ALGORITHMS FOR THE UNITARY EIGENVALUE PROBLEM

By

RODEN JASON A. DAVID

A dissertation submitted in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

WASHINGTON STATE UNIVERSITY
Department of Mathematics

MAY 2007

To the Faculty of Washington State University:

The members of the Committee appointed to examine the dissertation of RODEN JASON A. DAVID find it satisfactory and recommend that it be accepted.

Chair

ALGORITHMS FOR THE UNITARY EIGENVALUE PROBLEM

Abstract

by Roden Jason A. David, Ph.D.
Washington State University
May 2007

Chair: David S. Watkins

Eigenvalues of unitary matrices arise in a variety of contexts in applied mathematics. This dissertation present four new algorithms for computing the eigenvalues of unitary matrices. In chapter 1, we give an overview these algorithms, and then survey the major applications where eigenvalues of unitary matrices arise. In chapter 2, we present the unitary QR algorithm, an algorithm that can used to compute all of the eigenvalues of a unitary matrix. In chapter 3, we present two Krylov space algorithms that approximate some of the eigenvalues of a large unitary matrix. Finally, in chapter 4, we present an algorithm that compute the eigenvalues of a unitary matrix U when U is expressed as a product $U = U_1 \cdots U_n$ of unitary matrices of the same order. As a special case, we consider the generalized eigenvalue problem for unitary matrices.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

During the last ten years, there has been considerable progress in the development of specialized algorithms that compute the eigenvalues of unitary matrices. Algorithms that were developed include several techniques that make use of the Schur parametrization of a unitary matrix to efficiently implement the QR algorithm [25, 4], the divide-and-conquer methods [9, 27, 28], and the bisection method [13]. There was also an approach based on matrix pencils [12], and several algorithms designed for orthogonal matrices [8, 2]. The major applications that stimulate research in this area of eigenvalue computation lie in signal processing [32, 2, 14, 15], in time series analysis [6], in Gaussian quadrature on the unit circle [24, 25], and in trigonometric approximations [33, 23].

This dissertation is an effort to contribute to the growing body of specialized algorithms that compute eigenvalues of unitary matrices. In developing these new algorithms, we have incorporated techniques from several recent algorithms that compute eigenvalues in general. Among these were the Krylov-Schur algorithms [35], and the product QR algorithms [42].

In the following sections we will review some basic definitions, give an overview of the new algorithms, and motivate the problem of computing the eigenvalues of unitary matrices by considering some of its applications.

1.1 Basic Definitions

We denote by $M_n(\mathbb{C})$ the set of all $n \times n$ matrices with complex entries, and by $\mathbb{C}^n$ the set of all $n \times 1$ complex vectors. If $A = (a_{ij})$ is a matrix, its *Hermitian transpose* A^H is defined as $A^H = (\overline{a}_{ji})$ where $\overline{a}$ is the complex conjugate of a. A matrix $U \in M_n(\mathbb{C})$ is *unitary* if and only if $U^H U = I$ where I is the $n \times n$ identity matrix.

An *inner product* on $\mathbb{C}^n$ is a map $\langle , \rangle : \mathbb{C}^n \times \mathbb{C}^n \mapsto \mathbb{C}$ that satisfies

1. (*positive definiteness*) $\langle x, x \rangle \geq 0$ with $\langle x, x \rangle = 0$ if and only if $x = 0$
2. (*Hermitian symmetry*) $\langle x, y \rangle = \overline{\langle y, x \rangle}$
3. (*linearity in the first argument*) $\langle \alpha x + \beta y, z \rangle = \alpha \langle x, z \rangle + \beta \langle y, z \rangle$

for all $x, y, z \in \mathbb{C}^n$ and all scalars α and β. An example of an inner product in $\mathbb{C}^n$ is the *standard inner product* in $\mathbb{C}^n$ defined by $\langle x, y \rangle = y^H x$. This is the inner product that we will use throughout this dissertation.

A *norm* on $\mathbb{C}^n$ is a map $\| \cdot \| : \mathbb{C}^n \mapsto \mathbb{R}^+ \cup \{0\}$ which satisfies

1. (*positive definiteness*) $\| x \| \geq 0$ with $\| x \| = 0$ if and only if $x = 0$

2. (*absolute homogeneity*) $\| \alpha x \| = | \alpha | \| x \|$

3. (*triangle inequality*) $\| x + y \| \leq \| x \| + \| y \|$

for all $x, y \in \mathbb{C}^n$ and all scalars α. An example of a norm in $\mathbb{C}^n$ is the 2-norm defined by $\| x \|_2 = \sqrt{\langle x, x \rangle}$.

If $U \in M_n(\mathbb{C})$ is unitary, then it follows that $\langle Ux, Uy \rangle = \langle x, y \rangle$, and that $\| Ux \|_2 = \| x \|_2$ for all $x, y \in \mathbb{C}^n$.

A scalar λ is called an *eigenvalue* of a matrix $A \in M_n(\mathbb{C})$ if and only if there exists a nonzero vector $x \in \mathbb{C}^n$ such that

$$Ax = \lambda x.$$

Such a vector x is called an *eigenvector* of A associated with λ. We shall refer to the set of all eigenvalues of a square matrix A as the *spectrum* of A. If $U \in M_n(\mathbb{C})$ is unitary, and λ is an eigenvalue of U, then $| \lambda | = 1$.

Two matrices $A, B \in M_n(\mathbb{C})$ are said to be *similar* if and only if there exists a nonsingular matrix $P \in M_n(\mathbb{C})$ such that $B = P^{-1}AP$. A matrix $C \in M_n(\mathbb{C})$ is said to be *diagonalizable* if and only if there exists a diagonal matrix $D \in M_n(\mathbb{C})$ such that C is similar to D. A matrix $N \in M_n(\mathbb{C})$ is *normal* if and only if $N^H N = N N^H$. It is well-known that the set of normal matrices are precisely those that are unitarily diagonalizable. In particular, unitary matrices are normal, hence unitary matrices are unitarily diagonalizable.

1.2 Overview of the Algorithms

Topics that serve as background material for the whole dissertation are given in chapter 2. We consider the construction of elimination matrices and the Schur parametrization of unitary unreduced upper Hessenberg matrices.

After these introductory topics, we introduce the unitary QR algorithm. This is an implementation of the QR algorithm in terms of its Schur parametrization. The algorithm is not without precedent. An implementation of the QR algorithm for unitary upper Hessenberg matrices in terms of its Schur parameters has been introduced by Gragg in [25]. In this implementation, only one shift can be used. The implementation suffers from instability problems. In [36], M. Stewart showed how this instability can be remedied.

Our implementation of the unitary QR algorithm is an improvement over Gragg's implementation in the sense that it can do multi-shift QR iterations of arbitrary degree. We also prove its backward stability. Finally, our implementation is conceptually straightforward and easy to understand.

The unitary QR algorithm seeks out all of the eigenvalues of a unitary matrix. In applications where the order of the matrix is large, the usual goal is to approximate the eigenvalues which lie in a specified region of the spectrum. In this case, Krylov space methods are often used, where the original matrix is approximated by its restriction on a Krylov space.

We present two Krylov space algorithms in chapter 3. The first is an inexact Krylov-Schur algorithm that approximates the eigenvalues of a unitary matrix nearest a specified target point that lies in the unit circle. The algorithm is implemented purely in term of Schur parameters. The approximating Krylov space is constructed by a variant of Gragg's isometric Arnoldi process [24, 26] that uses a pair of two-term recurrence relations. The process of generating the Krylov space can be restarted implicitly using the same pair of recurrence relations.

The second Krylov space algorithm in chapter 3 performs a Cayley transform on the unitary matrix to give rise to an Hermitian matrix. A Lanczos-Schur algorithm is used to find the eigenvalues of the Hermitian matrix of largest magnitude. These dominant eigenvalues are mapped to the eigenvalues of the unitary matrix nearest the specified target by the inverse Cayley transform.

In chapter 4, we consider the product eigenvalue problem for unitary matrices. Given $U \in M_n(\mathbb{C})$ as a product of unitary matrices, we perform a modified product QR algorithm to the corresponding cyclic matrix, and then use the unitary QR algorithm presented in chapter 2 to compute all of the eigenvalues of U. We treat the generalized eigenvalue problem for unitary matrices as a special case for the algorithm developed for the product unitary eigenvalue problem.

1.3 Some Applications

In this section, we cover a sampling of applications where the eigenvalues of unitary matrices arise. The applications cited here are not meant to be exhaustive. We refer the reader to [3], and to the references cited below for details.

1.3.1 Frequency Estimation

As a first application, we consider the problem of estimating the frequencies of a wide sense stationary random process $x(n)$ given by

$$x(n) = \sum_{k=1}^{p} \Big(A_k e^{in\omega_k} + w(n)$$

where the amplitudes A_k are complex,

$$A_k = |\,A_k\,| e^{i\phi_k}$$

with ϕ_k uncorrelated random variables uniformly distributed on the interval $[-\pi, \pi]$, and where $w(n)$ is white noise with mean zero and variance σ^2. The frequencies ω_k are not random but unknown, and in typical applications, they are the "information bearing" part of the signal. In speech signals, for example, these frequencies correspond to the format frequencies [31], while in sonar signals they represent bearing or velocity information [29].

Pisarenko [32, 2] showed that if M_{p+1} is the autocorrelation matrix of $x(n)$, then the smallest eigenvalue $\lambda_{\min}$ of M_{p+1} is equal to the variance σ^2 of the white noise.

Further if $\lambda_{\min}$ is of algebraic multiplicity one, and $v = (v_j)_0^p$ is an eigenvector of M_{p+1} associated with $\lambda_{\min}$ normalized so that $v_p = 1$, then the polynomial

$$\psi_p(t) = \sum_{j=0}^{p} v_j \, t^j$$

has p distinct zeroes $e^{i\hat{\omega}_1}, \cdots, e^{i\hat{\omega}_p}$ on the unit circle where the phase angles $\{\hat{\omega}_j\}_1^p$ are estimates for the frequencies $\{\omega_j\}_1^p$. These estimates $\{\hat{\omega}_j\}_1^p$ are called *Pisarenko frequency estimates.*

The computational problem of Pisarenko frequency estimation is divided into two subproblems:

1. The determination of p, the determination of the autocorrelation matrix M_{p+1}, and the computation of $\lambda_{\min} = \sigma^2$.

2. The estimation of the frequencies $\{\omega_j\}_1^p$ and of the amplitudes $|\, A_k \,|$.

Some aspects of the first subproblem are discussed in the papers [16, 19, 18]. We are concerned with the second subproblem. To find the zeros of ψ_p, we can form the companion matrix

$$\begin{pmatrix} 0 & 0 & 0 & \cdots & -v_p \\ 1 & 0 & 0 & \cdots & -v_{p-1} \\ 0 & 1 & 0 & \cdots & -v_{p-2} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & -v_1 \end{pmatrix}$$

of ψ_p. This matrix however is not unitary, even if its eigenvalues lie on the unit circle.

In [2], the authors showed how to construct an upper Hessenberg unitary matrix H whose characteristic polynomial is $\psi_p(t)$. Using the Szegö recursion (or Levinson's algorithm), the unitary matrix H is constructed in terms of its Schur parameters, making it convenient to directly apply the unitary QR algorithm described in section 2.5.

1.3.2 Approximation by Trigonometric Polynomials

Given a set of m distinct node points $\{\theta_k\}_{k=1}^m$ on the interval $[0, 2\pi)$, and a set of positive weights $\{w_k^2\}_{k=1}^m$, consider a real-valued function $f(\theta)$ whose values at θ_k are known. In [33], the authors presented an algorithm that approximates $f(\theta)$ by the trigonometric polynomial

$$t(\theta) = a_0 + \sum_{j=1}^{l} \left(a_j \; \cos(j\theta) + b_j \; \sin(j\theta) \right)$$

where $l < m/2$ which minimizes the discrete least squares error

$$\| f - t \| = \left(\sum_{k=1}^{m} | f(\theta_k) - t(\theta_k) |^2 w_k \right)^{1/2} .$$

The algorithm is based on a scheme for solving an inverse eigenvalue problem for unitary Hessenberg matrices presented in [5]. The least-squares approximant is obtained by incorporating the node information one at a time.

1.3.3 Other Applications

We briefly cite other applications of eigenvalues of unitary matrices that are found in the literature. In the area of time series analysis and discrete-time control theory, the computation of the poles of a stable autoregressive model from the Schur parameters of a unitary Hessenberg matrix H is presented in [6]. The eigenvalues of H represent the poles of an associated lossless model, and from these eigenvalues, the zeroes of the desired Szegö polynomials are computed using a continuation method [7].

More recently, the distribution of the eigenvalues of unitary random matrices have been investigated [22], and have found applications in telephone encryption, and in connection with the Riemann zeta function [20].

Finally, all of these applications are rooted in the fact that the Schur parameters of a unitary Hessenberg matrix have an intimate connection with the theory of orthogonal polynomial on the unit circle and on Gaussian quadrature [38, 40, 24, 26].

Chapter 2

The Unitary QR Algorithm

We begin the development of algorithms for the unitary eigenvalue problem in this chapter. The basic algorithmic tools are the elementary elimination matrices which we introduce in sections 2.1 and 2.2. For unitary matrices, there is an elegant factorization of unreduced upper Hessenberg unitary matrices in term of Given's reflectors. The entries of the Given's reflector are the Schur parameters of the unitary matrix. We shall introduce this factorization and the Schur parametrization in section 2.3.

Fundamental to eigenvalue computations is the QR algorithm. In this chapter, we present an implementation of the QR for unitary matrices using the Schur parameters of the matrix. We review the QR algorithm in section 2.4, and finally present the unitary QR algorithm in section 2.5.

2.1 Elementary Matrices

Let $u, v \in \mathbb{C}^n$. A matrix E of the form

$$E = I - uv^H \tag{2.1.1}$$

is called an *elementary matrix*. If $v^H u \neq 1$, then

$$\begin{aligned}
(I - uv^H)\left(I - \frac{uv^H}{v^H u - 1}\right) &= I - uv^H - \frac{uv^H}{v^H u - 1} + \frac{uv^H uv^H}{v^H u - 1} \\
&= I + \frac{-uv^H(v^H u - 1) - uv^H + uv^H uv^H}{v^H u - 1} \\
&= I + \frac{-(v^H u)uv^H + uv^H - uv^H + uv^H uv^H}{v^H u - 1} \\
&= I.
\end{aligned}$$

Thus if $E = I - uv^H$ where $v^H u \neq 1$, then E is nonsingular and $E^{-1} = I - \dfrac{uv^H}{v^H u - 1}$.

Let $x, y \in \mathbb{C}^n$. We begin with the problem of finding an elementary matrix E such that $Ex = y$. This is an underdetermined system of n equations in n^2 unknowns. Since E has the form $E = I - uv^H$, we have

$$\begin{aligned}
(I - uv^H)x &= y \\
x - uv^H x &= y.
\end{aligned}$$

Solving for u we get

$$u = \frac{x - y}{v^H x} \tag{2.1.2}$$

provided $v^H x \neq 0$. We can choose $v \in \mathbb{C}^n$ such that $v^H x \neq 0$, and define u by (2.1.2). It follows that if $E = I - uv^H$, then $Ex = y$.

Proposition 2.1. *Let $x, y \in \mathbb{C}^n$. If $v \in \mathbb{C}^n$ such that $v^H x \neq 0$ and $u = \dfrac{x-y}{v^H x}$ then $E = I - uv^H$ satisfies $Ex = y$.*

Consider now an elementary matrix P defined by

$$P = I - 2uu^H$$

where $\| u \|_2 = 1$. In the notation of (2.1.2), we have $v = 2u$ where $u \in \mathbb{C}^n$, $\| u \|_2 = 1$. Such a matrix P is called a *Householder reflector.*

We note two basic properties of a Householder reflector. First

$$P^H = (I - 2uu^H)^H = I - 2uu^H = P$$

hence P is Hermitian. Further

$$\begin{aligned} P^H P = PP^H &= (I - 2uu^H)(I - 2uu^H) \\ &= I - 2uu^H - 2uu^H + 4uu^H uu^H \\ &= I. \end{aligned}$$

Thus P is unitary.

Proposition 2.2. *If $P = I - 2uu^H$ is a Householder reflector, where $u \in \mathbb{C}^n$, $\| u \|_2 =$ 1, then P is Hermitian and unitary.*

Let $x \in \mathbb{C}^n$, $x \neq 0$. We consider next the problem of finding a Householder reflector P such that $Px = \alpha e_1$ for some constant α. Since P is unitary, it follows that $\alpha = \beta \| x \|_2$ for some complex constant β with $|\beta| = 1$. Further, by proposition 2.1, u takes the form

$$u = \frac{1}{\gamma}(x - \alpha e_1)$$

where $\gamma = v^H x = 2u^H x$, from which we get

$$|\gamma|^2 = 2(\| x \|_2^2 - \overline{\alpha} x_1).$$

Since the left side of this equation is real, and the quantity $\| x \|_2$ is real, it follows that $\overline{\alpha} x_1$ must be real. This forces $\beta = \pm \frac{x_1}{|x_1|}$ provided $x_1 \neq 0$. If $x_1 = 0$, then we can take $\beta = 1$.

Computationally, we take $\beta = -\frac{x_1}{|x_1|}$ whenever $x_1 \neq 0$ to avoid a loss-of-significance error. To see this, let $c = -\frac{x_1}{|x_1|}$ so that

$$\alpha = c\| x \|_2 = -\frac{x_1}{|x_1|} \| x \|_2.$$

This implies that

$$\begin{aligned} \overline{\alpha} x_1 &= -\frac{\overline{x_1}}{|x_1|} \| x \|_2 x_1 \\ &= -\frac{|x_1|^2}{|x_1|} \| x \|_2 \\ &= -|x_1| \, \| x \|_2 \end{aligned}$$

from which we see that $\overline{\alpha} x_1$ is a nonpositive real number, and that $\overline{\alpha} x_1 = \overline{\overline{\alpha} x_1} = \alpha \overline{x_1}$.

Define $\xi = \| x - \alpha e_1 \|_2$ so that

$$\begin{aligned}
\xi^2 &= \| x - \alpha e_1 \|_2^2 \\
&= (x^H - \overline{\alpha} e_1^T)(x - \alpha e_1) \\
&= x^H x - \overline{\alpha} x_1 - \alpha \overline{x_1} + |\alpha|^2 \\
&= 2\| x \|_2 - 2\overline{\alpha} x_1 \\
&= 2(\| x \|_2 + |x_1|)\| x \|_2.
\end{aligned}$$

Hence the expression for $\| x - \alpha e_1 \|_2^2$ does not involve the subtraction of two positive real numbers. Further,

$$\begin{aligned}
(x^H - \overline{\alpha} e_1^T)x &= x^H x - \overline{\alpha} x_1 \\
&= \| x \|_2^2 - (-\, |x_1| \, \| x \|_2) \\
&= \xi^2/2.
\end{aligned}$$

Hence the expression for $(x^H - \overline{\alpha} e_1^T)x$ also does not involve the subtraction of two positive real numbers.

Finally we verify that if $P = I - 2uu^H$, where $u = (x - \alpha e_1)/\| x - \alpha e_1 \|_2$, then

$$\begin{aligned} Px &= (I - 2uu^H)x \\ &= x - \frac{2(x - \alpha e_1)(x^H - \overline{\alpha} e_1^T)x}{\| x - \alpha e_1 \|_2^2} \\ &= \frac{x\xi^2}{\xi^2} - \frac{2(x - \alpha e_1)\xi^2/2}{\xi^2} \\ &= x - (x - \alpha e_1) \\ &= \alpha e_1. \end{aligned}$$

Thus the choice $\beta = -\dfrac{x_1}{|x_1|}$ for $x_1 \neq 0$ computationally avoids a loss-of-significance error whenever we compute Px for which $P = I - 2uu^H$ where $u = (x - \alpha e_1)/\| x - \alpha e_1 \|_2$ with $\alpha = \beta \| x \|_2$.

Proposition 2.3. *Let $x = (x_i) \in \mathbb{C}^n$, $x \neq 0$. Define β by $\beta = -\dfrac{x_1}{|x_1|}$ if $x_1 \neq 0$ and $\beta = 1$ if $x_1 = 0$. Let $\alpha = \beta \| x \|_2$. If $P = I - 2uu^H$ where*

$$u = \frac{x - \alpha e_1}{\| x - \alpha e_1 \|_2},$$

then P is a Householder reflector that satisfies $Px = \alpha e_1$.

Finally we state Schur's triangularization theorem which guarantees what can be achieved with eigenvalue algorithms.

Proposition 2.4. *If $A \in M_n(\mathbb{C})$, then there exists a unitary $U \in M_n(\mathbb{C})$ and an upper-triangular $T \in M_n(\mathbb{C})$ such that $U^H A U = T$, where the diagonal entries of T are the eigenvalues of A.*

Proof. We use by induction on the order of the matrix n. The result holds for $n = 1$. Assume now that each $(n-1) \times (n-1)$ matrix is unitarily similar to an upper triangular matrix. Let $A \in M_n(\mathbb{C})$. Let (λ, x) be an eigenpair of A with $\|x\|_2 = 1$. By proposition 2.3, we can construct a Householder reflector $R = R^H = R^{-1}$ such that $Rx = e_1$ or equivalently $x = Re_1$. Hence the first column of R is x, and we have $R = (x \quad V)$ for some V such that $V^H x = 0$. We have

$$\begin{aligned} R^H A R &= \begin{pmatrix} x^H \\ V^H \end{pmatrix} \begin{pmatrix} \lambda x & AV \end{pmatrix} \\ &= \begin{pmatrix} x & x^H AV \\ 0 & V^H AV \end{pmatrix} \end{aligned}$$

The matrix $V^H AV$ is of order $n-1$. By the induction assumption, there exists a unitary matrix Q such that $Q^H(V^H AV)Q = \tilde{T}$ is upper triangular. Taking

$$U = R \begin{pmatrix} 1 & 0 \\ 0 & Q \end{pmatrix}$$

we get

$$U^H AU = \begin{pmatrix} \lambda & x^H AVQ \\ 0 & \tilde{T} \end{pmatrix} = T$$

which is upper triangular having the eigenvalues of A along the main diagonal.

□

2.2 Hessenberg Matrices

A matrix $H = (h_{ij}) \in M_n(\mathbb{C})$ is said to be an *upper Hessenberg matrix* if $h_{ij} = 0$ for $i > j+1$. Further if H is an upper Hessenberg matrix and $h_{j+1,j} \neq 0$ for $j = 1, \ldots, n-1$, then H is said to be an *unreduced* upper Hessenberg matrix.

Given a matrix $A = (a_{ij}) \in M_n(\mathbb{C})$, we consider the construction of a unitary matrix P such that PAP^H is upper Hessenberg. Take $x_1 = Ae_1$, the first column of A. Let $y_1 = (h_{i1}) \in \mathbb{C}^n, i = 1, \ldots, n$ where $h_{11} = a_{11},\ h_{21} = \beta_1\sqrt{|a_{21}|^2 + \cdots + |a_{n1}|^2}$, and $h_{31} = h_{41} = \cdots = h_{n1} = 0$, and where β_1 is chosen such that $\beta_1 = -a_{21}/|a_{21}|$ if $a_{21} \neq 0$ and $\beta_1 = 1$ if $a_{21} = 0$. By proposition 2.3, we can construct a unitary matrix P_1 such that $P_1x_1 = y_1$. Further since P_1 acts as an identity on the first row of x_1, it follows that P_1 has the form

$$P_1 = \begin{pmatrix} 1 & O \\ O & \hat{P}_1 \end{pmatrix} \tag{2.2.3}$$

where $\hat{P}_1$ is an $(n-1) \times (n-1)$ Householder matrix that satisfies

$$\hat{P}_1 \begin{pmatrix} a_{21} \\ \vdots \\ a_{n1} \end{pmatrix} = \begin{pmatrix} h_{21} \\ \vdots \\ h_{n1} \end{pmatrix}.$$

Thus the transformation $A \mapsto P_1A$ maps the first column of A to the first column of an upper Hessenberg matrix. The form of P_1 in (2.2.3) implies that the transformation $P_1A \mapsto P_1AP_1^H$ leaves the first column of P_1A unchanged, and hence the unitary similarity transformation $A \mapsto P_1AP_1^H$ maps the first column of A to upper Hessenberg form.

Similarly, if $x_2 = (x_{i2}) = P_1AP_1^He_2$, we let $y_2 = (h_{i2}) \in \mathbb{C}^n$ where $h_{12} = x_{12}, h_{22} = x_{22},\ h_{32} = \beta_2\sqrt{|x_{32}|^2 + \cdots + |x_{n2}|^2}$ and $h_{k2} = 0$ for $k = 4, \ldots, n$, and where β_2 is defined as $\beta_2 = -x_{32}/|x_{32}|$ if $x_{32} \neq 0$ and $\beta_2 = 1$ if $x_{32} = 0$. By proposition 2.3, we construct a unitary matrix P_2 having the form

$$P_2 = \begin{pmatrix} I_2 & O \\ O & \hat{P}_2 \end{pmatrix} \Bigg($$

such that the unitary similarity transformation $P_1AP_1^H \mapsto P_2(P_1AP_1^H)P_2^H$ maps the second column of $P_1AP_1^H$ to upper Hessenberg form, and leaves the first column unchanged.

Repeating this process, we construct unitary matrices $P_3, P_4, \ldots, P_{n-1}$ such that if $\hat{A}_{i-1} = P_{i-1} \cdots P_1AP_1^H \cdots P_{i-1}^H$, then the unitary similarity transformation $\hat{A}_{i-1} \mapsto P_i\hat{A}_{i-1}P_i^H$ maps column i of $\hat{A}_{i-1}$ to upper Hessenberg form. The final matrix

$$H = P_{n-1} \cdots P_1AP_1^H \cdots P_{n-1}^H$$

is an upper Hessenberg matrix which is unitarily similar to A. This reduction of A to upper Hessenberg form can be implemented using $\frac{10}{3}n^3 + O(n^2)$ flops [41].

2.3 Schur Parametrization of Unitary Matrices

Let $\hat{U} \in M_n(\mathbb{C})$ be unitary. We reduce $\hat{U}$ to upper Hessenberg form $\check{U}$ using the reduction algorithm described in the preceding section. Assume further that $\check{U} = (\check{u}_{ij})$ is unreduced, so that $\check{u}_{i+1,i} \neq 0$ for $i = 1, \ldots, n-1$. Performing a diagonal unitary similarity transformation, we can make the subdiagonals real positive to obtain a unitary matrix U similar to $\hat{U}$. We now describe an algorithm that expresses U as a product of matrices of a very simple form [25]:

$$U = G_1G_2 \cdots G_{n-1}G_n, \tag{2.3.4}$$

where $G_k = \operatorname{diag}\{I_{k-1}, \tilde{G}_k, I_{n-k-2}\}$,

$$\tilde{G}_k = \begin{pmatrix} \gamma_k & \sigma_k \\ \sigma_k & -\overline{\gamma}_k \end{pmatrix}, \quad \sigma_k > 0, \quad |\gamma_k|^2 + \sigma_k^2 = 1,$$

for $k = 1, \ldots, n-1$, and $G_n = \operatorname{diag}\{I_{n-1}, \gamma_n\}$ with $|\gamma_n| = 1$.

The matrices G_k are unitary and we will refer to them as *Givens reflectors*. We will refer to the numbers $\gamma_1, \ldots, \gamma_n, \sigma_1, \ldots, \sigma_{n-1}$ collectively as *Schur parameters*. For distinction, we will call $\gamma_1, \ldots, \gamma_n$ as the *Schur parameters* and $\sigma_1, \ldots, \sigma_{n-1}$ as the *complementary Schur parameters* of U.

Let

$$U = \begin{pmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1,n-1} & u_{n,1} \\ u_{21} & u_{22} & u_{23} & \cdots & u_{2,n-1} & u_{2,n} \\ 0 & u_{32} & u_{33} & \cdots & u_{3,n-1} & u_{3,n} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & u_{n,n-1} & u_{n,n} \end{pmatrix}$$

where $u_{i+1,i} > 0$ for $i = 1, \ldots, n-1$. Since the first column of U is a unit vector, we have $|u_{11}|^2 + u_{21}^2 = 1$. We define $\gamma_1 := u_{11}$, $\sigma_1 := u_{21}$ and

$$\tilde{G}_1 := \begin{pmatrix} \gamma_1 & \sigma_1 \\ \sigma_1 & -\overline{\gamma}_1 \end{pmatrix}$$

so that if $G_1 = \operatorname{diag}\{\tilde{G}_1, I_{n-2}\}$, then

$$G_1^{-1}U = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & u_{22}^{(1)} & u_{23}^{(1)} & \cdots & u_{2,n-1}^{(1)} & u_{2,n}^{(1)} \\ 0 & u_{32} & u_{33} & \cdots & u_{3,n-1} & u_{3,n} \\ 0 & 0 & u_{43} & \cdots & u_{4,n-1} & u_{4,n} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & u_{n,n-1} & u_{n,n} \end{pmatrix}$$

Similarly, the matrix $G_1^{-1}U$ is unitary, hence $\left|u_{22}^{(1)}\right|^2 + u_{32}^2 = 1$. Define $\gamma_2 := u_{22}^{(1)}$, $\sigma_2 :=$

u_{32} and

$$\tilde{G}_2 := \begin{pmatrix} \gamma_2 & \sigma_2 \\ \sigma_2 & -\overline{\gamma}_2 \end{pmatrix}$$

so that if $G_2 = \operatorname{diag}\{I_1, \tilde{G}_2, I_{n-3}\}$, then

$$G_2^{-1}G_1^{-1}U = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & u_{33}^{(2)} & \cdots & u_{3,n-1}^{(2)} & u_{3,n}^{(2)} \\ 0 & 0 & u_{43} & \cdots & u_{4,n-1} & u_{4,n} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & u_{n,n-1} & u_{n,n} \end{pmatrix}$$

We repeat this process and continue the construction of the matrices $G_3, \ldots, G_{n-1}$ that satisfy

$$G_{n-1}^{-1} \cdots G_2^{-1}G_1^{-1}U = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & u_{n,n}^{(n-1)} \end{pmatrix}$$

Note that $u_{n,n}^{(n-1)} = 1$, hence if $G_n = \operatorname{diag}\{1, 1, \ldots, 1, u_{n,n}^{(n-1)}\}$, then

$$G_n^{-1}G_{n-1}^{-1} \cdots G_2^{-1}G_1^{-1}U = I.$$

Thus we get the factorization

$$U = G_1G_2 \cdots G_n.$$

The construction of each $\tilde{G}_k$ gives us one algorithm to carry out the factorzation (2.3.4).

Alternatively, we can assume that each G_k has the form $G_k = \operatorname{diag}\{I_{k-1}, \tilde{G}_k, I_{n-k-2}\}$, where

$$\tilde{G}_k = \begin{pmatrix} \gamma_k & \sigma_k \\ \sigma_k & -\overline{\gamma}_k \end{pmatrix} \quad \sigma_k > 0, \quad |\gamma_k|^2 + \sigma_k^2 = 1,$$

for $k = 1, \ldots, n-1$, and $G_n = \operatorname{diag}\{I_{n-1}, \gamma_n\}$ with $|\gamma_n| = 1$. Multiplying explicitly the factors $G_k,\ k = 1, \ldots, n$, we have

$$U = G_1 \cdots G_n = \begin{pmatrix} \gamma_1 & \sigma_1\gamma_2 & \sigma_1\sigma_2\gamma_3 & \cdots & \sigma_1\cdots\sigma_{n-1}\gamma_n \\ \sigma_1 & -\overline{\gamma}_1\gamma_2 & -\overline{\gamma}_1\sigma_2\gamma_3 & \cdots & -\overline{\gamma}_1\sigma_2\cdots\sigma_{n-1}\gamma_n \\ & \sigma_2 & -\overline{\gamma}_2\gamma_3 & \cdots & -\overline{\gamma}_2\sigma_3\cdots\sigma_{n-1}\gamma_n \\ & & \sigma_3 & \cdots & \vdots \\ & & & \ddots & \vdots \\ & & & & -\overline{\gamma}_{n-1}\gamma_n \end{pmatrix}$$

We see that the entries of U and its Schur parametrization are completely determined by its diagonal and subdiagonal entries. With this observation, we have a second procedure to carry out the Schur parametrization of an unreduced upper Hessenberg unitary matrix with positive subdiagonal entries.

2.4 The Multi-Shift QR Algorithm

The algorithm that we will present in the next section is an efficient implementation of the multi-shift QR algorithm [11, 43]. We shall refer to that algorithm as the *unitary QR algorithm.* We will first begin with a brief review of how the multi-shift QR algorithm is implemented implicitly.

Given a matrix $A \in M_n(\mathbb{C})$ in unreduced upper Hessenberg form and shifts $\mu_i \in \mathbb{C}$ for $i = 1, 2, \ldots, m$, a multi-shift QR iteration of degree m carries out the steps

$$(A - \mu_i I) = \check{Q}_i\check{R}_i$$

$$\check{A}_i := \check{R}_i\check{Q}_i + \mu_i I$$

for $i = 1, 2, \ldots, m$ implicitly. The final matrix $\hat{A} := \check{A}_m$ is produced directly from A

and is unitarily similar to A by

$$\hat{A} = Q^H A Q \tag{2.4.5}$$

where $Q = \check{Q}_1 \check{Q}_2 \cdots \check{Q}_m$. It can be shown [11, 43] that Q is also the unitary factor in the unitary-upper triangular decomposition

$$(A - \mu_m I)(A - \mu_{m-1} I) \cdots (A - \mu_1 I) = QR.$$

The transformation (2.4.5) from A to $\hat{A}$ is carried out implicitly as follows:

1. Construct a unitary matrix $V \in M_n(\mathbb{C})$ that satisfies

   $$V\mathbf{e}_1 = \frac{1}{\alpha}(A - \mu_m I)(A - \mu_{m-1} I) \cdots (A - \mu_1 I)\mathbf{e}_1$$

   where $\alpha = \| (A - \mu_m I)(A - \mu_{m-1} I) \cdots (A - \mu_1 I)\mathbf{e}_1 \|_2$.

2. Reduce the matrix $V^H A V$ to upper Hessenberg form.

Since A is upper Hessenberg, the unitary matrix V has the block diagonal form $V = \text{diag}\{\tilde{V}_1, I_{n-m-1}\}$ where $\tilde{V}_1 \in \mathbb{C}^{(m+1)\times(m+1)}$ is unitary. In fact, the matrix V^H maps the vector

$$\mathbf{v} = (A - \mu_m I)(A - \mu_{m-1} I) \cdots (A - \mu_1 I)\mathbf{e}_1$$

to $\mathbf{y} = (\alpha, 0, \cdots, 0)^T \in \mathbb{C}^n$.

Because of the form of V, the transformation $A \mapsto V^H A$ acts only on the first $(m+1)$ rows and the transformation $V^H A \mapsto (V^H A)V$ acts only on the first $(m+1)$

columns. Hence the unitary similarity transformation $A \mapsto \tilde{A} := V^H A V$ introduces an *initial bulge* of size $(m+1) \times (m+1)$ given by the submatrix

$$\begin{pmatrix} \tilde{a}_{2,1} & \cdots & \tilde{a}_{2,m+1} \\ \vdots & & \vdots \\ \tilde{a}_{m+2,1} & \cdots & \tilde{a}_{m+2,m+1} \end{pmatrix} \tag{2.4.6}$$

To return $\tilde{A}$ to upper Hessenberg form, a unitary matrix P_1 is built such that the tranformation $\tilde{A} \mapsto P_1^H \tilde{A}$ acts only on rows $2, \cdots, m+2$ of $\tilde{A}$ to zero out the entries $\tilde{a}_{3,1}, \cdots, \tilde{a}_{m+2,1}$. Matrix P_1 has the block diagonal form $\text{diag}\{I_1, \tilde{P}_1, I_{n-m-2}\}$. The transformation $P_1^H \tilde{A} \mapsto (P_1^H \tilde{A}) P_1$ acts only on the columns $2, \cdots, m+2$, leaving the newly created zeros unaffected, and creates a new row to the bulge. Hence the unitary similarity transformation $\tilde{A} \mapsto P_1^H \tilde{A} P_1$ returns the first column to upper Hessenberg form and moves the bulge one row and one column down. A second unitary matrix P_2 is built so that the tranformation $P_1^H \tilde{A} P_1 \mapsto P_2^H (P_1^H \tilde{A} P_1) P_2$ returns the second column to Hessenberg form and moves the bulge one row and one column down. The process is repeated until the bulge is chased off the bottom of the matrix and $\tilde{A}$ is eventually returned to upper Hessenberg form. In all, unitary matrices $P_1, P_2, \cdots, P_{n-2}$ are created to carry out this reduction. The *kth* unitary matrix has the block diagonal form

$$P_k = \begin{pmatrix} I_k & & \\ & \tilde{P}_k & \\ & & I_{n-m-k-1} \end{pmatrix} \tag{2.4.7}$$

for $k = 1, 2, \ldots, n-m-2$ and

$$P_k = \begin{pmatrix} I_k & \\ & \tilde{P}_k \end{pmatrix} \tag{2.4.8}$$

for $k = n-m-1, \dots, n-2$. It can be shown [11, 43] that the upper Hessenberg matrix that is obtained at the end of this reduction of $\tilde{A}$ is the matrix $\hat{A}$ in (2.4.5). Hence matrices A and $\hat{A}$ are related by

$$\hat{A} = (P_{n-2}^H \cdots P_2^H P_1^H V^H) A (V P_1 P_2 \cdots P_{n-2}).$$

If A is unitary Hessenberg, we make the following modification in the scheme: The matrices $P_1, P_2, \cdots, P_{n-2}$ are constructed such that we get real, positive subdiagonal entries in $\hat{A}$. We require this so that $\hat{A}$ has a factorization of the form (2.3.4). Matrix P_1 for instance can be chosen as the unitary matrix that maps the first column of $\tilde{A}$ to the vector $(\tilde{a}_{11}, \xi, 0, 0, \cdots, 0)^T \in \mathbb{C}^n$ where $\xi = \| (\tilde{a}_{21}, \cdots, \tilde{a}_{m+2,1}) \|_2$.

2.5 Efficient Unitary Multi-Shift QR Iteration

We now show how to implement a multi-shift QR iteration efficiently on a unitary matrix in factored form. Let $U \in M_n(\mathbb{C})$ be a unitary matrix in upper Hessenberg form with $u_{j+1,j} > 0$ for $j = 1, \dots, n-1$. As described in section 2.3, U has a factorization

$$U = G_1 G_2 \cdots G_{n-1} G_n, \tag{2.5.9}$$

where

$$G_k = \begin{pmatrix} I_{k-1} & & \\ & \tilde{G}_k & \\ & & I_{n-k-2} \end{pmatrix} \tag{2.5.10}$$

with

$$\tilde{G}_k = \begin{pmatrix} \gamma_k & \sigma_k \\ \sigma_k & -\overline{\gamma}_k \end{pmatrix}, \quad \sigma_k > 0, \quad |\gamma_k|^2 + \sigma_k^2 = 1,$$

for $k = 1, \ldots, n-1$, and

$$G_n = \begin{pmatrix} I_{n-1} & \\ & \gamma_n \end{pmatrix}\Bigg(\tag{2.5.11}$$

with $|\gamma_n| = 1$. The implementation of a QR iteration which we shall describe will produce a new unitary matrix $\hat{U}$ in factored form:

$$\hat{U} = \hat{G}_1\hat{G}_2\cdots\hat{G}_{n-1}\hat{G}_n. \tag{2.5.12}$$

We define two vectors $\mathbf{g} = (\gamma_1, \cdots, \gamma_n) \in \mathbb{C}^n$ and $\mathbf{s} = (\sigma_1, \cdots, \sigma_{n-1}) \in \mathbb{R}^{n-1}$ to store matrix U. Let $\mu_i \in \mathbb{C}$ for $i = 1, \ldots, m$ be the shifts. The first part of the implicit algorithm is as follows. We construct a matrix $V = \text{diag}\{\tilde{V}_1, I_{n-m-1}\}$ as described in the preceding section. If $\tilde{U} := V^H UV$ then $\tilde{U}$ contains the initial bulge given by the submatrix $\tilde{U}(2 : m+2, 1 : m+1)$. The second part of the algorithm is to return $\tilde{U}$ to upper Hessenberg form $\hat{U}$ by chasing this bulge. The idea behind our implementation is to multiply together the first few of the G_i factors to build a leading submatrix of U that is big enough to accommodate the bulge. We then build the bulge and begin to chase it downward. As we do so, we must multiply in additional G_i factors to accommodate the progressing bulge. However, we also get to factor out matrices $\hat{G}_1$, $\hat{G}_2$, ..., from the top since, as soon as the bulge begins to move downward, we can begin to refactor the top part of the matrix, for which the iteration is complete. At any given point in the algorithm, the part of the matrix that contains the bulge can be stored in a work area of dimension $(m+2) \times (m+2)$. On each forward step we must factor in one new G_i at the bottom of the work area, and we get to factor out a $\hat{G}_j$ at the top. The total storage space needed by our algorithm is thus $O(n+m^2)$.

Let

$$W_1 = \begin{pmatrix} \tilde{G}_1 & \\ & I_m \end{pmatrix} \begin{pmatrix} I_1 & & \\ & \tilde{G}_2 & \\ & & I_{m-1} \end{pmatrix} \cdots \begin{pmatrix} I_m & \\ & \tilde{G}_{m+1} \end{pmatrix}.$$

Thus W_1 is the $(m+2)\times(m+2)$ leading principal submatrix of $G_1G_2\cdots G_{m+1}$. This goes into the work area initially. Note that the submatrix $W_1(:,1:m+1)$ consisting of the first $(m+1)$ columns of W_1 is the submatrix $U(1:m+2,1:m+1)$ of U. It follows that the submatrix $\tilde{U}(1:m+2,1:m+1)$ which contains the initial bulge is the first $m+1$ columns of $W_2 := V_1^H W_1 V_1$ where $V_1 = \text{diag}\{\tilde{V}_1, I_1\}$. In computing W_2, we have thus performed the transformation $U \mapsto V^H U V$ by working only with matrix W_1.

We now chase the bulge. The matrix $P_1 = \text{diag}\{I_1, \tilde{P}_1, I_{n-m-2}\}$ is constructed such that the transformation $\tilde{U} \mapsto P_1^H \tilde{U}$ returns the first column of $\tilde{U}$ to upper Hessenberg form. In terms of the working matrix, we perform the transformation $W_2 \mapsto W_2^{(1)} := \tilde{P}_1^{(1)*} W_2$ where $\tilde{P}^{(1)} = \text{diag}\{I_1, \tilde{P}_1\}$. Further, $\tilde{P}_1$ is constructed so that the entry $W_2^{(1)}(2,1) > 0$. Hence the first column of $W_2^{(1)}$ is $(\hat{\gamma}_1, \hat{\sigma}_1, 0, \cdots, 0)^T$. We can then perform the factorization

$$W_2^{(1)} = \begin{pmatrix} \tilde{G}_1^{(1)} & \\ & I_m \end{pmatrix} \begin{pmatrix} I_1 & \\ & \tilde{W}_2^{(1)} \end{pmatrix}. \tag{2.5.13}$$

where

$$\tilde{G}_1^{(1)} = \begin{pmatrix} \hat{\gamma}_1 & \hat{\sigma}_1 \\ \hat{\sigma}_1 & -\overline{\hat{\gamma}}_1 \end{pmatrix}$$

The matrix $\hat{G}_1 = \text{diag}\{\tilde{G}_1^{(1)}, I_{m-2}\}$ is the first matrix in the factorization (2.5.12) The first entries of the vectors $\mathbf{g}$ and $\mathbf{s}$ are replaced with the new Schur parameters $\hat{\gamma}_1$ and $\hat{\sigma}_1$. From (2.5.13), we see that $\tilde{W}_2^{(1)}$ is the trailing $(m+1) \times (m+1)$ principal submatrix of $\text{diag}\{\tilde{G}_1^{(1)}, I_m\}^H W_2^{(1)}$. We extract $\tilde{W}_2^{(1)}$ and let

$$W_2^{(2)} := \left(\begin{pmatrix} \tilde{W}_2^{(1)} & \\ & I_1 \end{pmatrix} \right..$$

This is our new working matrix. The next factor in (2.5.9) is multiplied in:

$$W_2^{(3)} := W_2^{(2)} \begin{pmatrix} I_m & \\ & \tilde{G}_{m+2} \end{pmatrix} \left(\right.$$

Finally to carry out the transformation $P_1^H \tilde{U} \mapsto \tilde{U}_1 := (P_1^H \tilde{U}) P_1$ we note from (2.4.7) that P_1 commutes with $G_{m+3}, \cdots, G_n$. Thus if

$$W_3 := W_2^{(3)} \begin{pmatrix} \tilde{P}_1 & \\ & I_1 \end{pmatrix}$$

then the first $(m+1)$ columns of W_3 form the submatrix $\tilde{U}_1(3 : m+3, 2 : m+2)$, which contains the new bulge. This completes the transformation $\tilde{U} \mapsto P_1^H \tilde{U} P_1$.

In general, for $k = 2, \ldots, n-m-2$, we have the working matrix W_{k+1} whose first $(m+1)$ columns contain the bulge. The matrix P_k having the form (2.4.7) is built. In this block diagonal form, the unitary matrix $\tilde{P}_k$ is constructed such that the transformation

$$W_{k+1} \mapsto W_{k+1}^{(1)} := \tilde{P}_k^{(1)*} W_{k+1}, \qquad (2.5.14)$$

where $\tilde{P}_k^{(1)} = \text{diag}\{I_1, \tilde{P}_k\}$, returns the first column of W_{k+1} to upper Hessenberg form and makes the entry $W_{k+1}^{(1)}(2,1) > 0$. Next, the factorization

$$W_{k+1}^{(1)} = \begin{pmatrix} \tilde{G}_k^{(1)} & \\ & I_m \end{pmatrix} \begin{pmatrix} I_1 & \\ & \tilde{W}_{k+1}^{(1)} \end{pmatrix} \tag{2.5.15}$$

is performed, where

$$\tilde{G}_k^{(1)} = \begin{pmatrix} \hat{\gamma}_k & \hat{\sigma}_k \\ \hat{\sigma}_k & -\bar{\hat{\gamma}}_k \end{pmatrix}$$

The *kth* entries in the vectors $\mathbf{g}$ and $\mathbf{s}$ are updated with $\hat{\gamma}_k$ and $\hat{\sigma}_k$ respectively. The submatrix $\tilde{W}_{k+1}^{(1)}$ is extracted and the working matrix

$$W_{k+1}^{(2)} := \begin{pmatrix} \tilde{W}_{k+1}^{(1)} & \\ & I_1 \end{pmatrix} \tag{2.5.16}$$

is formed. The next factor in (2.5.9) is multiplied in:

$$W_{k+1}^{(3)} := W_{k+1}^{(2)} \begin{pmatrix} I_m & \\ & \tilde{G}_{m+k+1} \end{pmatrix}$$

and a full working matrix is formed by

$$W_{k+2} := W_{k+1}^{(3)} \begin{pmatrix} \tilde{P}_k & \\ & I_1 \end{pmatrix}. \tag{2.5.17}$$

When $k = n - m - 1$, the working matrix begins to shrink. After the operations (2.5.14) and (2.5.15), there is no need to make the extension indicated by (2.5.16), because $\tilde{G}_n = [\gamma_n]$ is only 1×1, not 2×2. On subsequent steps the working matrix continues to shrink, because there are no more factors to multiply in. By the time the bulge chase is complete, the working matrix has been reduced to 2×2 and can be factored to form

$$\begin{pmatrix} \hat{\gamma}_{n-1} & \hat{\sigma}_{n-1} \\ \hat{\sigma}_{n-1} & -\bar{\hat{\gamma}}_{n-1} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & \hat{\gamma}_n \end{pmatrix}$$

The new Schur parameters $\hat{\gamma}_{n-1}$, $\hat{\sigma}_{n-1}$, and $\hat{\gamma}_n$ replace the old ones in $\mathbf{g}$ and $\mathbf{s}$, and the iteration is complete.

2.5.1 Enforcement of Unitarity

One other important detail needs to be mentioned. Each new pair of Schur parameters $\hat{\gamma}_k$, $\hat{\sigma}_k$ satisfies $|\hat{\gamma}_k|^2 + \hat{\sigma}_k^2 = 1$ in principle, but in practice roundoff errors will cause this equation to be violated by a tiny amount. Therefore the following normalization step is required:

$$\begin{aligned} \nu &\leftarrow \left(|\hat{\gamma}_k|^2 + \hat{\sigma}_k^2\right)^{1/2} \\ \hat{\gamma}_k &\leftarrow \hat{\gamma}_k/\nu \\ \hat{\sigma}_k &\leftarrow \hat{\sigma}_k/\nu \end{aligned}$$

This should be done even when $k = n$, taking $\hat{\sigma}_n = 0$. This enforcement of unitarity is essential to the stability of the algorithm. If it is not done, the matrix will (over the course of many iterations) drift away from being unitary, and the algorithm will fail. A proof of the backward stability of the unitary QR algorithm is found in [17].

2.5.2 Operation Count

The bulk of the arithmetic in our algorithm is contained in the steps (2.5.14) and (2.5.17). Each unitary transformation is taken to be the product of a reflector followed by a diagonal phase-correcting transformation to enforce the condition $\hat{u}_{k+1,k} > 0$. The latter costs $O(m)$ arithmetic; the real work is in applying the reflector. Each of these is at most $(m+1) \times (m+1)$ (smaller at the very end of the iteration), and the cost of applying it efficiently to the working matrix on left or right is about $4m^2$ flops

[41, § 3.2]. Since the reflector is applied only to the small work area and not to the full Hessenberg matrix, the amount of arithmetic is $O(m^2)$ instead of $O(nm)$; this is where we realize our savings. Since $n-1$ reflectors are applied (on left and right) in the whole iteration, the arithmetic cost is about $8nm^2$ flops.

If m is fixed and small, then we can say that the cost of an iteration is $O(n)$, in the sense that the arithmetic is bounded by $C_m n$, where C_m is independent of n. However, the fact that C_m grows like m^2 as m is increased shows that it will be inefficient to take m too large.

There is another important reason for keeping m fairly small. If m is made much bigger than 8 or 10, roundoff errors interfere with the mechanism of shift transmission and render the QR iteration ineffective [39]. This phenomenon is known as *shift blurring.*

2.5.3 Shift Strategies

Eberlein and Huang [21] presented a globally convergent shift strategy for the QR algorithm, and they showed that it converges at least quadratically. Wang and Gragg [37] proposed a family of strategies that includes that of Eberlein and Huang. They demonstrated global convergence and showed that the convergence rate is always at least cubic. These strategies are for single QR iterations, the case $m = 1$.

Since we are taking multiple steps, we need a different strategy. The most common way to obtain m shifts is to take the eigenvalues of the trailing $m \times m$ submatrix of

U. Watkins and Elsner [44] showed that this strategy is cubically convergent when it converges. However, it is not globally convergent, as the following well-known example shows. Let U be the unitary circulant shift matrix, which looks like

$$\begin{pmatrix} & & & 1 \\ 1 & & & \\ & 1 & & \\ & & 1 & \end{pmatrix}$$

in the 4×4 case. For any $m < n$, if we take the eigenvalues of the trailing submatrix as shifts, we get shifts $0, \ldots, 0$, which are equidistant from all of the eigenvalues. A QR iteration on U with these shifts goes nowhere.

Since the eigenvalues of a unitary matrix lie on the unit circle, it make sense to choose shifts that are on the unit circle. We tried two strategies. The first computes the eigenvalues of the trailing $m \times m$ submatrix and normalizes each of them by dividing it by its absolute value. If any of the tentative shifts happens to be zero, it is replaced by a random number on the unit circle. If we use this strategy on the circulant shift matrix, we get m random shifts.

A second strategy stems from the following observation. The last m rows of the unreduced Hessenberg matrix U are orthonormal. Since $u_{n-m+1,n-m} > 0$, the trailing $m \times m$ submatrix $U(n-m+1:n, n-m+1:n)$ is not unitary, but it is nearly unitary. Its rows are orthogonal, and they all have norm 1, except that the top row $U(n-m+1, n-m+1:n)$ has norm less than one. Unitarity can be restored by dividing this row by its norm. In the rare case when the whole top row is zero, a suitable first row can be generated by orthonormalizing a random row against rows 2 through m. The m eigenvalues of the modified matrix then give m shifts on the unit circle.

When this strategy is used on the circulant shift matrix, the orthonormalization process will generate a first row of the form $(0, \ldots, 0, \gamma)$ with $|\gamma| = 1$. The shifts are then the roots of the equation $z^m - \gamma = 0$, which are equally spaced points on the unit circle.

We found that these two strategies work about equally well. Both are cubically convergent. As $u_{n-m+1,n-m} \to 0$, the trailing $m \times m$ submatrix becomes closer and closer to unitary. Its eigenvalues become ever closer to the unit circle, and normalizing them as in the first strategy moves them only slightly. On the other hand, if we modify the matrix as in the second strategy by normalizing its first row, that also moves the eigenvalues only slightly, because the rescaling factor is very close to 1. Thus both strategies behave asymptotically the same as the strategy that simply takes the eigenvalues of the trailing submatrix as shifts; that is, if they converge, then they converge cubically.

We conjecture that both strategies converge globally.

2.5.4 Numerical Results

We implemented the unitary QR algorithm in MATLAB and tried it out on numerous unitary matrices. Test problems with known eigenvalues were generated as follows.

A unitary diagonal matrix D was generated and its eigenvalues noted. A unitary matrix Q, random with respect to Haar measure, was generated, and the random unitary matrix $B = QDQ^H$ formed. Then B was transformed to upper Hessenberg form to yield an upper Hessenberg unitary matrix A with known eigenvalues, which was then factored into the form (2.3.4).

The eigenvalues of unitary matrices are perfectly conditioned, so we always expect to be able to compute them to very high accuracy. We found that our algorithm was able to do this. The results in Table 2.1 are typical. We computed the eigenvalues with our code using $m = 2, 3, \ldots, 10$ and obtained accurate results in all cases. The test matrices included matrices with many repeated eigenvalues and others with tight clusters of eigenvalues. The eigenvalues of smaller matrices are computed with slightly more accuracy than are those of large ones.

For real orthogonal matrices one should always take $m \geq 2$, and the complex shifts should be taken in conjugate pairs. Then the matrix $(A - \mu_m I) \cdots (A - \mu_1 I)$ is real, and all operations can be done in real arithmetic.

Table 2.1: Maximum error in the computed eigenvalues

Size	$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$
200	9.12×10^{-15}	9.56×10^{-15}	6.36×10^{-15}	5.50×10^{-15}	1.01×10^{-14}
400	1.92×10^{-14}	1.72×10^{-14}	1.54×10^{-14}	1.31×10^{-14}	1.13×10^{-14}
600	2.83×10^{-14}	2.17×10^{-14}	2.21×10^{-14}	1.86×10^{-14}	1.63×10^{-14}
800	4.50×10^{-14}	3.00×10^{-14}	2.75×10^{-14}	2.62×10^{-14}	2.45×10^{-14}
1000	4.07×10^{-14}	3.02×10^{-14}	2.56×10^{-14}	2.24×10^{-14}	2.32×10^{-14}

Size	$m = 7$	$m = 8$	$m = 9$	$m = 10$
200	5.42×10^{-15}	4.31×10^{-15}	4.81×10^{-15}	4.00×10^{-15}
400	1.09×10^{-14}	8.53×10^{-15}	6.70×10^{-15}	1.01×10^{-14}
600	1.57×10^{-14}	1.52×10^{-14}	1.48×10^{-14}	1.54×10^{-14}
800	2.59×10^{-14}	2.29×10^{-14}	2.42×10^{-14}	2.55×10^{-14}
1000	2.03×10^{-14}	2.05×10^{-14}	1.51×10^{-14}	1.48×10^{-14}

Chapter 3

Krylov Space Algorithms

In this chapter, we present a family of related algorithms in which a unitary matrix U is approximated by its restriction on a Krylov space defined by

$$\mathcal{K}_m(U,q) = \langle q, Uq, U^2q, ..., U^{m-1}q \rangle$$

where $\| q \|_2 = 1$. One procedure that generates an orthonormal basis for this space is the *Arnoldi process* [10]. The equation that relates the Krylov space $\mathcal{K}_m(U,q)$, the matrix U, and its approximate restriction H_m on $\mathcal{K}_m(U,q)$ is

$$UQ_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^H.$$

The columns of Q_m form an orthonormal basis for $\mathcal{K}_m(U,q)$. Matrix H_m is upper Hessenberg ($h_{ij} = 0$ for $i > j+1$). The unit vector q_{m+1} is orthogonal to the columns of Q_m and $h_{m+1,m}$ is a scalar. If $h_{m+1,m} = 0$, then

$$UQ_m = Q_m H_m.$$

This implies that the space $\mathcal{K}_m(U, q)$ is invariant under U, that H_m represents the restriction of U on $\mathcal{K}_m(U, q)$, and that the spectrum of H_m is a subset of the spectrum of U. If $h_{m+1,m} \neq 0$, then the eigenvalues of H_m are approximations of the eigenvalues of U. We can improve the quality of the approximation on each iteration by choosing the generating vector q appropriately, and we can perform implicit restarts. We collectively refer to these related algorithms as *Krylov space eigenvalue algorithms.*

This chapter is organized as follows: We define Krylov spaces and the Arnoldi process in section 3.1. We also consider two special cases of the Arnoldi process: the isometric Arnoldi process for unitary matrices, and the Lanczos process for Hermitian matrices. The isometric Arnoldi process is used in an inexact Arnoldi-Schur algorithm in section 3.3. The Hermitian Lanczos process is used in a Cayley transformed Lanczos-Schur algorithm in section 3.4. The implicitly restarted Arnoldi method presented in section 3.2 ties up all the algorithms in the chapter to subspace iteration driven by polynomial filters.

3.1 The Arnoldi Process and its Variants

Fundamental to all the algorithms in this chapter is the notion of a Krylov space.

Definition 3.1. Let $A \in M_n(\mathbb{C}),\ q \in \mathbb{C}^n,\ q \neq 0$. Let m be a positive integer. The *mth Krylov space associated with A and q* is the space $\mathcal{K}_m(A, q)$ defined as

$$\mathcal{K}_m(A, q) = \langle q, Aq, \ldots, A^{m-1}q \rangle.$$

We briefly summarize a few basic properties of a Krylov space.

Proposition 3.1. *If $A \in M_n(\mathbb{C}),\ q \in \mathbb{C}^n,\ q \neq 0,$ then:*

1. $\mathcal{K}_m(A, cq) = \mathcal{K}_m(A, q)$ *for any nonzero scalar* c
2. $\mathcal{K}_m(A, q) \subseteq \mathcal{K}_{m+1}(A, q)$
3. $A\mathcal{K}_m(A, q) \subseteq \mathcal{K}_{m+1}(A, q)$
4. $\mathcal{K}_m(A, q) = \{p(A)q \mid p \in \mathcal{P}_{m-1}\}$
5. *If* $\{q, Aq, \ldots, A^{m-1}q\}$ *is a linearly independent set, then* $\mathcal{K}_m(A, q)$ *is invariant under* A *if and only if* $\{q, Aq, \ldots, A^{m-1}q, A^m q\}$ *is a linearly dependent set.*

While $\{q, Aq, \ldots, A^{m-1}q\}$ is a basis for $\mathcal{K}_m(A, q)$ whenever it is a linearly independent set, it is usually an ill-conditioned basis in the sense that the vectors $A^j q$ increasingly point to the direction of the dominant eigenvector of A. To obtain a well-conditioned orthonormal basis for $\mathcal{K}_m(A, q)$, we could apply the Gram-Schmidt process to $q, Aq, \ldots, A^{m-1}q$. An equivalent orthonormalization procedure is the Arnoldi process which we now introduce.

Given $q \in \mathbb{C}^n, q \neq 0$, such that $\{q, Aq, \ldots, A^{m-1}q\}$ is linearly independent, we shall obtain an orthonormal set $\{q_1, q_2, \ldots, q_m\}$ such that

$$\langle q_1, \ldots, q_k\rangle = \langle q, Aq, \ldots, A^{k-1}q\rangle = \mathcal{K}_k(A, q)$$

for $k = 1, 2, \ldots, m$. The first step is a normalization:

$$q_1 = q/\| q \|.$$

Next for $k = 1, 2, \ldots, m-1$, define

$$\acute{q}_{k+1} = Aq_k - \sum_{j=1}^{k} q_j h_{jk} \tag{3.1.1}$$

where $h_{jk} = \langle Aq_k, q_j \rangle$ and define

$$h_{k+1,k} = \| \acute{q}_{k+1} \|_2, \tag{3.1.2}$$

$$q_{k+1} = \acute{q}_{k+1}/h_{k+1,k}. \tag{3.1.3}$$

This is the *Arnoldi process.* It can be shown that in exact arithmetic, this procedure generates exactly the same sequence of vectors as the Gram-Schmidt process applied to $q, Aq, \ldots, A^k q$. To avoid the gradual loss of orthogonality inherent in numerical orthogonalization processes, each new q_{k+1} must be re-orthogonalized against every q_j that has been generated.

The following result [41] links the Arnoldi process with the Krylov space it generates.

Proposition 3.2. *Let $\{q, Aq, \ldots, A^{m-1}q\}$ be a linearly independent set. If $q_1, q_2, \ldots, q_m$ are orthonormal vectors generated by the Arnoldi process (3.1.1),(3.1.2),(3.1.3), then:*

1. $\langle q_1, q_2, \ldots, q_k \rangle = \mathcal{K}_k(A, q)$ *for* $k = 1, 2, \ldots, m$

2. $h_{k+1,k} > 0$ *for* $k = 1, 2, \ldots, m-1$

3. $h_{m+1,m} = 0$ *if and only if* $\{q, Aq, \ldots, A^{m-1}q, A^m q\}$ *is a linearly dependent set, which holds (by Prop. 3.1) if and only if* $\mathcal{K}_m(A, q)$ *is invariant under* A.

The Arnoldi orthonormalization procedure can be recast in matrix form. Equations (3.1.1) and (3.1.2) can be written as

$$Aq_k = \sum_{j=1}^{k+1} q_j h_{jk} \tag{3.1.4}$$

for $k = 1, 2, \ldots, m$. If

$$Q_m := (q_1 \; q_2 \; \cdots q_m) \;\in\; \mathbb{C}^{n \times m}$$

and H_m is the upper Hessenberg matrix

$$H_m := \begin{pmatrix} h_{11} & h_{12} & \cdots & h_{1,m-1} & h_{1,m} \\ h_{21} & h_{22} & \cdots & h_{2,m-1} & h_{2,m} \\ 0 & h_{32} & \cdots & h_{3,m-1} & h_{3,m} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & h_{m,m-1} & h_{m,m} \end{pmatrix} \in \mathbb{C}^{(m+1) \times m}$$

then (3.1.4) can be written as

$$AQ_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^H. \tag{3.1.5}$$

An equation of this form is called an *Arnoldi decomposition of order m.* The distinguishing property of this equation is that Q_m has orthonormal columns and H_m is upper Hessenberg with positive subdiagonals. If this is the case, then the columns of Q_m and the entries in H_m are uniquely determined by the vector q_1.

Proposition 3.3. *Let the columns of*

$$Q_m = (q_1 \ q_2 \ \cdots q_m) \ \in \ \mathbb{C}^{n\times m}$$

be orthonormal vectors, and let $H_m \in \mathbb{C}^{m\times m}$ be upper Hessenberg with $h_{j+1,j} > 0$ for $j = 1, 2, \ldots, m-1$. Let $q_{m+1} \in \mathbb{C}^n$ be a unit vector orthogonal to the columns of Q_m and let $h_{m+1,m} > 0$. If Q_m, H_m, q_{m+1} and $h_{m+1,m}$ satisfy (3.1.5), then $q_1, q_2, \ldots, q_m$ are the vectors produced by the Arnoldi process on the matrix A with starting vector q_1.

If Q_k consists of the the first k columns of Q_m, then (3.1.1) can be written as

$$\acute{q}_{k+1} = (I - Q_k Q_k^H)\ A q_k$$

from which we see that $\acute{q}_{k+1}$ is the orthogonal projection of Aq_k into the space $\mathcal{K}_k(A,q)^{\perp}$. Normalization is carried out by (3.1.2) and (3.1.3).

In equation (3.1.5), the eigenvalues of H_m are called the *Ritz values.* If $h_{m+1,m} = 0$ then H_m represents the restriction of A on $\mathcal{K}_m(A,q)$ and the Ritz values are exact eigenvalues of A. If $h_{m+1,m} \neq 0$, then the Ritz values approximate the eigenvalues of A. In either case, if (μ, x) is an eigenpair of H_m, then $(\mu, Q_m x)$ is an exact eigenpair of the perturbed matrix

$$A - h_{m+1,m} q_{m+1} q_m^H.$$

Hence, the pair $(\mu, Q_m x)$ is an approximate eigenpair of A and the following result gives the corresponding residual norm.

Proposition 3.4. *Let Q_m, H_m, and $h_{m+1,m}$ be generated by the Arnoldi process so that (3.1.5) holds. Let (μ, x) be an eigenpair of H_m. If $v = Q_m x$ then*

$$\| Av - \mu v \| = | h_{m+1,m} | | x_m |$$

where x_m is the mth component of x.

In the preceding proposition, if the eigenvector x of H_m is scaled such that $\| x \|_2 = 1$, then the quantity $\| Av - \mu v \|$ represents the *relative* error of the residual. Otherwise, the quantity represents the *absolute* error of the residual.

3.1.1 The Isometric Arnoldi Process

If the matrix A is unitary, then the Arnoldi process can be expressed as a pair of intertwining short-term recurrence relations that can be used to build the spaces $\mathcal{K}_m(A, q)$ and $\mathcal{K}_m(A^{-1}, q)$. The procedure is known as the *isometric Arnoldi process* of Gragg [24, 26].

To derive the isometric Arnoldi process, we shall assume $U \in M_n(\mathbb{C})$ is unitary. Let $q \in \mathbb{C}^n, q \neq 0$. Suppose we build the spaces $\mathcal{K}_m(U, q)$ and $\mathcal{K}_m(U^{-1}, q)$ simultaneously. Then by (3.1.1) and (3.1.2) we have

$$q_1 := q / \| q \| =: w_1$$

$$h_{j+1,j} \; q_{j+1} = Uq_j - \sum_{i=1}^{j} \langle Uq_j, q_i \rangle q_i \tag{3.1.6}$$

$$k_{j+1,j} \; w_{j+1} = U^{-1} w_j - \sum_{i=1}^{j} \langle U^{-1} w_j, w_i \rangle w_i \tag{3.1.7}$$

for $j = 1, \ldots, m$.

Thus

$$\mathcal{K}_j(U, q) = \langle q, Uq, \ldots, U^{j-1}q\rangle = \langle q_1, \ldots, q_j\rangle \tag{3.1.8}$$

for $j = 1, \ldots, m$ where $q_1, \ldots, q_j$ is an orthonormal basis for $\mathcal{K}_j(U, q)$. Similarly

$$\mathcal{K}_j(U^{-1}, q) = \langle q, U^{-1}q, \ldots, U^{-(j-1)}q\rangle = \langle w_1, \ldots, w_j\rangle \tag{3.1.9}$$

for $j = 1, \ldots, m$ where $w_1, \ldots, w_j$ is an orthonormal basis for $\mathcal{K}_j(U^{-1}, q)$.

Multiplying (3.1.9) by U^{j-1} we get

$$U^{j-1}\langle w_1, \ldots, w_j\rangle = \langle q_1, \ldots, q_j\rangle \tag{3.1.10}$$

from which it also follows that

$$U^{-(j-1)}\langle q_1, \ldots, q_j\rangle = \langle w_1, \ldots, w_j\rangle. \tag{3.1.11}$$

Hence if we generate q_{j+1} from (3.1.6) by orthogonalizing Uq_j against $U^{j-1}w_1, \ldots, U^{j-1}w_j$ instead of $q_1, \ldots, q_j$, we have

$$h_{j+1,j}q_{j+1} = Uq_j - \sum_{i=1}^{j} \Bigg(\langle Uq_j, U^{j-1}w_i\rangle \; U^{j-1}w_i. \tag{3.1.12}$$

Since $U^{j-2}\langle w_1, \ldots, w_{j-1}\rangle = \langle q_1, \ldots, q_{j-1}\rangle \;\perp q_j$, it follows that

$$U^{j-1}\langle w_1, \ldots, w_{j-1}\rangle \;\perp Uq_j.$$

Thus

$$\langle Uq_j, U^{j-1}w_i\rangle = 0$$

for $i = 1, \ldots, j-1$, and (3.1.12) simplifies to

$$\begin{aligned} h_{j+1,j} q_{j+1} &= U q_j - \langle U q_j, U^{j-1} w_j \rangle \, U^{j-1} w_j \\ &= U q_j - \gamma_j U^{j-1} w_j \qquad (3.1.13) \end{aligned}$$

where $\gamma_j := w_j^H (U^{j-1})^H U q_j = w_j^H U^{-(j-2)} q_j$. Similarly, (3.1.7) simplifies to

$$\begin{aligned} k_{j+1,j} w_{j+1} &= U^{-1} w_j - \langle U^{-1} w_j, U^{-(j-1)} q_j \rangle U^{-(j-1)} q_j \\ &= U^{-1} w_j - \delta_j U^{-(j-1)} q_j \qquad (3.1.14) \end{aligned}$$

where $\delta_j := q_j^H U^{j-1} U^{-1} w_j = q_j^H U^{j-2} w_j = \overline{\gamma}_j$. To summarize, equations (3.1.13) and (3.1.14) can be written as

$$h_{j+1,j} q_{j+1} = U q_j - \gamma_j U^{j-1} w_j \qquad (3.1.15)$$

$$k_{j+1,j} w_{j+1} = U^{-1} w_j - \delta_j U^{-(j-1)} q_j \qquad (3.1.16)$$

Further $q_{j+1} \perp \langle q_1, \ldots, q_j \rangle = U^{j-1} \langle w_1, \ldots, w_j \rangle$ which implies that $q_{j+1} \perp U^{j-1} w_j$. Hence by the Pythagorean theorem, equation (3.1.15) yields

$$\| h_{j+1,j} q_{j+1} \|_2^2 + \| \gamma_j U^{j-1} w_j \|_2^2 = \| U q_j \|_2^2 = 1$$

and therefore

$$(h_{j+1,j})^2 + |\gamma_j|^2 = 1.$$

Since $h_{j+1,j}$ is a nonnegative quantity, being the norm of a vector, we define $\sigma_j := h_{j+1,j} = \sqrt{1 - |\gamma_j|^2}$. Similarly $k_{j+1,j} = \sqrt{1 - |\delta_j|^2} = \sigma_j$. If we define $\tilde{q}_j := U^{j-1} w_j$

for $j = 1, \ldots, m$, then multiplying (3.1.16) by U^j we get

$$\sigma_j \tilde{q}_{j+1} = \tilde{q}_j - \delta_j U q_j.$$

Thus we can write (3.1.15) and (3.1.16) as a pair of intertwining short-term recurrences as

$$\begin{aligned} \sigma_j q_{j+1} &= U q_j - \gamma_j \tilde{q}_j && (3.1.17) \\ \sigma_j \tilde{q}_{j+1} &= \tilde{q}_j - \overline{\gamma}_j U q_j && (3.1.18) \end{aligned}$$

for $j = 1, 2, \cdots, m$. We generate q_{j+1} by (3.1.17) but generating $\tilde{q}_{j+1}$ using (3.1.18) can be numerically unstable when σ_j is near zero. Solving for Uq_j in (3.1.17) and substituting into (3.1.18), we get

$$\tilde{q}_{j+1} = \sigma_j \tilde{q}_j - \overline{\gamma}_j q_{j+1}. \qquad (3.1.19)$$

This is the expression that we will use to generate $\tilde{q}_{j+1}$.

Equation (3.1.6) can be written as

$$UQ_j = Q_j H_j + h_{j+1,j} q_{j+1} e_j^H$$

for $j = 1, \ldots, m$, which are Arnoldi decompositions of orders 1 through m. To complete the description of the isometric Arnoldi process, it remains to show that $\gamma_1, \ldots, \gamma_m$ and $\sigma_1, \ldots, \sigma_{m-1}$ are the Schur parameters of H_m.

Comparing (3.1.6) with (3.1.17), we see that

$$\begin{aligned}
\gamma_j \tilde{q}_j &= \sum_{i=1}^{j} \langle U q_j, q_i \rangle q_i \\
&= \sum_{i=1}^{j} h_{ij} q_i \qquad (3.1.20)
\end{aligned}$$

Since $q_1, \ldots, q_j$ are linearly independent, it follows that for each fixed j, the coefficient h_{ij} of q_i in (3.1.20) is uniquely determined by $\tilde{q}_j$.

We shall now find an expression for h_{ij} in terms of $\gamma_1, \ldots, \gamma_m$ and $\sigma_1, \ldots, \sigma_{m-1}$. First we shall prove by induction that

$$\tilde{q}_j = q_1 \prod_{k=1}^{j-1} \sigma_k + \sum_{i=2}^{j} \left(q_i \left(-\overline{\gamma}_{i-1} \prod_{k=1}^{j-1} \sigma_k \right) \right). \qquad (3.1.21)$$

Using the recursion (3.1.19), we verify that

$$\begin{aligned}
\tilde{q}_2 &= \sigma_1 q_1 - \overline{\gamma}_1 q_2 \\
\tilde{q}_3 &= \sigma_2 q_2 - \overline{\gamma}_2 q_3 \\
&= \sigma_2(\sigma_1 q_1 - \overline{\gamma}_1 q_2) - \overline{\gamma}_2 q_3 \\
&= \sigma_1 \sigma_2 q_1 - \overline{\gamma}_1 \sigma_2 q_2 - \overline{\gamma}_2 q_3 \\
\tilde{q}_4 &= \sigma_1 \sigma_2 \sigma_3 q_1 - \overline{\gamma}_1 \sigma_2 \sigma_3 q_2 - \overline{\gamma}_2 \sigma_3 q_3 - \overline{\gamma}_3 q_4
\end{aligned}$$

Assuming now (3.1.21), from (3.1.19) we get

$$\begin{aligned}
\tilde{q}_{j+1} &= \sigma_j \left(q_1 \prod_{k=1}^{j-1} \sigma_k + \sum_{i=2}^{j} q_i \left(-\overline{\gamma}_{i-1} \prod_{k=1}^{j-1} \sigma_k \right) \right) - \overline{\gamma}_j q_{j+1} \\
&= q_1 \prod_{k=1}^{j} \sigma_k + \sum_{i=2}^{j+1} q_i \left(-\overline{\gamma}_{i-1} \prod_{k=1}^{j-1} \sigma_k \right)
\end{aligned}$$

which proves (3.1.21).

Next, we introduce the simplifying notation

$$\tau_{ij} := \prod_{k=i}^{j-1} \sigma_k$$

with $\tau_{jj} := 1$. Equation (3.1.21) becomes

$$\tilde{q}_j = q_1 \tau_{1j} + \sum_{i=2}^{j} q_i \left(-\overline{\gamma}_{i-1} \tau_{ij} \right). \tag{3.1.22}$$

Rewriting (3.1.17) as $Uq_j = \gamma_j \tilde{q}_j + \sigma_j q_{j+1}$, and substituting the result from (3.1.22), we get

$$Uq_j = q_1 \tau_{1j} \gamma_j + \sum_{i=2}^{j} q_j \left(-\overline{\gamma}_{i-1} \tau_{ij} \right) + \sigma_j q_{j+1}.$$

Comparing the coefficients of q_i with those of (3.1.20) we find that

$$h_{ij} = \begin{cases} \tau_{ij}\gamma_j & \text{if} \quad i = 1 \\ -\overline{\gamma}_{i-1}\tau_{ij}\gamma_i & \text{if} \quad 2 \le i \le j \\ \sigma_j & \text{if} \quad i = j+1 \\ 0 & \text{if} \quad i > j+1 \end{cases}$$

Hence

$$H_m = \begin{pmatrix} \gamma_1 & \sigma_1\gamma_2 & \sigma_1\sigma_2\gamma_3 & \cdots & \sigma_1 \cdots \sigma_{m-1}\gamma_m \\ \sigma_1 & -\overline{\gamma}_1\gamma_2 & -\overline{\gamma}_1\sigma_2\gamma_3 & \cdots & -\overline{\gamma}_1\sigma_2 \cdots \sigma_{m-1}\gamma_m \\ & \sigma_2 & -\overline{\gamma}_2\gamma_3 & \cdots & -\overline{\gamma}_2\sigma_3 \cdots \sigma_{m-1}\gamma_m \\ & & \sigma_3 & \cdots & \vdots \\ & & & \ddots & \vdots \\ & & & & -\overline{\gamma}_{m-1}\gamma_m \end{pmatrix}$$

and we conclude that $\gamma_1, \ldots, \gamma_m$ and $\sigma_1, \ldots, \sigma_{m-1}$ are the Schur parameters of H_m.

Proposition 3.5. *Let $U \in M_n(\mathbb{C})$ be unitary such that*

$$UQ_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^H$$

is an Arnoldi decomposition of order m, where $Q_m e_1 = q/\| q \|_2$ for some nonzero vector $q \in \mathbb{C}^n$. If the vectors $q_1, \ldots, q_m$ and the numbers $\gamma_1, \ldots, \gamma_m$, and $\sigma_1, \ldots, \sigma_{m-1}$ are obtained from the isometric Arnoldi process (3.1.17), (3.1.19) with $q_1 = w_1 = q/\| q \|_2$, then $q_1, \ldots, q_m$ are the respective columns of Q_m, and $\gamma_1, \ldots, \gamma_m$, and $\sigma_1, \ldots, \sigma_{m-1}$ are the Schur parameters of H_m.

3.1.2 The Hermitian Lanczos Process

In section 3.4, we shall present a Cayley transformed Lanczos-Schur algorithm for large unitary matrices. Our interest in the Hermitian Lanczos process in this section lies in the fact that the Cayley transform of an unitary matrix is a Hermitian matrix. Hence an algorithm that seeks the dominant eigenvalue of a Hermitian matrix can be used to find the eigenvalues of a unitary matrix which are near a specified target on the unit circle.

If the matrix A in (3.1.5) is Hermitian, then the Arnoldi process also assumes a simpler form. The corresponding process is called the Hermitian Lanczos Process. Historically, it was the Lanczos process that was first used to find eigenvalues of linear and integral equations in 1950. It was Arnoldi who applied the method to non-symmetric matrices in 1951.

Let $A \in M_n(\mathbb{C})$ be Hermitian. Given a starting vector $q \in \mathbb{C}^n$, $q \neq 0$, after performing m steps of the Arnoldi process, we have

$$AQ_m = Q_m T_m + t_{m+1,m} q_{m+1} e_m^H \tag{3.1.23}$$

where Q_m has orthonormal columns, and T_m is upper Hessenberg with positive subdiagonal entries. Pre-multiplying Q_m^H, and using the fact that q_{m+1} is orthogonal to the columns of Q_m, we get

$$Q_m^H A Q_m = T_m.$$

Since A is Hermitian, so is T_m. Hence T_m is tridiagonal:

$$T_m = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & & & \beta_{m-1} \\ & & & \beta_{m-1} & \alpha_m \end{pmatrix}$$

where $\beta_i > 0$ for $i = 1, \ldots, m-1$. The entries of T_m can be stored in $O(m)$ memory locations. The Arnoldi decomposition (3.1.23) simplifies to a three-term recurrence known as the *Hermitian Lanczos Process*:

$$\beta_k q_{k+1} = Aq_k - \alpha_k q_k - \beta_{k-1} q_{k-1}$$

where $\alpha_k = \langle Aq_k, q_k \rangle$. Since A is Hermitian, the expression for α_k implies that α_k is in fact real, and hence T_m is in fact a real symmetric matrix.

3.2 The Implicitly Restarted Arnoldi Method

In typical applications, only a few eigenvalues of a large matrix are desired. The other eigenvalues are *unwanted* in the sense that they are not of immediate use, and no excessive computational effort need be expended in inadvertently calculating them. In the following sections, we shall describe Krylov space algorithms that seek precisely those eigenvalues that are desired.

The Krylov space eigenvalue algorithms for large matrices generally consist of an expansion phase and a contraction phase. In the initial expansion phase, the Krylov space $\mathcal{K}_m(A, q)$ is built for some staring vector $q \neq 0$. If an orthonormal basis $\{q_1, \ldots, q_m\}$ for $\mathcal{K}_m(A, q)$ is to be used, the Arnoldi process described in the preceding section can be employed to construct this basis. The procedure is expressed by the equation

$$AQ_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^H.$$

The vectors $q_1, \ldots, q_m$ are stored as columns of Q_m, and are kept for two reasons: for re-orthogonalization when constructing the space, and for computing the eigenspaces associated with the desired eigenvalues. The upper Hessenberg matrix H_m approximates A on the space $\mathcal{K}_m(A, q)$. If $h_{m+1,m} \neq 0$, then the Ritz values are used to approximate the eigenvalues of A.

In the contraction phase, the approximating Krylov space $\mathcal{K}_m(A, q)$ is purged of eigenvectors associated with the unwanted eigenvalues. The result is a Krylov space of a smaller dimension k, from which a new starting vector $\check{q}$ is selected. The space is then expanded to dimension m, giving rise to a new set of Ritz values that approximate the eigenvalues of A.

The construction of the new space $\mathcal{K}_m(A, \check{q})$ does not begin from the one-dimensional space $\langle \check{q} \rangle$. Instead the construction begins from the space $\mathcal{K}_k(A, \check{q})$. This k dimensional space is implicitly constructed during the contraction phase, and amounts to what is called an *implicit restart*.

In this section, we shall describe one implicit restart strategy that uses the QR algorithm and is based on a paper by Sorensen [34]. The resulting eigenvalue algorithm is known as *implicitly restarted Arnoldi* (IRA) method. Our interest in IRA lies in the fact that it effects a simultaneous subspace iteration through a polynomial filter, a notion which unifies the theory of the Krylov space algorithms in this chapter.

Suppose we have an Arnoldi decomposition of order m :

$$AQ_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^H \tag{3.2.24}$$

For some fixed k, we choose $j := m - k$ shifts $\mu_1, \ldots, \mu_j$ and use them to perform j steps of the implicit QR algorithm on H_m. From the discussion in section 2.4, this is equivalent to generating a unitary matrix Q such that $\check{H} := Q^H H_m Q$ is upper Hessenberg. Such a matrix Q in fact satisfies

$$g(H_m) = (H_m - \mu_1 I_m) \cdots (H_m - \mu_j I_m) = QR \tag{3.2.25}$$

where R is upper triangular and g is the j degree polynomial

$$g(t) = (t - \mu_1) \cdots (t - \mu_j).$$

Post-multiply Q to (3.2.24) to get

$$A\check{Q}_m = \check{Q}_m \check{H}_m + h_{m+1,m} q_{m+1} e_m^H Q \tag{3.2.26}$$

where $\check{Q}_m := Q_m Q$.

From (3.2.25), we notice that Q is $j-$Hessenberg, hence $e_m^H Q$ has $m-j-1$ leading zeroes. If we drop the last j entries from this vector, we get a vector of the form βe_k^H for some scalar β. Thus extracting the first k columns of (3.2.26) we get

$$A\check{Q}_k = \check{Q}_k \check{H}_k + (\check{h}_{k+1,k} \check{q}_{k+1} + \beta h_{m+1,m} q_{m+1}) e_k^H. \tag{3.2.27}$$

Let $\check{q}_{k+1} := \alpha(\check{h}_{k+1,k} \check{q}_{k+1} + \beta h_{m+1,m} q_{m+1})$ where α is the unique positive scalar chosen so that $\| \check{q}_{k+1} \|_2 = 1$. If $\check{h}_{k+1,k} = 1/\alpha$, then (3.2.27) becomes

$$A\check{Q}_k = \check{Q}_k \check{H}_k + \check{h}_{k+1,k} \check{q}_{k+1} e_k^H$$

which is an Arnoldi decomposition of order k. By Prop.3.3, the columns of $\check{Q}_k$ are exactly the vectors generated by the Arnoldi process using $\check{q}_1 = \check{Q}_k e_k$ as the starting vector. We have thus built the Krylov space $\mathcal{K}_k(A, \check{q}_1)$ by implicitly restarting the Arnoldi process with the vector $\check{q}_1$.

The space $\mathcal{K}_k(A, \check{q}_1)$ can now be expanded to $\mathcal{K}_m(A, \check{q}_1)$. A new $\check{H}_m$ would approximate A over $\mathcal{K}_m(A, \check{q}_1)$, from which a new set of Ritz values would approximate the eigenvalues of A. The quality of the new approximating Ritz values depends on the starting vector $\check{q}_1$. We now consider how to construct the vector $\check{q}_1$ so that $\mathcal{K}_m(A, \check{q}_1)$ would contain the eigenspaces corresponding to the desired eigenvalues.

It can be shown by induction that (3.2.24) implies

$$g(A)Q_m = Q_m g(H_m) + E_j \tag{3.2.28}$$

where the first $k = m - j$ columns of $E_j \in \mathbb{C}^{n\times m}$ are zero. Using the factorization (3.2.25) we have

$$g(A)Q_m = Q_m QR + E_j. \tag{3.2.29}$$

Since R is upper triangular, and $\check{q}_1 = Q_m Q e_1$, the first column of (3.2.29) is

$$g(A)q_1 = \check{q}_1 r_{11}$$

from which we get an expression for the new starting vector $\check{q}_1$ as

$$\check{q}_1 = \frac{1}{r_{11}} g(A) q_1.$$

This expression gives us a rationale for choosing the shifts $\mu_1, \ldots, \mu_j$. The information about the eigenvalues of A are coming from the Ritz values of H_m. Suppose the Ritz values are $\rho_1, \ldots, \rho_m$. If we are seeking k eigenvalues of A and $\rho_{k+1}, \ldots, \rho_m$ are j Ritz values that lies in the part of the spectrum of A that is unwanted, then choosing $\mu_1 = \rho_{k+1}, \ldots, \mu_j = \rho_m$ deemphasizes the components of $\check{q}_1$ along the eigenspace associated with the unwanted part of the spectrum. Such a choice of shifts is called *exact* in the sense that shifts are eigenvalues of H_m. More generally, if we choose the shifts to lie in the part of the spectrum of A that is unwanted, then the implicit restart process deemphasizes those very eigenvalues.

Further, since

$$\mathcal{K}_i(A, \check{q}_1) = g(A)\mathcal{K}_i(A, q_1),$$

we see that

$$\langle \check{q}_1, \ldots, \check{q}_i \rangle = g(A)\langle q_1, \ldots, q_i \rangle \tag{3.2.30}$$

for $i = 1, \ldots, k$ which is a simultaneous subspace iteration on nested subspaces of dimensions 1 through k. We can loosely describe the IRA as a simultaneous subspace iteration driven by the polynomial g.

3.3 An Inexact Arnoldi-Schur Algorithm

In this section, we describe a second strategy for implicitly restarting the Arnoldi process. The strategy has been generalized to arbitrary bases for the Krylov space, not just the orthonormal ones, and has been called the *Krylov-Schur algorithm* [35]. Since we restrict our attention to orthonormal bases generated by the Arnoldi process, we shall refer to the algorithm as an *Arnoldi-Schur algorithm.* Further we shall limit our attention to unitary matrices.

We begin with an overview of the algorithm. Then we discuss the details in the subsections. Let U be an $n \times n$ unitary matrix. Suppose n is large and we are seeking

k eigenvalues of U nearest a target τ where $k \ll n$. Let q be a vector of 2-norm one. We construct an orthonormal basis $q_1, q_2, ..., q_k$ for the Krylov space

$$\mathcal{K}_k(U, q) = \langle q, Uq, U^2q, ..., U^{k-1}q \rangle$$

using the Arnoldi process. In matrix form, we can write this orthonormalization procedure as

$$UQ_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^H \tag{3.3.31}$$

where $q_1, q_2, ..., q_k$ are the columns of Q_k, and H_k is an upper Hessenberg matrix that approximates U on the space $\mathcal{K}_k(U, q)$. Like all implicitly-restarted Krylov space algorithms, the algorithm consists of an expansion phase, where the underlying Krylov space is extended, and a contraction phase, where the unwanted approximations to the eigenvalues of U are purged from the decomposition. We initially build the Krylov space $\mathcal{K}_k(U, q)$ of dimension k, and then expand the space by an additional dimension of j. If $m := k + j$, we then have a Krylov decomposition of order m given by

$$UQ_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^H. \tag{3.3.32}$$

The eigenvalues of H_m are estimates of eigenvalues of U. Since H_m is not unitary, we modify it slightly to make a unitary matrix $\tilde{H}_m$ and use the eigenvalues of $\tilde{H}_m$ as our estimates of eigenvalues of U. In the contraction phase, we trim down the space to dimension k by picking out the k-dimensional Krylov subspace associated with the k eigenvalues of $\tilde{H}_m$ nearest the target τ. The space is again expanded by an additional dimension j from which we get a new set of eigenvalues that approximate those of U. The new Krylov space is $\mathcal{K}_m(U, \hat{q})$ where $\hat{q}$ is a new starting vector chosen so that $\mathcal{K}_m(U, \hat{q})$ is a better approximating space than $\mathcal{K}_m(U, q)$ in a sense that we will describe in Section 3.3.7.

We describe details in the following subsections.

3.3.1 Initial Construction of the Krylov Space

The construction of $\mathcal{K}_k(U, q)$ and its initial expansion to $\mathcal{K}_m(U, q)$ can be combined in one construction. Hence we will begin by considering the Krylov decomposition (3.3.32). We will assume that H_m is unreduced upper Hessenberg ($h_{i,i+1} \neq 0$) and that its subdiagonal entries are real and positive. Thus we can write H_m as a product of reflectors:

$$H_m = G_1 G_2 \cdots G_{m-1} G_m,$$

where $G_r = \mathrm{diag}\{I_{r-1}, \tilde{G}_r, I_{m-r-2}\}$,

$$\tilde{G}_r = \begin{pmatrix} \gamma_r & \sigma_r \\ \sigma_r & -\overline{\gamma}_r \end{pmatrix} \Bigg(\quad \sigma_r > 0, \quad |\gamma_r|^2 + \sigma_r^2 = 1,$$

for $r = 1, \ldots, m-1$, and $G_m = \mathrm{diag}\{I_{m-1}, \gamma_m\}$.

Since U is unitary, we use the isometric Arnoldi process to carry out the orthonormalization procedure. The columns of Q_m and the Schur parameters of H_m

are generated by the short-term recurrence

$$\sigma_j q_{j+1} = U q_j - \gamma_j \tilde{q}_j \tag{3.3.33}$$

$$\tilde{q}_{j+1} = \sigma_j \tilde{q}_j - \overline{\gamma}_j q_{j+1}. \tag{3.3.34}$$

for $j = 1, 2, \cdots, m$ as described in section 3.1. Each new q_j is re-orthogonalized against the columns of Q_{j-1}. The orthonormality of the columns of H_m is enforced by making sure that the Schur parameters of H_m satisfy $|\gamma_k|^2 + \sigma_k^2 = 1$ for $k = 1, 2, \ldots, m-1$. Thus in the initial expansion, given a unit vector q, we generate the matrix Q_m and the Schur parameters of H_m by the following algorithm:

```
$q_1 \leftarrow q$
$\tilde{q}_1 \leftarrow q$
$Q \leftarrow [\,]$
for $j = 1 : m$
    $Q \leftarrow [Q \;\; q_j]$
    $\gamma_j \leftarrow \tilde{q}_j^H U q_j$
    $q_{j+1} \leftarrow U q_j - \gamma_j \tilde{q}_j$
    {re − orthogonalization}
    $d \leftarrow Q^H q_{j+1}$
    $q_{j+1} \leftarrow q_{j+1} - Qd$
    $\sigma_j \leftarrow \| q_{j+1} \|_2$
    {enforce unitarity}
    $\nu \leftarrow \left(|\gamma_j|^2 + \sigma_j^2\right)^{1/2}$
    $\gamma_j \leftarrow \gamma_j/\nu$
    $\sigma_j \leftarrow \sigma_j/\nu$
    $q_{j+1} \leftarrow q_{j+1}/\sigma_j$
    $\tilde{q}_{j+1} \leftarrow \sigma_j \tilde{q}_j - q_{j+1}\overline{\gamma_j}$
    $\tilde{q}_{j+1} \leftarrow \tilde{q}_{j+1}/\| \tilde{q}_{j+1} \|_2$
```
(3.3.35)

3.3.2 Contraction Phase

In the exact Krylov-Schur algorithm [35], we would use the eigenvalues of H_m as approximations to the eigenvalues of U. However, since the eigenvalues of U lie on the unit circle, we would prefer to use approximants which also lie on the unit circle. Now the columns of H_m are orthogonal and the first $m-1$ columns are orthonormal. The matrix H_m fails to be unitary because its last column is not of unit length. We normalize this last column to form a unitary matrix $\tilde{H}_m$ having Schur parameters γ_1, γ_2, $\ldots$, γ_{m-1}, $\gamma_m/|\gamma_m|$ and σ_1, σ_2, $\ldots$, σ_{m-1}. Thus $H_m = \tilde{H}_m + pe_m^H$ for some vector p and we have an inexact Krylov configuration

$$\begin{aligned} UQ_m &= Q_m H_m + h_{m+1,m} q_{m+1} e_m^H \\ &= Q_m(\tilde{H}_m + pe_m^H) + h_{m+1,m} q_{m+1} e_m^H \\ &= Q_m \tilde{H}_m + \tilde{p} e_m^H \qquad (3.3.36) \end{aligned}$$

where $\tilde{p} := Q_m p + h_{m+1,m} q_{m+1}$. The eigenvalues of $\tilde{H}_m$, called *isometric Ritz values* [30], all lie on the unit circle and we will use them to approximate the eigenvalues of U. It is in this sense that we refer to the algorithm as *inexact.*

The unitary matrix $\tilde{H}_m$ has several well-known properties. For instance, the eigenvalues of $\tilde{H}_m$ interlace with those of U [30], and that with respect to the $2-$norm, $\tilde{H}_m$ is the closest unitary matrix to H_m [7].

Since $\tilde{H}_m$ is normal, its Schur decomposition takes the form

$$S^H \tilde{H}_m S = D$$

where D is diagonal and S is unitary. Further we assume that the eigenvalues of $\tilde{H}_m$ along the main diagonal of D are ordered such that the leading k entries of the diagonal of D are the eigenvalues of $\tilde{H}_m$ which are nearest the target τ. The reduction of $\tilde{H}_m$ to D uses the unitary QR algorithm presented in section 2.5. This algorithm manipulate only the Schur parameters of $\tilde{H}_m$ to yield the Schur parameters of D.

The main-diagonal entries of D are the isometric Ritz values, and the first k are the ones we wish to retain. Write

$$D = \operatorname{diag}\{D_{11}, D_{22}\}, \tag{3.3.37}$$

where $D_{11} \in \mathbb{C}^{k \times k}$. Thus D_{11} contains the isometric Ritz values we want to keep. Partition S as $S = \left[\begin{array}{cc} S_1 & S_2 \end{array}\right]$, where $S_1 \in \mathbb{C}^{n \times k}$. Then the columns of S_1 are eigenvectors of $\tilde{H}_m$ corresponding to the isometric Ritz values we wish to retain. In other words, $\mathcal{S} = \mathcal{R}(S_1)$ is the invariant subspace of $\tilde{H}_m$ associated with these values.

Post-multiplying S to (3.3.36) gives

$$U\tilde{Q}_m = \tilde{Q}_m D + \tilde{p} z^H$$

where $\tilde{Q}_m := Q_m S$ and $z^H := e_m^H S$. Extracting the first k columns, we get

$$U\tilde{Q}_k = \tilde{Q}_k D_{11} + \tilde{p}\tilde{z}^H \tag{3.3.38}$$

where $\tilde{Q}_k$ is the first k columns of $\tilde{Q}_m$, and $\tilde{z}$ consists of the first k entries of z. Note that

$$\tilde{Q}_k = Q_m S_1. \tag{3.3.39}$$

In retaining $\tilde{Q}_k$, we are keeping that portion of the space that corresponds to the isometric Ritz values in D_{11}.

Let W be a unitary matrix such that $\tilde{z}^H W = \alpha e_k^H$ where $\alpha := \|\tilde{z}\|_2$. Post-multiplying W, we get

$$U\tilde{Q}_k W = \tilde{Q}_k W F + \alpha \tilde{p} e_k^H$$

where $F := W^H D_{11} W$ is a full matrix. Finally we return F to upper Hessenberg form $\hat{H}_k$ from the bottom up with $V^H F V = \hat{H}_k$ where V is unitary. In doing so, the last term is not affected, and hence we have

$$U\hat{Q}_k = \hat{Q}_k \hat{H}_k + \alpha \tilde{p} e_k^H \tag{3.3.40}$$

where

$$\hat{Q}_k := \tilde{Q}_k W V. \tag{3.3.41}$$

We can carry out the reduction from $W^H D_{11} W$ to $\hat{H}_k$ by an upward bulge chase procedure that manipulates only the Schur parameters of D_{11} in a manner analogous to the one presented in section 2.5. This procedure yields the Schur parameters of $\hat{H}_k$ obtained from the Schur parameters of D_{11} and the information contained in $\tilde{z}^H$. Details are given in 3.3.4.

Thus in (3.3.40), $\hat{H}_k$ is expressed in terms of its Schur parameters. Equation (3.3.40) however is not a Krylov decomposition of order k since $\tilde{p}$ is not necessarily orthogonal to the columns of $\hat{Q}_k$. Noting that the last term of (3.3.40) only affects the last column, we throw away the final column and get

$$\begin{aligned} U\hat{Q}_{k-1} &= \hat{Q}_k \hat{H}_{k,k-1} \\ &= \hat{Q}_{k-1}\hat{H}_{k-1} + \hat{h}_{k,k-1}\hat{q}_k e_{k-1}^H \end{aligned} \tag{3.3.42}$$

which is a Krylov decomposition of order $k-1$. By Prop. 3.3, equation (3.3.42) is precisely the result of an Arnoldi process with $\hat{q} := \hat{Q}_k e_1$ as the starting vector. We have thus built the space $\mathcal{K}_k(U, \hat{q})$ implicitly.

3.3.3 Expansion Phase

All that we need to restart the isometric Arnoldi process are the new vectors $\tilde{q}_{k-1}$ and $\tilde{q}_k$ that would have been generated by the isometric Arnoldi process if $\hat{q}$ had been used as the starting vector. The vectors $\hat{q}_{k-1}$ and $\hat{q}_k$, and the quantities $\hat{\sigma}_{k-1}$, and $\hat{\gamma}_{k-1}$ are known from (3.3.42). We now drop the hats and refer to these quantities as the new q_{k-1}, q_k, σ_k, and γ_k. We can generate the new $\tilde{q}_{k-1}$ using (3.3.33) with j replaced by $k-1$:

$$\tilde{q}_{k-1} = (Uq_{k-1} - \sigma_{k-1}q_k)/\gamma_{k-1}.$$

Next we generate the new $\tilde{q}_k$ using (3.3.34) with $j+1$ replaced by k:

$$\tilde{q}_k = \sigma_{k-1}\tilde{q}_{k-1} - \overline{\gamma}_{k-1}q_k.$$

With the new $\tilde{q}_k$ and the current q_k we can carry out the subspace expansion using the loop in (3.3.35), letting j run from k to m.

3.3.4 Reduction to Hessenberg Form

We now fill in the details of the transformation from (3.3.38) to (3.3.40). The unitary matrix W is completely determined by the vector $\tilde{z}$. With the information contained in the entries of $\tilde{z}^H$, we will obtain the Schur parameters of $\hat{H}_k$ from the Schur parameters of D_{11}, bypassing the explicit construction of the matrix $F = W^H D_{11} W$ and of the unitary matrix V. While there are several variations on how this reduction can be done, we will describe one which immediately returns all intermediate matrices to the upper Hessenberg form.

Let $D_{11} = \mathrm{d}iag(d_1, d_2, \ldots, d_k)$ having Schur parameters $\gamma_1^{(0)}, \gamma_2^{(0)}, \ldots, \gamma_k^{(0)}$ and $\sigma_1^{(0)}, \sigma_2^{(0)}, \ldots, \sigma_{k-1}^{(0)}$. Since D_{11} is diagonal, we have $\sigma_i^{(0)} = 0$ for $i = 1, 2, \ldots, k-1$, and the main diagonal entries are given by $d_i = -\overline{\gamma}_{i-1}^{(0)} \gamma_i^{(0)}$ for $i = 2, 3, \ldots, k$, and $d_1 = \gamma_1^{(0)}$. Let $\tilde{z}^H = (z_1 \ z_2 \ z_3 \cdots z_k)$. Construct a unitary matrix W_1 such that $\tilde{z}^H W_1 = z_{(1)}^H := (0 \ z_2^{(1)} \ z_3 \cdots z_k)$ where $z_2^{(1)} := \sqrt{|\, z_1 \,|^2 + |\, z_2 \,|^2}$. Matrix W_1 takes the form

$$W_1 = \begin{pmatrix} \hat{W}_1 & \\ & I_{k-2} \end{pmatrix}$$

where $\hat{W}_1$ is a 2×2 unitary matrix that maps (z_1, z_2) to $(0, z_2^{(1)})$ by post-multiplication.

Performing a unitary similarity transformation to D_{11}, we have

$$D_{11}^{(1)} := W_1^H D_{11} W_1 = \begin{pmatrix} d_{11}^{(1)} & d_{12}^{(1)} & & & \\ d_{21}^{(1)} & d_{22}^{(1)} & & & \\ & & d_3 & & \\ & & & \ddots & \\ & & & & d_k \end{pmatrix}$$

If the subdiagonal entry $d_{21}^{(1)}$ is not real positive, we do an additional similarity transformation by a diagonal unitary matrix to make it positive. Thus without loss of generality, we can assume $d_{21}^{(1)} > 0$. The Schur parameters of $D_{11}^{(1)}$ are $\gamma_1^{(1)}, \gamma_2^{(1)}, \gamma_3^{(0)}, \ldots, \gamma_k^{(0)}$ and $\sigma_1^{(1)}, 0, 0, \ldots, 0$, where $\gamma_1^{(1)} = d_{11}^{(1)}$, $\gamma_2^{(1)} = -d_{22}^{(1)}/\overline{\gamma}_1$ and $\sigma_1^{(1)} = d_{21}^{(1)}$. Only the first two Schur parameters $\gamma_1^{(1)}, \gamma_2^{(1)}$ differ from those of D_{11}, and only the first complementary Schur parameter $\sigma_1^{(1)}$ differ from that of D_{11}.

Next we construct a unitary matrix W_2 such that

$$z_{(1)}^H W_2 = z_{(2)}^H := (0 \;\; 0 \;\; z_3^{(2)} \;\; z_4 \cdots z_k),$$

where $z_3^{(2)} := \sqrt{|z_2^{(1)}|^2 + |z_3|^2}$. Matrix W_2 takes the form

$$W_2 = \begin{pmatrix} I_1 & & \\ & \hat{W}_2 & \\ & & I_{k-3} \end{pmatrix}$$

where $\hat{W}_2$ is a 2×2 unitary matrix that maps $(z_2^{(1)}, z_3)$ to $(0, z_3^{(2)})$. The transformation $D_{11}^{(1)} \mapsto D_{11}^{(1)} W_2$ affects only the second and third columns of $D_{11}^{(1)}$, creating possibly nonzero entries in the $(1,3)$ and $(2,3)$ positions, and the transformation $D_{11}^{(1)} W_2 \mapsto W_2^H D_{11}^{(1)} W_2$ affects only the second and third rows of $D_{11}^{(1)} W_2$. We thus have a unitary similarity transformation

$$\hat{D}_{11}^{(2)} := W_2^H D_{11}^{(1)} W_2 = \begin{pmatrix} \hat{d}_{11}^{(2)} & \hat{d}_{12}^{(2)} & \hat{d}_{13}^{(2)} & & & \\ \hat{d}_{21}^{(2)} & \hat{d}_{22}^{(2)} & \hat{d}_{23}^{(2)} & & & \\ \hat{d}_{31}^{(2)} & \hat{d}_{32}^{(2)} & \hat{d}_{33}^{(2)} & & & \\ & & & d_4 & & \\ & & & & \ddots & \\ & & & & & d_k \end{pmatrix}$$

We now return $\hat{D}_{11}^{(2)}$ to upper Hessenberg form. We construct a unitary matrix $V_1^{(2)}$ such that the transformation $\hat{D}_{11}^{(2)} \mapsto \hat{D}_{11}^{(2)} V_1^{(2)}$ zeroes out the $(3,1)$ entry. Hence the matrix

$$D_{11}^{(2)} := (V_1^{(2)})^H \hat{D}_{11}^{(2)} V_1^{(2)} = \begin{pmatrix} d_{11}^{(2)} & d_{12}^{(2)} & d_{13}^{(2)} & & & \\ d_{21}^{(2)} & d_{22}^{(2)} & d_{23}^{(2)} & & & \\ & d_{32}^{(2)} & d_{33}^{(2)} & & & \\ & & & d_4 & & \\ & & & & \ddots & \\ & & & & & d_k \end{pmatrix}$$

is in upper Hessenberg form, and by an additional unitary similarity transformation, we can assume that the subdiagonals $d_{21}^{(2)}$ and $d_{32}^{(2)}$ are positive. We thus obtain the Schur parameters of $D_{11}^{(2)}$ as $\gamma_1^{(2)}, \gamma_2^{(2)}, \gamma_3^{(2)}, \gamma_4^{(0)} \ldots, \gamma_k^{(0)}$ and $\sigma_1^{(2)}, \sigma_2^{(2)}, 0, \ldots, 0$. In particular $\gamma_1^{(2)} = d_{11}^{(2)}$,$\gamma_2^{(2)} = -d_{22}^{(2)}/\overline{\gamma}_1^{(2)}$,$\gamma_3^{(2)} = -d_{33}^{(2)}/\overline{\gamma}_2^{(2)}$, and $\sigma_1^{(2)} = d_{21}^{(2)}$,$\sigma_2^{(2)} = d_{32}^{(2)}$.

In general, for $i = 2, 3, \ldots, k-1$, given the vector $z_{(i-1)}^H = (0 \ \ldots 0 \ z_i^{(i-1)} \ z_{i+1} \ldots z_k)$ where $z_i^{(i-1)} := \sqrt{|\,z_1\,|^2 + \cdots + |\,z_i\,|^2}$, we construct a unitary matrix W_i that maps, by post-multiplication, the vector $z_{(i-1)}^H$ to $z_{(i)}^H := (0 \ \ldots 0 \ z_{i+1}^{(i)} \ z_{i+2} \ldots z_k)$ where $z_{i+1}^{(i)} := \sqrt{|\,z_1\,|^2 + \cdots + |\,z_{i+1}\,|^2}$. The matrix W_i takes the form

$$W_i = \begin{pmatrix} I_{i-1} & & \\ & \hat{W}_i & \\ & & I_{k-i-1} \end{pmatrix} \qquad (3.3.43)$$

where $\hat{W}_i$ is a 2×2 unitary matrix that maps $(z_i^{(i-1)}, z_{i+1})$ to $(0, z_{i+1}^{(1)})$ by post multiplication. The upper Hessenberg matrix $D_{11}^{(i-1)}$ has the form

$$D_{11}^{(i-1)} = \begin{pmatrix} d_{11}^{(i-1)} & d_{12}^{(i-1)} & \cdots & d_{1,i-1}^{(i-1)} & d_{1,i}^{(i-1)} & & & \\ d_{21}^{(i-1)} & d_{22}^{(i-1)} & \cdots & d_{2,i-1}^{(i-1)} & d_{2,i}^{(i-1)} & & & \\ & d_{32}^{(i-1)} & \cdots & d_{3,i-1}^{(i-1)} & d_{3,i}^{(i-1)} & & & \\ & & \ddots & \vdots & \vdots & & & \\ & & & d_{i,i-1}^{(i-1)} & d_{i,i}^{(i-1)} & & & \\ & & & & & d_{i+1} & & \\ & & & & & & \ddots & \\ & & & & & & & d_k \end{pmatrix}$$

with Schur parameters $\gamma_1^{(i-1)}, \ldots, \gamma_i^{(i-1)}, \gamma_{i+1}^{(0)}, \ldots, \gamma_k^{(0)}$, and $\sigma_1^{(i-1)}, \ldots, \sigma_{i-1}^{(i-1)}, 0, \ldots, 0$. The transformation $D_{11}^{(i-1)} \mapsto D_{11}^{(i-1)} W_i$ affects only columns i and $i+1$ of $D_{11}^{(i-1)}$, and the transformation $D_{11}^{(i-1)} W_i \mapsto W_i^H D_{11}^{(i-1)} W_i$ affects only rows i and $i+1$. Hence the unitary similarity transformation $D_{11}^{(i-1)} \mapsto W_i^H D_{11}^{(i-1)} W_i =: \hat{D}_{11}^{(i-1)}$ introduces a bulge in the Hessenberg structure of $D_{11}^{(i-1)}$. Specifically, the bulge is the 3×3 submatrix $\hat{D}_{11}^{(i-1)}(i-1 : i+1, i-1 : i+1)$. We chase this bulge upward by a series of unitary similarity transformations that act on two consecutive columns and two consecutive rows at a time. Specifically, the unitary matrix

$$V_1^{(i)} = \begin{pmatrix} I_{i-2} & & \\ & \hat{V}_1^{(i)} & \\ & & I_{k-i} \end{pmatrix} \quad \hat{V}_1^{(i)} \in M_2 \tag{3.3.44}$$

is constructed such that the map $\hat{D}_{11}^{(i)} \mapsto \hat{D}_{11}^{(i)} V_1^{(i)}$ acts only on columns $i-1$ and i of $\hat{D}_{11}^{(i)}$ and zeroes out the $(i+1, i-1)$ entry. The map $\hat{D}_{11}^{(i)} V_1^{(i)} \mapsto (V_1^{(i)})^H \hat{D}_{11}^{(i)} V_1^{(i)}$ acts only on rows $i-1$ and i, and moves the bulge one row left and one column up. The form of $V_1^{(i)}$ in (3.3.44) implies that $z_{(i)}^H V_1^{(i)} = z_{(i)}^H$. A second unitary matrix

$$V_2^{(i)} = \begin{pmatrix} I_{i-3} & & \\ & \hat{V}_2^{(i)} & \\ & & I_{k-i+1} \end{pmatrix} \quad \hat{V}_2^{(i)} \in M_2$$

is constructed such that the similarity transformation

$$(V_1^{(i)})^H \hat{D}_{11}^{(i)} V_1^{(i)} \mapsto (V_2^{(i)})^H (V_1^{(i)})^H \hat{D}_{11}^{(i)} V_1^{(i)} V_2^{(i)}$$

chases the bulge one row and one column up. In general, the unitary matrix

$$V_j^{(i)} = \begin{pmatrix} I_{i-j-1} & & \\ & \hat{V}_j^{(i)} & \\ & & I_{k-i+j-2} \end{pmatrix} \quad \hat{V}_j^{(i)} \in M_2$$

is constructed to chase the bulge one row and one column up, and each $V_j^{(i)}$ acts as an identity on $z_{(i)}^H$ by post-multiplication. After a series of such unitary similarity transformations, we would have returned $\hat{D}_{11}^{(i)}$ to upper Hessenberg form $D_{11}^{(i)}$.

Each of the unitary matrices constructed must also be applied by post-multiplication to the matrix $\tilde{Q}_k$. Since each of these unitary matrices only affects two consecutive columns at a time, the implementation can be done in a way that we only manipulate two columns of $\tilde{Q}_k$ at a time. The resulting matrix in the end is $\hat{Q}_k = \tilde{Q}_k W V$.

3.3.5 Implicit Upward Bulge Chase

The reduction described in the preceding section can be done without forming matrix D_{11} explicitly. Only the Schur parameters of D_{11} are manipulated. The idea is to build from the Schur parameters the submatrix that contains the bulge. As we chase the bulge upward, we multiply in additional Givens reflectors which contain the old Schur parameters, and factor out Given reflectors that contain updated Schur parameters. Since the unitary matrices that we construct to carry out the Hessenberg reduction only affect two consecutive columns and rows at a time, the part of the matrix that are being manipulated can be stored in a working area of size 3×3. This technique is identical to the one presented in section 2.5 except that we will chase the bulge upward instead of downward.

To illustrate the technique, we consider the matrix

$$D_{11}^{(i-1)} = \begin{pmatrix} d_{11}^{(i-1)} & d_{12}^{(i-1)} & \cdots & d_{1,i-1}^{(i-1)} & d_{1,i}^{(i-1)} & & & \\ d_{21}^{(i-1)} & d_{22}^{(i-1)} & \cdots & d_{2,i-1}^{(i-1)} & d_{2,i}^{(i-1)} & & & \\ & d_{32}^{(i-1)} & \cdots & d_{3,i-1}^{(i-1)} & d_{3,i}^{(i-1)} & & & \\ & & \ddots & \vdots & \vdots & & & \\ & & & d_{i,i-1}^{(i-1)} & d_{i,i}^{(i-1)} & & & \\ & & & & & d_{i+1} & & \\ & & & & & & \ddots & \\ & & & & & & & d_k \end{pmatrix}$$

having Schur parameters $\gamma_1^{(i-1)}, \ldots, \gamma_i^{(i-1)}, \gamma_{i+1}^{(0)}, \ldots, \gamma_k^{(0)}$ stored in a vector $\mathbf{g}$, and complementary Schur parameters $\sigma_1^{(i-1)}, \ldots, \sigma_{i-1}^{(i-1)}, 0, \ldots, 0$ stored in vector $\mathbf{s}$. The last $k-i$ Schur parameters in $\mathbf{g}$, and the last $k-i-1$ complementary Schur parameters in $\mathbf{s}$ are those of D_{11}. Since W_i in (3.3.43) only affects columns i and $i+1$, we build the initial working matrix

$$\begin{aligned} B_i &= \begin{pmatrix} \gamma_{i-1}^{(i-1)} & \sigma_{i-1}^{(i-1)} & \\ \sigma_{i-1}^{(i-1)} & -\overline{\gamma}_{i-1}^{(i-1)} & \\ & & 1 \end{pmatrix} \begin{pmatrix} 1 & & \\ & \gamma_i^{(i-1)} & 0 \\ & 0 & -\overline{\gamma}_i^{(i-1)} \end{pmatrix} \begin{pmatrix} 1 & & \\ & 1 & \\ & & \gamma_{i+1}^{(0)} \end{pmatrix} \\ &= \begin{pmatrix} \gamma_{i-1}^{(i-1)} & \sigma_{i-1}^{(i-1)} \gamma_i^{(i-1)} & 0 \\ \sigma_{i-1}^{(i-1)} & -\overline{\gamma}_{i-1}^{(i-1)} \gamma_i^{(i-1)} & 0 \\ & & -\overline{\gamma}_i^{(i-1)} \gamma_{i+1}^{(0)} \end{pmatrix} \end{aligned}$$

The bulge introduced in the unitary similarity $D_{11}^{(i-1)} \mapsto W_i^H D_{11}^{(i-1)} W_i$ is contained in the working area by the transformation

$$B_i^{(1)} := \begin{pmatrix} 1 & \\ & \hat{W}_i^H \end{pmatrix} B_i \begin{pmatrix} 1 & \\ & \hat{W}_i \end{pmatrix}$$

The transformation

$$\hat{D}_{11}^{(i)} \mapsto \hat{D}_{11}^{(i)} V_1^{(i)} \tag{3.3.45}$$

that zeroes out the $(i+1, i-1)$ entry of $\hat{D}_{11}^{(i)}$ is done in the working area by

$$B_i^{(2)} := B_i^{(1)} \begin{pmatrix} \hat{V}_1^{(i)} & \\ & 1 \end{pmatrix}$$

This zeroes out the $(3,1)$ entry of $B_i^{(1)}$. Further $\hat{V}_1^{(i)}$ can be constructed so that it leaves the $(3,2)$ entry of $B_i^{(2)}$ positive. Hence we can construct a Givens reflector

$$\tilde{G}_i^{(i)} = \begin{pmatrix} \gamma_i^{(i)} & \sigma_i^{(i)} \\ \sigma_i^{(i)} & -\overline{\gamma}_i^{(i)} \end{pmatrix} \quad \sigma_i^{(i)} > 0, \quad |\gamma_i^{(i)}|^2 + (\sigma_i^{(i)})^2 = 1,$$

such that

$$B_i^{(2)} = \begin{pmatrix} \tilde{B}_i^{(2)} & \\ & 1 \end{pmatrix} \begin{pmatrix} 1 & \\ & \tilde{G}_i^{(i)} \end{pmatrix}$$

The entries $\gamma_i^{(i)}$ and $\sigma_i^{(i)}$ of $\tilde{G}_i^{(i)}$ update the Schur parameters $\gamma_i^{(i-1)}$ and $\sigma_i^{(i-1)} = 0$ respectively. The next Givens reflector is multiplied in:

$$\tilde{B}_i^{(3)} := \begin{pmatrix} \tilde{G}_{i-1}^{(i-1)} & \\ & 1 \end{pmatrix} \begin{pmatrix} 1 & \\ & \tilde{B}_i^{(2)} \end{pmatrix}$$

where

$$\tilde{G}_{i-1}^{(i-1)} = \begin{pmatrix} \gamma_{i-1}^{(i-1)} & \sigma_{i-1}^{(i-1)} \\ \sigma_{i-1}^{(i-1)} & -\overline{\gamma}_{i-1}^{(i-1)} \end{pmatrix}$$

The transformation (3.3.45) is completed to a similarity $\hat{D}_{11}^{(i)} \mapsto (V_1^{(i)})^H \hat{D}_{11}^{(i)} V_1^{(i)}$ by

$$B_i^{(3)} := \begin{pmatrix} 1 & \\ & \hat{V}_1^{(i)} \end{pmatrix}^H \tilde{B}_i^{(3)}.$$

Thus the bulge has been chased one row and one column up. This process is repeated until the bulge has been chased off the top of the matrix, and the all of the old Schur parameters have been updated.

3.3.6 Convergence and Locking

We use the same convergence and locking procedure as the standard Krylov-Schur algorithm uses. Because U is unitary, the procedure is simpler than it is in general. The natural opportunity to check for convergence occurs in the contraction phase at (3.3.38). In the equation

$$U\tilde{Q}_k = \tilde{Q}_k D_{11} + \tilde{p}\tilde{z}^H$$

write $\tilde{z}^H = (z_1 \ z_2 \ \cdots \ z_k)$, as before. If any of the entries z_j is zero, then the jth column of $\tilde{Q}_m^{(k)}$ is an eigenvector of U with eigenvalue d_j. In practice we count z_j as zero whenever $|\, z_j \,| < \epsilon$ for some specified tolerance ϵ. Any eigenvectors so detected can be permuted to the front of the decomposition. Thus, if i eigenpairs have been detected, then after the permutation, the new permuted version of $\tilde{z}^H$ will have the form $(0 \ \cdots \ 0 \ z_{i+1} \ \cdots \ z_k)$. The first i columns of $\tilde{Q}_m^{(k)}$ will then be eigenvectors. These can remain locked in place, that is, left untouched, from now on. They do not participate in the subsequent reduction to Hessenberg form. The Hessenberg matrix

$\hat{H}_k$ in (3.3.40) has the form $\hat{H}_k = \text{diag}\{d_1, \ldots, d_i, H\}$, where H is an unreduced upper Hessenberg matrix of dimension $k - i$. On subsequent contraction phases they will remain unchanged, because in the reduction of H_m to diagonal form, the top part of the matrix is already diagonal. Once the desired number of eigenpairs has been locked in, the algorithm halts.

3.3.7 Inexact Arnoldi-Schur as Subspace Iteration

As seen in (3.2.30), the effectiveness of the implicitly restarted Arnoldi process can be attributed to the fact that each restart cycle effects nested subspace iterations driven by $g(U)$, where g is a filter polynomial that amplifies eigenvectors associated with part of the spectrum while suppressing unwanted eigenvectors. The Krylov-Schur algorithm [35] is justified by showing that it is equivalent to the implicitly restarted Arnoldi process. In this section we show that the inexact Arnoldi-Schur method also does nested subspace iterations driven by a filter polynomial, even though it is "inexact". We proceed by two stages, proving first a basic result (Proposition 3.6), then a refined result (Proposition 3.8).

Before the contraction phase of our process we have a matrix Q_m with m orthonormal columns. Let Q_k denote the submatrix of Q_m consisting of the first k columns. After the contraction phase we have a new matrix $\hat{Q}_k \in \mathbb{C}^{n \times k}$ satisfying $\mathcal{R}(\hat{Q}_k) \subseteq \mathcal{R}(Q_m)$, which will be used to start the next expansion step.

Proposition 3.6. *Suppose the eigenvalues of D_{11} are disjoint from those of D_{22}*

in (3.3.37). Let $g(t) = (t - \mu_1)(t - \mu_2) \cdots (t - \mu_j)$*, where* $\mu_1, \mu_2, \ldots, \mu_j$ *are the eigenvalues of* D_{22}*. Then*

$$\mathcal{R}(\hat{Q}_k) = g(U)\mathcal{R}(Q_k).$$

The zeros of g are exactly the isometric Ritz values that we are discarding in the contraction phase. The effect of $g(U)$ is to suppress components corresponding to eigenvalues of U near $\mu_1, \ldots, \mu_j$ and to enhance components associated with eigenvalues away from $\mu_1, \ldots, \mu_j$, including the eigenvalues closest to the target τ. Thus filtering is achieved.

Proof. Let $\tilde{H}_m$ be the unitary matrix defined in Section 3.3.2, and consider $g(\tilde{H}_m)$. Since $\mu_1, \ldots, \mu_j$ are eigenvalues of $\tilde{H}_m$, $g(\tilde{H}_m)$ is highly rank deficient. In fact, $g(\tilde{H}_m) = Sg(D)S^H = S\text{diag}\{g(D_{11}), 0\}S^H$, so the rank of $g(\tilde{H}_m)$ is exactly k, and $\mathcal{R}(g(\tilde{H}_m))$ is exactly the invariant subspace associated with the eigenvalues of D_{11}. We named this subspace $\mathcal{S}$ in Section 3.3.2. The eigenvalues of D_{11} are the isometric Ritz values that are not discarded in the contraction phase. Since $\tilde{H}_m$ is an unreduced upper Hessenberg matrix, and g has degree $j = m - k$, the first k columns of $g(\tilde{H}_m)$ are linearly independent and therefore span $\mathcal{S}$.

Now consider a decomposition $g(\tilde{H}_m) = PN$, where P is unitary and N is upper triangular. Partition P as $P = \begin{bmatrix} P_1 & P_2 \end{bmatrix}$, where P_1 has k columns. Then $\mathcal{R}(P_1) = \mathcal{R}(g(\tilde{H}_m)) = \mathcal{S}$. Let $\check{Q}_m = Q_m P$ and

$$\check{Q}_k = Q_m P_1. \tag{3.3.46}$$

Starting from (3.3.36), one easily proves by induction that

$$g(U)Q_m = Q_m g(\tilde{H}_m) + E_j,$$

where the first k columns of E_j are zero. Then, using the decomposition $g(\tilde{H}_m) = PN$, we obtain

$$g(U)Q_m = \check{Q}_m N + E_j.$$

Now, retaining only the first k columns of this equation, we obtain

$$g(U)Q_k = \check{Q}_k \check{N}, \tag{3.3.47}$$

where $\check{N}$ is the $k \times k$ leading principal submatrix of N and is nonsingular. Therefore $\mathcal{R}(\check{Q}_k) = g(U)\mathcal{R}(Q_k)$.

Finally we notice that since $\mathcal{R}(P_1) = \mathcal{S} = \mathcal{R}(S_1)$, equations (3.3.39), (3.3.41), and (3.3.46) show that $\mathcal{R}(\hat{Q}_k) = \mathcal{R}(\check{Q}_k)$. Thus $\mathcal{R}(\hat{Q}_k) = g(U)\mathcal{R}(Q_k)$. $\square$

In the case when $\mathcal{R}(\hat{Q}_k)$ is not invariant under U (which is always the case up until convergence has been achieved), we can get a sharper result.

Proposition 3.7. *Let $A \in \mathbb{C}^{n\times n}$ and let $\mathcal{V}$ be a subspace of $\mathbb{C}^n$ that is not invariant under A. Suppose $\mathcal{V} = \mathcal{K}_k(A, \check{q}) = \mathcal{K}_k(A, \hat{q})$. Then $\check{q}$ and $\hat{q}$ are multiples of one another.*

Proof. Since $\mathcal{V}$ is not invariant, $\hat{q}$, $A\hat{q}$, …, $A^{k-1}\hat{q}$ are linearly independent, and $A^k\hat{q} \notin \mathcal{V}$. $\check{q} \in \mathcal{K}_k(A, \hat{q})$, so

$$\check{q} = c_1\hat{q} + c_2 A\hat{q} + \cdots + c_k A^{k-1}\hat{q}$$

for some uniquely determined $c_1, \ldots, c_k$, not all of which are zero. Let r be the largest integer for which $c_r \neq 0$. If $r > 1$, then $A^{k-r+1}\check{q} \in \mathcal{K}_k(A, \check{q}) = \mathcal{V}$. On the other hand,

$$A^{k-r+1}\check{q} = c_1 A^{k-r+1}\hat{q} + \cdots + c_{r-1}A^{k-1}\hat{q} + c_r A^k \hat{q},$$

so

$$A^k \hat{q} = c_r^{-1}\left(A^{k-r+1}\check{q} - c_1 A^{k-r+1}\hat{q} - \cdots - c_{r-1}A^{k-1}\hat{q}\right) \in \mathcal{V}.$$

This contradicts the non-invariance of $\mathcal{V}$ under A. Therefore we must have $r = 1$ and $\check{q} = c_1 \hat{q}$. □

For $i = 1, \ldots, k-1$, let Q_i denote the matrix consisting of the first i columns of Q_k, and likewise for $\hat{Q}_k$.

Proposition 3.8. *Under the conditions of Proposition 3.6, assume further that $\mathcal{R}(\hat{Q}_k)$ is not invariant under U. Then*

$$\mathcal{R}(\hat{Q}_i) = g(U)\mathcal{R}(Q_i), \qquad i = 1,\ 2,\ ,\ldots,\ k.$$

In particular, taking $i = 1$, we see that the original and the restarted starting vector are related by $\hat{q} = \alpha g(U) q$ for some nonzero constant α.

Proof. Equation (3.3.42) implies that $\mathcal{R}(\hat{Q}_k)$ is a Krylov subspace: $\mathcal{R}(\hat{Q}_k) = \mathcal{K}_k(U, \hat{q})$, where $\hat{q}$ is the first column of $\hat{Q}_k$. Better yet,

$$\mathcal{R}(\hat{Q}_i) = \mathcal{K}_i(U, \hat{q}), \qquad i = 1,\ 2,\ \ldots,\ k. \tag{3.3.48}$$

We now wish to establish that similar relationships hold for $\check{Q}_k$ as defined in (3.3.46). Using the transforming matrix P from the decomposition $g(\tilde{H}_m) = PN$, define $\check{H}_m = P^H \tilde{H}_m P$. The equations

$$g(\tilde{H}_m) = PN \quad \text{and} \quad \check{H}_m = P^H \tilde{H}_m P$$

together constitute an iteration of the QR algorithm of degree j. Since all of the shifts $\mu_1, \ldots, \mu_j$ are eigenvalues of $\tilde{H}_m$, $\check{H}_m$ has the special form [43]

$$\check{H}_m = \begin{bmatrix} \check{H}_k & X \\ 0 & Y \end{bmatrix},$$

where $\check{H}_k$ is $k \times k$ and unreduced upper Hessenberg.

Multiply equation (3.3.36) by P on the right to obtain

$$U\check{Q}_m = \check{Q}_m \check{H}_m + \tilde{p} e_m^T P.$$

The first $k-1$ entries of $e_m^T P$ are zero so, retaining the first $k-1$ columns of this equation, we have

$$U\check{Q}_{k-1} = \check{Q}_k \check{H}_{k,k-1}, \tag{3.3.49}$$

where $\check{H}_{k,k-1}$ is the $k \times (k-1)$ obtained by deleting the last column of $\check{H}_k$. Since $\check{H}_k$ is upper Hessenberg, (3.3.49) implies that

$$\mathcal{R}(\check{Q}_i) = \mathcal{K}_i(U, \check{q}), \qquad i = 1,\ 2,\ \ldots,\ k, \tag{3.3.50}$$

where $\check{q}$ is the first column of $\check{Q}_k$.

In the proof of Theorem 3.6 we found that $\mathcal{R}(\hat{Q}_k) = \mathcal{R}(\check{Q}_k)$, so $\mathcal{K}_k(U,\hat{q}) = \mathcal{K}_k(U,\check{q})$. Since this space is not invariant, we can invoke proposition 3.7 to deduce that $\hat{q}$ and $\check{q}$ are multiples of one another. Thus, by (3.3.48) and (3.3.50), $\mathcal{R}(\hat{Q}_i) = \mathcal{R}(\check{Q}_i)$ for $i = 1, \ldots, k$.

Now revisit (3.3.47). Since $\check{N}$ is nonsingular and upper triangular, this equation shows that $\mathcal{R}(\check{Q}_i) = g(U)\mathcal{R}(Q_i)$ for $i = 1, \ldots, k$. Since $\mathcal{R}(\check{Q}_i) = \mathcal{R}(\hat{Q}_i)$ for $i = 1, \ldots, k$, we are done. □

The algorithmic import of proposition 3.8 is as follows: On one hand, suppose we perform an Arnoldi-Schur iteration on the inexact Krylov configuration

$$UQ_m = Q_m\tilde{H}_m + \tilde{p}e_m^H \tag{3.3.51}$$

to yield an Arnoldi decomposition

$$U\hat{Q}_{k-1} = \hat{Q}_{k-1}\hat{H}_{k-1} + \hat{h}_{k,k-1}\hat{q}_k e_{k-1}^H.$$

Such a procedure is carried out in equations (3.3.36) through (3.3.42). On the other hand, suppose we perform an IRA iteration on (3.3.51) using the polynomial filter

$$g(t) = (t-\mu_1)\cdots(t-\mu_j)$$

where $\mu_1, \ldots, \mu_j$ are the eigenvalues of the matrix D_{22} defined in (3.3.37), to yield an Arnoldi decomposition

$$U\check{Q}_k = \check{Q}_k\check{H}_k + \check{h}_{k+1,k}\check{q}_{k+1}e_k^H.$$

If $\check{Q}_{k-1}$ is the matrix that consists of the first $k-1$ columns of $\check{Q}_k$, then proposition 3.8 states that

$$\hat{Q}_{k-1} = \check{Q}_{k-1},$$

provided D_{11} and D_{22} have disjoint spectra. This shows that an Arnoldi-Schur iteration applied to (3.3.51) yields the same result as an IRA iteration applied to (3.3.51) using the polynomial filter g.

The careful reader will have noticed that the proofs given in this section contain all of the same elements as the proof that ordinary Krylov-Schur is equivalent to ordinary implicitly restarted Arnoldi. We have included the details for completeness and because we believe our viewpoint may help to improve understanding of this class of methods. The main enabling equation is (3.3.51) which, while not an Arnoldi decomposition, is enough like one to allow us to draw our conclusions.

3.3.8 Numerical Results

We tested the inexact Arnoldi-Schur algorithm on a variety of unitary matrices. Without loss of generality, we used diagonal unitary matrices. The eigenvalues were selected on the unit circle. We then used the algorithm to seek 20 of the eigenvalues nearest a specified target τ that also lies on the unit circle. The initial dimension of the Krylov space is $k+j$. Then the space is contracted to dimension k, and then re-expanded to dimension $k+j$. A tolerance of $\epsilon = 10^{-8}$ was used, where ϵ is as defined in Section 3.3.6. We computed the residual norm and the difference between each computed eigenvalue and the actual eigenvalue it approximated. We also noted the number of iterations and the number of Arnoldi steps taken. We report the results on three types of unitary matrices.

Table 3.1: Uniformly Distributed Eigenvalues

Size	No. of Iters	No. of Arnoldi steps	Max. Residual	Max. Error
2000	28	2852	1.10×10^{-8}	5.20×10^{-15}
4000	55	5579	1.21×10^{-8}	5.78×10^{-15}
6000	114	11538	1.19×10^{-8}	9.31×10^{-15}
8000	161	16285	1.22×10^{-8}	9.30×10^{-15}
10000	188	19012	1.17×10^{-8}	9.04×10^{-15}

The first type has eigenvalues which are uniformly distributed around the unit circle. The target τ is a random point on the unit circle. We used $k = 25$ and $j = 100$. The results are shown in Table 3.1.

The residuals were about as expected, given the tolerance that was used. The column labeled "Max. Error" gives the maximum error in the 20 computed eigenvalues. This too was about what one would expect, given that unitary matrices are normal. The Arnoldi process is best at finding eigenvalues on the periphery of the spectrum and has a harder time with eigenvalues in the "interior". Thus the problems in this class are "hard" problems, in the sense that every eigenvalue is essentially an "interior" eigenvalue. This is reflected in the large number of iterations needed to get convergence. The number of iterations increases in a fairly regular way as the matrix dimension goes up.

The second type of matrix that we considered has eigenvalues clustered near the

Figure 3.1: Eigenvalues of a Unitary Matrix of the Second Type

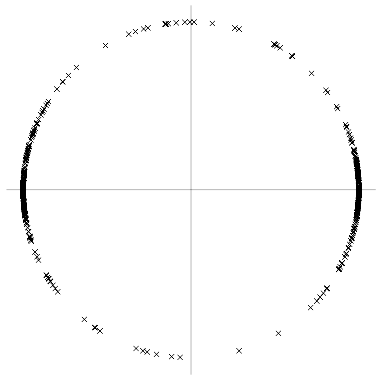

Table 3.2: Eigenvalues Clustered near Real Line

Size	No. of Iters	No. of Arnoldi steps	Max. Residual	Max. Error
2000	4	328	1.01×10^{-9}	5.35×10^{-15}
4000	8	632	1.16×10^{-8}	3.41×10^{-15}
6000	13	1012	1.04×10^{-8}	5.03×10^{-15}
8000	23	1772	1.09×10^{-8}	4.00×10^{-15}
10000	26	2000	1.18×10^{-8}	5.54×10^{-15}

real line. The eigenvalues were generated using the MATLAB commands

```
lambda = 10*randn(n,1)+i*randn(n,1);

lambda = lambda./abs(lambda);
```

The spectrum of a matrix of this type is shown in Figure 3.1. The eigenvalues near i are more like isolated or peripheral eigenvalues, so we expect to be able to compute them relatively quickly. The results in Table 3.2, for which we took $\tau = \mathrm{i}$, confirm this. We used $k = 25$ and $j = 75$.

Table 3.3: Quadrant I Limited Eigenvalues

Size	No. of Iters	No. of Arnoldi steps	Max. Residual	Max. Error
2000	7	556	1.59×10^{-9}	3.68×10^{-15}
4000	10	784	5.95×10^{-9}	5.11×10^{-15}
6000	13	1012	1.01×10^{-8}	9.35×10^{-15}
8000	14	1088	8.94×10^{-9}	1.05×10^{-14}
10000	16	1240	1.01×10^{-8}	9.55×10^{-15}

The third type of matrices has eigenvalues limited to the first quadrant. The eigenvalues were generated by:

```
lambda = exp(i*pi/2*rand(n,1));
```

Setting $\tau = \mathrm{i}$, we hoped to pick off the eigenvalues at one edge of the spectrum fairly quickly. Table 3 confirms that we were able to do this. We used $k = 25$ and $j = 75$.

3.4 A Cayley Transformed Lanczos-Schur Algorithm

As discussed in the numerical results of the preceding section, when a unitary matrix U has a tight cluster of eigenvalues, and the target τ is chosen to be inside the cluster, the inexact Arnoldi-Schur algorithm suffers from the fact that all eigenvalues are interior eigenvalues. When the target is inside a tight cluster of eigenvalues, what is required is a way to separate the eigenvalues near a target from the rest of the spectrum. A method to do this is to perform a Cayley transform on the unitary matrix. The resulting matrix H is Hermitian, in which the eigenvalues of U nearest the target are mapped to the largest eigenvalues of H. Moreover we can use the Hermitian Lanczos process to generate the approximating Krylov space for this

Hermitian matrix. If we use a Schur decomposition to carry out the implicit restart, the decomposition would involve a diagonal matrix having real entries.

The Cayley transformation, however, involves matrix inversion. If the unitary matrix is extremely large, then this shift-and-invert method is not even an option. For that case, we can resort to the inexact Arnoldi-Schur algorithm

3.4.1 The Cayley Transform

Let τ be a complex number such that $|\tau|_2 = 1$. The *Cayley transform with respect to* τ of a unitary matrix $U \in M_n(\mathbb{C})$ is defined as

$$C(U) = \mathrm{i}\ (U + \tau I_n)(U - \tau I_n)^{-1},$$

provided τ is not an eigenvalue of U, and $\mathrm{i} = \sqrt{-1}$.

Given a unitary matrix $U \in M_n(\mathbb{C})$, let $H := C(U)$. Since U is normal, there exists a diagonal matrix $D = \mathrm{diag}\{\lambda_1, \dots, \lambda_n\}$ and a unitary matrix X such that

$$U = XDX^{-1}.$$

Then

$$
\begin{aligned}
H &= \mathrm{i}\ (U+\tau I)\,(U-\tau I)^{-1} \\
&= \mathrm{i}\ \left(XDX^{-1}+\tau XX^{-1}\right)\left(XDX^{-1}-\tau XX^{-1}\right)^{-1} \\
&= \mathrm{i}\ X\,(D+\tau I)\,X^{-1}\left[X\,(D-\tau I)\,X^{-1}\right]^{-1} \\
&= \mathrm{i}\ X\,(D+\tau I)\,(D-\tau I)^{-1}\,X^{-1} \\
&= X\hat{D}X^{-1}
\end{aligned}
$$

where

$$\hat{D} := \operatorname{diag}\left\{\mathrm{i}\ \frac{\lambda_1+\tau}{\lambda_1-\tau},\ldots,\mathrm{i}\ \frac{\lambda_n+\tau}{\lambda_n-\tau}\right\}\Bigg($$

This proves that H is normal. It also shows that if λ is an eigenvalue of U with associated eigenvector x, then x is an eigenvector of H associated with the eigenvalue

$$\mu = \mathrm{i}\ \frac{\lambda+\tau}{\lambda-\tau}.$$

Further if we define the Möbius transformation f by

$$f(z) = \mathrm{i}\ \frac{z+\tau}{z-\tau},$$

then f maps the unit circle into the extended real line, with τ mapped to infinity and to negative infinity, while $-\tau$ is mapped to 0. In particular, f maps the eigenvalues of U to the eigenvalues of H. Thus the eigenvalues of H are real, and by continuity of f on $\mathbb{C}\backslash\{\tau\}$, the eigenvalues of U nearest τ are mapped to the eigenvalues of H of largest magnitude. Finally since H is normal and its eigenvalues are real, it follows that H is Hermitian. This result enables us to use the Lanczos-Schur algorithm to find the eigenvalues of H.

Proposition 3.9. *If $U \in M_n(\mathbb{C})$ is unitary, then the Cayley transformed matrix $H = \mathrm{i}\ (U + \tau I_n)(U - \tau I_n)^{-1}$ is Hermitian.*

3.4.2 The Cayley Transformed Lanczos-Schur Algorithm

We now present an algorithm that makes use of the Cayley transform to find the eigenvalues of a large unitary matrix nearest a specified target τ. Let $U \in M_n(\mathbb{C})$ be unitary, where n is large. Suppose we are seeking k eigenvalues of U nearest a target τ where $\|\tau\|_2 = 1$, and where $k \ll n$. We further assume that τ is not an eigenvalue of U. This is not a severe restriction. With probability zero, τ is an eigenvalue of U, and if this is the case, then we have found an eigenvalue and work with a deflated matrix.

The matrix H defined by the Cayley transform

$$H := \mathrm{i}\ (U + \tau I)(U - \tau I)^{-1}$$

is Hermitian. Let the eigenvalues $\mu_1, \ldots, \mu_n$ of H be ordered so that

$$|\mu_n| \leq |\mu_{n-1}| \leq \cdots \leq |\mu_1|.$$

From the results of the preceding subsection, it follows that

$$\mu_i = \mathrm{i}\ \frac{\lambda_i + \tau}{\lambda_i - \tau} \qquad i = 1, \ldots, k$$

where $\lambda_1, \ldots, \lambda_k$ are the k eigenvalues of U nearest τ. To compute $\lambda_1, \ldots, \lambda_k$, we need to compute the eigenvalues $\mu_1, \ldots, \mu_k$ of H, and then compute each λ_i by the transformation

$$\lambda_i = \tau \, \frac{\mu_i + \mathrm{i}}{\mu_i - \mathrm{i}} \qquad i = 1, \ldots, k. \tag{3.4.52}$$

To find $\mu_1, \ldots, \mu_k$, we will use a Lanczos-Schur algorithm on H. The Krylov space $\mathcal{K}_m(H, q), q \neq 0$, is generated by the Hermitian Lanczos process. The restart strategy makes use of the Schur decomposition. This is equivalent to the thick restart strategy proposed by Wu and Simon [45] for symmetric matrices. We briefly describe the details of the Lanczos-Schur algorithm below. The implementation is straightforward.

Let $q \in \mathbb{C}^n,\ q \neq 0$. We begin with an initial expansion. Performing the Lanczos process on H with q as the starting vector, we set $q_1 = q/\|\, q \,\|_2$ and generate the vectors q_{i+1} for $i = 1, \ldots, m$ by

$$\beta_i q_{i+1} = H q_i - \alpha_i q_i - \beta_{i-1,i} q_{i-1}. \tag{3.4.53}$$

Each q_{i+1} is re-orthogonalized against $q_1, \ldots, q_i$. The parameters α_i are given by

$$\alpha_i = \langle H q_i, q_i \rangle \qquad i = 1, \ldots, m.$$

For each $i = 1, \ldots, m$, the parameter β_i is chosen as the unique positive scalar such that $\|\, q_{i+1} \,\|_2 = 1$.

After m steps of the Lanczos process, we have the Arnoldi decomposition

$$H Q_m = Q_m T_m + t_{m+1,m} q_{m+1} e_m^H \tag{3.4.54}$$

where T_m is tridiagonal:

$$T_m = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & & & \beta_{m-1} \\ & & & \beta_{m-1} & \alpha_m \end{pmatrix}$$

and where q_{m+1} is a unit vector orthogonal to the columns of Q_m. Matrix T_m is stored into two vectors **a** and **b** containing the parameters $\alpha_1, \dots, \alpha_m$ and $\beta_1, \dots, \beta_{m-1}$ respectively. As noted in section 3.1.2, the matrix T_m is real symmetric, and the parameters $\alpha_1, \dots, \alpha_m$ and $\beta_1, \dots, \beta_{m-1}$ are real.

The contraction phase makes use of the Schur decomposition. Let

$$S^H T_m S = D$$

be the Schur decomposition of T_m. Since T_m is symmetric, it follows that D is diagonal. We further assume that the eigenvalues of T_m appear along the diagonal of D in decreasing order of magnitude. Thus if

$$D = \begin{pmatrix} D_{11} & 0 \\ 0 & D_{22} \end{pmatrix} \tag{3.4.55}$$

where $D_{11} \in \mathbb{C}^{k \times k}$, then D_{11} contains k eigenvalues of T_m of largest magnitude. These are precisely the Ritz values we want to keep.

Post-multiply S to (3.4.54) to get

$$H\hat{Q}_m = \hat{Q}_m D + t_{m+1,m} q_{m+1} e_m^H S$$

where $\hat{Q}_m = Q_m S$. Extracting the first k columns, we get

$$H\hat{Q}_k = \hat{Q}_k D_{11} + t_{m+1,m} q_{m+1} s_k^H \tag{3.4.56}$$

where s_k^H consists of the first k entries of the mth row of S. Construct a unitary matrix W such that $t_{m+1,m} s_k^H W = \beta e_k^H$ where $\beta = \| t_{m+1,m} s_k \|_2$. Post-multiply W to (3.4.56) to get

$$H\hat{Q}_k W = \hat{Q}_k W W^H D_{11} W + \beta q_{m+1} e_k^H. \tag{3.4.57}$$

Finally construct a unitary matrix V such that $W^H D_{11} W V =: \check{T}_k$ is upper Hessenberg with positive subdiagonals. Post-multiplying the matrix V to (3.4.57), and noting that V acts an identity on the last term, we get

$$H\check{Q}_k = \check{Q}_k \check{T}_k + \beta q_{m+1} e_k^H \tag{3.4.58}$$

where $\check{Q}_k := \hat{Q}_k W V$. Since the columns of $\check{Q}_k$ are orthonormal and the subdiagonals of $\check{T}_k$ are positive, (3.4.58) is an Arnoldi decomposition by Prop. 3.3. It follows that $\check{T}_k$ is symmetric tridiagonal. The entries in the vectors $\mathbf{a}$ and $\mathbf{b}$ can now be replaced with the new diagonal and subdiagonal entries of $\check{T}_k$ respectively.

Restarting the Lanczos process is surprisingly easy. Set $\mathbf{b}(k) = \beta$ and set $q_{k+1} = q_{m+1}$. These are known from the last term of (3.4.58). Generate the new vectors using (3.4.53) with i running from $k+1$ to m. Re-orthogonalization is done against the columns of $\check{Q}_k$. Locking and convergence are identical to those of the Krylov-Schur algorithm. With the ordering of the eigenvalues done in (3.4.55), the implicit restart performs a polynomial filter that suppresses the eigenvalues of small magnitude. Hence the Lanczos-Schur algorithm seeks the eigenvalues $\mu_1, \ldots, \mu_k$ of H of largest magnitude. The corresponding eigenvalues of U nearest τ are computed using (3.4.52).

3.4.3 Numerical Results

We used the Cayley transformed Lanczos-Schur algorithm to seek 10 eigenvalues nearest a target τ. We report the results on the same types of matrices as in section 3.3.8: unitary matrices with uniformly distributed eigenvalues, matrices with eigenvalues clustered near the real line, and matrices with quadrant 1 limited eigenvalues. The size of the test matrices for this algorithm is at least one order of magnitude larger than those of 3.3.8. The initial size of the Krylov space was $k + j$, which was then contracted to j in each iteration, and then re-expanded to $k + j$. We used the values $k = 10$ and $j = 10$ in each case. The tolerance $\epsilon = 10^{-8}$ was used for the relative error of the residual.

Table 3.4 shows the results for unitary matrices with uniformly distributed eigenvalues. The target τ was a randomly chosen point on the unit circle. Within a few iterations, the locking tolerance of $\epsilon = 10^{-8}$ was attained by the Lanczos-Schur algorithm, the usual figure being in the order of 10^{-12}. This represents the relative error of the residual norm $\| H\hat{v} - \hat{\mu}\hat{v} \|_2$, where $(\hat{\mu}, \hat{v})$ is the computed eigenpair of the Cayley transformed Hermitian matrix H. Since the largest eigenvalues of H are usually in the order of 10^3, the absolute error of the residual norm is in the order of 10^{-15}. When we transform these eigenvalues to the eigenvalues of the unitary matrix, the absolute error remains in the order of 10^{-15}. Moreover since each eigenvalue of the unitary matrix is of modulus one, the relative error of the residual norm of the unitary matrix is also the absolute error. We further note that the maximum errors (both relative and absolute) between the computed eigenvalues and the true eigenvalues are in the order of MATLAB's `eps` permanent variable, representing the distance from 1.0 to the next largest floating point number.

Table 3.4: Uniformly Distributed Eigenvalues

Size	No. of Iters	No. of Arnoldi steps	Max. Residual	Max. Error
100000	5	60	8.08×10^{-15}	2.48×10^{-16}
200000	3	40	2.53×10^{-16}	2.22×10^{-16}
300000	5	60	2.78×10^{-16}	2.29×10^{-16}
400000	3	40	1.78×10^{-16}	1.57×10^{-16}
500000	7	80	1.31×10^{-15}	3.34×10^{-16}

For unitary matrices with spectra having a thick cluster near the real axis, the eigenvalues were generated by the MATLAB commands

```
lambda = 100*randn(n,1)+i*randn(n,1);

lambda = lambda./abs(lambda);
```

The factor of 100 was used to exaggerate the clustering of the eigenvalues near the real axis. We report the results of two runs of the algorithm with different targets. On the first run, the target was $\tau_1 = 1$, in the center of the thick cluster. The results are summarized in table 3.5. The table shows that the algorithm can find eigenvalues nearest a target which is embedded in the center of a thick cluster. The residual norms are in the order of `eps`. Some of the maximum errors are already less than `eps`, and we expect them to be smaller if multi-precision arithmetic is used.

Table 3.5: Real Clustered Eigenvalues, target $\tau_1 = 1$

Size	No. of Iters	No. of Arnoldi steps	Max. Residual	Max. Error
100000	5	60	2.24×10^{-16}	1.11×10^{-16}
200000	5	60	2.25×10^{-16}	2.22×10^{-16}
300000	7	80	4.76×10^{-16}	1.11×10^{-16}
400000	10	110	2.48×10^{-15}	2.22×10^{-16}
500000	6	70	2.65×10^{-16}	2.22×10^{-16}

On the second run, the target was $\tau_2 = \mathrm{i}$, far from the thick cluster, and in the region of the spectrum where the eigenvalues are isolated. The results are presented in table 3.6. The table shows that the algorithm also performed well in seeking eigenvalues in the region of isolated eigenvalues. It is interesting to note that the residual norms are larger for this situation where the eigenvalues being sought are far from the thick cluster. Some of the maximum errors are smaller than `eps`.

Table 3.6: Real Clustered Eigenvalues, target $\tau_2 = \mathrm{i}$

Size	No. of Iters	No. of Arnoldi steps	Max. Residual	Max. Error
100000	4	50	1.92×10^{-12}	2.22×10^{-16}
200000	4	50	8.84×10^{-12}	2.22×10^{-16}
300000	4	50	1.61×10^{-11}	1.11×10^{-16}
400000	4	50	6.88×10^{-13}	1.11×10^{-16}
500000	4	50	5.09×10^{-12}	2.23×10^{-16}

Table 3.7 shows the results for unitary matrices with eigenvalues limited in quadrant I. The target was $\tau = 1$. The table shows that the algorithm can seek eigenvalues at the edge of the spectrum.

Table 3.7: Quadrant I Limited Eigenvalues

Size	No. of Iters	No. of Arnoldi steps	Max. Residual	Max. Error
100000	6	70	3.84×10^{-14}	2.22×10^{-16}
200000	7	80	1.35×10^{-14}	2.22×10^{-16}
300000	7	80	3.39×10^{-14}	2.22×10^{-16}
400000	5	60	2.97×10^{-14}	1.11×10^{-16}
500000	8	90	6.51×10^{-15}	1.11×10^{-16}

Chapter 4

The Product Unitary Eigenvalue Problem

In this chapter, we consider the problem of finding the eigenvalues of the product

$$U = U_k U_{k-1} \cdots U_1 \tag{4.0.1}$$

where each $U_i \in M_n(\mathbb{C})$ is unitary. Such a problem arises in quantum Schubert calculus, for instance, where the product

$$I = U_k U_{k-1} \cdots U_1$$

is considered [1]. We shall employ the strategy used in product eigenvalues in general [42].

4.1 A Product Unitary QR Algorithm

The algorithm that computes the eigenvalues of the product (4.0.1) is based on the following result.

Proposition 4.1. *Let $U_i \in M_n(\mathbb{C})$ be unitary for $i = 1, \ldots, k$. Then there exist unitary matrices $Q_i \in M_n(\mathbb{C})$, $i = 1, \ldots, k$ such that*

$$\begin{aligned} Q_1 U_1 Q_k^H &= I_n \\ Q_2 U_2 Q_1^H &= I_n \\ Q_3 U_3 Q_2^H &= I_n \\ &\vdots \\ Q_{k-1} U_{k-1} Q_{k-2}^H &= I_n \\ Q_k U_k Q_{k-1}^H &= H \end{aligned}$$

where $H \in M_n(\mathbb{C})$ is upper Hessenberg with positive subdiagonal entries.

We shall illustrate a constructive proof of proposition 4.1 using $n = 4$ and $k = 3$. The construction of H serves as an algorithm which we shall refer to as the *reduction algorithm.*

Let $U_i \in M_4(\mathbb{C})$ be unitary for $i = 1, 2, 3$. Consider the cyclic matrix

$$C = \begin{pmatrix} & & U_3 \\ U_1 & & \\ & U_2 & \end{pmatrix} \Bigg($$

The objective is to reduce U_3 to upper Hessenberg form by performing unitary similarity transformations to C that simultaneously reduces U_1 and U_2 to the identity. Since the first column of U_1 is a unit vector, we can construct a unitary matrix $Q_1^{(1)} \in M_n(\mathbb{C})$ such that $Q_1^{(1)} U_1 e_1 = e_1$. Forming the block diagonal unitary matrix

$\mathrm{diag}\{I, Q_1^{(1)}, I\}$ and performing a unitary similarity transformation to C, we get

$$\begin{pmatrix} I & & \\ & Q_1^{(1)} & \\ & & I \end{pmatrix} \begin{pmatrix} & & U_3 \\ U_1 & & \\ & U_2 & \end{pmatrix} \begin{pmatrix} I & & \\ & Q_1^{(1)} & \\ & & I \end{pmatrix}^H = \begin{pmatrix} & & U_3 \\ Q_1^{(1)}U_1 & & \\ & U_2Q_1^{(1)H} & \end{pmatrix} =: C^{(1)}.$$

The first column of $C^{(1)}$ is a block vector of the form $\begin{pmatrix} 0 \\ e_1 \\ 0 \end{pmatrix}$. The first column of $U_2Q_1^{(1)H}$ is a unit vector, hence we can construct a unitary matrix $Q_2^{(1)} \in M_n(\mathbb{C})$ such that $Q_2^{(1)}U_2Q_1^{(1)H}e_1 = e_1$. Forming the block diagonal matrix $\mathrm{diag}\{I, I, Q_2^{(1)}\}$ and performing a unitary similarity transformation to $C^{(1)}$, we get

$$\begin{aligned} &\begin{pmatrix} I & & \\ & I & \\ & & Q_2^{(1)} \end{pmatrix} C^{(1)} \begin{pmatrix} I & & \\ & I & \\ & & Q_2^{(1)} \end{pmatrix}^H \\ = &\begin{pmatrix} & & U_3Q_2^{(1)H} \\ Q_1^{(1)}U_1 & & \\ & Q_2^{(1)}U_2Q_1^{(1)H} & \end{pmatrix} =: C^{(2)}. \end{aligned}$$

The fifth column of $C^{(2)}$ is a block vector of the form $\begin{pmatrix} 0 \\ e_1 \\ 0 \end{pmatrix}$. Let $U_3 = (u_{ij}^{(3)})$ $i = 1, 2, 3,$ $j = 1, 2, 3$. To reduce $U_3Q_2^{(1)H}$ to upper Hessenberg form, we construct a unitary matrix

$$Q_3^{(1)} = \begin{pmatrix} 1 & \\ & \hat{Q}_3^{(1)} \end{pmatrix}$$

such that $Q_3^{(1)}U_3Q_2^{(1)H} = u_{13}^{(3)}e_1 + \alpha e_2$ where $\alpha > 0$. The form of $Q_3^{(1)}$ shows that premultiplying $Q_3^{(1)}$ to $U_3Q_2^{(1)H}$ does not affect the first row. Similarly, postmultiplication of $Q_1^{(1)}U_1$ by $Q_3^{(1)H}$ does not affect the first column. Thus in carrying out the similarity transformation

$$\begin{pmatrix} Q_3^{(1)} & & \\ & I & \\ & & I \end{pmatrix} C^{(2)} \begin{pmatrix} Q_3^{(1)} & & \\ & I & \\ & & I \end{pmatrix}^H$$

$$= \begin{pmatrix} & & Q_3^{(1)} U_3 Q_2^{(1)H} \\ Q_1^{(1)} U_1 Q_3^{(1)H} & & \\ & Q_2^{(1)} U_2 Q_1^{(1)H} & \end{pmatrix} =: C_2$$

we have reduced the first column of $U_3 Q_1^{(1)H}$ to upper Hessenberg form, and preserved the first columns of $U_1 Q_3^{(1)H}$ and $Q_2^{(1)} U_2 Q_1^{(1)H}$ both of which are the first column the identity matrix. Schematically, the form of C_2 is

$$C_2 = \begin{pmatrix} & & & & & & & & x & x & x & x \\ & & & & & & & & x & x & x & x \\ & & & & & & & & 0 & x & x & x \\ & & & & & & & & 0 & x & x & x \\ 1 & 0 & 0 & 0 & & & & & & & & \\ 0 & x & x & x & & & & & & & & \\ 0 & x & x & x & & & & & & & & \\ 0 & x & x & x & & & & & & & & \\ & & & & 1 & 0 & 0 & 0 & & & & \\ & & & & 0 & x & x & x & & & & \\ & & & & 0 & x & x & x & & & & \\ & & & & 0 & x & x & x & & & & \end{pmatrix}$$

We repeat the process of cycling through the block matrices to generate the unitary matrices $Q_i^{(2)}$ for $i = 1, 2, 3$ and obtain a block matrix of the form

$$C_3 = \begin{pmatrix} & & \begin{pmatrix} x & x & x & x \\ x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{pmatrix} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{pmatrix} & & \\ & \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{pmatrix} & \end{pmatrix}$$

Repeating this process, we obtain the unitary matrices

$$\begin{aligned} Q_1 &= Q_1^{(4)} Q_1^{(3)} Q_1^{(2)} Q_1^{(1)} \\ Q_2 &= Q_2^{(4)} Q_2^{(3)} Q_2^{(2)} Q_2^{(1)} \\ Q_3 &= Q_3^{(4)} Q_3^{(3)} Q_3^{(2)} Q_3^{(1)} \end{aligned}$$

that satisfy

$$\begin{aligned} Q_1 U_1 Q_3^H &= I_3 \\ Q_2 U_2 Q_1^H &= I_3 \\ Q_3 U_3 Q_2^H &= H \end{aligned}$$

where H is upper Hessenberg with positive subdiagonals. This completes the description of the reduction algorithm.

Consider now the product

$$U = U_k U_{k-1} \cdots U_1$$

where each $U_i \in M_n(\mathbb{C})$ is unitary. Performing the reduction algorithm, we obtain unitary matrices $Q_1, \ldots, Q_k$ such that

$$\begin{aligned} Q_1 U_1 Q_k^H &= I_n \\ &\vdots \\ Q_k U_k Q_{k-1}^H &= H \end{aligned}$$

where H is upper Hessenberg with positive subdiagonals. Notice that

$$\begin{aligned} H &= I \cdot I \cdots I \cdot H \\ &= (Q_k U_k Q_{k-1}^H) \cdot (Q_{k-1} U_{k-1} Q_{k-2}^H) \cdots (Q_1 U_1 Q_k^H) \\ &= Q_k U_k \cdots U_1 Q_k^H. \end{aligned}$$

Thus H is unitarily similar to U. To find the eigenvalues of U, we perform the unitary QR algorithm to H as described in chapter 2. This amounts to an eigenvalue algorithm which we will call the *product unitary QR algorithm.* The reduction algorithm implicitly performs the multiplication $U = U_k U_{k-1} \cdots U_1$ and the reduction of U to upper Hessenberg form, at a total cost of $O(\frac{8}{3}kn^3)$ flops. Performing the multiplication explicitly costs $O(2kn^3)$ flops and a separate reduction of U to upper Hessenberg form costs $O(\frac{10}{3}n^3)$ flops. The reduction algorithm is cheaper when $k \leq 5$.

4.2 Numerical Results

We tested the unitary product QR algorithm on the product $U = U_k \cdots U_1$ for $k = 2, \ldots, 6$ and the size $n = 100,\ 200,\ 300,\ 400$. The following table shows the maximum error of the computed eigenvalues from the true eigenvalues of the product. The individual factors U_i have randomly distributed eigenvalues around the unit circle.

Table 4.1: Product Unitary Eigenvalues

Size	k=2	k=3	k=4	k=5	k=6
100	9.44×10^{-14}	9.90×10^{-14}	1.30×10^{-13}	5.50×10^{-14}	4.83×10^{-14}
200	5.79×10^{-13}	5.73×10^{-13}	4.08×10^{-13}	3.24×10^{-13}	5.22×10^{-13}
300	5.77×10^{-13}	1.32×10^{-12}	1.32×10^{-12}	9.84×10^{-13}	1.03×10^{-12}
400	1.44×10^{-12}	2.30×10^{-12}	2.69×10^{-12}	1.42×10^{-12}	1.61×10^{-12}

4.3 The Generalized Eigenvalue Problem for Unitary Matrices

As a special case of the unitary product eigenvalue algorithm, we consider the generalized eigenvalue problem

$$Ax = \lambda Bx \tag{4.3.2}$$

where $A, B \in M_n(\mathbb{C})$ are both unitary. Multiplying by B^{-1}, we convert this to an ordinary eigenvalue problem

$$B^{-1}Ax = B^H Ax = \lambda x.$$

We then use the product eigenvalue algorithm to find the eigenvalues of $B^H A$.

We tested this method for n ranging from 500 to 1000 in increments of 100. The following table shows the maximum error of the computed eigenvalues. Both A and B have uniformly distributed eigenvalues.

Table 4.2: Generalized Eigenvalues

	500	600	700	800	900	1000
k=2	1.34×10^{-12}	2.10×10^{-12}	2.72×10^{-12}	4.10×10^{-12}	2.75×10^{-12}	1.90×10^{-12}

Bibliography

[1] S. Agnihotri and C. Woodward, *Eigenvalues of products of unitary matrices and quantum Schubert calculus*, Mathematical Research Letters, 5 (1998), pp. 817–836.

[2] G. Ammar, W. Gragg, and L. Reichel, *Detemination of Pisarenko frequency estimates as eigenvalues of an orthogonal matrix*, in Proc. SPIE, Advanced Algorithms and Architectures for Signal Processing II, F. T. Luk, ed., vol. 826, San Diego, 1987, pp. 143–145.

[3] ——, *Direct and inverse unitary eigenproblems in signal processing: an overview*, in Linear Algebra for Large Scale and Real-Time Applications, F. T. Moonen, G. H. Golub, and B. L. D. Moor, eds., The Netherlands, 1993, pp. 341–343.

[4] G. Ammar, W. B. Gragg, and C. He, *An efficient QR algorithm for a Hessenberg submatrix of a unitary matrix*, in New Directions and Applications in Control Theory, Springer, 2005.

[5] G. Ammar, W. B. Gragg, and L. Reichel, *Constructing a unitary Hessenberg matrix from spectral data*, in Numerical Linear Algebra, Digital Signal Processing, and Parallel Algorithms, Springer, 1991, pp. 385–396.

[6] G. S. Ammar, D. Calvetti, and L. Reichel, *Computing the poles of autoregressive models from the reflection coefficients*, in Proc. 31st Annual Allerton

Conference on Communication, Control, and Computing, Monticello, IL, 1993, pp. 255–264.

[7] ——, *Continuation methods for the computation of zeros of szego polynomials*, Lin. Alg. Appl., (1996), pp. 125–155.

[8] G. S. Ammar, W. B. Gragg, and L. Reichel, *On the eigenproblem for orthogonal matrices*, in Proc. 25th IEEE Conference on Decision and Control, Athens, Greece, 1986, pp. 1963–1966.

[9] G. S. Ammar, L. Reichel, and D. C. Sorensen, *An implementation of a divide and conquer algorithm for the unitary eigenproblem*, ACM Trans. Math. Software, 18 (1992), pp. 292–307.

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

[11] Z. Bai and J. Demmel, *On a block implementation of the Hessenberg multishift QR iteration*, Internat. J. High Speed Comput., 1 (1989), pp. 97–112.

[12] A. Bunse-Gerstner and L. Elsner, *Schur parameter pencils for the solution of the unitary eigenproblem*, Linear Algebra Appl., 154–156 (1991), pp. 741–778.

[13] A. Bunse-Gerstner and C. He, *On a Sturm sequence of polynomials for unitary Hessenberg matrices*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 1043–1055.

[14] G. Cybenko, *A general orthogonalization technique with applications to time series analysis and signal processing*, Math. Comp., 40 (1983), pp. 323–336.

[15] ——, *Pisarenko frequency estimates*, in Proc. 18th Annual Princeton Conference on Information Systems and Sciences, Princeton, New Jersey, 1984, pp. 587–591.

[16] G. Cybenko and C. van Loan, *Computing the minimum eigenvalue of a symmetric positive definite Toeplitz matrix*, SIAM J. Sci. Stat. Comp., 7 (1986), pp. 123–131.

[17] R. J. David and D. S. Watkins, *Efficient implementation of the multi-shift QR algorithm for the unitary eigenvalue problem*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 623–633.

[18] P. Delsarte and Y. Genin, *An efficient algorithm for computing Pisarenko's harmonic decomposition using Levinson's recursion*, IEEE Trans. Acoust. Speech Signal Process, 34 (1986), pp. 485–491.

[19] ——, *The split Levinson algorithm*, IEEE Trans. Acoust. Speech Signal Process, 34 (1986), pp. 470–478.

[20] P. Diaconis, *Patterns in eigenvalues: The 70th Josiah Willard Gibbs lecture*, Bull. Amer. Math. Soc., 40 (2003), pp. 155–178.

[21] P. J. Eberlein and C. P. Huang, *Global convergence of the QR algorithm for unitary matrices with some results about normal matrices*, SIAM J. Numer. Anal., 12 (1975), pp. 421–453.

[22] A. Edelman and N. R. Rao, *Random matrix theory*, Acta Num., (2005), pp. 233–297.

[23] H. Fassbenber, *On numerical methods for discrete least-squares approximation for trigonometric polynomials*, Math. Comp., 66 (1997), pp. 719–741.

[24] W. B. Gragg, *Positive definite Toeplitz matrices, the Arnoldi process for isometric operators, and Gaussian quadrature on the unit circle*, in Numerical Methods in Linear Algebra, E. S. Nikolaev, ed., Moscow University Press, 1982, pp. 16–23. (in Russian).

[25] ——, *The QR Algorithm for unitary Hessenberg matirces*, J. Comput. Appl. Math., 16 (1986), pp. 1–8.

[26] ——, *Positive definite Toeplitz matrices, the Arnoldi process for isometric operators, and Gaussian quadrature on the unit circle*, J. Comput. Appl. Math., 46 (1993), pp. 183–198. (English translation of [24]).

[27] W. B. Gragg and L. Reichel, *A divide and conquer algorithm for the unitary and orthogonal eigenproblems*, Numer. Math., 57 (1990), pp. 695–718.

[28] M. Gu, R. Guzzo, X.-B. Chi, and X.-Q. Cao, *A stable divide and conquer algorithm for the unitary eigenproblem*, SIAM J. Matrix Anal. Appl., 25 (2003), pp. 385–404.

[29] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*, John Wiley and Sons, 1996.

[30] S. Helsen, A. B. Kuilaars, and M. van Barel, *Convergence of the isometric Arnoldi process*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 782–809.

[31] R. J. McAulay and T. F. Quatieri, *Speech analysis/synthesis based on sinusoidal representation*, IEEE Tran. Acoust. Speech Signal Process., 34 (1986), pp. 744–754.

[32] V. F. Pisarenko, *Retrieval of harmonics from a covariance function*, Geophys. J. R. Astr. Soc., 33 (1973), pp. 347–366.

[33] L. Reichel, G. S. Ammar, and W. B. Gragg, *Discrete least squares approximation by trigonometric polynomials*, Math. Comp., 57 (1991), pp. 273–289.

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

[35] G. W. Stewart, *A Krylov-Schur algorithm for large eigenproblems*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 601–614.

[36] M. Stewart, *An error analysis of a unitary Hessenberg QR algorithm*, Tech. Rep. TR-CS-98-11, Department of Computer Science, Australian National University, 1998. http://eprints.anu.edu.au/archive/00001557/.

[37] T.-L. Wang and W. B. Gragg, *Convergence of the shifted QR algorithm for unitary hessenberg matrices*, Math. Comp., 71 (2002), pp. 1473–1496.

[38] D. S. Watkins, *Some perspectives on the eigenvalue problem*, SIAM Review, (1993), pp. 430–471.

[39] D. S. Watkins, *The transmission of shifts and shift blurring in the QR algorithm*, Linear Algebra Appl., 241–243 (1996), pp. 877–896.

[40] D. S. Watkins, *Unitary orthogonalization processes*, J. Comp. Appl. Math., (1997), pp. 335–345.

[41] D. S. Watkins, *Fundamentals of Matrix Computations*, John Wiley and Sons, Second ed., 2002.

[42] D. S. Watkins, *Product eigenvalue problems*, SIAM Review, (2005), pp. 3–40.

[43] D. S. Watkins and L. Elsner, *Chasing algorithms for the eigenvalue problem*, SIAM J. Matrix Anal. Appl., 12 (1991), pp. 374–384.

[44] ——, *Convergence of algorithms of decomposition type for the eigenvalue problem*, Linear Algebra Appl., 143 (1991), pp. 19–47.

[45] K. Wu and H. Simon, *Thick-restart Lanczos method for large symmetric eigenvalue problems*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 602–616.