Abstract

Fixed-Polynomial Approximate Spectral Transformations for Preconditioning the Eigenvalue Problem

by

Heidi K. Thornquist

Arnoldi's method is often used to compute a few eigenvalues and eigenvectors of large, sparse matrices. When the eigenvalues of interest are not dominant or well-separated, this method may suffer from slow convergence. Spectral transformations are a common acceleration technique that address this issue by introducing a modified eigenvalue problem that is easier to solve than the original. This modified problem accentuates the eigenvalues of interest, but requires solving a linear system, which is computationally expensive for large-scale eigenvalue problems.

This thesis shows how this expense can be reduced through a preconditioning scheme that uses a fixed-polynomial operator to approximate the spectral transformation. Implementation details and accuracy heuristics for employing a fixed-polynomial operator with Arnoldi's method are discussed. The computational results presented indicate that this preconditioning scheme is a promising approach for solving large-scale eigenvalue problems. Furthermore, this approach extends the domain of applications for current Arnoldi-based software. Future research directions include development of convergence theory, accuracy bounds for computed eigenpairs, and alternative constructions of the fixed-polynomial operator.

This work was supported in whole or in part by the Los Alamos National Laboratory Computer Science Institute (LACSI) through LANL contract number 03891-001-99-49, as part of the prime contract (W-7405-ENG-36) between the Department of Energy and the Regents of the University of California.

Acknowledgments

First and foremost, I would like to thank my advisor, Danny Sorensen, who introduced me to the intriguing idea of approximating a spectral transformation with matrix polynomials. The discoveries presented in this thesis, as well at those yet to come, are largely due to to his support and patience, for which I am extremely grateful.

I also wish to thank the rest of my committee for their interest in this thesis, especially Mark Embree. Our many discussions enabled the creation of a more complete and accurate document than would have been possible without his help.

The mere existence of this thesis has to be attributed to the wonderful mathematics teachers I have encountered over the years at Humboldt State University and Rice University. This unquestionably includes my undergraduate advisor, Charles Biles. I am forever indebted to him for his never-ending encouragement and guidance.

I am also grateful for my fellow graduate students, friends, and family. They made the fun of being in graduate school a lot more bearable. I would especially like to thank Carrie Jessen, Fayne Martini, Rusty Burnett, Keith Berrier, Dara Farmer, Michele Hendrix, Shannon Scott, and Summer Husband for their friendship and emotional support. Furthermore, I would like to thank Daria Lawrence for keeping us graduate students in line and providing those much-needed hugs.

Lastly, and most importantly, I wish to thank my parents, Geron and Marilyn Thornquist for their love, encouragement, and constant support. Without them, I would not have the solid foundation that allowed me to achieve this accomplishment.

Contents

List of Figures

Chapter 1

Introduction

Arnoldi's method [2] is often used to compute the eigenvalues and eigenvectors of large, sparse matrices. Various techniques can be employed to improve the convergence of this method, some resulting in considerable computational expense. This thesis will introduce an efficient preconditioning scheme that greatly accelerates convergence to the right-most eigenvalues of non-Hermitian matrices. Furthermore, this scheme can easily be implemented with existing Arnoldi-based software, extending the domain of application.

Before laying out the motivation for this preconditioning scheme, some notation and definitions will be discussed in Section 1.1. Following this, the large-scale eigenvalue problem will be introduced in Section 1.2. The foundation of most current methods for solving this problem, the *power method*, is presented in Section 1.3. A generalization, Arnoldi's method, is examined in Section 1.4, in addition to some common acceleration and computational techniques. This is followed by the introduction of the *Implicitly Restarted Arnoldi* [43] and *rational Krylov* [32] methods in Section 1.5 and 1.6, respectively.

The discussion shows the need for a spectral transformation to improve the convergence of these methods to the right-most eigenvalues. This acceleration technique results in a linear solve for each basis vector of the Krylov subspace. Utilizing iter-

ative methods for these linear solves requires a high degree of accuracy to guarantee that the Krylov subspace has been generated. For large-scale problems this may be necessary, but computationally expensive.

This computational expense can be reduced through a preconditioning scheme that uses a fixed-polynomial operator to approximate a spectral transformation. The fixed-polynomial can then be used to produce a modified problem with approximately the same solution as the original. Most importantly, this modified problem is one that the Krylov subspace method can solve rapidly.

1.1 Notation

The following notation will be used throughout this thesis:

- $\mathbb{R}$ and $\mathbb{C}$ will denote the real and complex numbers, respectively.
- Capital and lower case Latin letters denote matrices and vectors, respectively. Lower case Greek letters will denote scalars.
- The transpose of a matrix A is denoted by A^T and the conjugate-transpose is denoted by A^H; likewise for vectors.
- Unless otherwise specified, $\| \cdot \|$ is the Euclidean norm of a vector and the induced two-norm of a matrix.
- The jth canonical basis vector is denoted by e_j.
- A diagonal matrix whose diagonal elements are α_i, $i = 1, \cdots, n$ is denoted by $diag(\alpha_1, \alpha_2, \cdots, \alpha_n)$.
- Given a matrix A, $\mathcal{R}(A)$ and $\mathcal{N}(A)$ denote the range and null space of A, respectively.

- Given a scalar $\alpha \in \mathbb{C}$, $\Re(\alpha)$ and $\Im(\alpha)$ are the real and imaginary parts of α, respectively.

1.2 The Eigenvalue Problem

The algebraic eigenvalue problem

$$Ax = x\lambda \tag{1.1}$$

is a fundamental problem in scientific computing. It appears in disciplines like economics, dynamical systems, control theory and mechanics. The computation of solutions to equation (1.1) is important for performing structural and stability analysis on mathematical models in these disciplines.

Large eigenvalue problems commonly arise from a finite dimensional approximation of a continuous model

$$\mathcal{L}u = u\lambda, \tag{1.2}$$

where $\mathcal{L}$ is a linear differential operator. Often, $\mathcal{L}$ is the linearization of a nonlinear operator about a steady state. If the continuous model (1.2) is discretized by finite differences, the result is a *standard eigenvalue problem* (1.1). Discretization by a finite element method will lead to a *generalized eigenvalue problem* of the form

$$Ax = Bx\lambda. \tag{1.3}$$

Usually A is referred to as the *stiffness* matrix and B is the *mass* matrix. Sometimes the matrices A and B in the generalized eigenvalue problem (1.3) are referred to as the matrix pencil (A,B).

This thesis will address both the standard and generalized eigenvalue problem. The scalar, λ, and vector, x, that satisfy (1.1) or (1.3) are called the *eigenvalue* and (*right*) *eigenvector*, respectively. Often they can also be referred to as an *eigenpair*

(λ,x). Furthermore, an eigenvalue may also have a corresponding nonzero *left eigenvector* y that satisfies

$$y^H A = y^H B\lambda, \tag{1.4}$$

where $B = I$ if the original eigenvalue problem is in standard form (1.1).

Standard and generalized eigenvalue problems are classified by the structure of the matrix A or the matrix pencil (A,B), respectively. These classifications guide the design of algorithms through theoretical insight. If A is Hermitian, $A = A^H$, then (1.1) is referred to as a Hermitian eigenvalue problem. Furthermore, if A and B are both Hermitian, then (1.3) is referred to as a generalized Hermitian eigenvalue problem. The eigenvalues are always real for Hermitian eigenvalue problems and both their left and right eigenvectors are the same. This also holds true for the generalized case when B is positive definite. Moreover, any standard or generalized Hermitian eigenvalue problem is guaranteed to have an orthonormal basis of eigenvectors associated with it.

When A is non-Hermitian, $A \neq A^H$, then (1.1) is referred to as a non-Hermitian eigenvalue problem. Likewise, if A and/or B is non-Hermitian, then (1.3) is referred to as a generalized non-Hermitian eigenvalue problem. In both cases, the eigenvalues may be real, complex, or appear as complex conjugate pairs and the left and right eigenvectors are usually not the same. Even worse, a complete set of independent eigenvectors may not exist, complicating the theory and algorithms for non-Hermitian eigenvalue problems [3]. When a matrix or matrix pencil does not have an orthonormal basis of eigenvectors, it will be referred to as *non-normal*.

Numerical methods that solve the standard (1.1) and generalized (1.3) eigenvalue problems without direct factorizations or similarity transformations are important for large-scale problems since the discretization of the operator $\mathcal{L}$ results in matrices that are likely to be sparse. Thus, matrix-vector products can cost much less than n^2 floating point operations. The focus of this thesis will be on methods that require only matrix-vector products. A sparse, direct factorization might be used in some

cases as a preconditioner for an iterative linear solver or eigensolver.

Most current eigensolvers can be categorized as either a Krylov subspace method or a subspace iteration method. However, both of these categories are generalizations of the classic power method. The behavior of this single-vector iteration is the foundation for understanding the methods presented here. Thus, the power method will first be introduced in Section 1.3 and some important observations will be made before looking at Krylov subspace methods in Section 1.4.

1.3 The Power Method

The power method is a simple and efficient way to compute a single eigenvalue and its corresponding eigenvector of the standard eigenvalue problem (1.1). The method, described in Figure 1.1, only requires the application of a fixed operator A. It is

Input: (A, v)
Output: An eigenvalue, λ, of A and its corresponding eigenvector v.

1. **for** $k = 1, 2, \cdots$, **until** convergence
   - **1.1.** $w = Av$
   - **1.2.** $j = i\_max(w)$
   - **1.3.** $\lambda = w(j)$
   - **1.4.** $v \leftarrow v/\lambda$
2. **end**

Figure 1.1: The Power method

well known that the power method tends to converge to the eigenvalue of A with the largest magnitude and its corresponding eigenvector [49]. However, with the use of a spectral transformation, this method can be made to converge to any specified eigenvalue in the spectrum of A.

The convergence analysis for this method is straightforward when A is diagonalizable and has n distinct eigenvalues which can be ordered such that

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|.$$

Generally, the convergence rate of v to the eigenvector q_1 corresponding to λ_1 is linear and proportional to $|\frac{\lambda_2}{\lambda_1}|$. Then, for some constant c,

$$\|v^{(k)} - q_1\| \leq c \left|\frac{\lambda_2}{\lambda_1}\right|^k \|v^{(1)} - q_1\|.$$

This may be slow if λ_1 is not well separated from the rest of the spectrum. However, if v is orthogonal to q_1 and A is normal, the power method will converge to the eigenvector q_2 corresponding to λ_2 at a rate proportional to $|\frac{\lambda_3}{\lambda_2}|$. This realization motivates the idea of *deflation*, by which another eigenpair of A is exposed.

Convergence to any eigenvalue of A near a shift σ can be achieved through the use of a shift-invert spectral transformation, $(A-\sigma I)^{-1}$. This is called the *inverse power method* and is useful for computing interior eigenvalues of A, but it still observes only linear convergence. There is a natural extension of this method through the variation of the shift at each step. By choosing σ to be the Rayleigh quotient of v, $\frac{v^H A v}{v^H v}$, the *Rayleigh Quotient Iteration* (RQI) is obtained. Convergence is generally quadratic for this method, but increases to cubic if A is Hermitian.

The main drawback of the power method is that it only computes one eigenpair at a time. Deflation using the spectral projector of λ_1,

$$P_1 = q_1 \hat{q}_1^H,$$

where $\hat{q}_1$ is a left eigenvector of λ_1 normalized so that $\hat{q}_1^H q_1 = 1$, can be used to continue the computation. By restricting A to the complementary invariant subspace of λ_1, $(I - q_1\hat{q}_1^H)A(I - q_1\hat{q}_1^H)$, convergence to q_2 can be achieved. This mechanism allows for the computation of more than one eigenpair, but can be dangerous when A is non-normal. Krylov subspace methods provide a better approach by considering a sequence of vectors generated by the power method.

1.4 The Arnoldi Method

A single-vector iteration like the power method does not take advantage of the information provided by the successive vectors in the sequence

$$v, Av, A^2v, A^3v, \cdots .$$

This sequence may be rich with information along eigenvector directions that correspond to the dominant eigenvalues of A. A natural consequence is to consider the k-dimensional *Krylov subspace*

$$\mathcal{K}_k(A, v) = span\{v, Av, A^2v, \cdots, A^{k-1}v\}. \tag{1.5}$$

The *Arnoldi method* [2], described in Figure 1.2, is used to construct an orthonormal basis of $\mathcal{K}_k(A, v)$. The result is the *k-step Arnoldi factorization* of A

$$AV_k = V_kH_k + fe_k^T \tag{1.6}$$

where $V_k \in \mathbb{C}^{n\times k}$ has orthonormal columns and $V_k^H f = 0$. If A is non-Hermitian then $H_k = V_k^H AV_k$ is a $k \times k$ upper Hessenberg matrix. If A is Hermitian then H_k is a $k \times k$ real, symmetric, tridiagonal matrix and the factorization (1.6) is called the *k-step Lanczos factorization* of A.

This factorization can be used to obtain approximate eigenvalues and eigenvectors for A. Using the eigenpairs (θ, y) of H_k, the vector $x = V_ky$ satisfies

$$\|Ax - x\theta\| = \|(AV_k - V_kH_k)y\| = |\beta e_k^T y| \tag{1.7}$$

where $\beta = \|f\|$. The approximate eigenpair (θ, x) is called the *Ritz value* and *Ritz vector*, respectively, and the number $|\beta e_k^T y|$ is called the *Ritz estimate*. If A is Hermitian, the Ritz estimate can be used to provide a rigorous bound on the accuracy of the eigenvalues of H_k as estimates for the eigenvalues of A [30]. In the non-Hermitian case, a small Ritz estimate does not imply an accurate answer due to the possible non-normality of A.

Input: (A, v_0, k)
Output: (V_k, H_k, f) where $AV_k = V_k H_k + f e_k^T, V_k^H V_k = I_k$, and $V_k^H f = 0$.

1. $v_1 = \frac{v_0}{\|v_0\|}$

2. $w = Av_1$, $\alpha = v_1^H w$

3. $H_1 = [\alpha]$, $V_1 = [v_1]$, $f = w - v_1\alpha$

4. **for** $j = 1, 2, \cdots, k-1$

4.1. $\beta = \|f\|$, $v_{j+1} = \frac{f}{\beta}$

4.2. $H_j \leftarrow \begin{bmatrix} H_j \\ \beta e_j^T \end{bmatrix}$, $V_{j+1} = [V_j \; v_{j+1}]$

4.3. $w = Av_{j+1}$, $h = V_{j+1}^H w$

4.4. $H_{j+1} \leftarrow [H_j \; h]$, $f = w - V_{j+1} h$

5. **end**

Figure 1.2: The Arnoldi method

The implementation of the Arnoldi method given in Figure 1.2 uses the classical Gram-Schmidt (CGS) algorithm to build the orthogonal Krylov basis. For eigenvalue calculations an orthogonal basis is essential and CGS is notoriously unstable. Replacing classical Gram-Schmidt with modified Gram-Schmidt (MGS) does not always ensure orthogonality. Furthermore, MGS is not as computationally efficient as CGS. The best solution to this problem is to reorthogonalize the Krylov basis using the Daniel, Gragg, Kaufman, and Stewart (DGKS) correction [7] whenever necessary.

It is impossible to know in advance how large the Krylov subspace must be before the eigenvalues of H_k are good approximations to the eigenvalues of interest. For large-scale problems, storing a sufficient number of basis vectors may not be possible. Incorporating an acceleration scheme with the Arnoldi method helps keep k small, while still converging to the selected eigenvalues. Two techniques briefly discussed here are *restarting* and *spectral transformations*. These are essential for practical

implementations, like the Implicitly Restarted Arnoldi (IRA)[43] and rational Krylov [32] methods. More information on acceleration techniques for the Arnoldi method can be found in Yang's thesis [50].

1.4.1 Restarting the Arnoldi Method

Restarting accelerates convergence of the Arnoldi method by modifying the starting vector v using eigenvector information obtained in the previous Krylov subspace. Simple restarting replaces the starting vector with a single Ritz vector [38]. Polynomial restarting generalizes this by updating the starting vector v with

$$v \leftarrow \psi(A)v,$$

where $\psi(\lambda)$ is constructed to filter out unwanted eigenvector components. The easiest method uses a linear combination of Ritz vectors

$$v^+ = \sum_{j=1}^{k} q_j \gamma_j, \tag{1.8}$$

with some appropriate choice of weights γ_j [36],[39].

Another approach for building ψ is to specify its roots as points in a region containing unwanted eigenvalues. A straightforward choice would be to select a set of eigenvalues from the current H_m and use them as the roots of the polynomial ψ. This method of selection, suggested by Sorensen [43], is called *exact shifts*. If K is a compact set containing the unwanted eigenvalues, then *Leja points* [22] can also be used as roots of the polynomial ψ. Given a real-valued, positive function $w(z)$ defined on K, the sequence of Leja points z_j satisfies

$$\begin{aligned} w(z_0)|z_0| &= \max_{z\in K}\ w(z)|z| \\ w(z_j)\textstyle\prod_{k=0}^{j-1}|z_j - z_k| &= \max_{z\in K}\ w(z)\prod_{k=0}^{j-1}|z - z_k|, \end{aligned}$$

for $j = 1, 2, \cdots$. These points are uniformly distributed in K with respect to $w(z)$ and are typically expensive to compute.

Other possibilities for constructing ψ require knowledge about the region containing the wanted eigenvalues. A Chebyshev polynomial could be constructed [37] if the containment region is a line segment or an ellipse. This idea was based upon the acceleration scheme of Manteuffel [23]. Faber polynomials [10] can be constructed for polygonal containment regions [16]. This is an adaptation of the hybrid Arnoldi-Faber iterative method proposed by Starke and Varga [44] using a Schwarz-Christoffel conformal mapping [47].

1.4.2 Spectral Transformations

Spectral transformations improve the convergence of the Arnoldi method through the substitution of A with $\psi(A)$, where $\psi(\lambda)$ is some simple function [50]. This transformation serves to map interior or clustered eigenvalues of A to dominant, well-separated eigenvalues of $\psi(A)$. The most common of these is the *shift-invert* spectral transformation

$$\psi_{SI}(\lambda) = \frac{1}{\lambda - \sigma},$$

which emphasizes the eigenvalues of A around the shift σ (Figure 1.3). The Arnoldi

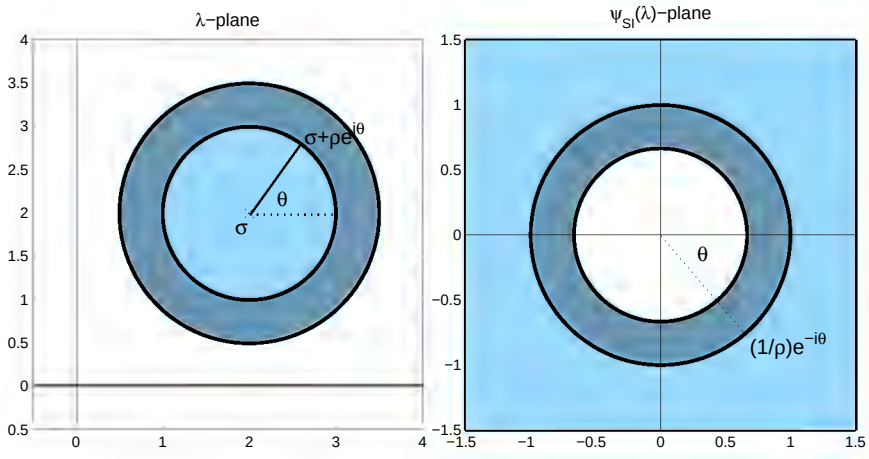

Figure 1.3: Shift-invert transformation

method applied to this transformation, called *shift-invert Arnoldi*, is a generalization

of the inverse power method. If the shift σ is varied at each step, the shift-invert Arnoldi method becomes a generalization of RQI called *rational Krylov* [32].

Another interesting spectral transformation is the Möbius transformation

$$\psi_M(\lambda) = \frac{\lambda - \mu}{\lambda - \sigma}.$$

This transformation maps the eigenvalues of A close to σ far from the unit circle, and those close to μ to eigenvalues of $\psi_M(A)$ with small modulus (Figure 1.4). In

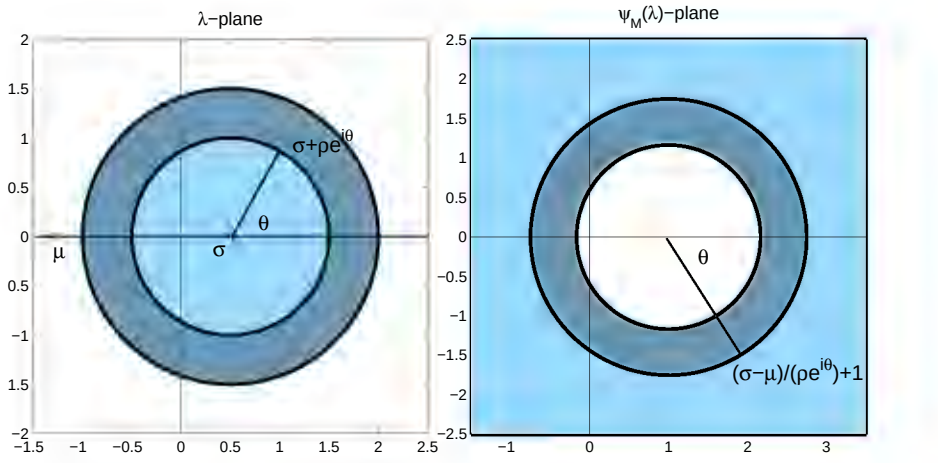

Figure 1.4: Möbius transformation

exact arithmetic there is no advantage to using this transformation with a translation-invariant Arnoldi method because $\psi_M(A) = I + (\sigma - \mu)\psi_{SI}(A)$. However, it can be beneficial to use ψ_M with an inexact Krylov subspace method.

1.4.3 Eigenvector Purification

Generalized eigenvalue problems of the form

$$Ax = Bx\lambda, \tag{1.9}$$

where B is positive semidefinite, arise in many physical situations. The most common way to this solve problem (1.9) is to transform it back to a standard eigenvalue

problem using a spectral transformation, like the shift-invert [9]. Then the original problem becomes

$$Sx = x\theta,$$

where $S = (A - \sigma B)^{-1}B$ and $\theta = \frac{1}{\lambda - \sigma}$. This technique works well with both rational Krylov and restarted Arnoldi methods. However, this transformation can cause numerical problems when B is singular.

If B is singular then it is possible for Arnoldi's method to find approximations to the zero eigenvalues of S. These eigenvalues usually correspond to *infinite* eigenvalues of the generalized eigenvalue problem (1.9), which are often considered uninteresting. However, the eigenvectors corresponding to these uninteresting eigenvalues typically corrupt the invariant subspace corresponding to the nonzero eigenvalues of S. The first way to prevent this is to ensure the starting vector, v_1, is in $\mathcal{R}(S)$ by replacing v_1 with the normalized vector $Sv_1/\|Sv_1\|$. However, the final approximate eigenvector, x, may still have components from the zero eigenvalues of S, which can again be purged by replacing x by Sx and normalizing it. Unfortunately, this incurs an additional matrix-vector product.

Purification of the computed eigenvectors is an efficient solution for purging these undesirable components and is equivalent to performing a formal step of the power method with S [9, 29]. Given a k-step Arnoldi factorization for S, where $H_k y = y\theta$ and $x = V_k y$, then

$$Sx = (A - \sigma B)^{-1}Bx = V_k H_k y + f e_k^T y = x\theta + f e_k^T y. \tag{1.10}$$

So by replacing the computed eigenvector x with $x + f\frac{e_k^T y}{\theta}$ and renormalizing it, any undesirable components have been purged without additional matrix-vector products.

1.5 Implicitly Restarted Arnoldi Method

The Implicitly Restarted Arnoldi (IRA) method [43], described in Figure 1.5, provides a numerically stable way to extract the desired eigenvalues and eigenvectors from a high-dimensional Krylov subspace. It implements polynomial restarting by applying a sequence of p implicit QR updates to an m-step Arnoldi factorization, where $m = k + p$. These updates apply p exact shifts implicitly to reduce the factorization back to order k.

Input: (A, v_0, m, k) where $p = m - k$
Output: (V_k, H_k, f) where $AV_k = V_k H_k + f e_k^T, V_k^H V_k = I_k$, and $V_k^H f = 0$.

1. Compute an m-step Arnoldi factorization: $AV_m = V_m H_m + f e_m^T$

2. **for** $iter = 1, 2, \cdots,$**until** convergence

2.1. Compute $\sigma(H_m)$ and select p shifts $\mu_1, \mu_2, \cdots, \mu_p$

2.2. $Q = I_m$

2.3. **for** $j = 1, 2, \cdots, p$

2.3.1. $[Q_j, R_j] = \mathrm{qr}(H_m - \mu_j I)$

2.3.2. $H_m \leftarrow Q_j^H H_m Q_j, Q \leftarrow Q Q_j$

2.4. **end**

2.5. $\beta_k = H_m(k+1, k), \sigma_k = Q(m, k)$

2.6. $f \leftarrow v_{k+1}\beta_k + f\sigma_k$

2.7. $V_k \leftarrow V_m Q(:, 1:k), H_k \leftarrow H_m(1:k, 1:k)$

2.8. Apply p additional steps of the Arnoldi process to $AV_k = V_k H_k + f e_k^T$ to obtain a new m-step Arnoldi factorization $AV_m = V_m H_m + f e_m^T$

3. **end**

Figure 1.5: The Implicitly Restarted Arnoldi method

A shift $\mu \in \mathbb{R}$ is applied implicitly using the QR factorization of $H_m - \mu I$,

$$H_m - \mu I = Q_m R_m$$

where Q_m is unitary and R_m is upper triangular. Multiplying the Arnoldi factorization on the right by Q_m results in

$$A(V_m Q_m) = (V_m Q_m) Q_m^H H_m Q_m + f_m e_m^T Q_m. \tag{1.11}$$

Since H_m is Hessenberg, Q_m will also have Hessenberg structure and $e_m^T Q_m$ will have $m-1$ nonzero entries. Thus, the first $m-1$ columns of (1.11) are a new Arnoldi factorization. After p shifts have been applied, the new k-step Arnoldi factorization is the first k columns of

$$A(V_m Q) - (V_m Q) Q^T H_m Q = f e_m^T Q,$$

where $Q = \prod_{j=1}^{p} Q_j$. This can be expanded, through p steps of the Arnoldi process, back to an m-step factorization.

1.5.1 Inexact Implicitly Restarted Arnoldi

Accelerating a Krylov subspace method with a spectral transformation requires a linear system solve for each basis vector of the Krylov subspace. It is preferable to use a direct method for these solves. However, for large problems it is often necessary to use an iterative solver. This also has its drawbacks since each iterative solve must be highly accurate to ensure that a Krylov subspace is being constructed. Usually methods that incorporate an iterative solver with a Krylov subspace method are called *inexact*.

Inexact Implicitly Restarted Arnoldi accelerated by a Möbius transformation is effective in practice. The Möbius transformation has properties that make it preferable over shift-invert for linear stability analysis in computational fluid dynamics (CFD) problems [13, 14, 15]. This method has been employed in the stability analysis of a Chemical Vapor Deposition (CVD) reactor simulation with over 4 million variables [21]. More recently, techniques based on these findings have allowed researchers to perform stability analyses on systems with 16 million variables [6].

1.6 The Rational Krylov Method

The rational Krylov method [32, 33, 34] is a generalization of shift-invert Arnoldi that allows for the shift to be varied at each step. It also allows the iteration to continue with a linear combination of the basis vectors already computed instead of just the last one. This method builds an orthogonal basis for the k-dimensional rational Krylov subspace

$$span\{v, (A-\sigma_1 B)^{-1}Bv, (A-\sigma_2 B)^{-1}Bv, \cdots, (A-\sigma_{k-1}B)^{-1}Bv\}.$$

The use of several shifts σ_j can improve convergence, but may result in more computationally expensive linear solves.

Input: (A, B, v_0) where $\|v_0\| = 1$

1. $V_1 = v_0$
2. **for** $j = 1, 2, \cdots$,**until** convergence
   - **2.1.** Select a shift σ_j and continuation vector $r = V_j t_j$ where $\|t_j\| = 1$
   - **2.2.** $w = (A - \sigma_j B)^{-1} Br$
   - **2.3.** $w \leftarrow w - V_j h_j$, where $h_j = V_j^H w$
   - **2.4.** $\beta = \|w\|$, $v = \frac{w}{\beta}$
   - **2.5.** $V_{j+1} = [V_j \; v]$
   - **2.6.** $\hat{h}_j = \begin{bmatrix} h_j \\ \beta \end{bmatrix}$, $\hat{k}_j = \sigma_j \hat{h}_j + \begin{bmatrix} t_j \\ 0 \end{bmatrix}$
   - **2.7.** **if** $j > 1$ **then**
     $H_{j+1,j} \leftarrow \begin{bmatrix} H_{j,j-1}\hat{h}_j \\ 0 \end{bmatrix}$ and $K_{j+1,j} \leftarrow \begin{bmatrix} K_{j,j-1}\hat{k}_j \\ 0 \end{bmatrix}$
     else
     $H_{j+1,j} = \hat{h}_j$ and $K_{j+1,j} = \hat{k}_j$
   - **2.8.** Compute approximate solution and test for convergence
3. **end**

Figure 1.6: The Rational Krylov method

The rational Krylov iteration illustrated in Figure 1.6 produces a generalized shift-invert Arnoldi factorization. By eliminating w in the orthogonalization step (2.3), it can be seen that the relationship at step j is

$$(A - \sigma_j B)^{-1} B V_j t_j = V_{j+1} \hat{h}_j \tag{1.12}$$

where $\hat{h}_j = \begin{bmatrix} h_j \\ \beta \end{bmatrix}$. Let $\hat{t}_j = \begin{bmatrix} t_j \\ 0 \end{bmatrix}$, then $V_{j+1}\hat{t}_j = V_j t_j$ and (1.12) can be rewritten as

$$A V_{j+1} \hat{h}_j = B V_{j+1} (\sigma_j \hat{h}_j + \hat{t}_j).$$

Collecting these relations over a sequence of iterations, $j = 1, 2, \cdots, k$, results in

$$A V_{k+1} H_{k+1,k} = B V_{k+1} K_{k+1,k},$$

where $H_{k+1,k}$ and $K_{k+1,k}$ are upper Hessenberg matrices, and

$$K_{k+1,k} = H_{k+1,k} diag(\sigma_1, \sigma_2, \cdots, \sigma_k) + T_{k+1,k}.$$

The columns of the upper triangular matrix $T_{k+1,k}$ contain the continuation vectors from step (2.1). If the shift is held constant, $\sigma_j = \sigma$, and the continuation vector is the latest vector, $t_j = e_j$, for all iterations $j = 1, 2, \cdots, k$, the resulting factorization is the shift-invert Arnoldi

$$(A - \sigma B)^{-1} B V_k = V_{k+1} H_{k+1,k}.$$

Extracting approximate eigenvalues and eigenvectors from the rational Krylov subspace can be done in a variety of ways [35]. If the approximate eigenvector is assumed to have the form $x \equiv V_{k+1} H_{k+1,k} y$, then (θ, x) is a Ritz pair for the matrix pencil (A,B) with respect to the subspace $\mathcal{R}(V_{k+1} H_{k+1,k})$ if and only if

$$H_{k+1,k}^H K_{k+1,k} y = \theta H_{k+1,k}^H H_{k+1,k} y. \tag{1.13}$$

The accuracy of a Ritz pair (θ, x) is estimated by $\|Ax - Bx\theta\|$, which is approximately $\|(K_{k+1,k} - \theta H_{k+1,k}) y\|$. Calculating the Ritz values is only one approach to

approximating the eigenvalues and eigenvectors from the rational Krylov subspace. This theory can also be extended to compute harmonic Ritz values.

While the rational Krylov method can be very effective, there are many practical implementation issues that need to be addressed. Choosing and deciding when to use a new shift σ_j is subjective. If direct factorizations are used to solve the linear system in step (2.2), then it might be necessary to use the same shift for multiple iterations to reduce the computational cost. If the rational Krylov subspace becomes large, then the cost of orthogonalization becomes computationally expensive. A strategy for locking and purging converged Schur vectors from the rational Krylov subspace has been proposed along with an implicit restarting scheme [35]. Both of these ideas were based on techniques developed for the Arnoldi method [19, 43].

1.6.1 Inexact Rational Krylov

The variation of the shift in the rational Krylov method is very important. If the shift is held constant, the resulting method is the shift-invert Arnoldi method. However, if the shift is varied every iteration and a direct factorization is used to solve the linear system (2.2), then this method can be computationally expensive. One approach to dealing with this issue is to use the same shift for multiple iterations. Another would be to solve the linear system (2.2) using an iterative method. The later approach, called *inexact rational Krylov*, is more effective in dealing with large eigenvalue problems.

As opposed to the shift-invert Arnoldi method, the rational Krylov method is not as demanding on the accuracy of the iterative solver. This is because the rational Krylov method does not rely on the correct representation of an underlying Krylov basis. An inexact rational Krylov method accelerated by a Möbius transformation (IC-RKS) has been proposed by Lehoucq and Meerbergen [18]. This method has shown that a Möbius transformation is more robust than a shift-invert transformation when the linear systems are solved inexactly within the rational Krylov method.

Chapter 2

IRA with a Fixed-Polynomial Operator

Accelerating the Implicitly Restarted Arnoldi (IRA) method with a spectral transformation requires a linear system solve for each basis vector of the Krylov subspace. It is preferable to use a direct method for these solves. However, due to storage constraints and computational cost, large-scale eigenvalue problems require the use of an iterative method for these linear solves. Unless stringent accuracy is specified, this results in the application of a different operator ϕ for every linear solve with no guarantee that the Krylov subspace has been generated. Alternatively, this thesis will consider computing the operator ϕ once and applying it instead of using an iterative solver. This approach is very similar to a class of linear solvers called *hybrid methods*, which are discussed in Section 2.1.

The fixed-polynomial operator used in this thesis will be constructed using a common iterative solver for non-Hermitian linear systems: the General Minimum Residual (GMRES) method [40]. This method will be introduced in Section 2.2 and the construction of the GMRES fixed-polynomial operator will follow in Section 2.3.

Using a fixed-polynomial operator to approximate the spectral transformation, thus replacing the iterative solver, is a promising approach. The eigenpairs of the original problem can be easily reconstructed from the Ritz vectors of the Krylov subspace, as discussed in Section 2.4. The current accuracy and convergence heuristics presented in Section 2.5 will show that the accuracy of these eigenpairs is dependent

upon the norm of the residual polynomial, $\|p(A_\sigma M_\sigma^{-1})\|$. If the fixed-polynomial operator closely approximates the spectral transformation, this norm should be small. A starting vector that is equally weighted in all eigendirections seems to facilitate the construction of such an operator. Furthermore, the summary of the algorithm in Section 2.6 will show that once the fixed-polynomial operator is constructed the cost of its application and storage is constant.

2.1 Hybrid Methods

Using a fixed-polynomial operator in the place of an iterative linear solver is not a novel idea. This approach is the basis for a class of linear solvers known as hybrid iterative methods which generally consist of a two-phase implementation. Phase I acquires information about the spectrum of the matrix, which is used in Phase II to design a polynomial iteration. Most hybrid algorithms rely directly on eigenvalue estimates obtained from the Arnoldi process in Phase I. However, there are limitations to this approach since eigenvalues may be misleading for non-normal matrices. Alternative hybrid methods have been proposed by Nachtigal, Reichel, and Trefethen [27] and Manteuffel and Starke [24] that circumvent computing eigenvalue estimates in Phase I.

2.2 Generalized Minimum Residual Method

The Generalized Minimum Residual (GMRES) method [40] is a Krylov subspace method for solving non-Hermitian linear systems

$$Ax = b. \tag{2.1}$$

Given an initial guess x_0 and residual $r_0 = b - Ax_0$, this method computes iterates of the form $x_k = x_0 + \hat{x}_k$, where $\hat{x}_k \in \mathcal{K}_k(A, r_0)$ solves

$$\min_{\hat{x} \in \mathcal{K}_k(A, r_0)} \|r_0 - A\hat{x}\|^2.$$

Then $\hat{x}_k = \phi_{k-1}(A)r_0$ where $\phi_{k-1}(A)$ is called the *iteration polynomial* and has degree $k-1$. Now the residual at step k can be written in the form

$$r_k = r_0 - A\hat{x}_k = (I - A\phi_{k-1}(A))r_0 = p_k(A)r_0 \tag{2.2}$$

where $p_k(\tau) = 1 - \tau\phi_{k-1}(\tau)$ is the *residual polynomial*. As a result, $\|p_k(A)\|$ is a bound on the size of $\|r_k\|$.

For $\|p_k(A)\|$ to be small, the magnitude of p_k must be small over some region in the complex plane containing the spectrum of A, $\sigma(A)$. Suppose $\|p_k(A)\| = 0$; then

$$A\phi_{k-1}(A) = I,$$

which means that $A^{-1} = \phi_{k-1}(A)$. So, for any $\lambda_j \in \sigma(A)$,

$$\phi_{k-1}(\lambda_j) = \frac{1}{\lambda_j}.$$

Now suppose that $\|p_k(A)\|$ is small; then it may be assumed that $\phi_{k-1}(\lambda_j) \approx 1/\lambda_j$ for any $\lambda_j \in \sigma(A)$, which means that $\phi_{k-1}(A) \approx A^{-1}$. This makes the iteration polynomial ϕ_{k-1} a good candidate for a fixed-polynomial operator to approximate A^{-1}.

If the spectrum of A is tightly clustered, then $\|p_k(A)\|$ can be small for a low-degree polynomial. Since this is not an attribute of most matrices, preconditioning must be employed to enable the construction of a low-degree polynomial p_k that is small in norm. *Right preconditioning* modifies the original linear system (2.1) by multiplying A, on the right, by a matrix M^{-1} which gives

$$AM^{-1}\hat{x} \;=\; b. \tag{2.3}$$

To obtain the approximate solution to the original system one preconditioner solve $Mx = \hat{x}$ is required. The residual (2.2) can now be written as

$$r_k = (I - AM^{-1}\phi_{k-1}(AM^{-1}))r_0 = p_k(AM^{-1})r_0.$$

The norm of the residual polynomial $p_k(AM^{-1})$ can be made small if the spectrum of AM^{-1} is tightly clustered. This requires a decent preconditioner for A that has similar spectral properties, resulting in AM^{-1} having a few sets of tightly clustered eigenvalues. Ideally the spectrum of the preconditioner M would closely approximate the spectrum of A.

Right preconditioning GMRES slightly changes the form of the fixed-polynomial operator. If $\|p_k(AM^{-1})\|$ is small, then $\|I - AM^{-1}\phi_{k-1}(AM^{-1})\|$ is small. Using the earlier argument, it can be concluded that A^{-1} can be approximated by $M^{-1}\phi_{k-1}(AM^{-1})$.

2.3 The GMRES Fixed-Polynomial Operator

The GMRES iteration polynomial ϕ_{k-1} is constructed through the k-step Arnoldi factorization (1.6). Consider the right-preconditioned linear system (2.3) and assume the zero vector as an initial guess, then $r_0 = b$ and

$$M\hat{x}_k \in \mathcal{K}_k(AM^{-1}, b).$$

The approximate solution $\hat{x}_k$ can be written in terms of the Krylov orthogonal basis V_k for $\mathcal{K}_k(AM^{-1}, b)$, $\hat{x}_k = M^{-1}V_k y$ for some $y = [\eta_1, \eta_2, \cdots, \eta_k]^T \in \mathbb{R}^k$. Assume the right-hand side vector b is of unit length, then each Krylov basis vector can be computed using the equation

$$v_{j+1} = \frac{1}{\gamma_{j+1,j}}[AM^{-1}v_j - \sum_{i=1}^{j} v_i\gamma_{i,j}] = \hat{\phi}_j(AM^{-1})b, \tag{2.4}$$

for $j = 1, 2, \cdots, k-1$, where $\gamma_{i,j} = H_k(i, j)$. Now it can be seen that

$$\hat{x}_k = M^{-1}\phi_{k-1}(AM^{-1})b,$$

where $\phi_{k-1}(AM^{-1}) = \sum_{j=0}^{k-1} \eta_{j+1}\hat{\phi}_j(AM^{-1})$.

Examining the GMRES iteration polynomial in this way illuminates the idea that it can be easily reconstructed from just H_k and y. The coefficients of the polynomial

$\hat{\phi}_j(AM^{-1})$ are determined by Gram-Schmidt and stored in the j-th column of H_k. These polynomials can then be linearly combined using the weights contained in y to obtain the GMRES iteration polynomial. It should be noted that H_k and y are determined by the right-hand side vector b. Therefore, the choice of this vector greatly affects the polynomial $\phi_{k-1}(AM^{-1})$. While this is an issue to be explored in future research, b is chosen to be a random unit vector in this thesis.

The ability to easily reconstruct the GMRES iteration polynomial makes it a promising fixed-polynomial operator for approximating A^{-1}. The polynomial can be constructed through the solution of (2.3) with a random unit vector b. After solving this linear system to a specified tolerance, H_k and y are stored. Then applying $M^{-1}\phi_{k-1}(AM^{-1})$ to any arbitrary vector v is the same as building the original Krylov basis, as illustrated in Figure 2.1. Then

$$w = M^{-1}\phi_{k-1}(AM^{-1})v$$

is an approximate solution to the linear system $Ax = v$.

Input: (H_k, y, v)
Output: $w = M^{-1}\phi_{k-1}(AM^{-1})v$

1. $\hat{v}_1 = v$

2. **for** $j = 1, 2 \cdots, k-1$

2.1. $\hat{v}_{j+1} = \frac{1}{\gamma_{j+1,j}}[AM^{-1}\hat{v}_j - \sum_{i=1}^{j} \hat{v}_i \gamma_{i,j}]$

3. **end**

4. $w = M^{-1} \sum_{j=1}^{k} \eta_j \hat{v}_j$

Figure 2.1: Applying the GMRES iteration polynomial

There are some important observations about the application of the GMRES iteration polynomial. The vectors constructed through this process (Figure 2.1) are not orthogonal. Furthermore, all of them must be stored during the construction.

This is a result of the long-term recurrence used by GMRES. An iteration polynomial constructed from a short-term recurrence method, like Bi-Conjugate Gradient (BiCG) [48] or Quasi-Minimal Residual (QMR) [12], would only require the two most recent vectors to be stored. A short-term recurrence is clearly a more efficient approach for constructing a fixed-polynomial operator, so it will be considered in future work.

2.4 Reconstructing Eigenpairs

Reconstructing approximate eigenpairs of A from a fixed-polynomial operator is straightforward. Let σ be the shift in a shift-invert transformation of the standard eigenvalue problem, where $A_\sigma = A - \sigma I$ and $M_\sigma = M - \sigma I$. If IRA is applied to the fixed-polynomial operator $M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})$, the resulting k-step Arnoldi factorization is

$$M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})V_k = V_k H_k + f e_k^T.$$

To find the eigenvalues closest to the shift, select the eigenvalues of largest magnitude of $H_k y = y\theta$ and let $x = V_k y$. Then for each eigenpair (θ, y) of H_k,

$$M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})x = x\theta + f e_k^T y = \hat{x}\theta, \tag{2.5}$$

where $\hat{x} = x + f\frac{e_k^T y}{\theta}$ is the purified eigenvector. Now an approximate eigenpair (λ, q) of A can be obtained from

$$A_\sigma \hat{x} - \hat{x}\frac{1}{\theta} = A\hat{x} - \hat{x}(\sigma + \frac{1}{\theta}), \tag{2.6}$$

where $q = \hat{x}/\|\hat{x}\|$ and $\lambda = \sigma + \frac{1}{\theta}$.

The reconstruction is similar for the generalized eigenvalue problem. Let σ be the shift, $A_\sigma = A - \sigma B$, and $M_\sigma = M - \sigma B$. The fixed-polynomial operator for this problem is

$$(A - \sigma B)^{-1}B \approx M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})B.$$

Applying IRA to this operator results in the factorization

$$M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})BV_k = V_k H_k + fe_k^T.$$

Using the same algebraic manipulation as before, for each eigenpair (θ, y) of H_k,

$$M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})Bx = x\theta + fe_k^T y = \hat{x}\theta, \tag{2.7}$$

where $\hat{x}$ is the purified eigenvector. Then the approximate eigenpair (λ, q) can be obtained from

$$A_\sigma\hat{x} - B\hat{x}\frac{1}{\theta} = A\hat{x} - B\hat{x}(\sigma + \frac{1}{\theta}), \tag{2.8}$$

where $q = \hat{x}/\|\hat{x}\|$ and $\lambda = \sigma + \frac{1}{\theta}$.

There are some theoretical and practical issues that should be mentioned. Clearly, the formula for directly calculating the approximate eigenvalues (2.6) cannot be obtained from equation (2.5). This is because the computational formulas discussed here ignore important error terms which will be addressed in the next section. Furthermore, in practice it has been found that computing the eigenvalue with a Rayleigh quotient using the purified eigenvector is better than calculating the eigenvalue directly using (2.6) or (2.8). This can be attributed to specific properties of the Rayleigh quotient as discussed in [49].

2.5 Accuracy and Convergence Heuristics

While computing an approximate eigenpair of A from a fixed-polynomial operator is straightforward, it is unclear how good the approximation is. Furthermore, it is also unclear whether the eigenpairs of a fixed-polynomial operator will converge to the eigenpairs of A, even if they are computed exactly. This section will discuss accuracy and convergence heuristics, which indicate a large dependence upon the residual polynomial of the iterative method used to construct the fixed-polynomial

operator. These heuristics are derived from equation (2.5) to obtain the error bounds that are ignored in (2.6) and (2.8).

If IRA is applied to a fixed-polynomial operator $M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})$, reconstructing the eigenpairs of A results in an equation (2.5) which can be rewritten as

$$\hat{x}\theta - A_\sigma^{-1}\hat{x} = M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})x - A_\sigma^{-1}\hat{x}. \tag{2.9}$$

Multiplying both sides of (2.9) by A_σ and dividing by θ results in

$$A_\sigma\hat{x} - \hat{x}\frac{1}{\theta} = \frac{1}{\theta}\,[A_\sigma M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})x - \hat{x}].$$

Using the definition of the purified eigenvector $\hat{x} = x + f\frac{e_k^T y}{\theta}$, this equation can be rewritten in terms of the residual polynomial p

$$A_\sigma\hat{x} - \hat{x}\frac{1}{\theta} = -\frac{1}{\theta}\,p(A_\sigma M_\sigma^{-1})x - f\frac{e_k^T y}{\theta^2}, \tag{2.10}$$

where $p(A_\sigma M_\sigma^{-1}) = I - A_\sigma M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})$. Now the left-hand side of (2.10) can be rewritten using (2.6) in terms of the approximate eigenpair (λ, q) of A

$$Aq - q\lambda = -\frac{1}{\theta\|\hat{x}\|}\,p(A_\sigma M_\sigma^{-1})x - f\frac{e_k^T y}{\theta^2\|\hat{x}\|}, \tag{2.11}$$

The accuracy bound for the computed eigenpair (λ, q) of A is obtained by taking the norm of (2.11)

$$\begin{aligned} \|Aq - q\lambda\| &= \left\| \frac{1}{\theta\|\hat{x}\|}\,p(A_\sigma M_\sigma^{-1})x + f\frac{e_k^T y}{\theta^2\|\hat{x}\|} \right\| \\ &\leq \frac{1}{|\theta|\|\hat{x}\|}\|p(A_\sigma M_\sigma^{-1})\|\|x\| + \frac{\|f\||e_k^T y|}{|\theta|^2\|\hat{x}\|}. \end{aligned} \tag{2.12}$$

An accuracy bound for the generalized eigenvalue problem can be similarly obtained. Again, for a shift σ, define $A_\sigma = A - \sigma B$, and $M_\sigma = M - \sigma B$. Then equation (2.7) can be rewritten as

$$\hat{x}\theta - A_\sigma^{-1}B\hat{x} = M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})Bx - A_\sigma^{-1}B\hat{x}. \tag{2.13}$$

Multiplying both sides of (2.13) by A_σ and dividing by θ results in

$$A_\sigma\hat{x} - B\hat{x}\frac{1}{\theta} = \frac{1}{\theta}\,[A_\sigma M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})Bx - B\hat{x}].$$

Using the definition of the purified eigenvector, this equation can be rewritten in terms of the residual polynomial p

$$A_\sigma \hat{x} - B\hat{x}\frac{1}{\theta} = -\frac{1}{\theta}\, p(A_\sigma M_\sigma^{-1})Bx - Bf\frac{e_k^T y}{\theta^2}. \tag{2.14}$$

Now the left-hand side of (2.14) can be rewritten using (2.8) in terms of the approximate eigenpair (λ, q) of the matrix pencil (A, B)

$$Aq - Bq\lambda = -\frac{1}{\theta\|\hat{x}\|}\, p(A_\sigma M_\sigma^{-1})Bx - Bf\frac{e_k^T y}{\theta^2\|\hat{x}\|}. \tag{2.15}$$

The accuracy bound for the computed eigenpair (λ, q) of (A, B) is

$$\begin{aligned} \|Aq - Bq\lambda\| &= \left\| \frac{1}{\theta\|\hat{x}\|}\, p(A_\sigma M_\sigma^{-1})Bx + Bf\frac{e_k^T y}{\theta^2\|\hat{x}\|} \right\| \\ &\leq \frac{1}{|\theta|\|\hat{x}\|}\|p(A_\sigma M_\sigma^{-1})\|\|Bx\| + \frac{\|Bf\||e_k^T y|}{|\theta|^2\|\hat{x}\|}. \end{aligned} \tag{2.16}$$

Since the magnitude of the Ritz value θ is expected to be large, the first term in the bounds (2.12) and (2.16) are of primary concern. Unfortunately, the iterative method used to construct p can only offer a bound for $\|p(A_\sigma M_\sigma^{-1})v_1\|$, where v_1 is the initial vector used to construct the polynomial. It is not obvious how to bound the residual polynomial $\|p(A_\sigma M_\sigma^{-1})\|$. However, this is necessary for determining the accuracy of the approximate eigenpairs computed by IRA applied to a fixed-polynomial operator. The tolerance used to build the fixed-polynomial operator can only serve as a rough estimate of how well the spectral transformation is being approximated. Furthermore, nothing is known at this time about the convergence of these fixed-polynomial operators to the spectral transformation they are approximating.

2.6 Summary of Algorithm

The simplicity of implementing this preconditioning scheme with any Krylov subspace method is appealing (Figure 2.2). The GMRES fixed-polynomial operator is constructed through one linear solve with a random right-hand side. It is then applied during every Arnoldi step until convergence. So, the cost and storage for this

Input: A, B, M, σ, nev, tol, $maxdeg$
Output: $(\lambda_j, \hat{x}_j)\ j = 1, \cdots, nev$ eigenpairs of (A, B) closest to σ.

Let $A_\sigma = A - \sigma B$, $M_\sigma = M - \sigma B$.

1. **Construct the operator**
   - **1.1.** Let v_1 be a random vector, where $\|v_1\| = 1$.
   - **1.2.** Use GMRES to solve $A_\sigma x = Bv_1$ with M_σ as a right preconditioner. Run GMRES until relative residual falls below specified tolerance tol, or $deg(\phi(A_\sigma M_\sigma^{-1})) > maxdeg$.
   - **1.3.** Output H_k, and y.
2. **Compute eigenvectors of the operator**
   Use the fixed-polynomial operator $M_\sigma^{-1}\phi(A_\sigma M_\sigma^{-1})$ with IRA to compute the eigenvectors corresponding to the nev eigenvalues of (A, B) closest to σ.
3. **Recover eigenvalues of A**
   For each purified approximate eigenvector $\hat{x}_j$, use the Rayleigh quotient $\frac{\hat{x}_j^H A \hat{x}_j}{\hat{x}_j^H B \hat{x}_j}$ to compute the approximate eigenvalue λ_j.

Figure 2.2: Algorithm for implementing IRA with GMRES fixed-polynomial operator

operator is fixed once it is constructed. In practice, the dimension of the polynomial is restricted to keep the cost of the operator low.

Some observations can be made about this algorithm. It can be extended so that any iteration polynomial can be constructed in Step 1 and used in Step 2. Short-term recurrence methods like BiCG and QMR construct iteration polynomials that are less computationally expensive, but it is not known how effective these polynomials would be as a fixed operator. Restriction of the dimension of the polynomial in Step 1 emphasizes the need for a high quality preconditioner. Furthermore, it requires that provisions are made if the spectral transformation is not adequately approximated by the fixed-polynomial operator. These two issues are prime candidates for future research.

Chapter 3

Numerical Results

Several test cases are presented to illustrate the promising numerical behavior of this preconditioning scheme. These include both Hermitian and non-Hermitian eigenvalue problems. For some of these test cases, results are obtained for two other current eigenvalue methods: Jacobi-Davidson [41] and Locally Optimal Block Preconditioned Conjugate Gradient (LOBPCG) [17]. These methods will be introduced in Section 3.1.

To provide a full comparison of these methods both performance and accuracy are analyzed. Performance is gauged by matrix-vector products and accuracy is measured by the Ritz residual or relative error for each approximate eigenvalue. When possible, the results are accumulated over an increasing problem size to detect any trends. To provide consistency to these results, a set of parameters are chosen and held constant for each method. These parameters and their settings are given in Section 3.2.

In Section 3.3 a large-scale Laplacian operator is used to compare the GMRES fixed-polynomial operator to preconditioned GMRES started anew for each linear solve. Then IRA with the GMRES fixed-polynomial operator is compared to the Jacobi-Davidson method and LOBPCG using a small-scale Laplacian operator in Section 3.4.

Two non-Hermitian eigenvalue problems were chosen to test the ability of this

preconditioning scheme in finding the rightmost eigenvalues or eigenvalues of largest imaginary part. In Section 3.5, linear stability analysis is performed on a wind-driven barotropic ocean model. In Section 3.6, the eigenvalue of largest imaginary part from a discretized sequence of 2D scalar wave equations is computed. This information can be used to accelerate the solution of a 2D dynamic rate equation model for a semiconductor laser. For both these applications, the results are compared with the Jacobi-Davidson method.

3.1 Current Methods

There are several eigenvalue methods that can be used for comparison in illustrating the promising numerical behavior of this preconditioning scheme. Due to the availability of software, only two of the most widely used methods were chosen for comparison: Jacobi-Davidson and Locally-Optimal Block Preconditioned Conjugate Gradient (LOBPCG).

3.1.1 Jacobi-Davidson

The Jacobi-Davidson [41] method is a subspace iteration method for computing a few eigenvalues of a non-Hermitian matrix, and their corresponding eigenvectors. It is best understood through the discussion of subspace iteration and Davidson's method. This will motivate the introduction of Jacobi's orthogonal component correction as an approach for addressing the weakness of the correction vector in Davidson's method. The end result is a combination of the two methods, known as Jacobi-Davidson.

Subspace iteration is another generalization of the power method that treats a block of vectors simultaneously. The shift-invert variant of this iteration is described in Figure 3.1. By combining steps (3.2) and (3.3), it can be seen that $H = V^H AV$. The Ritz pairs (θ, x) can then be computed from H exactly like those computed in Arnoldi's method.

Input: $(A, W \in \mathbb{C}^{n\times k}$, tol)
Output: k approximate eigenpairs (θ_j, x_j) of A.

1. Factor $[V, R] = \text{qr}(W)$

2. Set $H = 0$

3. **while** $(\|AV - VH\| > \text{tol}\|H\|)$,

3.1. $\mu = \text{Select\_shift}(H)$

3.2. Solve $(A - \mu I)W = V$

3.3. Factor $[V_+, R] = \text{qr}(W)$

3.4. $H = V_+^H V R^{-1} + \mu I$

3.5. $V \leftarrow V_+$

4. **end**

5. Compute k Ritz pairs (θ_j, x_j) of A.

Figure 3.1: The Generalized Shifted Inverse Power method

The most important observation about subspace iteration is that the construction of the subspace $\mathcal{S} = \mathcal{R}(V)$ is separate from the construction of the Ritz vectors. This means that the linear solve in step (3.2) can be computed using an iterative method. As a result, the projected matrix H in step (3.4) would have to be obtained directly by forming $H = V_+^H A V_+$. Using an inexact solve for step (3.2) also means abandoning the Krylov structure, resulting in the loss of efficiency in obtaining error estimates for the Ritz pairs.

Subspace iteration also has some significant advantages over Krylov subspace projection. Introducing inexact solves in step (3.2) may affect convergence, but the theoretical properties of this iteration do not require a set accuracy. This is contrary to the effect of inexact solves in the Krylov setting. For a sequence of closely related problems, the entire subspace basis from the previous problem can be used at the initial basis of the next problem. Meanwhile, Krylov subspace methods need a single starting vector so they can only use a linear combination of these basis vectors.

Finally, constructing vectors to augment the subspace can be done very generally. Instead of solving the shift-invert equations directly, defect corrections can be constructed to augment the subspace.

Davidson's Method [8] is based on this idea. Given a k-dimensional subspace $\mathcal{S}_k = \mathcal{R}(V_k)$, where the columns of V_k are orthogonal and $\theta \in \sigma(V_k^H A V_k)$ is the largest Ritz value of the projected matrix A, expand the subspace with a residual defect correction to improve the approximation of θ. Suppose the Ritz pair $(\theta, \hat{x})$ is the current approximation to the eigenpair (λ, x) and (δ, z) is the correction pair that solves

$$A(\hat{x} + z) - (\theta + \delta)(\hat{x} + z) = 0, \tag{3.1}$$

where $\hat{x}^H z = 0$. Using the residual $r = A\hat{x} - \theta\hat{x}$, equation (3.1) can be rewritten as

$$(A - \theta I)z = -r + \hat{x}\delta + z\delta. \tag{3.2}$$

Davidson suggested approximating $(A - \theta I)$ with $(D_A - \theta I)$, where D_A is the diagonal of A. Furthermore, he chose to ignore the first- and second-order terms on the right-hand side of equation (3.2). An approximate residual correction z to expand the subspace $\mathcal{S}_k$ can then be obtained by solving

$$(D_A - \theta I)z = -r. \tag{3.3}$$

Davidson's method can be quite successful in finding the dominant eigenvalues of strongly diagonally dominant matrices. This can be attributed to the fact that $(D_A - \theta I)^{-1}$ is a good preconditioner for r if A is diagonally dominant. However, if $(D_A - \theta I)$ is replaced with $(A - \theta I)$ in equation (3.3) and exactly solved, then $z = \hat{x}$. This solution does not expand the search space, contradicting the idea that solving equation (3.3) is a preconditioning step.

The *Jacobi-Davidson* method improves upon Davidson's method by restricting the residual correction to the orthogonal complement of the existing space. Consider the projection onto the orthogonal complement of the current Ritz vector $(I - \hat{x}\hat{x}^H)$ and

the current Ritz value as a Rayleigh quotient $\theta = \hat{x}^H A\hat{x}$. If the correction equation (3.2) is multiplied on the left by this projection and $\hat{x}^H z = 0$, the resulting equation is

$$\begin{aligned}(I - \hat{x}\hat{x}^H)(A - \theta I)(I - \hat{x}\hat{x}^H)z &= (I - \hat{x}\hat{x}^H)(-r + \hat{x}\delta + z\delta) && (3.4)\\ &= -r + z\delta && (3.5)\\ &\approx -r && (3.6)\end{aligned}$$

Only the second-order term $z\delta$ appears in the equality (3.5), proving that this formulation is a second-order correction $z = (I - \hat{x}\hat{x}^H)z$ that is orthogonal to $\hat{x}$.

The Jacobi-Davidson method is presented in Figure (3.2) for a Hermitian oper-

Input: (A, v_0, tol)
Output: Approximate eigenpair (θ, x) such that $\|Ax - x\theta\| < tol$.

1. $x = v_0/\|v_0\|$, $w = Ax$, $\theta = x^H w$

2. $H_1 = [\theta]$, $V_1 = [x]$, $r = w - \theta x$

3. **while** ($\|r\|$ >tol),

3.1. Solve (approximately) for $z \perp x$
$$(I - xx^H)(A - \theta I)(I - xx^H)z = -r$$

3.2. $c = V_j^H z$, $z = z - V_j c$

3.3. $v_{j+1} = z/\|z\|$, $V_{j+1} = [V_j\ v_{j+1}]$

3.4. $w = Av_{j+1}$

3.5. $\begin{bmatrix} h \\ \alpha \end{bmatrix} = V_{j+1}^H w$, $H_{j+1} = \begin{bmatrix} H_j & h \\ h^H & \alpha \end{bmatrix}$

3.6. Compute all eigenpairs of H_{j+1}, select desired pair (θ, y).

3.7. $x \leftarrow V_{j+1} y$

3.8. $r = Ax - x\theta$

4. **end**

Figure 3.2: The Jacobi-Davidson method for $A = A^H$

ator A. The projected correction equation (3.1.1) is solved using a preconditioned

iterative method. This correction is then used to expand the search space. The presented algorithm expands the subspace until convergence, which is not computationally practical. Restarting techniques are usually employed to keep the memory usage down. Furthermore, deflation techniques allow this method to find more than one eigenpair by forcing the subspace to be orthogonal to the converged eigenvectors. *JDQR* and *JDQZ* [11] are implementations of the Jacobi-Davidson method that employ restarting and deflation in solving standard and generalized eigenvalue problems, respectively.

3.1.2 Locally-Optimal Block Preconditioned Conjugate Gradient

LOBPCG [17] is a method for computing a few eigenvalues of a Hermitian matrix, and their corresponding eigenvectors. Based on the local optimization of a three-term recurrence, this method is best understood through the introduction of preconditioned simultaneous iteration. This general framework requires an iterative solver like the preconditioned Conjugate Gradient (PCG) method to enlarge the trial subspace for updating the current eigenvector approximations. The scalar iteration parameters needed to compute this update are then chosen to be locally optimal using the Rayleigh-Ritz procedure.

Consider the generalized eigenvalue problem

$$Ax = Bx\lambda, \tag{3.7}$$

where A is Hermitian positive definite. If B is not positive definite, then some of the eigenvalues of the generalized eigenvalue problem (3.7) may be infinite. In this case, an alternative form of the generalized eigenvalue problem is considered,

$$Bx = Ax\mu. \tag{3.8}$$

The infinite eigenvalues of the original problem (3.7) are now the zero eigenvalues of the alternative form (3.8). The alternative form of the eigenvalue problem will be

used in this section for the development of the LOBPCG method. Furthermore, the matrix M will refer to a preconditioning matrix that approximates A^{-1}, like those used with preconditioned iterative solvers for linear systems.

Preconditioned Simultaneous iteration can be defined as a generalized polynomial method that treats a group of vectors simultaneously

$$x_j^{(k)} = P_{m_k}(M^{-1}A, M^{-1}B)x_j^{(0)}, j = 1, \cdots, m, \tag{3.9}$$

where P_{m_k} is a polynomial of degree m_k and $x_j^{(0)}$ are the initial vectors. The polynomial P_{m_k} does not have to be the same for different values of j. When it is, preconditioned simultaneous iteration (3.9) is equivalent to subspace iteration. A recursive procedure (Figure 3.3) can be developed using information from the subspace spanned by the group of vectors $x_j^{(k)}$,

$$\mathcal{S}^{(k)} = span\{x_1^{(k)}, x_2^{(k)}, \cdots, x_m^{(k)}\},$$

to construct new initial vectors for simultaneous iteration (3.9). Usually, the Rayleigh-Ritz procedure is used to extract information from the subspace $\mathcal{S}^{(k)}$.

Input: $(A, B, M, x_1^{(0)}, x_2^{(0)}, \cdots, x_m^{(0)})$
Output: Approximate eigenpairs $(\mu_j^{(k)}, x_j^{(k)})$ of the largest eigenvalues and corresponding eigenvectors of $Bx = Ax\mu$.

1. **for** $i = 1, 2, \cdots,$ **until** convergence
   - **1.1.** Select $x_j^{(0)}, j = 1, \cdots, m$
   - **1.2.** Iterate k steps to compute $\hat{x}_j^{(k)} = P_{m_k}(M^{-1}A, M^{-1}B)x_j^{(0)}$, $j = 1, \cdots, m$.
   - **1.3.** Use the Rayleigh-Ritz procedure on $B - \mu A$ with the subspace $\mathcal{S}^{(k)} = span\{\hat{x}_1^{(k)}, \cdots, \hat{x}_m^{(k)}\}$ to compute Ritz pairs $(\mu_j^{(k)}, x_j^{(k)})$.
2. **end**

Figure 3.3: The block preconditioned eigensolver

For Hermitian matrices, most iterative procedures for computing m eigenvectors corresponding to a chosen set of m eigenvalues construct a matrix $V \in \mathbb{C}^{n\times m}$ with orthonormal columns, which are approximations to the wanted eigenvectors. The *Rayleigh-Ritz procedure* (Figure 3.4) can be used to increase the accuracy of these eigenvector estimates. In fact, the Ritz values and Ritz vectors obtained with this procedure are the optimal approximations given only the information contained in V [30].

Input: $(A, V \in \mathbb{C}^{n\times m})$, where $A = A^H$ and $V^H V = I$
Output: Approximate eigenpairs (μ_j, x_j) of A.

1. $W = AV$
2. $H = V^H W$
3. Compute the m eigenpairs (μ_j, y_j) of H
4. Compute the m Ritz pairs $(\mu_j, x_j = Vy_j)$ of A and corresponding residuals $r_j = Ax_j - x_j\mu_j$

Figure 3.4: The Rayleigh-Ritz procedure

Under the general framework of block preconditioned eigensolvers (Figure 3.3), the defining factor for any method is the polynomial used in step (1.2). One option is to use the polynomial constructed through the preconditioned Conjugate Gradient (PCG) method. For finding the smallest eigenvalues of $A - \lambda B$, it is commonly implemented using two linked two-term recurrences

$$p^{(i)} = w^{(i)} + \beta^{(i)} p^{(i-1)}, \;\; x^{(i+1)} = x(i) + \alpha^{(i)} p^{(i)},$$

where the scalar $\alpha^{(i)}$ is chosen using a line search to minimize the Rayleigh quotient of $x^{(i+1)}$ and $\beta^{(i)}$ is computed to make the directions $p^{(i)}$ conjugate. LOBPCG (Figure 3.5) uses a variant of the PCG polynomial to find the largest eigenvalues of $B - \mu A$,

$$p^{(i)} = x^{(i)} - \tau^{(i-1)} x^{(i-1)}, \;\; x^{(i+1)} = \alpha^{(i)} w^{(i)} + \tau^{(i)} x^{(i)} + \gamma^{(i)} p^{(i)},$$

Input: $(A, B, M, x_1^{(0)}, x_2^{(0)}, \cdots, x_m^{(0)})$
Output: Approximate eigenpairs $(\mu_j^{(k)}, x_j^{(k)})$ of the largest eigenvalues and corresponding eigenvectors of $Bx = Ax\mu$.

1. Select $x_j^{(0)}$ and set $p_j^{(0)} = 0$, $j = 1, \cdots, m$
2. **for** $i = 1, 2, \cdots,$ **until** convergence
   - **2.1.** $\mu_j^{(i)} = ((x_j^{(i)})^H B x_j^{(i)})/((x_j^{(i)})^H A x_j^{(i)}), j = 1, \cdots, m$
   - **2.2.** $r_j = Bx_j^{(i)} - \mu_j^{(i)} Ax_j^{(i)}, j = 1, \cdots, m$
   - **2.3.** $w_j^{(i)} = M^{-1} r_j, j = 1, \cdots, m$
   - **2.4.** Use the Rayleigh-Ritz procedure on $B - \mu A$ using the subspace $span\{w_1^{(i)}, \cdots, w_m^{(i)}, x_1^{(i)}, \cdots, x_m^{(i)}, p_1^{(i)}, \cdots, p_m^{(i)}\}$.
   - **2.5.** $x_j^{(i+1)} = \sum_{k=1}^{m} \alpha_k^{(i)} w_k^{(i)} + \tau_k^{(i)} x_k^{(i)} + \gamma_k^{(i)} p_k^{(i)}, j = 1, \cdots, m$ (the j-th Ritz vector corresponding to the j-th largest Ritz value)
   - **2.6.** $p_j^{(i+1)} = \sum_{k=1}^{m} \alpha_k^{(i)} w_k^{(i)} + \gamma_k^{(i)} p_k^{(i)}$
3. **end**

Figure 3.5: The Locally-Optimal Block Preconditioned Conjugate Gradient method

where $w^{(i)} = M^{-1}(Bx^{(i)} - \mu^{(i)} Ax^{(i)})$ and $\mu^{(i)} = ((x^{(i)})^H B x^{(i)})/((x^{(i)})^H A x^{(i)})$. The scalars $\tau^{(i)}$ and $\gamma^{(i)}$ are chosen to maximize the Rayleigh quotient $x^{(i)}$ using the Rayleigh-Ritz procedure. This method exploits the fact that the Rayleigh-Ritz procedure is an inexpensive way to compute the local minimization of the Rayleigh quotient for larger dimensional subspaces.

3.2 Parameter Settings

With one exception, the MATLAB implementation of each of these methods is used to obtain the results presented in this thesis. They were treated as "out-of-the-box" type routines. No prior knowledge about these methods is assumed, so these routines

are not tweaked to obtain the best performance on each problem. The presented results are meant to be impartial and fair comparisons illustrating the characteristics of using a fixed-polynomial operator with IRA.

For some numerical examples, a comparison of these three methods is made by obtaining the most accurate approximate eigenvalues for a set computational cost. LOBPCG and JDQR adhere to a set cost by setting the maximum number of iterations accordingly. However, JDQR computes desired eigenvalues one-at-a-time to the desired tolerance. If the tolerance is set too high, not all of the desired eigenvalues will be calculated. Thus, the tolerance for JDQR *varies* for each problem. The setting for any parameter that *varies* will be defined in each section.

- *Jacobi-Davidson (JDQR/JDQZ)*

Parameter	Setting	Description
k	5	Number of desired eigenvalues
sigma	0	Shift
Tol	*varies*	Convergence tolerance
jmin	10	Minimum dimension search space
jmax	25	Maximum dimension search space

There are many more parameters that can be set with JDQR and especially with JDQZ. The default setting can be assumed for any parameter that is not stated here.

- *Locally Optimal Block Preconditioned Conjugate Gradient (LOBPCG)*

Parameter	Setting	Description
Uini	randn(n,5)	Initial guess, n is dimension of matrix
mytol	10^{-12}	Convergence tolerance
maxit	*varies*	Maximum number of iterations

- *IRA with GMRES Fixed-Polynomial*

Parameter	Setting	Description
k	5	Number of desired eigenvalues
m	25	Maximum dimension of Krylov subspace
sigma	0	Shift
maxit	*varies*	Maximum number of iterations
gtol	*varies*	Residual tolerance of GMRES polynomial
maxdeg	25	Maximum degree of GMRES polynomial

3.3 Large-Scale Laplacian

In this example the smallest five eigenvalues of a discrete approximation to the self-adjoint linear operator Δu are computed using ARPACK [20]. This operator is restricted to the unit square

$$\Omega := \{(x, y) \in \mathbb{R}^2 : 0 < x < 1, 0 < y < 1\}$$

and subject to Dirichlet boundary conditions. Discretization by finite differences with a grid size $h = \frac{1}{N+1}$ gives a matrix of the form

$$A = \begin{bmatrix} T_N & -I_N & 0 & \cdots & 0 \\ -I_N & T_N & -I_N & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -I_N & T_N & -I_N \\ 0 & \cdots & 0 & -I_N & T_N \end{bmatrix}$$

where I_N is the identity matrix in $\mathbb{R}^{N \times N}$ and

$$T_N = \begin{bmatrix} 4 & -1 & 0 & \cdots & 0 \\ -1 & 4 & -1 & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -1 & 4 & -1 \\ 0 & \cdots & 0 & -1 & 4 \end{bmatrix}.$$

The results are compared for IRA in regular mode, where the larger Ritz values are used as the roots of the restarting polynomial ψ, and shift-invert mode. For the shift-invert mode, the GMRES fixed-polynomial operator is compared to preconditioned GMRES started anew for each linear solve. A block Jacobi preconditioner is used for both the fixed-polynomial operator and preconditioned GMRES solver. This preconditioner is applied symmetrically so that an efficient three-term recurrence can be employed. For the preconditioned GMRES solver, results are compared when it is restarted, GMRES(25), and not restarted, GMRES(∞). Both GMRES(25) and GMRES(∞) are applied with a relative tolerance of 10^{-6}. The fixed-polynomial operator is constructed with a relative tolerance of 10^{-2}.

For this example, it is observed that IRA with the GMRES fixed-polynomial operator is the best blend of performance and accuracy. As the problem size increases, it uses far fewer matrix-vector products (Figure 3.6) than IRA in regular mode, GM-

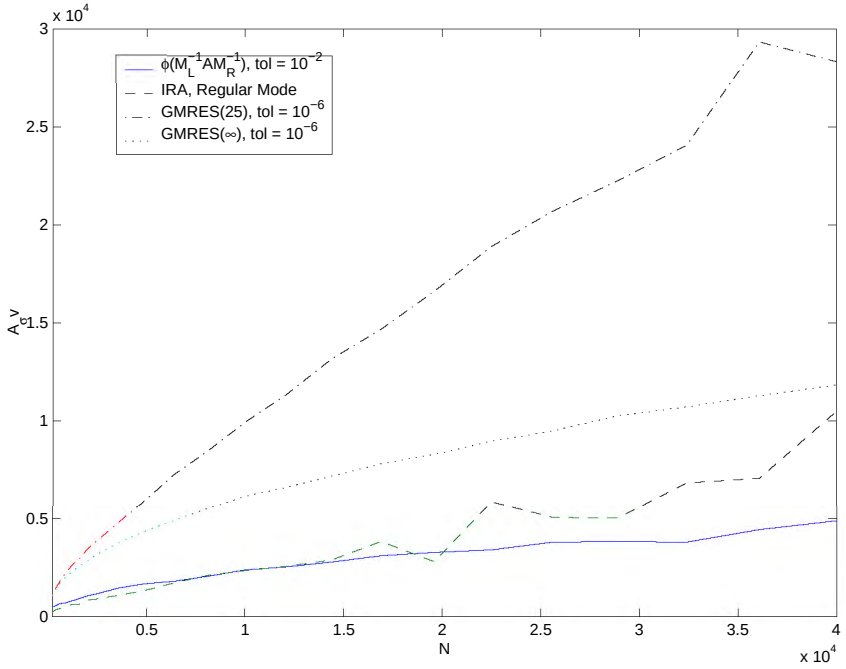

Figure 3.6: Matrix-vector products needed to compute smallest five eigenvalues

RES(25), or GMRES(∞). Moreover, IRA applied to the GMRES fixed-polynomial

operator is just as accurate in computing all five eigenvalues (Figure 3.7) as the regular

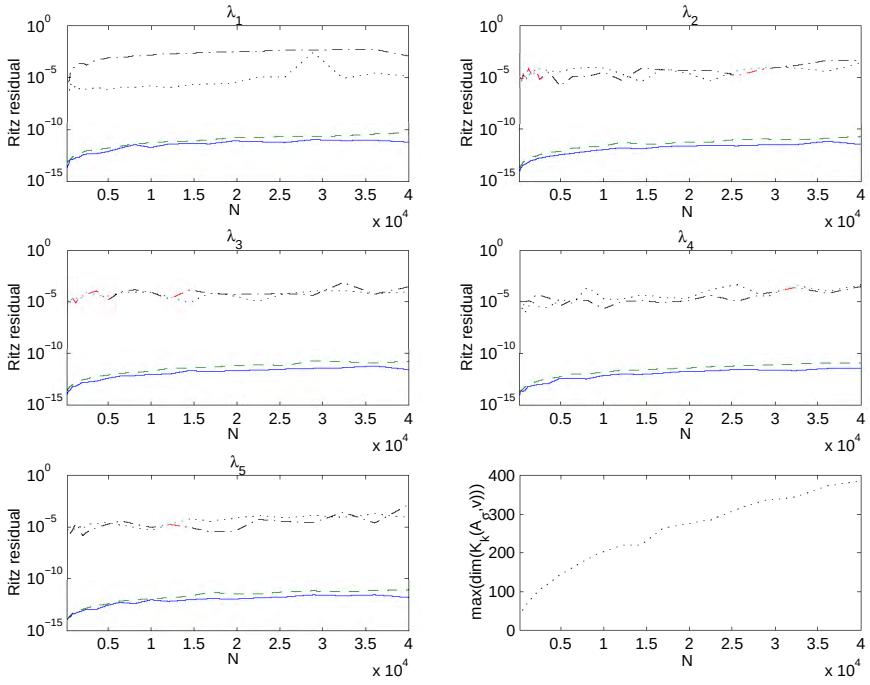

Figure 3.7: Ritz residual for the smallest five approximate eigenvalues

mode and more accurate than applying preconditioned GMRES.

The comparison of both preconditioned GMRES solvers serves to illustrate the tradeoff between storage constraints and computational cost. As a Krylov subspace method, it is impossible for GMRES to know in advance how large the subspace must be to obtain a good solution to the linear system. GMRES(∞) assumes that there is enough storage to keep all the basis vectors for the subspace, while GMRES(25) assumes a moderate amount of available storage. While restarting GMRES slows down the convergence, it requires a fixed amount of storage. Not restarting GMRES requires an increasing amount of storage for the Krylov basis, especially as the problem size grows (Figure 3.7). Regardless, neither preconditioned GMRES solver compares to using the GMRES fixed-polynomial operator.

3.4 Small-Scale Laplacian

In this example the smallest five eigenvalues of a discrete approximation to the self-adjoint linear operator Δu are computed using IRA with the GMRES fixed-polynomial operator and the well-known eigensolvers Jacobi-Davidson (JDQR) and LOBPCG. The discretization and preconditioning of the operator are the same as in the previous section (3.3). However, the dimension of the matrices in this example are slightly smaller, varying from 625 to 19600.

For this example there are two cost levels determined by the number of matrix-vector products: 500 (Figure 3.8) and 1000 (Figure 3.9). LOBPCG and JDQR

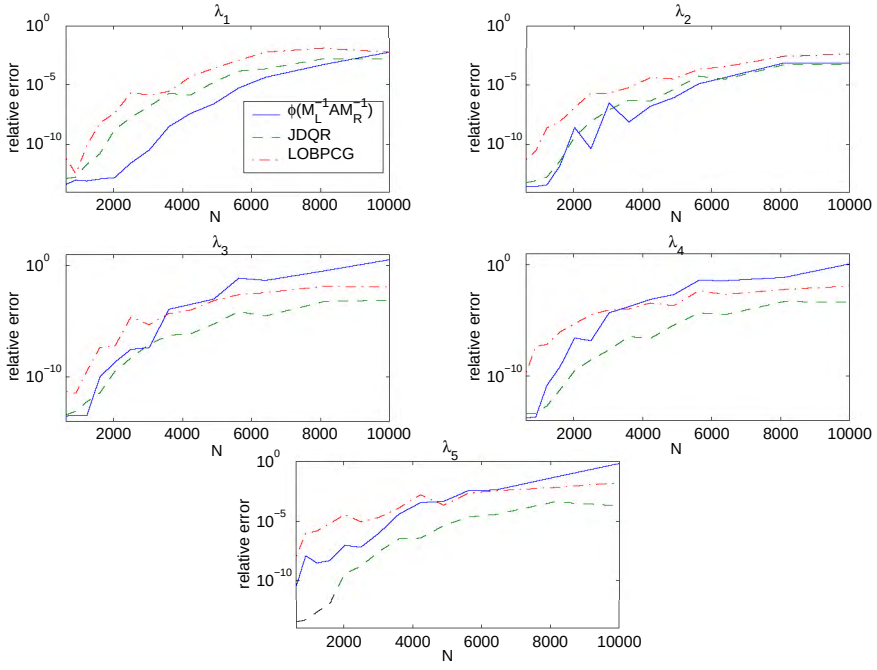

Figure 3.8: Relative error for the smallest five approximate eigenvalues
(500 matrix-vector products)

adhere to these cost levels by setting the maximum number of iterations accordingly. However, JDQR computes the five eigenvalues one-at-a-time to the desired tolerance. If the tolerance is set too high, not all five eigenvalues will be calculated. Thus, the

tolerance for JDQR is different for each matrix dimension so that all five eigenvalues were calculated to the highest accuracy possible. The GMRES fixed-polynomial operator is constructed with a relative tolerance of 10^{-2}. Other cost levels were analyzed, but these two are representative of the overall observed trends.

The results indicate that if five eigenvalues are required, then IRA with the fixed-polynomial operator computes the smallest eigenvalue more accurately than JDQR or LOBPCG, especially when the matrix-vector products are held to 500 (Figure 3.8). However, the relative error is higher for eigenvalues further away from the shift. This trend is similarly seen with LOBPCG. As would be expected, JDQR computes all five of the eigenvalues to the same accuracy.

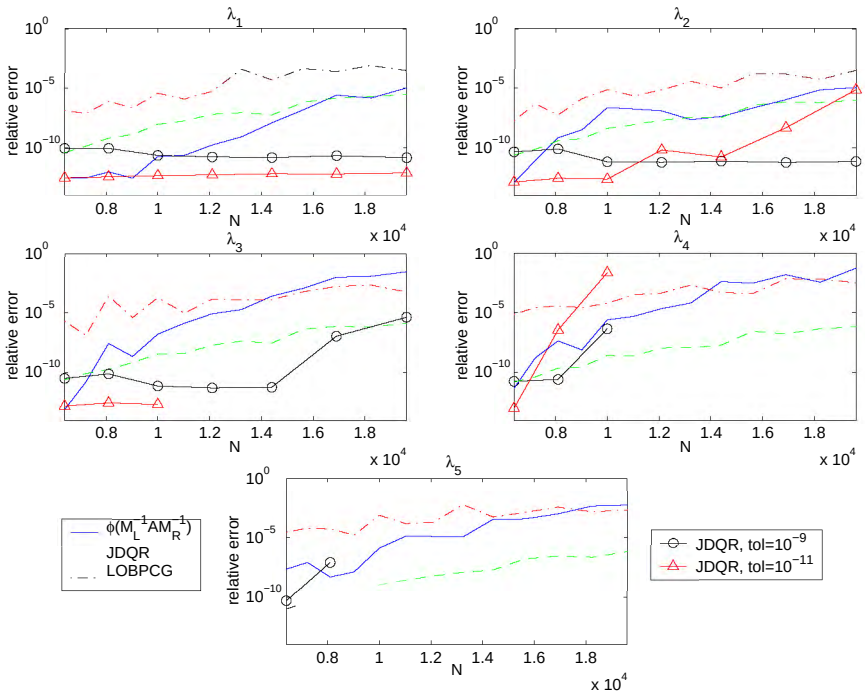

Figure 3.9: Relative error for the smallest five approximate eigenvalues
(1000 matrix-vector products)

This experiment is designed to compare these three methods with one particular goal: compute the same number of eigenvalues for the same cost. These methods are then compared by analyzing the accuracy of those computed eigenvalues. The

design of this experiment illustrates a perspective that may be unfair to JDQR. This is because Jacobi-Davidson is intended to compute one eigenvalue at a time to a prescribed accuracy, not multiple eigenvalues to varying precisions.

Another perspective is to compute a *few* eigenvalues for a fixed accuracy, where the number of computed eigenvalues is less important and the computational cost is still fixed. Figure 3.9 includes information about the accuracy and number of eigenvalues computed when the tolerance is held constant with JDQR. As the problem size increases, JDQR cannot compute all five eigenvalues to the set tolerance. Only the information from the computed eigenvalues is displayed in Figure 3.9. If no data is plotted, then it can be assumed that the eigenvalue was not computed for this matrix dimension. The results indicate that with JDQR the accuracy can be maintained for the smallest eigenvalue if the number of computed eigenvalues is not as important.

3.5 Wind-Driven Barotropic Ocean Model

This example will consider the stability of a single homogeneous layer of fluid driven by a sinusoidal wind stress. This fluid is enclosed by a rectangular basin of uniform depth on a mid-latitude β-plane, where the effect of the earth's sphericity is modeled by a linear variation of the planetary vorticity [31]. IRA with the GMRES fixed-polynomial and Jacobi-Davidson (JDQR) will be used to perform the stability analyses on this wind-driven barotropic ocean model. The model will be introduced first, followed by the definition of its parameters. Some interesting behavior of the fixed-polynomial operator is presented before results are discussed.

Consider the potential vorticity/streamfunction formulation of the barotropic circulation problem [25]

$$\frac{\partial \psi}{\partial t} + J[\psi, \nabla^2 \psi] + \beta \frac{\partial \psi}{\partial x} = \nu \nabla^4 \psi + \frac{1}{\rho_0 H} \nabla \times \tau. \tag{3.10}$$

Here, ψ is the streamfunction, τ is the wind stress, β is the meridional gradient of the planetary vorticity, ν is the eddy viscosity, ρ_0 is the uniform ocean density, and

H is the uniform depth of the ocean. J is the Jacobian operator

$$J[\psi, \nabla^2\psi] = -\frac{\partial\psi}{\partial y}\frac{\partial\nabla^2\psi}{\partial x} + \frac{\partial\psi}{\partial x}\frac{\partial\nabla^2\psi}{\partial y}.$$

The boundary conditions for (3.10) are free-slip

$$\nabla^2\psi = 0 \ and \ \frac{\partial^2\nabla^2\psi}{\partial n^2} = 0 \ on \ \partial\Omega,$$

where n is normal to the boundary.

This model idealizes the domain to a closed rectangular ocean basin Ω where the wind forcing τ is sinusoidal (Figure 3.10). This forcing produces a double-gyre with an anticyclonic gyre in the northern basin and a cyclonic gyre in the southern basin, as depicted by the streamfunctions in Figure 3.11.

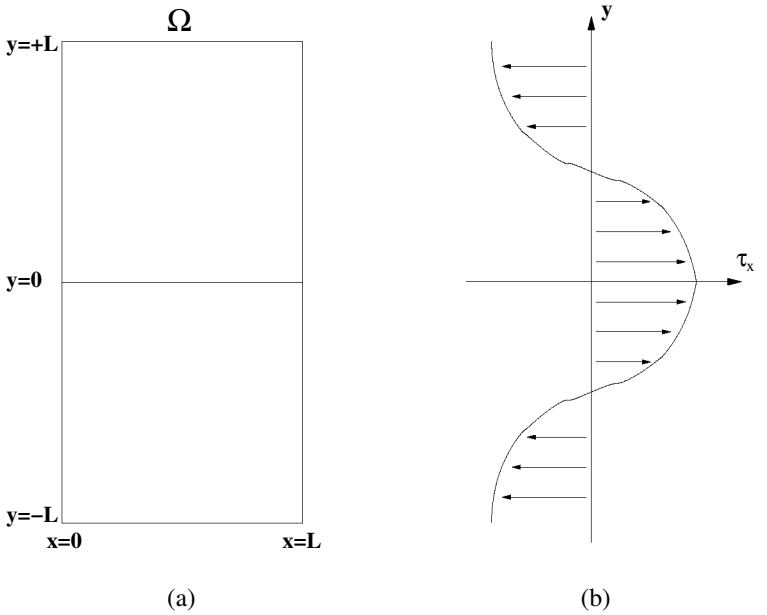

Figure 3.10: Layout of (a) domain and (b) wind stress forcing

The barotropic model (3.10) is nondimensionalized using a length scale corresponding to the horizontal extent of the domain L and a velocity scale U given by the Sverdrup relation [28]

$$U = \frac{\pi\tau_0}{\rho_0 H\beta L}.$$

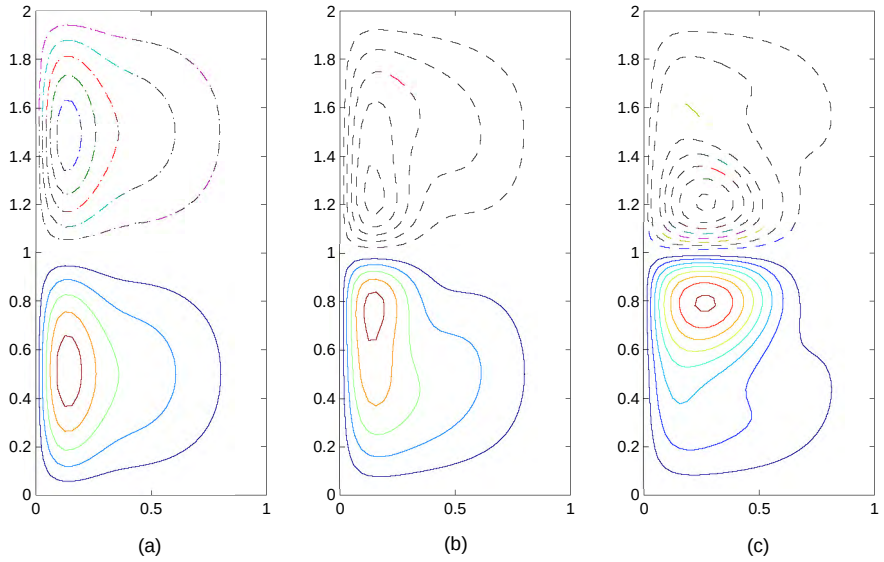

Figure 3.11: Steady-state streamfunctions depicting double-gyre structure; (a) $Ro = .0001$, (b) $Ro = .0016$, (c) $Ro = .0036$

A time-scale corresponding to L/U allows (3.10) to be rewritten as

$$\frac{\partial q}{\partial t} + J[\psi, q] = \sin(\pi y) - \frac{\nu}{\beta L^3}\nabla^4\psi, \tag{3.11}$$

where q is the potential vorticity

$$q = Ro\nabla^2\psi + y,$$

and $Ro = \frac{U}{\beta L^2}$ is the Rossby number.

A linear stability analysis of this barotropic flow model requires linearizing (3.11) about a steady state. This linearized problem is then discretized on the rectangular domain using the Arakawa discretization [1] for the Jacobian and centered differences for the other terms. Dirichlet boundary conditions are imposed. The resulting standard eigenvalue problem is sparse and the spectrum of A is complex. The stability of this model is determined by the right-most eigenvalues of the discretized operator

A. An example of the spectrum and sparsity pattern of the linearized model is given in Figure 3.12.

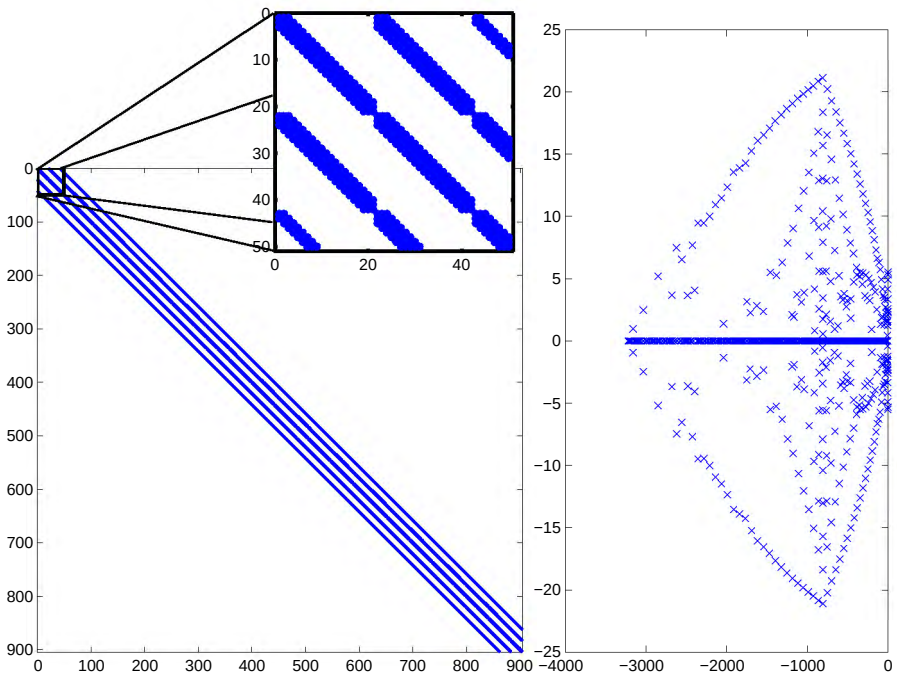

Figure 3.12: Sparsity pattern and spectrum of linearized model; dim = 903

The success of this preconditioning scheme depends upon the ability of the fixed-polynomial operator to approximate the spectral transformation. Restricting the degree of the polynomial has a definite affect on this. Furthermore, the quality of the preconditioner can profoundly impact the accuracy of this operator. A comparison of spectra from a sequence of restricted-degree, fixed-polynomial operators is displayed for various preconditioners. The base case is to analyze spectra from polynomials constructed without any preconditioner (Figure 3.13). For this example, it is obvious that a low-degree polynomial does not contain enough information about the spectrum of the original matrix.

The quality and availability of a preconditioner is very important in the construction of a low-degree, fixed-polynomial operator. Previous results have shown that by using a modest preconditioner, the relative residual tolerance used to construct

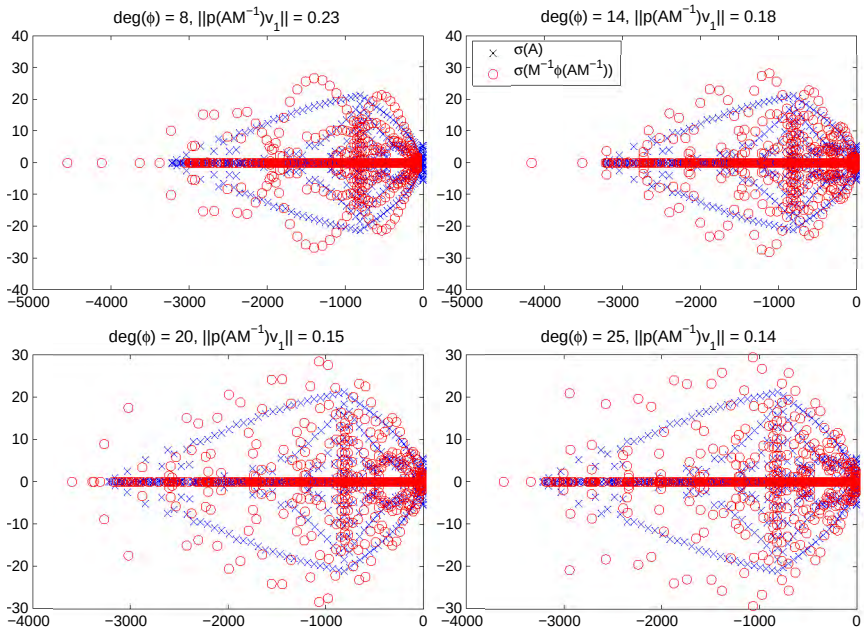

Figure 3.13: Spectra of restricted-degree, fixed-polynomial operators; dim = 903; no preconditioner

this operator does not have to be very low for the fixed-polynomial operator to have enough information about the spectrum. A good illustration of this trend is shown in Figure 3.14, where the same matrix previously used to construct the fixed-polynomial operator with no preconditioner (Figure 3.13) is now preconditioned using an incomplete LU factorization with a drop tolerance of 10^{-2}. This sequence of spectra confirms that this preconditioning scheme has the ability to construct a low-dimensional, fixed-polynomial operator that approximates a spectral transformation very well.

For this example IRA with the GMRES fixed-polynomial is compared to Jacobi-Davidson (JDQR) for computing the five right-most eigenvalues of the nondimensionalized model (3.11) linearized about a steady state. The discretization of the linearized equation yields matrices that range in dimension from 700 to 3000. An incomplete LU factorization was used as a preconditioner for both methods. To determine any dependence on preconditioner quality, two drop tolerances were used: 10^{-3} and 10^{-4}. The previous discussion used an incomplete LU preconditioner with

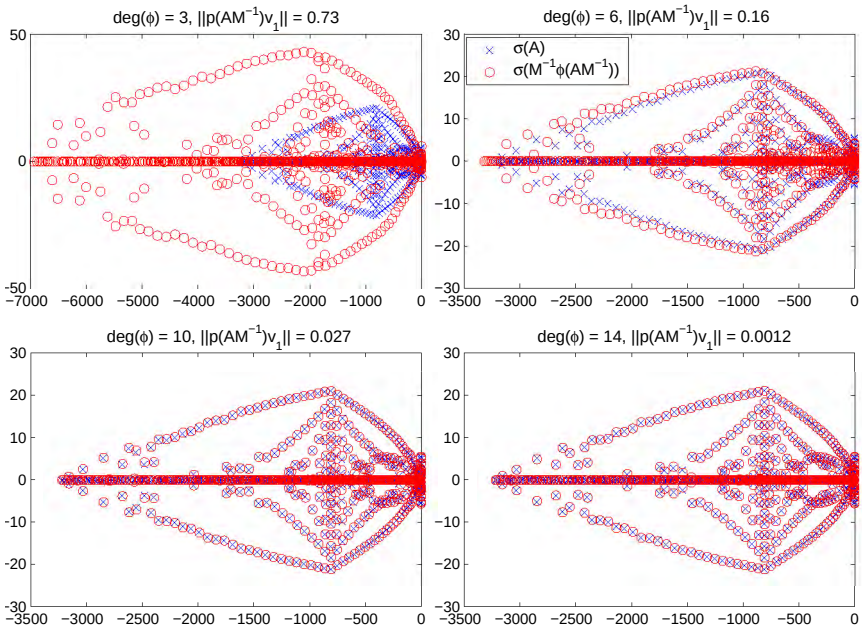

Figure 3.14: Spectra of restricted-degree, fixed-polynomial operators; dim = 903; $M = luinc(A_\sigma, 10^{-2})$

a drop tolerance of 10^{-2}, which was sufficient for a problem of a smaller dimension. However, as the matrix dimension increases, a preconditioner constructed using that drop tolerance is insufficient for the convergence of either JDQR or IRA with the GMRES fixed-polynomial operator.

Similar to the small-scale Laplacian example, these two methods are to compute the same number of eigenvalues for the same cost. Then the comparison is in the accuracy of the computed eigenvalues. Depending upon the quality, each preconditioner is assigned a cost level which is determined by the number of matrix-vector products. The lower quality incomplete LU preconditioner has a drop tolerance of 10^{-3} and allows 1000 matrix-vector products (Figure 3.15), while the higher quality preconditioner has a drop tolerance of 10^{-4} and allows 500 matrix-vector products (Figure 3.16).

JDQR computes the five eigenvalues one-at-a-time to the desired tolerance. If the tolerance is set too high, not all five eigenvalues will be calculated. Thus, the

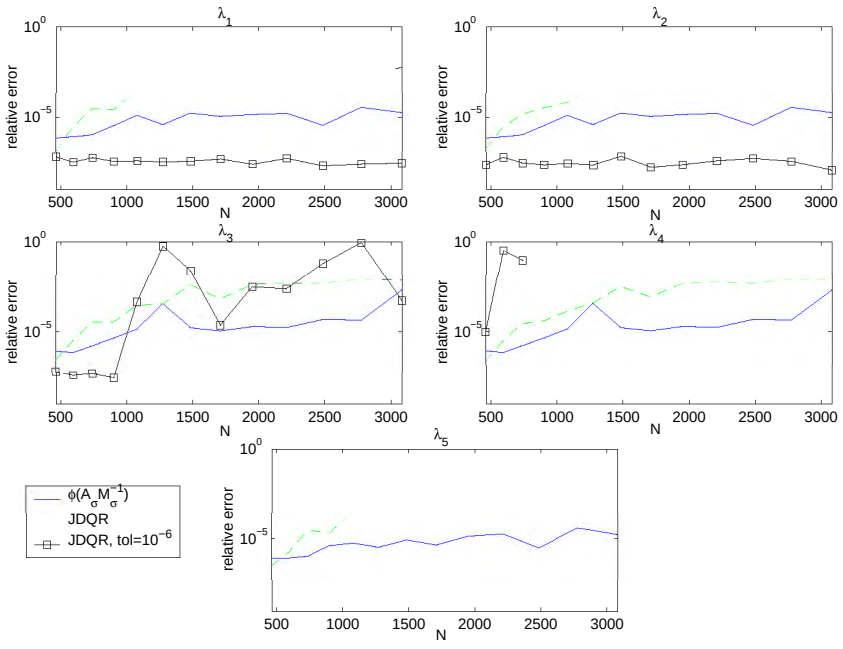

Figure 3.15: Relative error for the five rightmost eigenvalues; $M = luinc(A_\sigma, 10^{-3})$ (1000 matrix-vector products)

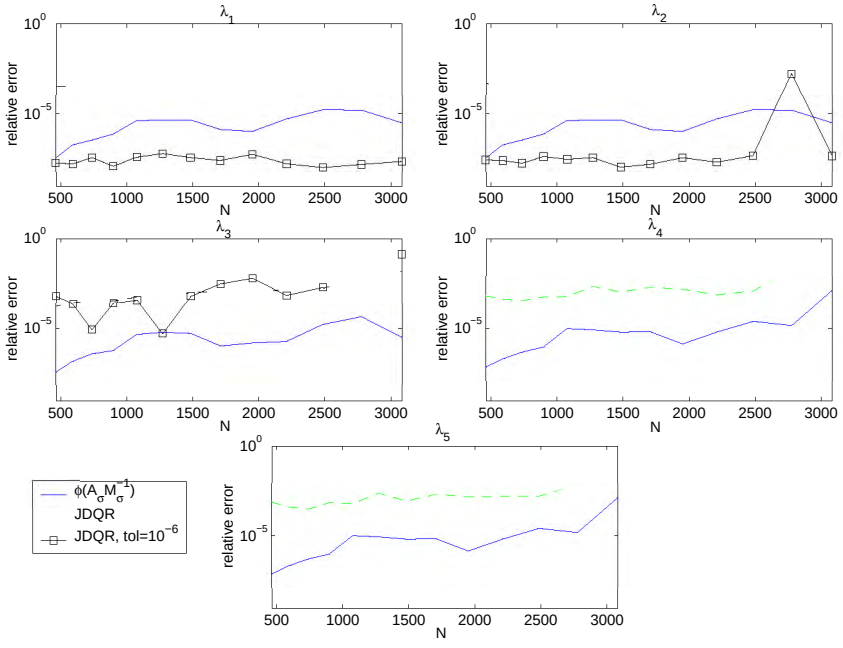

Figure 3.16: Relative error for the five rightmost eigenvalues; $M = luinc(A_\sigma, 10^{-4})$ (500 matrix-vector products)

tolerance for JDQR is different for each matrix dimension so that all five eigenvalues are calculated to the highest accuracy possible. These problems are then rerun using a fixed tolerance 10^{-6} with JDQR to analyze the number of eigenvalues it could compute at a higher tolerance. The GMRES fixed-polynomial operator is constructed with a relative tolerance of 10^{-6}.

The results indicate that if five eigenvalues are required, then IRA with the fixed-polynomial operator computes all the eigenvalues more accurately than JDQR. However, if accuracy is more important than the number of computed eigenvalues, then JDQR with a fixed tolerance can compute the two rightmost eigenvalues more accurately. These observations are independent of preconditioner.

3.6 Semiconductor Laser Model

For a 2D dynamic rate equation model of a semiconductor laser, the time dependent field interaction is crucial. It is necessary to calculate the electric field and confinement factor with time during the evolution of the pulses. This requires the solution to the 2D scalar wave equation

$$\frac{\partial^2 E(x,y)}{\partial x^2} + \frac{\partial^2 E(x,y)}{\partial y^2} + (\eta(x,y)^2 k_o^2 - \beta^2) E(x,y) = 0 \tag{3.12}$$

on the rectangular domain $\Omega = [0, L_x] \times [0, L_y]$, subject to Dirichlet boundary conditions, every pico-second. For a time period of 5 nano-seconds, the scalar wave equation needs to be solved 5000 times. In equation (3.12), $E(x,y)$ is the electric field, η is the complex refractive index, β is the mode propagation constant, $k_o = 2\pi/\lambda$ is the wave number, and λ is the wavelength [51].

The matrix eigenvalue equation is obtained by using Galerkin's method on the continuous equation (3.12), expanding the electric field using basis functions which are the products of sine functions,

$$\phi_i(x,y) = \frac{2}{(L_x L_y)^{1/2}} \sin\left(\frac{m_i \pi x}{L_y}\right) \sin\left(\frac{n_i \pi y}{L_y}\right). \tag{3.13}$$

Here the integers m_i and n_i are the wave numbers for the basis i in the x and y direction, respectively. Furthermore, N_x and N_y are the number of waves in the x and y direction, respectively, and $N = N_x N_y$. The basis functions (3.13) vanish at the boundary of Ω and form an orthonormal set of functions

$$\langle \phi_i(x,y), \phi_j(x,y) \rangle = \int_0^{L_y} \int_0^{L_x} \phi_i(x,y)\phi_j(x,y)\, dx\, dy = \delta_{ij}.$$

The electric field can be expanded in terms of this orthogonal basis as

$$E(x,y) = \sum_{i=1}^{N} a_i \phi_i(x,y).$$

This expansion can be substituted into the scalar wave equation (3.12), multiplied on the left side of by $\phi_j(x,y)$, and integrated over the domain Ω to obtain the matrix equation

$$\sum_{i=1}^{N} (A_{ij} - \beta \delta_{ij}) a_i = 0 \tag{3.14}$$

where

$$A_{ij} = (\left(\frac{m_i \pi}{L_x}\right)^2 + \left(\frac{n_i \pi}{L_y}\right)^2)\delta_{ij} + k_o^2 \langle \phi_i(x,y)\eta(x,y), \phi_j(x,y)\eta(x,y) \rangle.$$

The matrix equation (3.14) is complex and non-Hermitian due to the refractive index η in the second term of A_{ij}.

The eigenvalues of the matrix equation (3.14) represent both bound and continuum modes of the waveguide. Thus, the eigenvalues of interest are those whose real part lie within a range of values determined by the semiconductor laser. Within that range, the eigenvalue with largest imaginary part is of interest because it corresponds to the mode with the largest modal gain. It is this eigenpair that can be used to accelerate the solution of the 2D dynamic rate equation model.

For this example, 15 waves are used in each of the x and y directions, resulting in a complex non-Hermitian matrix of dimension 225. Although the dimension of this matrix is small and computing a direct factorization may be inexpensive, there

are a couple of reasons to approach this problem with an iterative eigensolver. One reason is the length of the sequence. It is computationally expensive to compute the direct factorization of 5000 matrices, even if they are of small dimension. If only a couple eigenvalues are necessary, then iterative eigensolvers may be more cost effective. Depending upon the problem, the eigenvalues of interest for a matrix sequence may be related from one time-step to the next. This information can be exploited by IRA with the GMRES fixed-polynomial operator or Jacobi-Davidson (JDQR) to accelerate convergence.

This example originated from a researcher, Canice O'Brien, from the Physics Department at Trinity College Dublin who recognized that direct methods were too computationally expensive. The short sequence of 20 matrix equations provided through this collaboration depicts an extremely coarse time discretization of a nano-second time period. The results from this short sequence will allow speculation about the performance of IRA with the GMRES fixed-polynomial on matrix sequences and how it compares to JDQR. An example of the spectrum is given in Figure 3.17. The eigenvalues of interest are well-separated, but not in an easy part of the spectrum for IRA to compute eigenvalues without a spectral transformation. However, a shift can be obtained using knowledge about the range where the eigenvalues of interest are expected to lie.

IRA with the GMRES fixed-polynomial and JDQR are used to track the two eigenvalues of largest imaginary part whose real parts are in the range of interest. An LU factorization of the first matrix in the sequence is used to compute the eigenvalues and eigenvectors of interest, then employed as a preconditioner for all of the subsequent matrices in the sequence. The quality of the LU factorization as a preconditioner for these matrices is tracked to see if it degrades and should be refactored. The shift for the spectral transformation is held constant throughout the sequence of matrices ($\sigma = 11.30$). Both these methods are accelerated by using the eigenvectors corresponding to the eigenvalues of interest from one matrix in the sequence as an

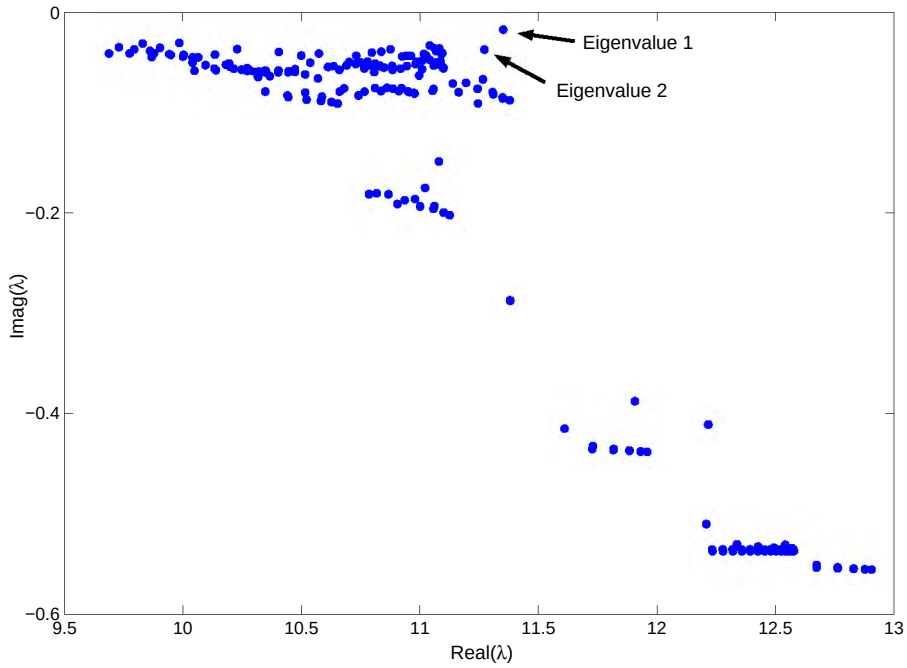

Figure 3.17: Spectrum of complex, non-Hermitian matrix from sequence

initial guess for eigenvectors of the next matrix in the sequence. The tolerance for JDQR is set at 10^{-6} and the GMRES fixed-polynomial operator is constructed with a relative tolerance of 10^{-4}.

First, it should be noted that the eigenvalues of the operators do not vary much throughout the sequence (Figure 3.18). The eigenvalues change color from dark to light to show the progression through the sequence of matrices which appears to have an elliptical path. This periodic behavior is a result of the periodic pulsed light output from the semiconductor laser [51]. Thus the LU factorization of the initial operator is a great preconditioner for the later matrices in the sequence.

The relative errors of the computed eigenvalues for both methods are illustrated in Figure 3.19. These errors are about the same for both methods. The dotted line in this figure indicates the tolerance used in constructing the GMRES fixed-polynomial. It is interesting to note that the relative errors of the eigenvalues computed using IRA with the GMRES fixed-polynomial are more dependent upon the residual reduction

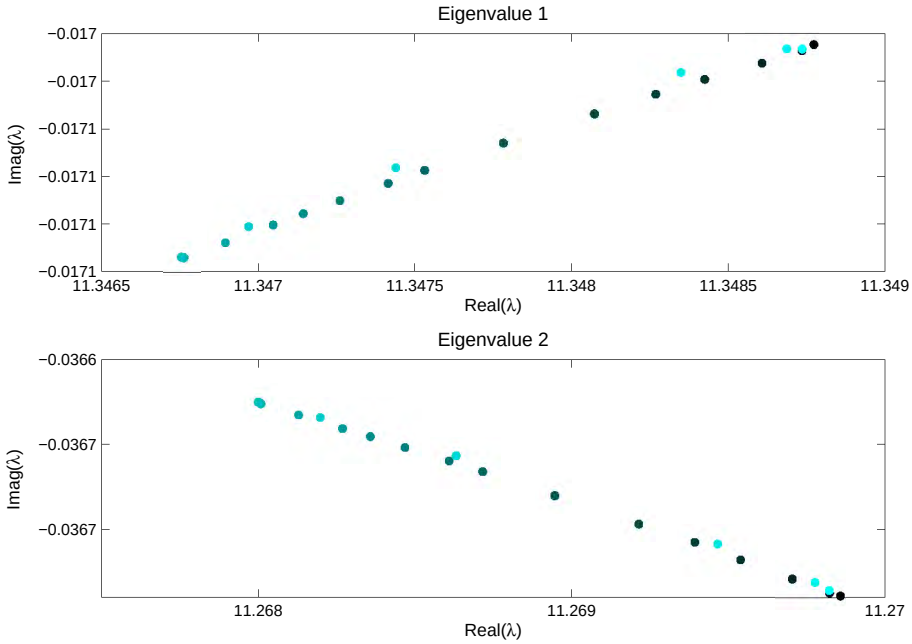

Figure 3.18: Tracking eigenvalues of interest throughout sequence

of GMRES than the imposed tolerance. JDQR has residuals that are more consistent with the imposed tolerance.

The periodicity in the quality of the preconditioner is apparent in the top plot of Figure 3.20. As would be expected, the initial LU factorization becomes a better preconditioner at the end of the matrix sequence. Refactoring the matrix at the peak of this period to obtain a better preconditioner might be considered. The cost of this refactorization should not be ignored, especially when the current preconditioner improves in quality later in the sequence.

The performance of both methods in computing these two eigenvalues for each matrix in the sequence is indicated by the bottom plot of Figure 3.20. The computational cost of JDQR and IRA with the GMRES fixed-polynomial operator is about the same for all the matrices in the sequence. There are a few instances where JDQR is less expensive, but overall the two methods are comparable. It is interesting to note that these instances coincide with the deterioration in the quality of the

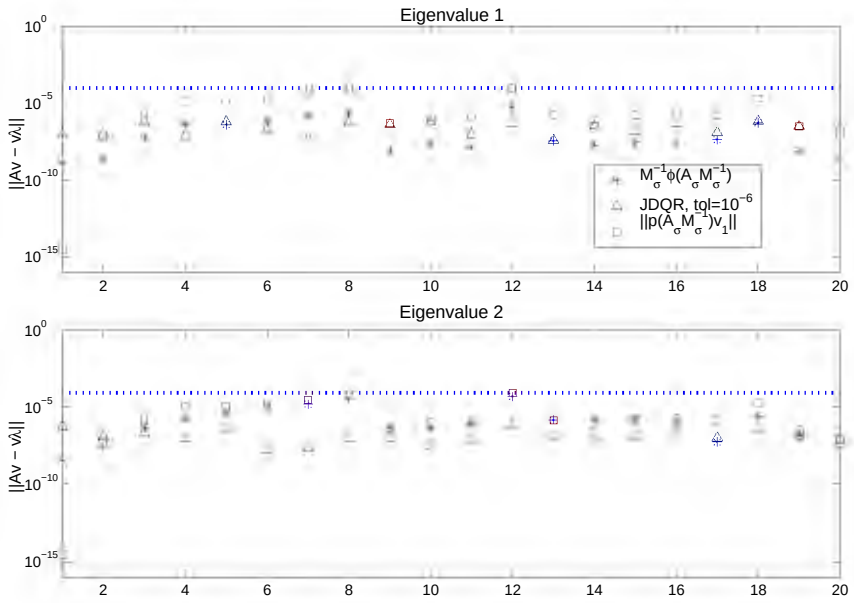

Figure 3.19: Relative errors for eigenvalues of interest

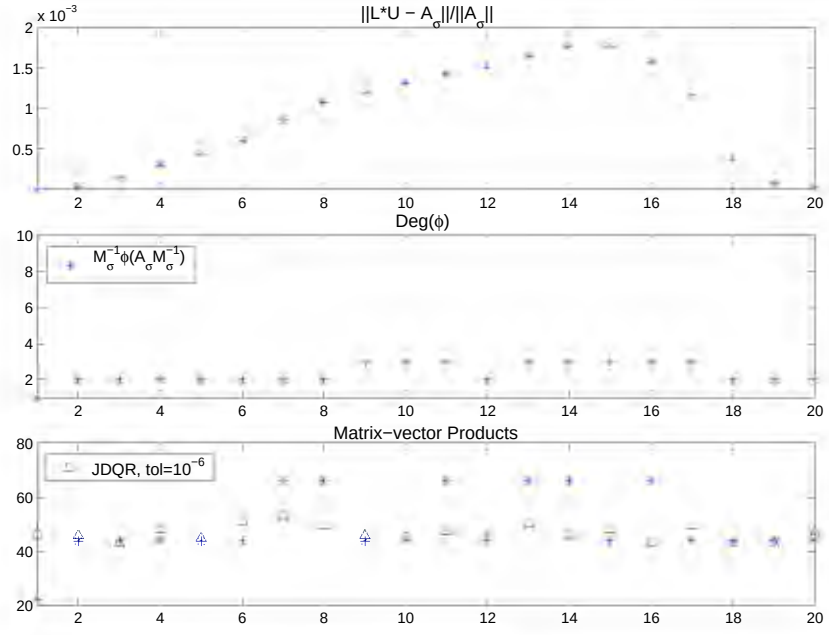

Figure 3.20: Quality of preconditioner and cost of IRA with GMRES fixed-polynomial operator

preconditioner, which may indicate that GMRES fixed-polynomial is sensitive to this. It also is possible that JDQR takes more advantage of the information from the previous eigenvectors in computing the eigenvectors of the next operator. Either way, both methods are more efficient than employing a dense method to track the eigenvalues of interest for a matrix sequence.

Chapter 4

Summary and Future Work

Accelerating a Krylov subspace method with a spectral transformation requires a linear system solve for each basis vector of the Krylov subspace. It is preferable to use a direct method for these solves. However, due to storage constraints and computational cost, large-scale eigenvalue problems require the use of an iterative method for these linear solves. Unfortunately, unless stringent accuracy is specified, this results in the application of a different operator ϕ for every linear solve. Another option is to compute the operator ϕ once and apply it instead of using an iterative solver.

Numerical results presented in this thesis indicate that constructing a GMRES fixed-polynomial operator to approximate the spectral transformation is promising. The eigenpairs of the original problem can be easily reconstructed from the Ritz vectors of the Krylov subspace method. The accuracy of these eigenpairs is dependent upon the norm of the residual polynomial, $\|p(A_\sigma M_\sigma^{-1})\|$. If the fixed-polynomial operator closely approximates the spectral transformation, this norm should be small. A starting vector that is equally weighted in all eigendirections seems to facilitate the construction of such an operator.

Numerical results show that IRA with this preconditioning scheme performs better and is more accurate for large-scale problems than using preconditioned GMRES

started anew for each linear solve. It compares favorably against Jacobi-Davidson and LOBPCG on small-scale Hermitian problems. However, the accuracy of IRA with this preconditioning scheme suffers for eigenvalues further away from the shift. The most important result comes from the linear stability analysis of a wind-driven barotropic ocean circulation model. If multiple eigenvalues are required, then IRA applied to a fixed-polynomial operator is more accurate than Jacobi-Davidson for a fixed computational cost.

Most of the results presented in this thesis are acquired using MATLAB codes because fully-developed C or Fortran implementations of JDQR and LOBPCG are not publically available. This limited the dimension of the eigenvalue problems to those on the order of 10^4. In research, most large-scale eigenvalue problems of interest have dimensions on the order of 10^7. These large problems are solved on supercomputers, using numerical software packages such as parallel ARPACK (PARPACK). This thesis introduces a preconditioning scheme that extends the domain of applications for ARPACK. Evidence of this is presented in the discussion of the implementation and numerical results. Because of the dominance of ARPACK in solving large-scale eigenvalue problems, extending the domain of applications for this software package has a significant impact.

These results are encouraging, but many questions about this preconditioning scheme remain.

- The feasibility of a convergence theory for Krylov subspace methods applied to this fixed-polynomial operator is uncertain. An attempt to develop a convergence theory will require a subspace approximation theory approach [4] for IRA. Meerbergen formulates an argument for inexact Möbius transformations with rational Krylov [26], which may provide some guidance.

- Short-term recurrence methods, like BiCG-stab [48] and QMR [12], build fixed-polynomial operators that require less storage and computation. These operators may or may not be as accurate as approximating the spectral transforma-

tion with a fixed-polynomial operator built from a long-term recurrence method. An ideal fixed-polynomial operator is one that uses a short-term recurrence and is as accurate as the one built from GMRES.

- The quality of the fixed-polynomial operator depends heavily upon the preconditioner. The current results only explore some basic preconditioners: block Jacobi, incomplete LU, and incomplete Cholesky. More advanced preconditioning methods should be studied for building the fixed-polynomial operator: multigrid, approximate inverse [5], and multilevel ILU [42].

- Restricting the degree of the fixed-polynomial operator impacts its ability to approximate the spectral transformation. Techniques should be developed to improve the accuracy of this operators' approximation, subject to a dimensional constraint.

- It is uncertain if this preconditioning scheme can aid IRA in computing interior eigenvalues. The current results are only encouraging for the computation of the right-most eigenvalues. An error bound should be found for the right-most eigenvalues of the fixed-polynomial as approximations to those of the spectral transformation.

- Current heuristics for the accuracy of a computed eigenpair include a bound that is dependent upon $\|p(A_\sigma M_\sigma^{-1})\|$. Unfortunately, the iterative method used to construct p can only offer a bound for $\|p(A_\sigma M_\sigma^{-1})v_1\|$, where v_1 is the initial vector. However, current results show that $\|p(A_\sigma M_\sigma^{-1})\| \approx \|p(A_\sigma M_\sigma^{-1})v_1\|$ if v_1 is a random starting vector of unit length. This phenomenon may be explainable using probability theory.

- The linear stability analysis of fluid flow models can be affected by the non-normality of the linearized problem [46]. Relatively small perturbations of this problem can result in large perturbations of its eigenvalues. The pseudospec-

trum [45] of the linearized problem provides information about the behavior of the eigenvalues when the problem is subject to perturbations. Furthermore, it even provides information about the transient growth of solutions to the unperturbed problem. Therefore, pseudospectra will be used to analyze the non-normality of these fluid flow models.

Bibliography

[1] A. Arakawa. Computational design for long-term numerical integration of the equations of fluid motion: Two dimensional incompressible flow, part 1. *J. Comput. Phys.*, 1:119–143, 1966.

[2] W. E. Arnoldi. The principle of minimized iteration in the solution of the matrix eigenproblem. *Quart. Appl. Math.*, 9:17–29, 1951.

[3] Zhaojun Bai, James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst. *Templates for the solution of algebraic eigenvalue problems: a practical guide*. SIAM, Philadelphia, 2000.

[4] C. Beattie, M. Embree, and J. Rossi. Convergence of restarted Krylov subspaces to invariant subspaces. Preprint Numerical Analysis TR 01/02, OUCL, Oxford, UK, 2001.

[5] Michele Benzi, John C. Haws, and Miroslav Tůma. Preconditioning highly indefinite and nonsymmetric matrices. *SIAM J. Sci. Comput.*, 22:1333–1353, 2000.

[6] E. A. Burroughs, L. A. Romero, R. B. Lehoucq, and A. J. Salinger. Large scale eigenvalue calculations for computing the stability of buoyancy driven flows. Technical Report 2001-0113J, Sandia National Laboratories, 2001. Submitted to the Journal of Computational Physics.

[7] J. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization. *Math. Comp.*, 30:772–795, 1976.

[8] E.R. Davidson. The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real symmetric matrices. *J. Comp. Phys.*, 17:87–94, 1975.

[9] T. Ericsson and A. Ruhe. The spectral transformation Lanczos method for the numerical solution of large sparse generalized symmetric eigenvalue problems. *Math. Comp.*, 35(152):1251–1268, 1980.

[10] G. Faber. Über polynomische Entwicklungen. *Math. Ann.*, 57:389–408, 1903.

[11] Diederik R. Fokkema, Gerard L. G. Sleijpen, and Henk A. van der Vorst. Jacobi-Davidson style QR and QZ algorithms for the reduction of matrix pencils. *SIAM J. Sci. Comput.*, 20:94–125, 1998.

[12] R. W. Freund and N. M. Nachtigal. QMR: a quasi-minimal residual method for non-Hermitian linear systems. *Num. Math.*, 60:315–339, 1991.

[13] T. J. Garratt. *The Numerical Detection of Hopf Bifurcations in Large Systems Arising in Fluid Mechanics*. PhD thesis, University of Bath, School of Math. Sciences, Bath,UK, 1991.

[14] T.J. Garratt, G. Moore, and A. Spence. Two methods for the numerical detection of Hopf bifurcations. In *Bifurcation and Chaos: Analysis, Algorithms, and Applications*, pages 119–123. Birkhäuser, 1991.

[15] T.J. Garratt, G. Moore, and A. Spence. A generalised Cayley transform for the numerical detection of Hopf bifurcations in large systems. Technical report, School of Mathematical Sciences, University of Bath, 1992.

[16] Vincent Heuveline and Miloud Sadkane. Arnoldi–Faber method for large non-hermitian eigenvalue problems. *ETNA*, 5:62–76, 1997.

[17] Andrew V. Knyazev. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. Report CCM 149, University of Colorado, Denver, 2000.

[18] R. B. Lehoucq and Karl Meerbergen. Using generalized Cayley transformations within an inexact rational Krylov sequence method. *SIAM J. Matrix Anal. Appl.*, 20:131–148, 1998.

[19] R. B. Lehoucq and D. C. Sorensen. Deflation techniques for an implicitly restarted Arnoldi iteration. *SIAM J. Matrix Analysis and Applications*, 17(4):789–821, 1996.

[20] R.B. Lehoucq, D.C. Sorensen, and C. Yang. *ARPACK Users Guide: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi methods*. SIAM Publications, Philadelphia, PA, 1998.

[21] Richard B. Lehoucq and Andrew G. Salinger. Large-scale eigenvalue calculations for stability analysis of steady flows on massively parallel computers. *Int. J. Numer Methods Fluids*, 36:309–327, 2001.

[22] F. Leja. Sur certaines suits liées aux ensembles plan et leur application à la représentation conforme. *Ann. Polon. Math.*, 4:8–13, 1957.

[23] T. A. Manteuffel. Adaptive procedure for estimating parameters for the non-symmetric Tchebychev iteration. *Numer. Math.*, 31:183–208, 1978.

[24] Thomas A. Manteuffel and Gerhard Starke. On hybrid iterative methods for nonsymmetric systems of linear equations. *Numerische Math.*, 73:489–506, 1996.

[25] S.P. Meacham and P.S. Berloff. Barotropic, wind-driven circulation in a small basin. *J. Mar. Res.*, 55:523–563, 1997.

[26] K. Meerbergen. *Robust methods for the calculation of rightmost eigenvalues of nonsymmetric eigenvalue problems*. PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium, 1996.

[27] Noel M. Nachtigal, Lothar Reichel, and Lloyd N. Trefethen. A hybrid GMRES algorithm for nonsymmetric linear systems. *SIAM J. Matrix Anal. Appl.*, 13:796–825, 1992.

[28] Balasubramanya T. Nadiga and Benjamin P. Luce. Global bifurcation of Shilnikov type in a double-gyre ocean model. *J. Phys. Ocean.*, 31:2669–2690, 2001.

[29] Bahram Nour-Omid, Bersford N. Parlett, Thomas Ericsson, and Paul S. Jensen. How to implement the spectral transformation. *Math. Comp.*, 48:663–673, 1987.

[30] B. N. Parlett. *The Symmetric Eigenvalue Problem*. Prentice-Hall, Englewood Cliffs, NJ, 1980.

[31] J. Pedlosky. *Geophysical Fluid Dynamics*. Springer-Verlag, 1987.

[32] A. Ruhe. Rational Krylov sequence methods for eigenvalue computation. *Lin. Alg. Appl.*, 58:391–405, 1984.

[33] A. Ruhe. The rational Krylov algorithm for nonsymmetric eigenvalue problems. III: Complex shifts for real matrices. *BIT*, 34:165–176, 1994.

[34] A. Ruhe. Rational Krylov algorithms for nonsymmetric eigenvalue problems. II. matrix pairs. *Lin. Alg. Appl.*, 197, 198:283–295, 1994.

[35] A. Ruhe. Rational Krylov: A practical algorithm for large sparse nonsymmetric matrix pencils. *SIAM J. Sci. Comput.*, 19:1535–1551, 1998.

[36] Y. Saad. Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices. *Lin. Alg. Appl.*, 34:269–295, 1980.

[37] Y. Saad. Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems. *Math. Comp.*, 42:567–588, 1984.

[38] Y. Saad. Numerical solution of large nonsymmetric eigenvalue problems. *Comput. Phys. Commun.*, 53:71–90, 1989.

[39] Y. Saad. *Numerical methods for large eigenvalue problems*. Manchester University Press, Manchester, UK, 1992.

[40] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7:856–869, 1986.

[41] G. L. G. Sleijpen and H.A. van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 17:401–425, 1996.

[42] G. L. G. Sleijpen and F. W. Wubs. Exploiting multilevel preconditioning techniques in eigenvalue computations. Technical Report 1117, University Utrecht, Department of Mathematics, 1999.

[43] D. C. Sorensen. Implicit application of polynomial filters in a k-step Arnoldi method. *SIAM J. Matrix Anal. Appl.*, 13(1):357–385, 1992.

[44] Gerhard Starke and Richard S. Varga. A hybrid Arnoldi–Faber iterative method for nonsymmetric systems of linear equations. *Numerische Math.*, 64:213–240, 1993.

[45] L. N. Trefethen. Pseudospectra of matrices. In D. F. Griffiths and G. A. Watson, editors, *Numerical Analysis 1991*, pages 234–266. Longman, 1992.

[46] Lloyd N. Trefethen, Anne E. Trefethen, Satish C. Reddy, and Tobin A. Driscoll. Hydrodynamic stability without eigenvalues. *Science*, 261:578–583, 1993.

[47] L.N. Trefethen. Numerical computation of the Schwarz–Christoffel transformation. *SIAM J. Sci. Statist. Comput.*, 1:82–101, 1980.

[48] H. van der Vorst. Bi–CGSTAB: A fast and smoothly converging variant of Bi–CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 13:631–644, 1992.

[49] J. H. Wilkinson. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, 1965.

[50] C. Yang. *Accelerating the Arnoldi Iteration–Theory and Practice*. PhD thesis, Rice University, Houston, Texas, 1998.

[51] Masaaki Yuri, James S. Harris Jr., Toru Takayama, Osamu Imafuji, Hiroki Naito, Masahiro Kume, Kunio Itoh, and Takaaki Baba. Two-dimensional analysis of self-sustained pulsation for narrow-stripe AlGaAs lasers. *IEEE J. Quantum Electron.*, 1:473–479, 1995.