|V_{n+1}\|_2 \le \sqrt{n+1}.$$

Next, we note that it is possible to recover BCG iterates, when they exist, from the corresponding QMR iterates. We have

Theorem 3.8 (Freund and Nachtigal, 1991)
Let $n = n_k - 1$, $k = 1, \ldots$. Then,

$$\mathbf{x}_n^{\mathrm{BCG}} = \mathbf{x}_{n-1}^{\mathrm{QMR}} + \frac{\tau_n}{c_n}\mathbf{p}_n,$$
$$\|\mathbf{r}_n^{\mathrm{BCG}}\|_2 = \|\mathbf{r}_0\|_2 |s_1 \cdots s_n| \frac{1}{c_n}.$$

Here, $\mathbf{p}_n$ is the nth column of $V_n R_n^{-1}$ and is computed anyway as part of step 3) of the QMR Algorithm 3.4, s_n and c_n are the sine and cosine of the nth Givens rotation involved in the QR decomposition of $H_n^{(e)}$, and τ_n is the $(n+1)$st component of $Q_n\mathbf{d}_n$. The point is that the BCG iterate and the norm of the BCG residual are both by-products of the QMR algorithm, available at little extra cost, and the existence of the BCG iterate can be checked by monitoring the size of the Givens rotation cosine c_n. Thus, QMR can also be viewed as a stable implementation of BCG.

Finally, we remark that for the special case of Hermitian matrices, the QMR Algorithm 3.4 (with $\mathbf{w}_1 = \overline{\mathbf{v}_1}$) is mathematically equivalent to MINRES. Hence, QMR can also be viewed as an extension of MINRES to non-Hermitian matrices.

3.4. Transpose-free methods

In contrast to Krylov subspace methods based on the Arnoldi Algorithm 2.2, which only require matrix-vector multiplications with A, algorithms such as BCG and QMR, which are based directly on the Lanczos process, involve matrix-vector products with A and A^T. This is a disadvantage for certain applications, where A^T is not readily available. It is possible to devise Lanczos-based Krylov subspace methods that do not involve the transpose of A. In this section, we give an overview of such transpose-free schemes.

First, we consider the QMR algorithm. As pointed out by Freund and

Zha (1991), in principle it is always possible to eliminate A^T altogether, by choosing the starting vector $\mathbf{w}_1$ suitably. This observation is based on the fact that any square matrix is similar to its transpose. In particular, there always exists a nonsingular matrix P such that

$$A^T P = PA. \tag{3.15}$$

Now suppose that in the QMR Algorithm 3.4 we choose the special starting vector $\mathbf{w}_1 = P\mathbf{v}_1/\|P\mathbf{v}_1\|_2$. Then, with (3.15), one readily verifies that the vectors generated by look-ahead Lanczos Algorithm 3.3 satisfy

$$\mathbf{w}_n = P\mathbf{v}_n/\|P\mathbf{v}_n\|_2 \quad \text{for all} \quad n. \tag{3.16}$$

Hence, instead of updating the left Lanczos vectors $\{w_n\}$ by means of the recursions in (3.7) or (3.8), they can be computed directly from (3.16). The resulting QMR algorithm no longer involves the transpose of A; in exchange, it requires one matrix-vector multiplication with P in each iteration step. Therefore, this approach is only viable for special classes of matrices A, for which one can find a matrix P satisfying (3.15) easily, and for which, at the same time, matrix-vector products with P can be computed cheaply. The most trivial case are complex symmetric matrices (see Section 6), which fulfill (3.15) with $P = I$. Another simple case are matrices A that are symmetric with respect to the antidiagonal. These so-called centrosymmetric matrices, by their very definition, satisfy (3.15) with $P = J$, where

$$J = \begin{bmatrix} 0 & \cdots & 0 & 1 \\ \vdots & \iddots & 1 & 0 \\ 0 & \iddots & \iddots & \vdots \\ 1 & 0 & \cdots & 0 \end{bmatrix}$$

is the $N \times N$ antidiagonal identity matrix. Note that Toeplitz matrices (see Section 7) are a special case of centrosymmetric matrices. Finally, the condition (3.15) is also fulfilled for matrices of the form

$$A = TM^{-1}, \quad P = M^{-1},$$

where T and M are real symmetric matrices and M is nonsingular. Matrices A of this type arise when real symmetric linear systems $Tx = b$ are preconditioned by M. The resulting QMR algorithm for the solution of preconditioned symmetric linear system has the same work and storage requirements as preconditioned SYMMLQ or MINRES. However, the QMR approach is more general, in that it can be combined with any nonsingular symmetric preconditioner M, while SYMMLQ and MINRES require M to be positive definite M (see, e.g., Gill *et al.*, 1990). For strongly indefinite matrices T, the use of indefinite preconditioners M typically leads to considerably faster convergence; see Freund and Zha (1991) for numerical examples.

Next, we turn to transpose-free variants of the BCG method. Sonneveld (1989) with his conjugate gradients squared algorithm (CGS) was the first to devise a transpose-free BCG-type scheme. Note that, in the BCG Algorithm 3.2, the matrix A^T appears merely in the update formulas for the vectors $\tilde{\mathbf{r}}_n$ and $\tilde{\mathbf{q}}_n$. On the other hand, these vectors are then used only for the computation of the vector products $\rho_n = \tilde{\mathbf{r}}_n^T \mathbf{r}_n$ and $\sigma_n = \tilde{\mathbf{q}}_n^T A \mathbf{q}_n$. Sonneveld observed that, by rewriting these products, the transpose can be eliminated from the formulas, while at the same time one obtains iterates

$$\mathbf{x}_{2n} \in \mathbf{x}_0 + \mathcal{K}_{2n}(\mathbf{r}_0, A), \quad n = 1, 2, \ldots, \tag{3.17}$$

that are contained in a Krylov subspace of twice the dimension, as compared to BCG. First, we consider ρ_n. From Algorithm 3.2 it is obvious that

$$\mathbf{r}_n = \psi_n(A)\mathbf{r}_0 \quad \text{and} \quad \tilde{\mathbf{r}}_n = \psi_n(A^T)\tilde{\mathbf{r}}_0, \tag{3.18}$$

where ψ_n is the nth residual polynomials (recall (2.3) and (2.4)) of the BCG process. With (3.18), one obtains the identity

$$\rho_n = \tilde{\mathbf{r}}_0^T \left(\psi_n(A)\right)^2 \mathbf{r}_0, \tag{3.19}$$

which shows that ρ_n can be computed without using A^T. Similarly,

$$\mathbf{q}_n = \varphi_n(A)\mathbf{r}_0 \quad \text{and} \quad \tilde{\mathbf{q}}_n = \varphi_n(A^T)\tilde{\mathbf{r}}_0,$$

for some polynomial $\varphi_n \in \mathcal{P}_n$, and hence

$$\sigma_n = \tilde{\mathbf{r}}_0^T A \left(\varphi_n(A)\right)^2 \mathbf{r}_0. \tag{3.20}$$

By rewriting the vector recursions in Algorithm 3.2 in terms of ψ_n and φ_n and by squaring the resulting polynomial relations, Sonneveld showed that the vectors in (3.19) and (3.20) can be updated by means of short recursions. Furthermore, the actual iterates (3.17) generated by CGS are characterized by

$$\mathbf{r}_{2n}^{\mathrm{CGS}} = \mathbf{b} - A\mathbf{x}_{2n} = \left(\psi_n^{\mathrm{BCG}}(A)\right)^2 \mathbf{r}_0. \tag{3.21}$$

Hence the CGS residual polynomials $\psi_{2n}^{\mathrm{CGS}} = \left(\psi_n^{\mathrm{BCG}}\right)^2$ are just the squared BCG polynomials. As pointed out earlier, BCG typically exhibits a rather erratic convergence behavior. As is clear from (3.21), these effects are magnified in CGS, and CGS typically accelerates convergence as well as divergence of BCG. Moreover, there are cases for which CGS diverges, while BCG still converges.

For this reason, more smoothly converging variants of CGS have been sought. Van der Vorst (1990) was the first to propose such a method. His Bi-CGSTAB again produces iterates of the form (3.17), but instead of squaring the BCG polynomials as in (3.21), the residual vector is now of the form

$$\mathbf{r}_{2n} = \psi_n^{\mathrm{BCG}}(A)\chi_n(A)\mathbf{r}_0.$$

Here $\chi_n \in \mathcal{P}_n$, with $\chi_n(0) = 1$, is a polynomial that is updated from step to step by adding a new linear factor:

$$\chi_n(\lambda) \equiv (1 - \eta_n \lambda)\chi_{n-1}(\lambda). \tag{3.22}$$

The free parameter η_n in (3.22) is determined by a local steepest descent step, i.e., η_n is the optimal solution of

$$\min_{\eta \in \mathbb{C}} \|(I - \eta A)\chi_{n-1}(A)\psi_n^{\mathrm{BCG}}(A)\mathbf{r}_0\|_2.$$

Due to the steepest descent steps, Bi-CGSTAB typically has much smoother convergence behavior than BCG or CGS. However, the norms of the Bi-CGSTAB residuals may still oscillate considerably for difficult problems. Finally, Gutknecht (1991) has noted that, for real A, the polynomials χ_n will always have real roots only, even if A has complex eigenvalues. He proposed a variant of Bi-CGSTAB with polynomials (3.22) that are updated by quadratic factors in each step and thus can have complex roots in general.

In the CGS algorithm, the iterates (3.17) are updated by means of a formula of the form

$$\mathbf{x}_{2n}^{\mathrm{CGS}} = \mathbf{x}_{2(n-1)}^{\mathrm{CGS}} + \alpha_{n-1}(\mathbf{y}_{2n-1} + \mathbf{y}_{2n}). \tag{3.23}$$

Here the vectors $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{2n}$ satisfy

$$\mathrm{span}\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m\} = \mathcal{K}_m(\mathbf{r}_0, A), \quad m = 1, 2, \ldots, 2n.$$

In other words, in each iteration of the CGS algorithm two search directions $\mathbf{y}_{2n-1}$ and $\mathbf{y}_{2n}$ are available, while the actual iterate is updated by the one-dimensional step (3.23) only. Based on this observation, Freund (1991b) has proposed a variant of CGS that makes use of all available search directions. More precisely, instead of one iterate $\mathbf{x}_{2n}^{\mathrm{CGS}}$ per step it produces two iterates $\mathbf{x}_{2n-1}$ and $\mathbf{x}_{2n}$ of the form

$$\mathbf{x}_m = \mathbf{x}_0 + [\,\mathbf{y}_1 \quad \mathbf{y}_2 \quad \cdots \quad \mathbf{y}_m\,]\,\mathbf{z}_m, \quad \mathbf{z}_m \in \mathbb{C}^m. \tag{3.24}$$

Furthermore, the free parameter vector $\mathbf{z}_m$ in (3.24) can be chosen such that the iterates satisfy a quasi-minimal residual condition, similar to the quasi-minimization property of the QMR Algorithm 3.4. For this reason, the resulting scheme is called transpose-free quasi-minimal residual algorithm (TFQMR). For details, we refer the reader to Freund (1991b), where the following implementation of TFQMR is derived.

Algorithm 3.9 (TFQMR algorithm)

0) Choose $\mathbf{x}_0 \in \mathbb{C}^N$.

Set $\mathbf{w}_1 = \mathbf{y}_1 = \mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$, $\mathbf{v}_0 = A\mathbf{y}_1$, $\mathbf{d}_0 = 0$.

Set $\tau_0 = \|\mathbf{r}_0\|_2$, $\vartheta_0 = 0$, $\eta_0 = 0$.

Choose $\tilde{\mathbf{r}}_0 \in \mathbb{C}^N$, $\tilde{\mathbf{r}}_0 \neq 0$, and set $\rho_0 = \tilde{\mathbf{r}}_0^T \mathbf{r}_0$.

For $n = 1, 2, \ldots,$ do :

1) Compute

$$\begin{aligned}\sigma_{n-1} &= \tilde{\mathbf{r}}_0^T \mathbf{v}_{n-1},\\ \alpha_{n-1} &= \rho_{n-1}/\sigma_{n-1},\\ \mathbf{y}_{2n} &= \mathbf{y}_{2n-1} - \alpha_{n-1}\mathbf{v}_{n-1}.\end{aligned}$$

2) For $m = 2n-1, 2n$ do :

Compute

$$\begin{aligned}\mathbf{w}_{m+1} &= \mathbf{w}_m - \alpha_{n-1} A \mathbf{y}_m,\\ \vartheta_m &= \|\mathbf{w}_{m+1}\|_2/\tau_{m-1},\ c_m = 1/\sqrt{1+\vartheta_m^2},\\ \tau_m &= \tau_{m-1}\vartheta_m c_m,\ \eta_m = c_m^2 \alpha_{n-1},\\ \mathbf{d}_m &= y_m + (\vartheta_{m-1}^2 \eta_{m-1}/\alpha_{n-1})\mathbf{d}_{m-1},\\ \mathbf{x}_m &= \mathbf{x}_{m-1} + \eta_m \mathbf{d}_m.\end{aligned}$$

If $\mathbf{x}_m$ has converged, stop.

3) Compute

$$\begin{aligned}\rho_n &= \tilde{\mathbf{r}}_0^T \mathbf{w}_{2n+1},\\ \beta_n &= \rho_n/\rho_{n-1},\\ \mathbf{y}_{2n+1} &= \mathbf{w}_{2n+1} + \beta_n \mathbf{y}_{2n},\\ \mathbf{v}_n &= A\mathbf{y}_{2n+1} + \beta_n(A\mathbf{y}_{2n} + \beta_n \mathbf{v}_{n-1}).\end{aligned}$$

We would like to point out that the iterates generated by the QMR Algorithm 3.4 and the TFQMR Algorithm 3.9 are different in general.

Another transpose-free QMR method was proposed by Chan *et al.* (1991). Their scheme is mathematically equivalent to the QMR Algorithm 3.4, where the latter is based on the classical Lanczos process without look-ahead. The method first uses a transpose-free squared version of the Lanczos algorithm (see Gutknecht, 1990a) to generate the tridiagonal matrix (3.2). The right Lanczos vectors $\mathbf{v}_n$ are then computed by running the corresponding recursion in step 3) of Algorithm 3.1, and finally the QMR iterates are obtained as in Algorithm 3.4.

Freund and Szeto (1991) have derived yet another transpose-free QMR scheme, which is modeled after CGS and is based on squaring the residual polynomials of the standard QMR Algorithm 3.4.

However, the algorithm by Chan *et al.* and the squared QMR approach both require per iteration three matrix-vector products with A and hence they are more expensive than CGS, Bi-CGSTAB, or TFQMR, which involve only two such products per step.

Finally, we remark that none of the transpose-free methods considered in this section, except for Freund and Zha's simplified QMR algorithm based on (3.15), addresses the problem of breakdowns. Indeed, in exact arithmetic, all these schemes break down every time a breakdown (3.5) or (3.6) occurs in the BCG Algorithm 3.2. Practical look-ahead techniques for avoiding exact and near-breakdowns in these transpose-free methods still have to be developed.

3.5 Related work and historical remarks

The problem of breakdowns in the classical Lanczos algorithm has been known from the beginning. Although a rare event in practice, the possibility of breakdowns certainly has brought the method into discredit and has prevented many people from actually using the algorithm. On the other hand, it was also demonstrated (see Cullum and Willoughby, 1986) that the Lanczos process—even without look-ahead—is a powerful tool for sparse matrix computation.

The Lanczos method has intimate connections with many other areas of Mathematics, such as formally orthogonal polynomials (FOPs), Padé approximation, Hankel matrices, and control theory. The problem of breakdowns has a corresponding formulation in all of these areas, and remedies for breakdowns in these different settings have been known for quite some time. For example, the breakdown in the Lanczos process is equivalent to a breakdown of the generic three-term recurrence relation for FOPs, and it is well known how to overcome such breakdowns by modifying the recursions for FOPs (see Gragg, 1974, Draux, 1983, Gutknecht, 1990b, and the references given there). Kung (1977) and Gragg and Lindquist (1983) presented remedies for breakdowns in the context of the partial realization problem in control theory. The Lanczos process is also closely related to fast algorithms for the factorization of Hankel matrices, and again it is well known how to overcome possible breakdowns of these algorithms (see Heinig and Rost, 1984). However, in all these cases, only the problem of exact breakdowns has been addressed. Taylor (1982) and Parlett *et al.* (1985) were the first to propose a modification of the classical Lanczos process that remedies both exact and near-breakdowns.

In recent years, there has been a revival of the nonsymmetric Lanczos algorithm, and since 1990, in addition to the papers we have already cited in this section, there are several others dealing with various aspects of the Lanczos process. We refer the reader to the papers by Boley *et al.* (1991), Boley and Golub (1991), Brezinski *et al.* (1991), Gutknecht (1990c), Joubert (1990), Parlett (1990), and the references given therein.

Note that Algorithm 3.2 is only one of several possible implementations of the BCG approach; see Joubert (1990) and Gutknecht (1990a) for an overview of the different BCG variants. As for the nonsymmetric Lanczos process, exact and near-breakdowns in the BCG methods can be avoided by incorporating look-ahead procedures. Such look-ahead BCG algorithms have been proposed by Joubert (1990) and Gutknecht (1990c). Particularly attractive in this context is the algorithm called Lanczos/Orthodir in Joubert (1990). Instead of generating the search directions $\mathbf{q}_n$ and $\tilde{\mathbf{q}}_n$ by coupled two-term recursions as in Algorithm 3.2, in Lanczos/Orthodir they are computed by three-term recurrences. This eliminates the vectors $\mathbf{r}_n$ and $\tilde{\mathbf{r}}_n$, and hence the second of the two possible breakdowns (3.5) and (3.6). We note that Brezinski *et al.* (1991) have proposed an implementation of the BCG approach that is mathematically equivalent to Lanczos/Orthodir.

Finally, recall that the algorithms QMR, Bi-CGSTAB, and TFQMR are designed to generate iterates that converge more smoothly than BCG and CGS. A different remedy for the erratic convergence behavior of BCG or CGS was proposed by Schönauer (see Weiss, 1990). The approach used is to run plain BCG or CGS and then apply a smoothing procedure to the sequence of BCG or CGS iterates, resulting in iterates with monotonically decreasing residual norms. However, since the process is based directly on BCG or CGS, this approach inherits the numerical problems of BCG and CGS.

4. Solving the normal equations is not always bad

In this section, we consider CGNR and CGNE in more detail. Recall from Section 1 that CGNR and CGNE are the algorithms that result when (1.1) is solved by applying standard CG to either one of the normal equations (1.2) or (1.3), respectively. Clearly, for nonsingular A, both systems (1.2) and (1.3) are equivalent to the original system (1.1). From the minimization property (2.10) of CG, it follows that CGNR produces iterates

$$\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n\left(A^H\mathbf{r}_0, A^H A\right), \quad n = 1, 2, \ldots, \tag{4.1}$$

that are characterized by the minimal residual condition

$$\|\mathbf{b} - A\mathbf{x}_n\|_2 = \min_{\mathbf{x}\in\mathbf{x}_0+\mathcal{K}_n(A^H\mathbf{r}_0, A^H A)} \|\mathbf{b} - A\mathbf{x}\|_2.$$

Similarly, CGNE generates iterates (4.1) that satisfy the minimal error property

$$\|A^{-1}\mathbf{b} - \mathbf{x}_n\|_2 = \min_{\mathbf{x}\in\mathbf{x}_0+\mathcal{K}_n(A^H\mathbf{r}_0, A^H A)} \|A^{-1}\mathbf{b} - \mathbf{x}\|_2.$$

Note that the letters "R" and "E" in CGNR and CGNE indicate the minimization of the residual or of the error, respectively. We also remark that the LSQR algorithm of Paige and Saunders (1982) is mathematically equivalent to CGNR, but has better numerical properties.

Since the convergence of CG depends on the spectrum of the coefficient matrix, the convergence behavior of CGNR and CGNE depends on

$$\lambda(A^H A) = \{\sigma^2 \mid \sigma \in \sigma(A)\},$$

i.e., on the squares of the singular values of A. In particular, the worst-case convergence behavior of CGNR and CGNE is governed by

$$\kappa_2\left(A^H A\right) = (\kappa_2(A))^2,$$

which suggests that convergence can be very slow even for matrices A with moderate condition numbers. This is indeed true in many cases, and generally it is preferable to use CG-type methods that are applied directly to (1.1) rather than CGNR or CGNE.

Nevertheless there are special cases for which solving the normal equations is optimal. A simple case are unitary matrices A for which CGNR and CGNE find the exact solution after only one step, while Krylov subspace methods with iterates (2.1) tend to converge very slowly (see Nachtigal *et al.*, 1990a). More interesting are cases for which CGNR and CGNE are optimal, in that they are mathematically equivalent to ideal CG-type methods based on the MR or OR conditions (2.11) or (2.12). Typically, these situations arise when the spectrum of A has certain symmetries. Since these equivalences are not widely known, we collect here a few of these results. In the following, $\mathbf{x}_n^{\mathrm{MR}}$ and $\mathbf{x}_n^{\mathrm{OR}}$ denote iterates defined by (2.11) and (2.12), respectively.

First, we consider the case of real skew-symmetric matrices.

Theorem 4.1 (Freund, 1983)
Let $A = -A^T$ be a real $N \times N$ matrix, and let $\mathbf{b} \in \mathbb{R}^N$ and $\mathbf{x}_0 \in \mathbb{R}^N$. Then:

$$\begin{aligned} \mathbf{x}_{2n+1}^{\mathrm{MR}} &= \mathbf{x}_{2n}^{\mathrm{MR}} = \mathbf{x}_n^{\mathrm{CGNR}}, \quad n = 0, 1, \ldots, \\ \mathbf{x}_{2n}^{\mathrm{OR}} &= \mathbf{x}_n^{\mathrm{CGNE}}, \quad n = 1, 2, \ldots . \end{aligned}$$

Moreover, no odd OR iterate $\mathbf{x}_{2n-1}^{\mathrm{OR}}$ exists.

Next, we turn to shifted skew-symmetric matrices of the form (2.13). For this class of matrices Eisenstat (1983a, 1983b) has obtained the following result (see also Szyld and Widlund, 1989).

Theorem 4.2 Let $A = I - S$ where $S = -S^T$ is a real $N \times N$ matrix, and let $\mathbf{b} \in \mathbb{R}^N$ and $\mathbf{x}_0 \in \mathbb{R}^N$. Let $\mathbf{x}_n^{\mathrm{CGNE}}$ and $\tilde{\mathbf{x}}_n^{\mathrm{CGNE}}$ denote the iterates generated by CGNE started with initial guess $\mathbf{x}_0$ and $\tilde{\mathbf{x}}_0 = \mathbf{b} + S\mathbf{x}_0$, respectively. Then, for $n = 0, 1, \ldots$, it holds:

$$\begin{aligned} \mathbf{x}_{2n}^{\mathrm{OR}} &= \mathbf{x}_n^{\mathrm{CGNE}}, \\ \mathbf{x}_{2n+1}^{\mathrm{OR}} &= \tilde{\mathbf{x}}_n^{\mathrm{CGNE}}. \end{aligned}$$

We remark that for the MR and CGNR approaches a result corresponding to Theorem 4.2 does not hold (see Freund, 1983).

Finally, we consider linear systems (1.1) with Hermitian coefficient matrices A. Note that A has a complete set of orthonormal eigenvectors, and hence one can always expand the initial residual in the form:

$$\mathbf{r}_0 = \sum_{j=1}^{m} \rho_j \mathbf{z}_j \tag{4.2}$$

where

$$\rho_j \in \mathbb{C}, \quad A\mathbf{z}_j = \lambda_j \mathbf{z}_j, \quad \lambda_1 < \lambda_2 < \cdots < \lambda_m, \quad \mathbf{z}_j^H \mathbf{z}_k = \delta_{jk}.$$

In the following theorem, $\mathbf{x}_n^{\mathrm{ME}}$ denotes the iterate defined by (2.16).

Theorem 4.3 (Freund, 1983)
Assume that the expansion (4.2) is "symmetric", i.e., $m = 2l$ is even and

$$\lambda_j = -\lambda_{m+1-j}, \quad |\rho_j| = |\rho_{m+1-j}|, \quad j = 1, 2, \ldots, l.$$

Then, for $n = 0, 1, \ldots$, it holds:

$$\begin{aligned} \mathbf{x}_{2n+1}^{\mathrm{MR}} &= \mathbf{x}_{2n}^{\mathrm{MR}} = \mathbf{x}_n^{\mathrm{CGNR}}, \quad n = 0, 1, \ldots, \\ \mathbf{x}_{2n}^{\mathrm{ME}} &= \mathbf{x}_{2n-1}^{\mathrm{ME}} = \mathbf{x}_{2n}^{\mathrm{OR}} = \mathbf{x}_n^{\mathrm{CGNE}}, \quad n = 1, 2, \ldots . \end{aligned}$$

Moreover, no odd OR iterate $\mathbf{x}_{2n-1}^{\mathrm{OR}}$ exists.

5. Estimating spectra for hybrid methods

We now turn to parameter-dependent schemes. As mentioned in Section 2.1, methods in this class require a priori knowledge of some spectral properties of the matrix. Typically, it is assumed that some compact set $\mathcal{G}$ is known, which in a certain sense approximates the spectrum of A and in particular satisfies (2.7). Given such a set $\mathcal{G}$, one then constructs residual polynomials ψ_n as some approximations to the optimal polynomials in (2.9). We would like to stress that, in view of (2.4) and (2.9), it is crucial that $\mathcal{G}$ excludes the origin. Since in general one does not know in advance a suitable set $\mathcal{G}$, parameter-dependent methods combine an approach for estimating the set $\mathcal{G}$ with an approach for solving the approximation problem (2.9), usually cycling between the two parts in order to improve the estimate for $\mathcal{G}$. For this reason, the algorithms in this class are often called hybrid methods. In this section, we review some recent insights in the problem of how to estimate the set $\mathcal{G}$.

The standard approach for obtaining $\mathcal{G}$ is to run a few steps of the Arnoldi Algorithm 2.2 to generate the upper Hessenberg matrix H_n (2.24) and then compute its spectrum $\lambda(H_n)$. Saad (1980) showed that the eigenvalues of H_n are Ritz values for A and can be used to approximate the spectrum

$\lambda(A)$, hence one takes the convex hull of $\lambda(H_n)$ as the set $\mathcal{G}$. Once the set is obtained, a hybrid method turns to solving the complex approximation problem (2.9), and the possibilities here are numerous.

However, there are problems with the approach outlined above, originating from the use of the Arnoldi Ritz values. In general, there is no guarantee that the convex hull of $\lambda(H_n)$ does not include the origin, or indeed, that one of the Ritz values is not at or near the origin. For matrices with spectrum in both the left and right half-planes, the convex hull of the Ritz values might naturally enclose the origin. Nachtigal *et al.* (1990b) give an example of a matrix whose spectrum is symmetric with respect to the origin, so that the convex hull of $\lambda(H_n)$ will generally contain the origin, and on every other step, one of the Ritz values will be close to the origin. A second problem is that the approach aims to estimate the spectrum of A, which may be highly sensitive to perturbations, especially for nonnormal matrices. In these cases, a more natural concept is the pseudospectrum $\lambda_\epsilon(A)$, introduced by Trefethen (1991):

Definition 5.1 For $\epsilon \geq 0$ and $A \in \mathbb{C}^{N \times N}$, the ϵ-pseudospectrum $\lambda_\epsilon(A)$ is given by

$$\lambda_\epsilon(A) = \{\lambda \in \mathbb{C} \bigm| \lambda \in \lambda(A + \Delta),\ \|\Delta\|_2 \leq \epsilon\}. \tag{5.1}$$

As is apparent from (5.1), the pseudospectrum in general can be expected to be insensitive to perturbations of A, even for highly nonnormal matrices. For practical purposes, the sets $\lambda_\epsilon(A)$ of interest correspond to values of the parameter ϵ that are small relative to the norm of A but larger than round-off.

It is easy to construct examples where a hybrid algorithm using the exact spectrum $\lambda(A)$ of A will in fact diverge, while the same hybrid algorithm using the pseudospectrum $\lambda_\epsilon(A)$ will converge (see Nachtigal *et al.*, 1990b, and Trefethen, 1991). Unfortunately, in general the pseudospectrum $\lambda_\epsilon(A)$ cannot be easily computed directly. Fortunately, it turns out that one can compute approximations to the pseudospectrum; this is the approach taken in the hybrid introduced by Nachtigal *et al.* (1990b). They observed that the level curves—or lemniscates—of the GMRES residual polynomial bound regions that approximate the pseudospectrum $\lambda_\epsilon(A)$. Let

$$\mathcal{C}_n(\eta) = \{\lambda \in \mathbb{C} \bigm| |\psi_n(\lambda)| = \eta\}, \quad \eta \geq 0,$$

be any lemniscate of the residual polynomial ψ_n. Due to the normalization (2.4), the region bounded by $\mathcal{C}_n(\eta)$ with $\eta < 1$ will automatically exclude the origin; in particular, 0 cannot be a root of a residual polynomial. In addition, since GMRES is solving the minimization problem (2.11), ψ_n will naturally be small on an appropriate set $\mathcal{G}$. Motivated by these considerations, Nachtigal *et al.* observed that the region bounded by the lemniscate

$\mathcal{C}_n(\eta_n)$ with

$$\eta_n = \frac{\|\mathbf{r}_n\|_2}{\|\mathbf{r}_0\|_2} < 1$$

usually yields a suitable set $\mathcal{G}$.

The GMRES residual polynomials are kernel polynomials, and their roots are a type of Ritz values called pseudo-Ritz values by Freund (1989a). In fact, one is not restricted to using the GMRES residual polynomial, but could use the residual polynomials from other minimization methods, such as QMR or its transpose-free versions. In these cases, the residual polynomials are no longer kernel polynomials, but rather, they are quasi-kernel polynomials (Freund, 1991c). Nevertheless, their lemniscates still yield suitable sets $\mathcal{G}$, and their roots are also pseudo-Ritz values of A.

Freund has also proposed an algorithm to compute pseudo-Ritz values, using the upper Hessenberg matrix $H_n^{(e)}$ (2.19) appearing in the recurrence (2.17). One uses the fact that kernel and quasi-kernel polynomials can be defined by

$$\psi_n(\lambda) \equiv \frac{\det\left(\left(H_n^{(e)}\right)^H \left(H_n^{(e)}\right) - \lambda H_n^H\right)}{\det\left(\left(H_n^{(e)}\right)^H \left(H_n^{(e)}\right)\right)},$$

which makes it clear that the roots of ψ_n can be obtained from the generalized eigenvalue problem

$$\left(H_n^{(e)}\right)^H \left(H_n^{(e)}\right) \mathbf{z} = \lambda H_n^H \mathbf{z}, \quad \mathbf{z} \neq 0 \in \mathbb{C}^n.$$

Finally, we should point out another set that has been proposed as a candidate for $\mathcal{G}$ in (2.7), namely the field of values. Defined as

$$\mathcal{F} = \{\mathbf{z}^H A \mathbf{z} \;\big|\; \mathbf{z} \in \mathbb{C}^N,\ \|\mathbf{z}\|_2 = 1\},$$

the field of values has the advantage that it is easier to compute (see, e.g., Ruhe, 1987) than the pseudospectrum. However, the field of values is convex and always at least as large as the convex hull of $\lambda(A)$, and hence may once again enclose the origin. For more details on the field of values in iterative methods, we refer the reader to Eiermann (1991).

The literature on hybrid methods for non-Hermitian linear systems starts with an algorithm proposed by Manteuffel (1978), which combines a modified power method with the Chebyshev iteration. Since then, literally dozens of hybrid methods have been proposed, most of which use Arnoldi in the first phase and differ in the way they compute the residual polynomial ψ_n in (2.9). For an overview of some of these algorithms, see Nachtigal *et al.* (1990b). The hybrid by Nachtigal *et al.* was the first to avoid the explicit computation of an estimate of $\lambda(A)$. Instead, it explicitly obtains the GMRES residual polynomial and applies it using Richardson's method until convergence. Finally, hybrids recently introduced include an algorithm by Saylor and Smolarski (1991), which combines Arnoldi with Richardson's method, and an algorithm proposed by Starke and Varga (1991), which combines Arnoldi with Faber polynomials.

6. CG-type methods for complex linear systems

While most linear systems arising in practice are real, there are important applications that lead to linear systems with complex* coefficient matrices A. Partial differential equations that model dissipative processes usually involve complex coefficient functions or complex boundary conditions, and discretizing them yields complex linear systems. An important example for this category is the complex Helmholtz equations. Other applications that lead to complex linear systems include the numerical solution of Schrödinger's equation, underwater acoustics, frequency response computations in control theory, semiconductor device simulation, and numerical computations in quantum chromodynamics; for details and further references, see Bayliss and Goldstein (1983), Barbour *et al.* (1987), Freund (1989b, 1991a), and Laux (1985). In all these applications, the resulting linear systems are usually non-Hermitian. In this section, we review some recent advances in understanding the issues related to solving complex non-Hermitian systems.

Until recently, the prevailing approaches used when solving complex linear systems have consisted of either solving the normal equations or rewriting the complex system as a real system of dimension $2N$. However, as indicated already in Section 4, the normal equations often lead to systems with very poor convergence rates, and this has indeed been observed for many of the complex systems of interest. The other option is to split the original matrix A into its real and imaginary parts and combine them into a real system of dimension $2N$. There are essentially only two different possibilities for doing this, namely

$$A_\star \begin{bmatrix} \operatorname{Re} \mathbf{x} \\ \operatorname{Im} \mathbf{x} \end{bmatrix} = \begin{bmatrix} \operatorname{Re} \mathbf{b} \\ \operatorname{Im} \mathbf{b} \end{bmatrix}, \quad A_\star := \begin{bmatrix} \operatorname{Re} A & -\operatorname{Im} A \\ \operatorname{Im} A & \operatorname{Re} A \end{bmatrix}, \tag{6.1}$$

and

$$A_{\star\star} \begin{bmatrix} \operatorname{Re} \mathbf{x} \\ -\operatorname{Im} \mathbf{x} \end{bmatrix} = \begin{bmatrix} \operatorname{Re} \mathbf{b} \\ \operatorname{Im} \mathbf{b} \end{bmatrix}, \quad A_{\star\star} := \begin{bmatrix} \operatorname{Re} A & \operatorname{Im} A \\ \operatorname{Im} A & -\operatorname{Re} A \end{bmatrix}. \tag{6.2}$$

Unfortunately, it turns out that this option is not viable either. As Freund (1989b, 1992) pointed out, both $A_\star$ and $A_{\star\star}$ have spectral properties that make them far more unsuitable for Krylov space iterations than the original

* In this section, "complex" will imply the presence of imaginary components.

system was. The spectrum of $A_\star$ in (6.1) is given by

$$\lambda(A_\star) = \lambda(A) \cup \overline{\lambda(A)}, \tag{6.3}$$

while the spectrum of $A_{\star\star}$ in (6.2) is given by

$$\lambda(A_{\star\star}) = \{\lambda \in \mathbb{C} \mid \lambda^2 \in \lambda(\overline{A}A)\}. \tag{6.4}$$

In particular, note that the spectrum (6.3) is symmetric with respect to the real axis, while the spectrum (6.4) is symmetric with respect to both the real and imaginary axes. The point is that in both cases, the spectrum of the real system is either very likely to or is guaranteed to contain the origin, thus presenting a Krylov subspace iteration with the worst possible eigenvalue distribution. As a result, both approaches have had very poor results in practice, and have often led to the conclusion that complex systems are ill-suited for iterative methods.

Therefore, instead of solving the normal equations or either one of the equivalent real systems (6.1) or (6.2), it is generally preferable to solve the original linear system by a CG-like Krylov subspace methods. In particular, if A is a general non-Hermitian matrix, then we recommend to use the Lanczos-based algorithms discussed in Section 3 or GMRES. However, in many applications, the resulting complex linear systems have additional structure that can be exploited. For example, often complex matrices of the form (2.13) arise. Recall from Theorem 2.1 that ideal CG-type algorithms based on the MR or OR property (2.11) or (2.12) exist for such shifted Hermitian matrices. Freund (1990) has derived practical implementations of the MR and OR approaches for the class of matrices (2.13). These algorithms can be viewed as extensions of Paige and Saunders' MINRES and SYMMLQ for Hermitian matrices (see Section 2.3). As in the case of SYMMLQ, the OR implementation for shifted Hermitian matrices also generates auxiliary iterates $\mathbf{x}_n^{\mathrm{ME}} \in \mathbf{x}_0 + \mathcal{K}_n(A^H\mathbf{r}_0, A)$ that are characterized by the minimal error property

$$\|A^{-1}\mathbf{b} - \mathbf{x}_n^{\mathrm{ME}}\|_2 = \min_{\mathbf{x} \in \mathbf{x}_0 + \mathcal{K}_n(A^H\mathbf{r}_0, A)} \|A^{-1}\mathbf{b} - \mathbf{x}\|_2.$$

Hence the OR algorithm proposed by Freund also generalizes Fridman's method to shifted Hermitian matrices. Unfortunately, when matrices A of the form (2.13) are preconditioned by standard techniques, the special structure of A is destroyed. In Freund (1990) it is shown that the shift structure can be preserved when polynomial preconditioning is used, and results on the optimal choice of the polynomial preconditioner are given.

Another special case that arises frequently in applications are complex symmetric matrices $A = A^T$. For example, the complex Helmholtz equations leads to complex symmetric systems. As pointed out in Section 3.4, the QMR Algorithm 3.4 can take advantage of this special structure, and work and storage is roughly halved. We remark that the complex symmetry structure is preserved when preconditioning is used, if the preconditioner M is again symmetric, as is the case for all standard techniques. For an overview of other CG-type methods and further results for complex symmetric linear systems, we refer the reader to Freund (1991a, 1992).

7. Large dense linear systems

As mentioned in Section 1, there are certain classes of large dense linear systems for which it is possible to compute matrix-vector products cheaply; for these systems, iterative methods remain an attractive option. Typically, the matrix-vector product takes advantage of either some special structure of the matrix or of some special property of the underlying operator. We will briefly discuss one situation from each class.

The first case is the solution of integral equations involving a decaying potential, such as a gravitational or Coulombic potential. Some typical applications are the solution of N-body problems, vortex methods, potential theory, and others. In these problems, the effort required by a naive computation of a matrix-vector product is generally $\mathcal{O}(N^2)$, as it involves computing the influence of each of N points on all the other $N-1$ points. However, Carrier *et al.* (1988) noticed that it is possible to approximate the result of the matrix-vector product $A\mathbf{x}$ in only $\mathcal{O}(N)$ time, rather than $\mathcal{O}(N^2)$.

The main idea behind their algorithm is to group points in clusters and evaluate the influence of entire clusters on faraway points. The cumulative potential generated by m charges in a cluster can be expressed as a power series. If the power series is truncated after p terms, then the work required to compute it turns out to be $\mathcal{O}(mp)$. Applying the cumulative influence to n points in a cluster well-separated from the first requires an additional $\mathcal{O}(np)$ work, for a total of $\mathcal{O}(mp+np)$ work, as compared to the $\mathcal{O}(mn)$ work required to compute the individual interactions. Finally, the point is that the number p of terms in the series is determined by the preassigned precision to which the series is computed, and once chosen, p becomes a constant, giving an $\mathcal{O}(m+n)$ algorithm. For a complete description of the algorithm, see Carrier *et al.* (1988).

While the Carrier *et al.* algorithm is the first in the class of fast multipole methods, it falls in the bigger class of hierarchical algorithms. Several other algorithms have been proposed in this class; see for example Hockney and Eastwood (1981), Appel (1985), Rokhlin (1985). More recently, Hanrahan *et al.* (1991) have proposed a hierarchical algorithm for radiosity problems in computer graphics, and Greenbaum *et al.* (1991) have introduced an algorithm that uses the Carrier *et al.* algorithm combined with GMRES to solve Laplace's equation in multiply connected domains. We refer the reader to these papers for details of the algorithms, as well as further references.

Another important class of dense linear systems where the matrix-vector product can be computed cheaply are Toeplitz systems, where the coefficient matrix A has the form

$$A = \begin{bmatrix} t_0 & t_1 & \cdots & t_{N-1} \\ t_{-1} & t_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_1 \\ t_{-(N-1)} & \cdots & t_{-1} & t_0 \end{bmatrix}.$$

A matrix-vector product with an $N \times N$ Toeplitz matrix can be computed with $\mathcal{O}(N \log N)$ operations by means of the fast Fourier transform. Furthermore, as Chan and Strang (1989) observed, Toeplitz systems can be preconditioned efficiently using circulant matrices

$$M = \begin{bmatrix} c_0 & c_1 & \cdots & \cdots & c_{N-1} \\ c_{N-1} & c_0 & \ddots & & c_{N-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & c_1 \\ c_1 & \cdots & \cdots & c_{N-1} & c_0 \end{bmatrix}.$$

Recall from Section 2.5 that preconditioned iterative methods require the solution of linear systems with the preconditioner M in each iteration. In the case of circulant M, these systems can also be solved by means of fast Fourier transform with only $\mathcal{O}(N \log N)$ operations per system.

8. Concluding remarks

In conclusion, we have covered some of the developments in iterative methods, especially for non-Hermitian matrices. The introduction of CGS in the late eighties spurred renewed interest in the nonsymmetric Lanczos algorithm, with most of the effort directed towards obtaining a method with better convergence properties than BCG or CGS. Several BCG-based algorithms were proposed, such as Bi-CGSTAB, introduced by Van der Vorst (1990). The quasi-minimal residual technique was introduced by Freund (1989b) in the context of complex symmetric systems, then later coupled with a new variant of the look-ahead Lanczos approach to obtain a general non-Hermitian QMR algorithm. Finally, several transpose-free algorithms based on QMR have been introduced recently, which trade the multiplication by A^T for one or more multiplications by A. However, their convergence properties are not well understood, and none of these algorithms have been combined with look-ahead techniques yet. In general, it seems that the transpose-free methods have more numerical problems than the corresponding methods that use A^T, and more research is needed into studying their behavior.

With the advent of Lanczos-based methods that require little work and storage per step, the importance of preconditioners has decreased. With GMRES or similar algorithms, the high cost of the method makes restarts necessary in practice, which generally results in much slower convergence. As a result, preconditioned GMRES requires a very effective preconditioner, so that the preconditioned system requires few iterations to converge. The problem is that effective preconditioners are often too expensive, especially on parallel machines, where the inherently serial nature of many preconditioners makes their use unappealing. In contrast, restarts are not necessary with Lanczos-based methods, and hence a wider array of preconditioners—in particular, cheaper or more parallelizable preconditioners—becomes usable.

Finally, even though the field of iterative methods has made great progress in the last few years, it is still in its infancy, especially with regard to the packaged software available. Whereas there are well-established robust general-purpose solvers based on direct methods, the same cannot be said about solvers based on iterative methods. There are no established iterative packages of the same robustness and wide acceptance as, for example, the LINPACK library, and as a result many of the scientists who use iterative methods write their own specialized solvers. We feel that this situation needs to change, and we would like to encourage researchers to provide code for their methods.

Acknowledgments

We would like to thank Susanne Freund for her help with LaTeX and Marlis Hochbruck for her careful reading of this paper. The first and third authors are grateful to the San Jose radio station KSJO for keeping them awake late at night. The first author would like to thank the Printers Inc Book Store in Mountain View for providing countless mega espressos.

REFERENCES

A.W. Appel (1985), "An efficient program for many-body simulation", *SIAM J. Sci. Stat. Comput.* **6**, 85–103.

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

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

O. Axelsson (1980), "Conjugate gradient type methods for unsymmetric and inconsistent systems of linear equations", *Linear Algebra Appl.* **29**, 1–16.

O. Axelsson (1985), "A survey of preconditioned iterative methods for linear systems of algebraic equations", *BIT* **25**, 166–187.

O. Axelsson (1987), "A generalized conjugate gradient, least square method", *Numer. Math.* **51**, 209–227.

A. Bayliss and C.I. Goldstein (1983), "An iterative method for the Helmholtz equation", *J. Comput. Phys.* **49**, 443–457.

I.M. Barbour, N.-E. Behilil, P.E. Gibbs, M. Rafiq, K.J.M. Moriarty and G. Schierholz (1987), "Updating fermions with the Lanczos method", *J. Comput. Phys.* **68**, 227–236.

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

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

C. Brezinski, M. Redivo Zaglia and H. Sadok (1991), "A breakdown-free Lanczos type algorithm for solving linear systems", Technical Report ANO–239, Université des Sciences et Techniques de Lille Flandres-Artois, Villeneuve d'Ascq.

C.G. Broyden (1965), "A class of methods for solving nonlinear simultaneous equations", *Math. Comp.* **19**, 577–593.

J. Carrier, L. Greengard and V. Rokhlin (1988), "A fast adaptive multipole algorithm for particle simulations", *SIAM J. Sci. Stat. Comput.* **9**, 669–686.

R.H. Chan and G. Strang (1989), "Toeplitz equations by conjugate gradients with circulant preconditioner", *SIAM J. Sci. Stat. Comput.* **10**, 104–119.

T.F. Chan, L. de Pillis and H.A. Van der Vorst (1991), "A transpose-free squared Lanczos algorithm and application to solving nonsymmetric linear systems", Technical Report, University of California at Los Angeles.

R. Chandra (1978), "Conjugate gradient methods for partial differential equations", Ph.D. Dissertation, Yale University, New Haven.

P. Concus and G.H. Golub (1976), "A generalized conjugate gradient method for nonsymmetric systems of linear equations", in *Computing Methods in Applied Sciences and Engineering* (R. Glowinski and J.L. Lions, eds.), Lecture Notes in Economics and Mathematical Systems 134, Springer, Berlin, 56–65.

P. Concus, G.H. Golub and D.P. O'Leary (1976), "A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations", in *Sparse Matrix Computations* (J.R. Bunch and D.J. Rose, eds.), Academic Press, New York, 309–332.

E.J. Craig (1955), "The N-step iteration procedures", *J. Math. Phys.* **34**, 64–73.

J. Cullum and R.A. Willoughby (1985) *Lanczos algorithms for large symmetric eigenvalue computations, Volume 1, Theory*, Birkhäuser, Basel.

J. Cullum and R.A. Willoughby (1986) "A practical procedure for computing eigenvalues of large sparse nonsymmetric matrices", in *Large Scale Eigenvalue Problems* (J. Cullum and R.A. Willoughby, eds.), North-Holland, Amsterdam, 193–240.

P. Deuflhard, R.W. Freund and A. Walter (1990), "Fast secant methods for the iterative solution of large nonsymmetric linear systems", *IMPACT Comput. Sci. Eng.* **2**, 244–276.

A. Draux (1983), *Polynômes Orthogonaux Formels – Applications*, Lecture Notes in Mathematics 974, Springer, Berlin.

I.S. Duff, A.M. Erisman and J.K. Reid (1986), *Direct Methods for Sparse Matrices*, Oxford University Press, Oxford.

M. Eiermann (1991), "Fields of values and iterative methods", Preprint, IBM Scientific Center, Heidelberg.

M. Eiermann, W. Niethammer and R.S. Varga (1985), "A study of semiiterative methods for nonsymmetric systems of linear equations", *Numer. Math.* **47**, 505–533.

T. Eirola and O. Nevanlinna (1989), "Accelerating with rank-one updates", *Linear Algebra Appl.* **121**, 511–520.

S.C. Eisenstat (1983a), "A note on the generalized conjugate gradient method", *SIAM J. Numer. Anal.* **20**, 358–361.

S.C. Eisenstat (1983b), "Some observations on the generalized conjugate gradient method", in *Numerical methods, Proceedings, Caracas 1982* (V. Pereyra and A. Reinoza, eds.), Lecture Notes in Mathematics 1005, Springer, Berlin, 99–107.

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

H.C. Elman (1982), "Iterative methods for large sparse nonsymmetric systems of linear equations", Ph.D. Dissertation, Yale University, New Haven.

V. Faber and T. Manteuffel (1984), "Necessary and sufficient conditions for the existence of a conjugate gradient method", *SIAM J. Numer. Anal.* **21**, 352–362.

V. Faber and T. Manteuffel (1987), "Orthogonal error methods", *SIAM J. Numer. Anal.* **24**, 170–187.

B. Fischer and R.W. Freund (1990), "On the constrained Chebyshev approximation problem on ellipses", *J. Approx. Theory* **62**, 297–315.

B. Fischer and R.W. Freund (1991), "Chebyshev polynomials are not always optimal", *J. Approx. Theory* **65**, 261–272.

R. Fletcher (1976), "Conjugate gradient methods for indefinite systems", in *Numerical Analysis Dundee 1975* (G.A. Watson, ed.), Lecture Notes in Mathematics 506, Springer, Berlin, 73–89.

R.W. Freund (1983), "Über einige cg-ähnliche Verfahren zur Lösung linearer Gleichungssysteme", Doctoral Dissertation, Universität Würzburg.

R.W. Freund (1989a), "Pseudo-Ritz values for indefinite Hermitian matrices", RIACS Technical Report 89.33, NASA Ames Research Center, Moffett Field.

R.W. Freund (1989b), "Conjugate gradient type methods for linear systems with complex symmetric coefficient matrices", RIACS Technical Report 89.54, NASA Ames Research Center, Moffett Field.

R.W. Freund (1990), "On conjugate gradient type methods and polynomial preconditioners for a class of complex non-Hermitian matrices", *Numer. Math.* **57**, 285–312.

R.W. Freund (1991a), "Krylov subspace methods for complex non-Hermitian linear systems", Habilitation Thesis, Universität Würzburg.

R.W. Freund (1991b), "A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems", RIACS Technical Report 91.18, NASA Ames Research Center, Moffett Field.

R.W. Freund (1991c), "Quasi-kernel polynomials and their use in non-Hermitian matrix iterations", RIACS Technical Report 91.20, NASA Ames Research Center, Moffett Field.

R.W. Freund (1992), "Conjugate gradient-type methods for linear systems with complex symmetric coefficient matrices", *SIAM J. Sci. Stat. Comput.* **13**, to appear.

R.W. Freund, G.H. Golub and M. Hochbruck (1991a) "Krylov subspace methods for non-Hermitian p-cyclic matrices", Technical Report, RIACS, NASA Ames Research Center, Moffett Field.

R.W. Freund, M.H. Gutknecht and N.M. Nachtigal (1991b), "An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices", Technical Report 91.09, RIACS, NASA Ames Research Center, Moffett Field.

R.W. Freund and N.M. Nachtigal (1991), "QMR: a quasi-minimal residual method for non-Hermitian linear systems", *Numer. Math.*, to appear.

R.W. Freund and St. Ruscheweyh (1986), "On a class of Chebyshev approximation problems which arise in connection with a conjugate gradient type method", *Numer. Math.* **48**, 525–542.

R.W. Freund and T. Szeto (1991), "A quasi-minimal residual squared algorithm for non-Hermitian linear systems", Technical Report, RIACS, NASA Ames Research Center, Moffett Field.

R.W. Freund and H. Zha (1991), "Simplifications of the nonsymmetric Lanczos process and a new algorithm for solving indefinite symmetric linear systems", Technical Report, RIACS, NASA Ames Research Center, Moffett Field.

V.M. Fridman (1963), "The method of minimum iterations with minimum errors for a system of linear algebraic equations with a symmetrical matrix", *USSR Comput. Math. and Math. Phys.* **2**, 362–363.

P.E. Gill, W. Murray, D.B. Ponceleón and M.A. Saunders (1990), "Preconditioners for indefinite systems arising in optimization", Technical Report SOL 90-8, Stanford University.

G.H. Golub and D.P. O'Leary (1989), "Some history of the conjugate gradient and Lanczos algorithms: 1948–1976", *SIAM Review* **31**, 50–102.

G.H. Golub and C.F. Van Loan (1989), *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore.

G.H. Golub and R.S. Varga (1961), "Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods", *Numer. Math.* **3**, 147–168.

W.B. Gragg (1974), "Matrix interpretations and applications of the continued fraction algorithm", *Rocky Mountain J. Math.* **4**, 213–225.

W.B. Gragg and A. Lindquist (1983), "On the partial realization problem", *Linear Algebra Appl.* **50**, 277–319.

A. Greenbaum, L. Greengard and G.B. McFadden (1991), "Laplace's equation and the Dirichlet-Neumann map in multiply connected domains", Technical Report, Courant Institute of Mathematical Sciences, New York University.

M.H. Gutknecht (1990a), "The unsymmetric Lanczos algorithms and their relations to Padé approximation, continued fractions, and the QD algorithm", in *Proceedings of the Copper Mountain Conference on Iterative Methods*.

M.H. Gutknecht (1990b), "A completed theory of the unsymmetric Lanczos process and related algorithms, Part I", *SIAM J. Matrix Anal. Appl.*, to appear.

M.H. Gutknecht (1990c), "A completed theory of the unsymmetric Lanczos process and related algorithms, Part II", IPS Research Report No. 90–16, ETH Zürich.

M.H. Gutknecht (1991), "Variants of BiCGStab for matrices with complex spectrum", IPS Research Report No. 91–14, ETH Zürich.

P. Hanrahan, D. Salzman and L. Aupperle (1991), "A rapid hierarchical radiosity algorithm", *Computer Graphics (Proc. SIGGRAPH '91)* **25**, 197–206.

M.T. Heath, E. Ng and B.W. Peyton (1991), "Parallel algorithms for sparse linear systems", *SIAM Review* **33**, 420–460.

G. Heinig and K. Rost (1984), "Algebraic methods for Toeplitz-like matrices and operators", Birkhäuser, Basel.

M.R. Hestenes and E. Stiefel (1952), "Methods of conjugate gradients for solving linear systems", *J. Res. Nat. Bur. Stand.* **49**, 409–436.

R.W. Hockney and J.W. Eastwood (1981), "Computer Simulation Using Particles", McGraw-Hill, New York.

W.D. Joubert (1990), "Generalized conjugate gradient and Lanczos methods for the solution of nonsymmetric systems of linear equations", Ph.D. Dissertation, The University of Texas at Austin.

W.D. Joubert and D.M. Young (1987), "Necessary and sufficient conditions for the simplification of generalized conjugate-gradient algorithms", *Linear Algebra Appl.* **88/89**, 449–485.

I.M. Khabaza (1963), "An iterative least-square method suitable for solving large sparse matrices", *Comput. J.* **6**, 202–206.

S. Kung (1977), "Multivariable and multidimensional systems: analysis and design", Ph.D. Dissertation, Stanford University.

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

C. Lanczos (1952), "Solution of systems of linear equations by minimized iterations", *J. Res. Natl. Bur. Stand.* **49**, 33–53.

S.E. Laux (1985), "Techniques for small-signal analysis of semiconductor devices", *IEEE Trans. Electron Dev.* **ED-32**, 2028–2037.

D.G. Luenberger (1969), "Hyperbolic pairs in the method of conjugate gradients", *SIAM J. Appl. Math.* **17**, 1263–1267.

T.A. Manteuffel (1977), "The Tchebychev iteration for nonsymmetric linear systems", *Numer. Math.* **28**, 307–327.

T.A. Manteuffel (1978), "Adaptive procedure for estimating parameters for the nonsymmetric Tchebychev iteration", *Numer. Math.* **31**, 183–208.

T.A. Manteuffel (1991), "Recent developments in iterative methods for large sparse linear systems", Seminar presentation, Numerical Analysis/Scientific Computing Seminar, Stanford University.

N.M. Nachtigal (1991), "A look-ahead variant of the Lanczos algorithm and its application to the quasi-minimal residual method for non-Hermitian linear systems", Ph.D. Dissertation, Massachusetts Institute of Technology, Cambridge.

N.M. Nachtigal, S.C. Reddy and L.N. Trefethen (1990a), "How fast are nonsymmetric matrix iterations?", *SIAM J. Matrix Anal. Appl.*, to appear.

N.M. Nachtigal, L. Reichel and L.N. Trefethen (1990b), "A hybrid GMRES algorithm for nonsymmetric linear systems", *SIAM J. Matrix Anal. Appl.*, to appear.

C.C. Paige and M.A. Saunders (1975), "Solution of sparse indefinite systems of linear equations", *SIAM J. Numer. Anal.* **12**, 617–629.

C.C. Paige and M.A. Saunders (1982), "LSQR: an algorithm for sparse linear equations and sparse least squares", *ACM Trans. Math. Softw.* **8**, 43–71.

P.N. Parlett (1990), "Reduction to tridiagonal form and minimal realizations", Preprint, University of California at Berkeley.

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

D. Rapoport (1978), "A nonlinear Lanczos algorithm and the stationary Navier-Stokes equation", Ph.D. Dissertation, New York University.

J.K. Reid (1971), "On the method of conjugate gradients for the solution of large sparse systems of linear equations", in *Large Sparse Sets of Linear Equations* (J.K. Reid, ed.), Academic Press, New York, 231–253.

V. Rokhlin (1985), "Rapid solution of integral equations of classical potential theory", *J. Comp. Phys.* **60**, 187–207.

A. Ruhe (1987), "Closest normal matrix finally found!", *BIT* **27**, 585–598.

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

Y. Saad (1981), "Krylov subspace methods for solving large unsymmetric linear systems", *Math. Comp.* **37**, 105–126.

Y. Saad (1982), "The Lanczos biorthogonalization algorithm and other oblique projection methods for solving large unsymmetric systems", *SIAM J. Numer. Anal.* **19**, 485–506.

Y. Saad (1984), "Practical use of some Krylov subspace methods for solving indefinite and nonsymmetric linear systems", *SIAM J. Sci. Stat. Comput.* **5**, 203–227.

Y. Saad (1989), "Krylov subspace methods on supercomputers", *SIAM J. Sci. Stat. Comput.* **10**, 1200–1232.

Y. Saad and M.H. Schultz (1985), "Conjugate gradient-like algorithms for solving nonsymmetric linear systems", *Math. Comp.* **44**, 417–424.

Y. Saad and M.H. Schultz (1986), "GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems", *SIAM J. Sci. Stat. Comput.* **7**, 856–869.

P.E. Saylor and D.C. Smolarski (1991), "Implementation of an adaptive algorithm for Richardson's method", *Linear Algebra Appl.* **154–156**, 615–646.

P. Sonneveld (1989), "CGS, a fast Lanczos-type solver for nonsymmetric linear systems", *SIAM J. Sci. Stat. Comput.* **10**, 36–52.

G. Starke and R.S. Varga (1991), "A hybrid Arnoldi-Faber iterative method for nonsymmetric systems of linear equations", Technical Report ICM-9108-11, Kent State University, Kent.

E. Stiefel (1955), "Relaxationsmethoden bester Strategie zur Lösung linearer Gleichungssysteme", *Comm. Math. Helv.* **29**, 157–179.

J. Stoer (1983), "Solution of large linear systems of equations by conjugate gradient type methods", in *Mathematical Programming – The State of the Art* (A. Bachem, M. Grötschel and B. Korte, eds.), Springer, Berlin, 540–565.

J. Stoer and R.W. Freund (1982), "On the solution of large indefinite systems of linear equations by conjugate gradient algorithms", in *Computing Methods in Applied Sciences and Engineering V* (R. Glowinski and J.L. Lions, eds.), North-Holland, Amsterdam, 35–53.

D.B. Szyld and O.B. Widlund (1989), "Variational analysis of some conjugate gradient methods", Technical Report CS-1989-28, Duke University, Durham.

D.R. Taylor (1982), "Analysis of the look ahead Lanczos algorithm", Ph.D. Dissertation, University of California at Berkeley.

L.N. Trefethen (1991), "Pseudospectra of matrices", in *Proceedings of the 14th Dundee Biennial Conference on Numerical Analysis* (D.F. Griffiths and G.A. Watson, eds.), to appear.

H.A. Van der Vorst (1990), "Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems", Preprint, University of Utrecht.

P.K.W. Vinsome (1976), "Orthomin, an iterative method for solving sparse sets of simultaneous linear equations", in *Proceedings of the Fourth Symposium on Reservoir Simulation*, Society of Petroleum Engineers of AIME, Los Angeles.

V.V. Voevodin (1983), "The problem of a non-selfadjoint generalization of the conjugate gradient method has been closed", *USSR Comput. Math. and Math. Phys.* **23**, 143–144.

C. Vuik (1990), "A comparison of some GMRES-like methods", Technical Report, Delft University of Technology, Delft.

R. Weiss (1990), "Convergence behavior of generalized conjugate gradient methods", Doctoral Dissertation, Universität Karlsruhe.

O. Widlund (1978), "A Lanczos method for a class of nonsymmetric systems of linear equations", *SIAM J. Numer. Anal.* **15**, 801–812.

J.H. Wilkinson (1965), *The Algebraic Eigenvalue Problem*, Oxford University Press, Oxford.

D.M. Young and K.C. Jea (1980), "Generalized conjugate gradient acceleration of nonsymmetrizable iterative methods", *Linear Algebra Appl.* **34**, 159–194.