POLE PLACEMENT PRECONDITIONING

D. CALVETTI * AND L. REICHEL †

Abstract. The partial pole placement problem has received considerable attention in Control Theory, where it is applied to stabilize time-invariant linear control systems. This paper discusses application of the partial pole placement problem to the construction of preconditioners for linear systems of equations. Numerical examples show that these preconditioners can improve the rate of convergence of the restarted GMRES methods significantly.

Key words. iterative method, preconditioner, nonsymmetric linear system, Arnoldi process, implicitly restarted Arnoldi method, partial eigenvalue assignment problem

AMS subject classifications. 65F10, 65F15, 93B55

1. Introduction. Many problems in Science and Engineering give rise to linear systems of equations

\[ Ax = b, \quad A \in \mathbb{R}^{n \times n}, \quad x, b \in \mathbb{R}^n, \]

with a large sparse nonsymmetric nonsingular matrix $A$. It is often desirable, and sometimes necessary, to solve these systems by an iterative method. Let $x_0$ be an initial approximate solution of (1.1), and let $r_0 := b - Ax_0$ be the associated residual vector. Introduce the Krylov subspaces

\[ K_m(A, r_0) = \text{span}\{r_0, Ar_0, \ldots, A^{m-1}r_0\}, \quad m = 1, 2, 3, \ldots, \]

associated with the matrix $A$ and vector $r_0$. Many popular iterative methods determine the $m$th iterate, $x_m$, so that $x_m - x_0 \in K_m(A, r_0)$. We refer to such methods as Krylov subspace iterative methods; see, e.g., Axelsson [2] and Saad [17] for reviews.

Let the iterate $x_m$ be generated by a Krylov subspace iterative method. Then the residual error $r_m := b - Ax_m$ associated with $x_m$ satisfies

\[ r_m = p_m(A)r_0, \]

where $p_m$ is a polynomial of degree at most $m$, such that $p_m(0) = 1$, determined by the iterative method. The polynomial $p_m$ is referred to as the residual polynomial associated with the iterate $x_m$.

Let $\mathbb{P}_m^0$ denote the set of polynomials $p$ of degree at most $m$ and such that $p(0) = 1$, and let $\|\cdot\|$ denote the Euclidean vector norm, as well as the associated induced matrix norm. The GMRES method, by Saad and Schultz [18], is one of the most popular Krylov subspace iterative methods for the solution of linear systems of equations with a nonsymmetric matrix. The residual polynomials $p_m$ determined by the GMRES method satisfy

\[ \|p_m(A)r_0\| = \min_{p \in \mathbb{P}_m^0} \|p(A)r_0\|, \quad m = 1, 2, 3, \ldots. \]
Let \( m \leq n \). The standard implementation of the GMRES method is based on the Arnoldi decomposition

\[
(1.5) \quad A V_m = V_m H_m + \eta_m v_{m+1} e_m^T,
\]

where \( V_m \in \mathbb{R}^{n \times m} \), \( V_m^T V_m = I_m \), \( V_m e_1 = r_0 / \| r_0 \| \), \( H_m \in \mathbb{R}^{m \times m} \) is an upper Hessenberg matrix with positive subdiagonal entries, \( v_{m+1} \in \mathbb{R}^n \) satisfies \( V_m^T v_{m+1} = 0 \) and \( \eta_m \geq 0 \). If \( \eta_m > 0 \), then \( v_{m+1} \) is normalized so that \( \| v_{m+1} \| = 1 \); otherwise \( v_{m+1} = 0 \.

Here \( I_m \) denotes the \( m \times m \) identity matrix and \( e_j = [0, \ldots, 0, 1, 0, \ldots, 0]^T \) the \( j \)th axis vector. It follows from the relation (1.5) and the fact that the first column of \( V_m \) is a normalization of the vector \( r_0 \), that the columns of \( V_m \) span the Krylov subspace \( K_m(A, r_0) \). Here and throughout this paper, we assume that \( m \) is small enough so that a decomposition of the form (1.5) with the stated properties exists, because otherwise the numerical solution method simplifies. In fact, we also may assume that \( \eta_m > 0 \), because if \( \eta_m \) vanishes, then the solution \( x \) of (1.1) can be expressed as a linear combination of the columns of the matrix \( V_m \).

It is convenient to define the matrices

\[
(1.6) \quad H_{m+1, m} := \left[ \begin{array}{c} H_m \\ \eta_m e_m^T \end{array} \right] \in \mathbb{R}^{(m+1) \times m}, \quad V_{m+1} := [V_m, v_{m+1}] \in \mathbb{R}^{n \times (m+1)}
\]

and express the decomposition (1.5) as

\[
(1.7) \quad A V_m = V_{m+1} H_{m+1, m}.
\]

The minimization problem in the right-hand side of (1.4) can be expressed in terms of the Arnoldi decomposition (1.7),

\[
\min_{p \in \mathcal{P}_m} \| p(A) r_0 \| = \min_{y \in \mathbb{R}^m} \| A V_m y - r_0 \| = \min_{y \in \mathbb{R}^m} \| V_{m+1} H_{m+1, m} y - r_0 \|
\]

\[
= \min_{y \in \mathbb{R}^m} \| H_{m+1, m} y - r_0 \|_1.
\]

The solution \( y_m \in \mathbb{R}^m \) of the minimization problem (1.8) can be computed by QR-factorization of the matrix \( H_{m+1, m} \), and we obtain the new approximate solution of (1.1),

\[
(1.9) \quad x_m := x_0 + V_m y_m.
\]

The computation of \( x_m \) by (1.9) requires that the matrix \( V_m \) be available; however, for large-scale problems storage of this matrix in fast computer memory is often only possible when \( m \) is fairly small. Therefore, the GMRES method generally is implemented as a restarted algorithm. One computes \( x_m \) for a certain value of \( m \), that is small enough to allow storage of the matrix \( V_m \) in fast computer memory. If \( x_m \) is not a sufficiently accurate approximation of the solution of (1.1), then we let \( x_0 := x_m \) and improve \( x_0 \) by solving the minimization problem (1.8) with \( r_0 := b - Ax_0 \).

This scheme is referred to as the GMRES(\( m \)) method; see [18].

A bound for the residual error \( r_m := b - Ax_m \) can be derived by substituting the spectral factorization

\[
A = S \Lambda S^{-1}, \quad \Lambda = \text{diag}[\lambda_1, \lambda_2, \ldots, \lambda_n],
\]

into (1.4). We obtain from (1.3) and (1.4) that

\[
(1.11) \quad \| r_m \| \leq \kappa(S) \| r_0 \| \min_{p \in \mathcal{P}_m} \left( \max_{z \in \lambda(A)} |p(z)| \right),
\]

\[
\kappa(S) := \max_{\lambda \in \lambda(S)} \frac{\lambda_{\text{max}}(S)}{\lambda_{\text{min}}(S)}.
\]
where $\lambda(A)$ denotes the spectrum of $A$ and $\kappa(S) := \|S\|\|S^{-1}\|$ is the condition number of the eigenvector matrix $S$. We assume for definiteness that the columns of $S$ are scaled to be of unit Euclidean norm; see Demmel [10] for a discussion of this scaling.

The rate of convergence of the restarted GMRES($m$) method typically can be improved significantly by using a preconditioner. For instance, right preconditioners $C \in \mathbb{R}^{n \times n}$ often are designed so that the spectrum of $AC$ is more “clustered” in the complex plane around a point away from the origin than the spectrum of $A$. This property generally makes iterative methods applied to preconditioned linear system of equations

$$ACz = b$$

converge faster towards the solution $x = Cz$ of (1.1), where $z$ solves (1.12), than iterates generated by the same iterative method applied to the original (unpreconditioned) linear system of equations (1.1). For a review of many methods available for the construction of a preconditioner, we refer to Saad [17].

It is the purpose of the present paper to describe a new approach to preconditioning. It is based on replacing the linear system of equations (1.1) by the linear system

$$\hat{A}\hat{x} = b, \quad \hat{A} := A - gf^T, \quad f,g \in \mathbb{R}^n,$$

where the rank-one matrix $gf^T$ is chosen to increase the rate of convergence of the restarted GMRES method. We note that replacing the matrix $A$ by $\hat{A}$ corresponds to using the preconditioner $C := I - A^{-1}gf^T$ in (1.12). When an approximate solution of (1.13) has been determined, we compute an approximate solution of (1.1) with the Sherman-Morrison formula.

Let the matrix $\hat{A}$ have the spectral factorization

$$\hat{A} = \hat{S}\hat{\Lambda}\hat{S}^{-1}, \quad \hat{\Lambda} = \text{diag}([\hat{\lambda}_1, \hat{\lambda}_2, \ldots, \hat{\lambda}_n]),$$

where the columns of $\hat{S}$ are scaled to be of unit Euclidean norm.

Application of the GMRES method to the solution of (1.13) yields the following bound for the residual error, analogous to (1.11),

$$\|\hat{r}_m\| \leq \kappa(\hat{S})\|\hat{r}_0\| \min_{p \in \mathbb{P}_m} \left( \max_{z \in \lambda(\hat{A})} |p(z)| \right),$$

where $\hat{r}_m := b - \hat{A}\hat{x}_m$ is the residual vector associated with the $m$th iterate $\hat{x}_m$ generated by the GMRES method applied to the preconditioned linear system of equations (1.13). We would like the rank-one matrix $gf^T$ to be such that the matrix $\hat{A}$ has all eigenvalues clustered in the vicinity of a point on the real axis away from the origin. Then the factor

$$\min_{p \in \mathbb{P}_m} \left( \max_{z \in \lambda(\hat{A})} |p(z)| \right)$$

in the bound (1.15) would be small already for small values of $m$ and iterates computed by the restarted GMRES($m$) method would converge rapidly to the solution of (1.13), provided that the condition number $\kappa(\hat{S})$ is not very large.

In the computed examples reported in Section 4 most or all of the eigenvalues of the matrices $A$ used have positive real part. The numerical methods proposed in this
paper determines vectors \( f \) and \( g \), such that the matrix \( \hat{A} \) has no eigenvalues with negative real part (Examples 4.1 and 4.3) or fewer eigenvalues close to the origin than the matrix \( A \) (Example 4.2). In all examples the solution of the preconditioned linear system of equations (1.13) by the restarted GMRES method requires fewer iterations than the solution of the unpreconditioned linear system of equations (1.1).

The eigenvalues of the matrix \( \hat{A} = A - gf^T \) are in Control Theory sometimes referred to as poles and the determination of a rank-one matrix \( gf^T \) such that \( \hat{A} \) has an arbitrarily prescribed spectrum is a classical problem in Control Theory known as the pole placement problem or the eigenvalue assignment problem; see, e.g., Datta [9] or Wonham [20] for discussions of the mathematical background.

In typical applications of pole placement in Control Theory one of the vectors \( f \) or \( g \) is given and, if possible, the other one is determined so that the matrix \( \hat{A} \), defined by (1.13), has a prescribed spectrum. A survey of numerical methods for this pole placement problem is provided by Datta [9]; see also Arnold and Datta [1], Calvetti et al. [4], Datta [8], and Miminis and Roth [15]. Numerical methods for pole placement require \( \mathcal{O}(n^3) \) arithmetic floating point operations and storage of \( n \) \( n \)-vectors. They are therefore poorly suited for the solution of the pole placement problem when the matrix \( A \) is large. Below, we will discuss faster methods for the closely related partial pole placement problem.

The pole placement problem with one vector, \( f \) or \( g \), prescribed can be quite ill-conditioned, and this can make it difficult to determine the other vector with sufficient accuracy when the matrix \( A \) is large. The conditioning depends on the condition number of \( S \), the prescribed vector \( f \) or \( g \), and on the distribution of the eigenvalues of the matrices \( A \) and \( \hat{A} \). An analysis of the conditioning of the pole placement problem is presented by Mehrmann and Xu [13], and discussions on how to choose the poles, i.e., the eigenvalues of \( \hat{A} \), in prescribed sets in the complex plane, so that the ill-conditioning is reduced, are provided by Calvetti et al. [5] and Mehrmann and Xu [14].

Assume that all but a few eigenvalues of the matrix \( A \) have positive real part. Then the rate of convergence of the GMRES\((m)\) method generally can be improved significantly by choosing the rank-one matrix \( fg^T \) so that the matrix \( \hat{A} \), defined by (1.13), has no eigenvalues with negative real part or close to the origin. The remaining eigenvalues of \( \hat{A} \) and \( A \) may agree. The problem of choosing a rank-one matrix \( fg^T \) so that only a few eigenvalues of \( \hat{A} \) and \( A \) differ is known in Control Theory as the partial pole placement problem or the partial eigenvalue assignment problem. Unlike in applications in Control Theory, where in general one of the vectors \( f \) and \( g \) is given and the other one is computed, in our application to preconditioning we are free to determine both \( f \) and \( g \).

Numerical methods based on the Arnoldi decomposition (1.5) for the partial pole placement problem when the vector \( g \) is fixed are described by Saad [16], and related computational methods based on the Implicitly Restarted Arnoldi (IRA) method, due to Sorensen [19], are presented in [6, 7]. The latter references address the situations when either \( f \) or \( g \) are given, or when none of these vectors are prescribed.

This paper is organized as follows. Section 2 discusses the partial pole placement problem. Its application to preconditioning is described in Section 3, and a few numerical examples are presented in Section 4. Concluding remarks can be found in Section 5.

The preconditioner of the present paper uses spectral information of the matrix \( A \) gained from an Arnoldi decomposition (1.5) to determine suitable vectors \( f \) and \( g \).
in (1.13). It can be used together with other preconditioners since the linear system of equation (1.1) may be a preconditioned system.

Recently, several preconditioners, that are constructed during the iterations with the GMRES($m$) method and use spectral information of $A$ extracted from computed Arnoldi decompositions determined during the iterations, have been proposed. For instance, the methods discussed by Baglama et al. [3] and Erhel et al. [11] seek to determine an invariant subspace associated with eigenvalues close to the origin and then solve a linear system of equations with a modified matrix that has no eigenvalues close to the origin. While these methods work very well for large number of linear systems of equations, they do have the drawback of requiring much more computer storage than the method of the present paper, because they store an orthogonal basis of the invariant subspace associated with the eigenvalues close to the origin. We note that the preconditioner of the present paper only requires storage of two $n$-vectors, $f$ and $g$.

Recently, Kharchenko and Yeremin [12] proposed a preconditioner different from ours, which also is based on application of the pole placement problem. Both the preconditioners by Kharchenko and Yeremin and the one of the present paper are computed by extracting spectral information of $A$ from computed Arnoldi decompositions (1.5). The preconditioners differ in that the one of the present paper is based on the Implicitly Restarted Arnoldi method and applies the algorithm by Datta [8] for the small-scale pole placement problem that has to be solved to determine $f$ and $g$.

Our new preconditioning method is attractive because of its conceptual simplicity, and because the algorithm for the partial pole placement problem, on which the preconditioning method is based, is quite effective at determining vectors $f$ and $g$, such that the GMRES($m$) method applied to the preconditioned linear system of equations (1.13) converges faster than the GMRES($m$) method applied to the original linear system of equations (1.1).

2. The partial pole placement problem. This section recalls how the partial pole placement problem arises in the stabilization of single-input time-invariant linear control systems in Control Theory, and discusses a modification that is well suited for the determination of the matrix $A$ of the preconditioned linear system of equations (1.13).

Consider the single-input time-invariant linear control system

\[
\frac{d}{dt}z(t) = Az(t) + gu(t), \quad z(0) := z_0, \quad t \geq 0,
\]

where $A \in \mathbb{R}^{n \times n}$ is a large nonsymmetric matrix, $g, z_0 \in \mathbb{R}^n$, $z(t)$ is a vector-valued function with values in $\mathbb{R}^n$ and $u(t)$ is a real-valued function. Introduce the set

$\Psi = \{\psi_j\}_{j=1}^\ell$ of $\ell \leq n$ complex numbers and consider the problem of determining a vector $f \in \mathbb{R}^n$ that solves the partial pole placement problem

\[
\lambda(A - gf^T) = \Psi \cup \{\lambda_j\}_{j=t+1}^n.
\]

We remark that in Control Theory, the $\psi_j$ are often referred to as poles. The vector $f$ is known as the feedback gain vector, because substituting $u(t) := -f^T z(t)$ into (2.1) yields a closed-loop system with solution

\[
z(t) = \exp((A - gf^T)t)z_0, \quad t \geq 0.
\]
Assume that all but $\ell$ eigenvalues of the matrix $A$ have nonnegative real part. For definiteness, let
\[
\text{Re}(\lambda_j) > 0, \quad 1 \leq j \leq \ell, \\
\text{Re}(\lambda_j) < 0, \quad \ell < j \leq n.
\]
If $f = 0$ in (2.3), then there are initial vectors $z_0$, such that the closed-loop solution $z(t)$ becomes unbounded as $t$ increases. This is generally undesirable. Instead, we would like to choose $f$ so that the closed-loop solution satisfies
\[
\lim_{t \to \infty} z(t) = 0
\]
for any choice of initial vector $z_0$. In order to achieve this, let the poles $\psi_j$, $1 \leq j \leq \ell$, have negative real part and choose a feedback gain vector $f$, so that (2.2) holds. Then all eigenvalues of the matrix $A - gf^T$ have negative real part and (2.4) holds independently of the choice of $z_0$.

We turn to the construction of preconditioners via pole placement. Assume that the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_\ell$ of $A$ are close to the origin or have negative real part, and that the remaining eigenvalues of $A$ have positive real part and are not very close to the origin. Let the poles $\psi_j$ in the set $\Psi = \{\psi_j\}_{j=1}^\ell$ have positive real part and not be very close to the origin. We seek to determine vectors $f$ and $g$ that solve the partial pole placement problem
\[
\lambda(A - gf^T) = \Psi \cup \{\lambda_j\}_{j=\ell+1}^n.
\]
Then all eigenvalues of the matrix $A - gf^T$ have positive real part and are not very close to the origin. We therefore expect that the GMRES method applied to the solution of the preconditioned linear system of equations (1.13) will yield faster convergence than the GMRES method applied to (1.1). Numerical illustrations that support this expectation are presented in Section 4. We refer to the preconditioners determined in this manner as pole placement preconditioners.

The following results are concerned with the possibility of solving the partial pole placement problem (2.5). They have been shown in [7]. We assume that the poles $\psi_j$ in the set $\Psi$ are real or appear in complex conjugate pairs.

**Theorem 2.1.** ([7, Theorem 3.3]) Let $V \in \mathbb{R}^{m \times \ell}$ and let the upper Hessenberg matrix $H \in \mathbb{R}^{n \times n}$ with nonvanishing subdiagonal entries satisfy
\[
AV = VH, \quad V^T V = I_\ell, \quad \lambda(H) = \{\lambda_j\}_{j=1}^\ell.
\]
Assume that $\{\lambda_j\}_{j=1}^\ell \cap \{\lambda_j\}_{j=\ell+1}^n = \emptyset$ and let $\Psi = \{\psi_j\}_{j=1}^\ell$. Then there are vectors $f, b \in \text{range}(V)$, such that (2.5) holds.

We remark that the solution $\{f, b\}$ of the partial pole placement problem (2.5) is not unique.

**Theorem 2.2.** ([7, Theorem 3.4]) Let the matrices $H_\ell$ and $V_\ell$, the vector $\nu_{\ell+1}$ and the scalar $\eta_\ell$ make up an Arnoldi decomposition (1.5) with $m = \ell$. Let $g \in \mathbb{R}^\ell$ satisfy
\[
\lambda(H_\ell - \tilde{g}e_\ell^T) = \Psi,
\]
and define $f := V_\ell e_\ell$ and $g := V_\ell \tilde{g} + \eta_\ell \nu_{\ell+1}$. Then
\[
\Psi \subset \lambda(A - gf^T).
\]
Datta’s algorithm [8] is well suited for the solution of the pole placement problem (2.7) for the unreduced upper Hessenberg matrix \( H_\ell \) in Theorem 2.2. We remark that the inclusion (2.8) holds regardless of the size of \( \eta_\ell \geq 0 \), however, \( \lambda(A - gf^T) \) generally varies with \( \eta_\ell \). When \( \eta_\ell = 0 \), equation (2.5) holds according to Theorem 2.1.

In Algorithm 1 below, we apply the Implicitly Restarted Arnoldi method [19] to determine an Arnoldi decomposition (1.5) with a small value of \( \eta_\ell \geq 0 \). When \( \eta_\ell \) is “tiny,” the spectrum of the matrix \( A - gf^T \), with the vectors \( f \) and \( g \) determined according to Theorem 2.2, is close to \( \Psi \cup \{\lambda_j\}_{j=t+1}^n \). For definiteness, we assume throughout the remainder of this paper that the eigenvalues of the matrix \( A \) satisfy

\[
\text{Re}(\lambda_j) < 0, \quad 1 \leq j \leq \ell,
\]
\[
\text{Re}(\lambda_j) > 0, \quad \ell < j \leq n,
\]

and that we would like to determine vectors \( f \) and \( g \) that satisfy (2.5), where \( \Psi \) is a set of \( \ell \) poles with positive real parts.

**Algorithm 1** Partial pole placement.

1. Apply the Implicitly Restarted Arnoldi method, outlined in Algorithm 2 below, to the matrix \( A \) to compute an Arnoldi decomposition

\[
AV_\ell = V_\ell H_\ell + \eta_\ell e_{\ell+1}e_\ell^T,
\]

such that \( \eta_\ell \geq 0 \) is small and \( \lambda(H_\ell) \) is a good approximation of the eigenvalues \( \{\lambda_j\}_{j=1}^{\ell} \) of \( A \) with negative real part. The range of the matrix \( V_\ell \) provides an approximation of the invariant subspace associated with these eigenvalues.

2. Select a set of poles \( \Psi = \{\psi_j\}_{j=1}^{\ell} \).

3. Let \( f := V_\ell e_{\ell} \) and determine \( \bar{g} \in \mathbb{R}^{\ell} \) so that \( \lambda(H_\ell - \bar{g}e_\ell^T) = \Psi \), e.g., by the algorithm in [8].

4. Let \( g := V_\ell \bar{g} + \eta_\ell e_{\ell+1} \). The vector pair \( \{f, g\} \) is the computed approximate solution of the partial pole placement problem (2.5). \( \square \)

For future reference, we note that when the vectors \( f \) and \( g \) are chosen as in Algorithm 1, we can determine an Arnoldi decomposition of \( A \) from the Arnoldi decomposition (2.9) without evaluating matrix-vector products with \( A \). We have

\[
\bar{A}V_\ell = (A - gf^T)V_\ell = AV_\ell - gf^TV_\ell = V_\ell (H_\ell - g\bar{e}_\ell^T).
\]

We conclude this section with an outline of the Implicitly Restarted Arnoldi method for computing the Arnoldi decomposition of Step 1 of Algorithm 1. In addition to computing an approximation of the invariant subspace associated with the eigenvalues \( \{\lambda_j\}_{j=1}^{\ell} \) of \( A \) with negative real part, the algorithm applies the computed Arnoldi decompositions to improve an initial approximate solution \( x_0 \) of the linear system of equations (1.1). We denote the computed improved approximate solution by \( \hat{x} \) and the associated residual vector by \( \hat{r} \). The approximate solution \( \hat{x} \) will be improved further by using the pole placement preconditioner as described in Section 3. The input variable \( k \) of Algorithm 2 determines how many Arnoldi decompositions of the form (1.5) are computed before exit. Actual black-box implementations of the Implicitly Restarted Arnoldi method may use different stopping criteria; see Sorensen [19] for details on the Implicitly Restarted Arnoldi method. The value \( \mu \) determined in Step 5 of Algorithm 2 is used in Algorithm 4 below to locate the poles.
Algorithm 2 Implicitly Restarted Arnoldi method.

Input: \( A \in \mathbb{R}^{n \times n}, b, x_0 \in \mathbb{R}^n, \ell, k. \)

Output: \( \tilde{V}_\ell \in \mathbb{R}^{n \times \ell}, \tilde{H}_\ell \in \mathbb{R}^{\ell \times \ell}, \tilde{\epsilon}, \tilde{\eta} \in \mathbb{R}^n, \tilde{\eta}_k \geq 0, \mu. \)

1. \( \tilde{\epsilon} := x_0; \tilde{\epsilon} := b - A \tilde{\epsilon}; \tilde{v}_1 := \tilde{\epsilon}/\|\tilde{\epsilon}\|; \mu := 0; m := 2\ell; i := 1; \)
2. Compute the Arnoldi decomposition

\[
(2.11) \quad AV_m = V_m H_m + \eta_m v_{m+1} e_m^T,
\]

where \( V_m e_1 = v_1 \) is defined in Step 1.

3. Compute an improved approximate solution of (1.1) by solving

\[
\min_{y \in \mathbb{R}^n} \|AV_m y - \tilde{\epsilon}\|.
\]

The solution \( \hat{y} \in \mathbb{R}^n \) of this minimization problem is computed similarly as the solution of (1.8). The improved approximate solution of (1.1) and the associated residual vector are given by

\[
\tilde{\epsilon} := \tilde{\epsilon} + V_m \hat{y}_m; \quad \tilde{\epsilon} := \tilde{\epsilon} - V_m+1 H_{m+1,m} \hat{y}_m,
\]

where \( H_{m+1,m} \) and \( V_{m+1} \) are defined analogously to (1.6).

4. Compute the spectral factorization of \( H_m \). Denote the eigenvalues by \( \tilde{\lambda}_j \), \( 1 \leq j \leq m \). These eigenvalues are Ritz values of \( A \). Enumerate the Ritz values so that the subset \( \{\tilde{\lambda}_j\}_{j=1}^\ell \) consists of the \( \ell \) Ritz values of smallest real part.

5. Determine a value \( \mu \) in the convex hull of computed Ritz values of \( A \), not very close to the origin. For instance, let

\[
\mu_0 := \max_{1 \leq j \leq m} \Re(\tilde{\lambda}_j)
\]

and define \( \mu := \max\{\mu, \mu_0\} \).

6. Apply the recursion formulas of the Implicitly Restarted Arnoldi method with shifts \( \tilde{\lambda}_{\ell+1}, \tilde{\lambda}_{\ell+2}, \ldots, \tilde{\lambda}_m \) to update the Arnoldi decomposition (2.11) to obtain a new Arnoldi decomposition

\[
(2.12) \quad AV_{\ell} = \tilde{V}_{\ell} H_{\ell} + \tilde{\eta}_{\ell} \tilde{v}_{\ell+1} e_{\ell}^T
\]

with \( \lambda(\tilde{H}_{\ell}) = \{\tilde{\lambda}_j\}_{j=1}^{\ell} \). These computations do not require matrix-vector product evaluations with the matrix \( A \). The purpose of these computations is to determine a matrix \( \tilde{V}_{\ell} \) whose span provides a better approximation of the invariant subspace of \( A \) associated with the eigenvalues \( \lambda_1, \lambda_2, \ldots, \lambda_\ell \) than the space spanned by the first \( \ell \) columns of the matrix \( V_m \).

7. \( i := i + 1; \) if \( i = k \) then \textbf{exit}.

8. Carry out \( \ell \) steps of the Arnoldi process to compute an Arnoldi decomposition of the form (2.11) with \( V_m e_1 = \tilde{V}_\ell e_1 \) from the Arnoldi decomposition (2.12).

This requires the evaluation of \( \ell \) matrix-vector products with the matrix \( A \).

Go to 3. \( \square \)

For many matrices the Implicitly Restarted Arnoldi method gives adequate approximations of the invariant subspace of \( A \) associated with the set of eigenvalues \( \{\lambda_j\}_{j=1}^{\ell} \) with negative real part already for fairly small values of the input parameter \( k \). In our experience, the construction of preconditioners often does not require
that the invariant subspace associated with the set \( \{ \lambda_j \}_{j=1}^\ell \) be determined to high accuracy. This is illustrated in Section 4.

Algorithm 2 carries out \( m = 2\ell \) steps of the Arnoldi process in Step 2 and \( \ell \) steps of the Arnoldi process in Step 8. Thus, the algorithm requires the evaluation of \((k+1)\ell+1\) matrix-vector products with the matrix \( A \).

We remark that the relation \( m = 2\ell \) between the number of desired eigenvalues \( \ell \) and the number of steps \( m \) of the Arnoldi method between restarts is default in several available implementations of the IRA method, such as the implementations of the function \texttt{eigs} in Matlab versions 5.3 and 6.1. We therefore have chosen \( m = 2\ell \) in Algorithm 2. The algorithm easily can be modified to allow a different relation between \( \ell \) and \( m \). Assuming that all or almost all eigenvalues of \( A \) have positive real part, we would like \( \ell \) to be the number of eigenvalues with negative real part or of tiny magnitude. A suitable choice of \( \ell \) typically is not known a priori. However, the distribution of the eigenvalues of the Hessenberg matrix \( H_\ell \) in the complex plane is often suggestive for the choice of \( \ell \). We will comment on the choice of \( \ell \) further in Section 4.

3. Solution of the preconditioned linear system of equations. Having determined vectors \( f \) and \( g \) by Algorithm 1, and thereby the matrix \( \tilde{A} \) defined by (1.13), we compute an approximate solution of the given linear system of equations (1.1) by application of the Sherman-Morrison formula

\[
A^{-1} = \tilde{A}^{-1} - \frac{1}{1 + f^T\tilde{A}^{-1}g} \tilde{A}^{-1}gf^T\tilde{A}^{-1}.
\]

This requires the approximate solution of the two preconditioned linear systems of equations

\[
\tilde{A}u = b, \quad \tilde{A}w = g.
\]

If the computation of sufficiently accurate approximate solutions of (3.2) and (3.3) can be carried out faster with the GMRES\((m)\) method than the determination of an acceptable approximate solution of (1.1) by application of the GMRES method to (1.1), then preconditioning by pole placement is attractive. We illustrate with a few numerical examples in Section 4 that this indeed can be the case.

The following algorithm describes a very simple scheme for the computation and application of the preconditioned matrix \( \tilde{A} \). Numerical examples presented in Section 4 based on this algorithm illustrate that it is not always important to determine the invariant subspace associated with the eigenvalues of smallest real part with high accuracy; the algorithm only computes one Arnoldi decomposition to determine the vectors \( f \) and \( g \) that define the preconditioned matrix \( \tilde{A} \).

The input parameter \( \ell \) is the number of eigenvalues with negative or small real part. The Arnoldi decompositions computed determine orthogonal bases of Krylov subspaces of dimension \( m := 2\ell \). For large problems, limitations of fast computer memory may make it necessary to choose a small value of \( \ell \). The initial Arnoldi decomposition is used both to compute vectors \( f \) and \( g \) that define the preconditioned matrix \( \tilde{A} \) and to improve the available approximate solution of the linear system of equations (1.1). The vectors \( \tilde{x} \), \( \tilde{u} \) and \( \tilde{w} \) denote the best available approximate solutions of the linear systems of equations (1.1), (3.2) and (3.3), respectively, and \( \tilde{r} \), \( \tilde{r}_u \) and \( \tilde{r}_w \) are the corresponding residual vectors.
Algorithm 3 Solution of preconditioned linear system of equations.
Input: \( A \in \mathbb{R}^{n \times n} , \ b , x_0 \in \mathbb{R}^n \), \( m \).
Output: \( f , g , \hat{x} , \hat{r} \in \mathbb{R}^n \).
1. \( r_0 := b - Ax_0 \);
2. Compute the Arnoldi decomposition
   \[
   AV_m = V_m H_m + \eta_m v_{m+1} e_m^T , \quad V_m e_1 = r_0 / \| r_0 \|. 
   \]
   Define the matrices \( H_{m+1,m} \) and \( V_{m+1} \) according to (1.6).
3. Solve the minimization problem
   \[
   \min_{y \in \mathbb{R}^m} \| AV_m y - r_0 \| = \min_{y \in \mathbb{R}^m} \| H_{m+1,m} y - r_0 \| e_1 \|
   \]
   for \( \hat{y}_m \) and compute the improved approximate solution and associated residual vector
   \[
   \hat{x} := x_0 + V_m \hat{y}_m ; \quad \hat{r} := r_0 - V_{m+1} H_{m+1,m} \hat{y}_m .
   \]
   Define the set of \( m \) poles \( \Psi = \{ \psi_j \}_{j=1}^m \) by
   \[
   (3.5) \quad \psi_j := \max_{t \in \lambda(H_m)} \text{Re}(t) , \quad 1 \leq j \leq m .
   \]
4. Compute the vectors \( \hat{y} \in \mathbb{R}^m \) and \( f , g \in \mathbb{R}^n \) according to Steps 3 and 4 (with \( \ell = m \)) of Algorithm 1. These vectors determine the matrix \( \hat{A} \); cf. (1.13).
5. Use the Arnoldi decomposition (3.4) and the Arnoldi decomposition (2.10) of \( \hat{A} \) (with \( \ell = m \)) to determine initial approximate solutions of the linear systems of equations (3.2) and (3.3). Solve
   \[
   \min_{y \in \mathbb{R}^m} \| \hat{A} V_m y - b \|^2 = \min_{y \in \mathbb{R}^m} \| (H_m - \hat{g} e_m^T) y - V_m^T b \|^2 + \| (I - V_m V_m^T) \hat{b} \|^2
   \]
   for \( \hat{y} \), where \( \hat{g} \) is determined in Step 4. Let \( \hat{u} := V_m \hat{y} \) and \( \hat{r}_u := b - V_m (H_m - \hat{g} e_m^T) \hat{y} \). Also solve
   \[
   \min_{y \in \mathbb{R}^m} \| \hat{A} V_m y - g \|^2 = \min_{y \in \mathbb{R}^m} \| (H_m - \hat{g} e_m^T) y - V_m^T g \|^2 + \| (I - V_m V_m^T) g \|^2
   \]
   for \( \hat{y} \). Let \( \hat{w} := V_m \hat{y} \) and \( \hat{r}_w := g - V_m (H_m - \hat{g} e_m^T) \hat{y} \).
6. Apply the Sherman-Morrison formula (3.1) to compute the new approximate solution \( \hat{x} \) of (1.1),
   \[
   \hat{x} := \hat{u} - \frac{f^T \hat{u}}{1 + f^T \hat{w}} \hat{w} .
   \]
   Similarly as the residual vectors \( \hat{r}_u \) and \( \hat{r}_w \), the residual vector \( \hat{r} = b - A \hat{x} \) can be evaluated by using the Arnoldi decomposition (3.4). If \( \hat{x} \) is sufficiently accurate, then exit.
7. Compute the Arnoldi decomposition, analogous to (1.7),
   \[
   (3.6) \quad \hat{A} \hat{V}_m = \hat{V}_{m+1} \hat{H}_{m+1,m} , \quad \hat{V}_m e_1 = \hat{r}_u / \| \hat{r}_u \|. 
   \]
8. Compute an improved approximate solution of the linear system of equations (3.2) by solving
   \[
   \min_{y \in \mathbb{R}^m} \| \hat{A} \hat{V}_m y - \hat{r}_u \| = \min_{y \in \mathbb{R}^m} \| \hat{H}_{m+1,m} y - \| \hat{r}_u \| e_1 \|
   \]
   for \( \hat{y} \). Let \( \hat{u} := \hat{u} + \hat{V}_m \hat{y} \) and \( \hat{r}_u := \hat{r}_u - \hat{V}_{m+1} \hat{H}_{m+1,m} \hat{y} \).
9. Compute an improved approximate solution of the linear system of equations (3.3) by solving
\[
\min_{y \in \mathbb{R}^n} \| A \tilde{y}_m - \tilde{r}_w \|^2 = \min_{y \in \mathbb{R}^n} \| \tilde{H}_{m+1,m} y - \tilde{V}_{m+1}^T \tilde{r}_w \|^2 + \| (I - \tilde{V}_{m+1} \tilde{V}_{m+1}^T) \tilde{r}_w \|^2
\]
for \( \tilde{y} \). Let \( \tilde{w} := \tilde{w} + \tilde{V}_m \tilde{y} \) and \( \tilde{r}_w := \tilde{r}_w - \tilde{V}_{m+1} \tilde{H}_{m+1,m} \tilde{y} \).

10. Apply the Sherman-Morrison formula (3.1) to compute the new approximate solution \( \tilde{x} \) of (1.1),
\[
\tilde{x} := \tilde{u} - \frac{f^T \tilde{u}}{1 + f^T \tilde{w}}. 
\]

The residual vector \( \tilde{r} = b - A \tilde{x} \) can be evaluated inexpensively by using the Arnoldi decomposition (3.6). If \( \tilde{x} \) is sufficiently accurate, then exit, else go to 7. \( \square \)

We will illustrate in Section 4 that this algorithm may give significantly faster convergence than the restarted GMRES(\( m \)) algorithm applied to the solution of (1.1). Note that the GMRES(\( m \)) method and Algorithm 3 both determine orthogonal bases of Krylov subspaces of the same dimension.

Sometimes it is possible to modify Algorithm 3 to enhance its performance when it is applied to certain linear systems of equations (1.1). For instance, we may want to choose the set of poles \( \Psi \) differently from (3.5) or determine an invariant subspace associated with eigenvalues of \( A \) to be moved with higher accuracy. The latter can be achieved by applying Algorithm 2. This is illustrated by Algorithm 4 below. We also might want to solve the linear system (3.3) with higher accuracy than in Step 9 of Algorithm 3. This is discussed in Example 4.3 below.

**Algorithm 4 Solution of preconditioned linear system of equations with application of the Implicitly Restarted Arnoldi method.**

Input: \( A \in \mathbb{R}^{n \times n} \), \( b, x_0 \in \mathbb{R}^n \), \( \ell, k \).

Output: \( f, g, \tilde{x}, \tilde{r} \in \mathbb{R}^n \).

1. \( \tilde{r}_0 := b - A x_0; \tilde{m} := 2 \ell; \)
2. Apply Algorithm 2 with input parameters \( \ell \) and \( k \), and initial approximate solution \( x_0 \in \mathbb{R}^n \) of (1.1) to determine the approximation range(\( V_{\ell} \)) of the invariant subspace associated with \( \ell \) eigenvalues of \( A \) of smallest real part, an improved approximate solution \( \tilde{x} \) of (1.1) and the associated residual vector \( \tilde{r} \). Algorithm 2 also yields an approximation of \( \max_{t \in \lambda(A)} \text{Re}(t) \) denoted by \( \mu \).
3. Define the set of \( \ell \) poles \( \Psi = \{ \psi_j \}_{j=1}^\ell \), e.g., let \( \psi_j := \mu_j, 1 \leq j \leq \ell \).
4. Compute the vectors \( f, g \in \mathbb{R}^n \) according to Steps 3 and 4 of Algorithm 1. These vectors determine the matrix \( \tilde{A} \); cf. (1.13).
5. Apply Steps 7-10 of Algorithm 3. \( \square \)

**4, Numerical examples.** All numerical experiments presented in this section were carried out on an Intel Pentium computer with about 16 significant decimal digits using Matlab 5.3. In all examples, we chose the initial approximate solution \( x_0 = 0 \).

Algorithm 3 with input parameter \( m \) and Algorithm 4 with input parameter \( \ell := m/2 \) are compared to the restarted GMRES(\( m \)) method. These schemes generate Krylov subspaces of the same dimensions and therefore require about the same
amount of computer storage. We refer to Algorithms 3 and 4 as preconditioned GMRES($m$) methods. The GMRES($m$) method applied to (1.1) is referred to as the unpreconditioned GMRES($m$) method. The methods are compared in terms of the number of matrix-vector product evaluations with the matrices $A$ and $\hat{A}$ required to achieve a certain accuracy of the computed approximate solutions of (1.1). We refer to this number as the "number matrix-vector product evaluations." The evaluation of these matrix-vector products constitutes the dominant computational work when $n$ is large. In all examples $m$ is chosen small, a situation of interest when solving large linear systems of equations.

Example 4.1. Let the linear system of equations (1.1) be defined by a finite-difference discretization of a Dirichlet problem for the differential operator $-\Delta + \frac{1}{h^2} \frac{\partial}{\partial r}$. The operator is discretized on a uniform grid in the unit square with $15 \times 15$ interior grid points using a five-point stencil. The matrix $A \in \mathbb{R}^{25 \times 225}$ so obtained is nonsymmetric with real and positive eigenvalues. The right-hand side vector is $b = h^2 [1, 1, \ldots, 1]^T$ with $h = 1/16$.

We apply Algorithm 3 with $m = 4$ to the solution of (1.1). Figure 4.1 displays the spectra of $A$ and of the computed preconditioned matrix $\hat{A}$. Note that $A$ has eigenvalues closer to the origin than $\hat{A}$. Some of the eigenvalues of $\hat{A}$ appear in complex conjugate pairs. Figure 4.2 shows the convergence histories of the errors in the computed approximate solutions determined by the preconditioned and unpreconditioned iterative methods as function of the number of matrix-vector product evaluations with the matrices $A$ or $\hat{A}$. The graphs of the convergence histories for the associated residual errors look similar and therefore are not displayed. Algorithm 3 is seen to give much faster convergence than the GMRES(4) method applied to the linear system of equations (1.1).
Fig. 4.2. Example 4.1: Norm of the error $\|x_k - A^{-1}b\|$ as function of the number of matrix-vector product evaluations with the matrices $A$ or $\tilde{A}$, when the approximate solutions $x_k$ are computed either by the GMRES(4) method applied to (1.1) (dash-dotted curve) or by Algorithm 3 with $m = 4$ (solid curve).

Fig. 4.3. Example 4.2: Eigenvalues of $H_{20}$ determined by Algorithm 3 are marked by + and eigenvalues of $H_{20} - \eta_{20}e_0^T$ by o.
Fig. 4.4. Example 4.2: Eigenvalues of $A$ are marked by $+$ and eigenvalues of $\hat{A}$ by $\circ$. A magnification is shown in Figure 4.5.

Fig. 4.5. Example 4.2: Magnification of Figure 4.4 at the origin. Eigenvalues of $A$ are marked by $+$ and eigenvalues of $\hat{A}$ by $\circ$. Two eigenvalues of $A$ can be seen to be closer to the origin than any eigenvalue of $\hat{A}$.
Example 4.2. Let $A$ be the block bidiagonal matrix

$$
A = \begin{bmatrix}
\sigma_1 & \tau_1 & 0 & & & \\
-\tau_1 & \sigma_1 & \tau_2 & 0 & & \\
0 & -\tau_2 & \sigma_2 & \tau_3 & & \\
& & & \ddots & \ddots & \\
0 & & & & -\tau_{20} & \sigma_{20} \\
\end{bmatrix} \in \mathbb{R}^{100 \times 100},
$$

where $\sigma_j := \tau_j := 2j - 1$. The matrix has the eigenvalues $\lambda_{2j-1} = \sigma_j + i\tau_j$ and $\lambda_{2j} = \sigma_j - i\tau_j$, $1 \leq j \leq 200$, where $i := \sqrt{-1}$. Let the right-hand side vector be $b$ have components $b_j = \exp(2\pi(j-1)/399)$, $1 \leq j \leq 400$. This defines the linear system of equations (1.1).

Algorithm 3 with $m = 20$ yields the preconditioned matrix $\hat{A}$. Figure 4.3 displays the eigenvalues of the matrix $H_{20}$ (marked by $+$) determined in Step 2 of Algorithm 3. The eigenvalues of $H_{20} - \hat{g}_{20}e_{20}^T$ are marked by $\circ$ in Figure 4.3. Instead of having all eigenvalues at one point in the complex plane, the matrix $H_{20} - \hat{g}_{20}e_{20}^T$ has 20 distinct, not very close, eigenvalues. This depends on ill-conditioning of the pole placement problem for the matrix $H_{20}$. The ill-conditioning as well as possible remedies are discussed in [5, 13, 14]. We will not dwell on the ill-conditioning here, since it is of
minor importance for the computed examples of this section.

Figure 4.4 displays the eigenvalues of $A$ (marked by $+$) and of $\tilde{A}$ (marked by $o$). Figure 4.5 shows a blow-up of the region around the origin of Figure 4.4. In particular, Figure 4.5 shows that the eigenvalues of $\tilde{A}$ are not quite as close to the origin as the eigenvalues of $A$. The different distribution of the eigenvalues of $A$ and $\tilde{A}$ is important for the rate of convergence as is illustrated by Figure 4.6, which shows the convergence histories of the errors in the computed approximate solutions determined by the preconditioned and unpreconditioned iterative methods as function of the number of matrix-vector product evaluations with the matrices $A$ or $\tilde{A}$. The graphs for the convergence histories for the associated residual errors look similar and are therefore not shown. The preconditioned iterative method is seen to give faster convergence than the GMRES(20) method applied to the linear system of equations (1.1).

We remark that instead of storing the vectors $f$ and $g$ for the preconditioner, one might increase the size of the Krylov subspaces used for the restarted GMRES method. For instance, we might consider comparing Algorithm 3 with $m = 20$ with the GMRES(22) method. It turns out that the latter method converges only insignificantly faster than GMRES(20); the error in the computed approximate solution after 1400 matrix-vector product evaluations with the matrix $\tilde{A}$ obtained by the former method is $7 \cdot 10^{-7}$, which is close to the error obtained with the GMRES(20) method.

The restarted GMRES method is popular because of its ease of implementation and the absence of breakdown before the solution has been determined. Other iterative methods, such as the BiCG and BiCGstab methods, are not as reliable. For instance, due to breakdown the Matlab implementations of both the BiCG and BiCGstab methods failed to produce accurate approximate solutions when applied to the solution of the unpreconditioned linear systems of equations (1.1) of this example. □

Example 4.3. Let the entries of the matrix $N \in \mathbb{R}^{296 \times 296}$ and of the right-hand side vector $b$ in (1.1) be normally distributed random numbers with mean 0 and
Fig. 4.8. Example 4.3: Eigenvalues of the $A$ (marked by $+$) and eigenvalues of $\tilde{A}$ (marked by $\circ$).

Fig. 4.9. Example 4.3: Norm of the error $\|x_h - A^{-1}b\|$ as function of the number of matrix-vector product evaluations with the matrices $A$ and $\tilde{A}$, when the approximate solutions $x_h$ are computed either by the GMRES(8) method applied to (1.1) (dash-dotted curve) or by Algorithm 4 with $\ell = 4$ (solid curve).

variance 1. Define the circulant matrix

$$C = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{4 \times 4}$$
and let

\[ A = \begin{bmatrix}
C - 2I_4 & 0 \\
0 & 4I_{296} + \frac{1}{10} N
\end{bmatrix} \in \mathbb{R}^{300 \times 300}. \]

The matrix \( A \) has four eigenvalues with negative real part \( \lambda_1 = -3, \lambda_{2,3} = -2 \pm i \) and \( \lambda_4 = -1 \), where \( i := \sqrt{-1} \). We seek to determine vectors \( f \) and \( g \) such that the matrix \( A \) only has eigenvalues with positive real part. We achieve this by using Algorithm 4 with \( \ell = 4 \). In Step 2 of Algorithm 4, we apply Algorithm 2 with \( k = 2 \).

The vector \( g \) determined in Step 4 is of fairly large norm, \( ||g|| = 1 \cdot 10^4 \), and therefore we add Steps 7'-9' after Step 9 of Algorithm 3 to solve the linear system (3.3) more accurately. Step 7' is analogous to Step 7 with \( \hat{r}_u \) replaced by \( \hat{r}_w \). Steps 8' and 9' follow Step 7' and are analogous to Steps 8 and 9, respectively.

Figure 4.7 displays the eigenvalues of the matrix \( \tilde{H}_4 \) (marked by +) determined in Step 2 of Algorithm 4. They are \( \lambda_1 = -2.96, \lambda_{2,3} = -1.93 \pm 0.96i \) and \( \lambda_4 = 0.13 \). Thus, the eigenvalues of \( \tilde{H}_4 \) are not accurate approximations of the eigenvalues \( \lambda_j \), \( 1 \leq j \leq 4 \), of \( A \). The eigenvalues of \( \tilde{H}_4 - \hat{g}_4 e_4^T \) are marked by \( o \) in Figure 4.7. Thus, \( o \) marks the value of \( \mu \) determined in Step 5 of Algorithm 2. Figure 4.8 shows the eigenvalues of \( A \) (marked by +) and the eigenvalues of \( \tilde{A} \) (marked by o). Note that all eigenvalues of \( \tilde{A} \) have positive real part.

Figure 4.9 shows the convergence histories of the norm of the errors of approximate solutions determined by Algorithm 4 (solid curve) and by the GMRES(8) method (dash-dotted curve). Algorithm 4 can be seen to give much faster convergence. The graphs for the associated residual errors look similar and are omitted. \( \square \)

5. Conclusion and extensions. A new preconditioning method based on partial pole placement is presented. Numerical examples show that pole placement preconditioners can increase the rate of convergence of the restarted GMRES method significantly. The preconditioners do not require a priori knowledge of the spectrum of the matrix and can be used together with other preconditioners. Their storage requirement is fairly small; only two \( n \)-vectors have to be stored. The preconditioners are most effective when the matrix of the linear system of equations has a few eigenvalues, such that if removed, the rate of convergence of the restarted GMRES increases significantly. Numerical experiments suggest that it is not necessary to determine an invariant subspace associated with the eigenvalues to be moved to high accuracy. We have observed that the convergence of the preconditioned linear systems of equations are not very sensitive to implementation details of the preconditioners. For instance, if Steps 7'-9' of the modified Algorithm 3 described in Example 4.3 are applied in Examples 4.1 and 4.2, convergence histories very similar to those shown in Figures 4.2 and 4.6, respectively, are obtained. Nevertheless, further investigation of the properties of pole placement preconditioners is required before a black-box code can be written. Moreover, a careful comparison with other preconditioners, such as those discussed in [3, 11, 12], is also desirable.

The preconditioners of the present paper can also be used together with other iterative methods, such as the BiCG and BiCGstab methods. It may also be possible to construct preconditioners using a (nonsymmetric) Lanczos decomposition of \( A \) instead of an Arnoldi decomposition. These issues require further investigation. An advantage of the GMRES method is that breakdowns do not cause problems; see Example 4.2 for an illustration.
REFERENCES