University of Zagreb
Department of Mathematics

Doctor of Philosophy Dissertation

Krylov Type Methods for Large Scale Eigenvalue Computations

by

Zvonimir Bujanović

Supervisor: prof. Zlatko Drmač

Zagreb, 2011.

Contents

Introduction

Computing the eigenpairs of a given matrix is one of the most important tasks of numerical linear algebra. From the perspective of theoretical mathematics, this problem is relatively simple – the existence, the total number of eigenpairs and their mutual relations has been resolved by the end of the nineteenth century by the Jordan form. The first algorithms (Jacobi [33]) for computing the eigenpairs date from the same period.

From the viewpoint of the practical applications in numerical simulations, using computers whose architecture disables the exact evaluation of even the most basic arithmetic operations and which are limited by the processing power and the amount of memory, the eigenvalue problem remains very actual. The need for an efficient and accurate computation of eigenpairs for matrices of ever larger and larger sizes resulted in the advent of many new and sophisticated algorithms. The theoretical and the numerical understanding of the problem was deepened through the perturbation theory.

With today's computers, the problem of computing all eigenpairs can be solved in a reasonable amount of time for matrices of orders up to several thousands. On the other hand, high resolution modelling of physical phenomena may require eigenpairs of huge matrices with orders ranging from several hundred thousands to up to billions. Fortunately, in most cases only very few eigenvalues have significance that determines the behavior of the physical system – usually those that are in some sense extreme, like the ones with largest module, or the largest or smallest real part, or those closest to some given point τ. This opens the possibility of designing specialized algorithms for computing only the part of the spectrum that is of interest. Such algorithms have a limited set of permissible actions on the matrix involved, considering its size. If A is a large scale matrix, typically the only action available is the computation of Ax, for a given vector x. In some cases, solving a linear system $(A - \xi I)x = b$, with some scalar ξ and some vector b is also an option.

Krylov subspace methods have a distinct place in such a setting. These methods iteratively build orthonormal bases for subspaces

$$\mathcal{K}_m = \operatorname{span}\{v, Av, A^2v, \ldots, A^{m-1}v\}, \text{ for } m = 1, 2, \ldots,$$

where v is some initial vector. Eigenvector approximations are computed as elements of $\mathcal{K}_m$ satisfying some optimality constraints. Such approximations have proved to be very

successful in practice, and the most widely used software for large scale eigenvalue computations (ARPACK [41]) is based on Krylov subspaces. This work is devoted to studying such methods, in particular to the most prominent of them – the Arnoldi algorithm.

The thesis is structured as follows:

Chapter 1 is an introductory chapter. It gives an overview of the eigenvalue problem, together with the most important results of the perturbation theory for the eigenvalues and eigenspaces. Methods for evaluating the quality of a given subspace as an approximation of an eigenspace are presented. We also discuss several most commonly used options for extracting approximations from a subspace. This gives a theoretical basis for building algorithms.

Chapter 2 is devoted to the Arnoldi algorithm. We list well-known properties of the Krylov subspaces and the existing results of the convergence theory. The algorithmic aspect of the Arnoldi method is studied, as well as some known issues concerning the numerical stability of the Lanczos algorithm, the accompanying symmetric version of Arnoldi. Very important is the topic of restarting the Arnoldi algorithm. We present the method of Sorensen [65], resulting in the implicitly restarted Arnoldi algorithm (IRAM), and the method of Stewart [67], resulting in the Krylov–Schur algorithm.

In the following Chapters, the original work is presented.

Stewart demonstrated how the Krylov–Schur algorithm can be used for restarting so that selected (skew, such as harmonic) Ritz vectors are kept in the restarted subspace. In Chapter 3 we show how the Krylov–Schur algorithm can be used with arbitrary shifts, similarly to the implicitly restarted Arnoldi method. We also show that any method of Krylov–Schur restarting is necessarily connected with solving of a pole placement problem. The pole placement problem is generally considered very badly conditioned, and we demonstrate how this effects the Krylov–Schur restart.

As shown by Sorensen, the restarted Arnoldi method using the exact shifts converges when the matrix A is Hermitian. On the other hand, Embree [28] constructed a non-normal matrix for which this method can fail to converge. In Chapter 4 we discuss the case of a normal matrix A. Connected to this problem is the geometry of the Ritz values, which we study from two perspectives. First, we show that analogies with the Cauchy interlacing lemma for projection of the eigenvalues onto arbitrary lines in the complex plane hold. This implies certain monotonicity during the growth phase of the Arnoldi algorithm. However, we show that such monotonicity in general is not preserved during a restart. Second, we show that a set of k complex numbers is a set of Ritz values for a normal matrix A if and only if a certain linear system involving a Cauchy matrix has a positive solution. Using this fact, several known properties of the Ritz values can be very simply deduced. Also, an example where the restarted Arnoldi algorithm fails to compute the second largest eigenvalue is constructed.

In Chapter 5 we present a new algorithm for reducing a matrix to a m-Hessenberg form (a banded Hessenberg form with the lower bandwidth m). This reduction occurs in implementations of a block–Arnoldi algorithm, when the default m-Hessenberg form of the

Rayleigh quotient is perturbed because of the restart or a deflation. Another important application of the reduction is found in the control theory. We analyze the performance of the new algorithm and compare it to existing ones found in libraries LAPACK [2] and SLICOT [64]. Based on the work of Tomov et al. [71], we also develop a hybrid algorithm that computes the reduction using both the CPU and the GPU.

Acknowledgement.
I would like to thank prof. Zlatko Drmač for his guidance, patience, optimism, and many motivating discussions. I am also grateful to prof. Sanja and Saša Singer and prof. Mladen Rogina for making available the access to the computing hardware necessary for testing, and also to Vedran Novaković for suggestions and discussions on the topic of GPU computing.

Dedication.
I dedicate this thesis to my parents and to my brother and sister, for all their support.

Chapter 1

The Eigenvalue Problem

This introductory chapter gives a short overview of the available theory concerning the matrix spectral decomposition. Notation used throughout this work is listed in the first section.

1.1 Notation

Scalars, vectors and subspaces. In this work we will consider objects over fields of real ($\mathbb{R}$) and complex ($\mathbb{C}$) numbers. Elements of these fields (scalars) will be denoted with lowercase Greek letters ($\alpha, \beta, \gamma, \ldots$). Vectors belonging to the finite-dimensional spaces $\mathbb{R}^n$, $\mathbb{C}^n$ are denoted with lowercase Roman letters – for example, vectors of the canonical basis are $e_1, e_2, \ldots, e_n$. Exceptionally, vector components with regard to the canonical basis will be denoted with indexed Roman letters as well: $x = (x_1, x_2, \ldots, x_n)$. Spaces $\mathbb{R}^n$, $\mathbb{C}^n$ are equipped with the standard scalar product $\langle \cdot \,|\, \cdot \rangle$ which induces the Euclidean norm (denoted by $\|\cdot\|$): for $x = (x_1, x_2, \ldots, x_n)$, $y = (y_1, y_2, \ldots, y_n) \in \mathbb{C}^n$ we have

$$\begin{aligned} \|x\| &= \sqrt{|x_1|^2 + |x_2|^2 + \ldots + |x_n|^2}, \\ \langle x \,|\, y \rangle &= x_1\overline{y_1} + x_2\overline{y_2} + \ldots + x_n\overline{y_n}. \end{aligned}$$

We will often write $y^\star x$ instead of $\langle x \,|\, y \rangle$, using analogy with matrix notation where a vector in n-dimensional space is interpreted as a matrix with n rows and a single column. Scalar product and Euclidean norm are connected via Schwarz-Cauchy-Bunjakovski inequality:

$$|y^\star x| = |\langle x \,|\, y \rangle| \leq \|x\| \cdot \|y\| .$$

If the space is equipped with another scalar product or norm, it will be given special notation such as $\langle \cdot \,|\, \cdot \rangle_A$ or $\|\cdot\|_A$.

Subspaces of $\mathbb{R}^n$ or $\mathbb{C}^n$ will be typically denoted calligraphically: $\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \ldots$ and $\mathcal{X} \leqslant \mathcal{Y}$ is read "$\mathcal{X}$ is a subspace of $\mathcal{Y}$". For $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$, the set of all linear combinations

of vectors $z_1, z_2, \ldots, z_k$ is a subspace of $\mathbb{K}^n$ given by

$$\operatorname{span}\{z_1, z_2, \ldots, z_k\} = \{\alpha_1 z_1 + \alpha_2 z_2 + \ldots + \alpha_k z_k \ : \ \alpha_1, \alpha_2, \ldots, \alpha_k \in \mathbb{K}\}.$$

Subspaces $\mathcal{X}$ and $\mathcal{Y}$ are orthogonal (denoted $\mathcal{X} \perp \mathcal{Y}$) if $\langle x \,|\, y \rangle = 0$ for all $x \in \mathcal{X},\ y \in \mathcal{Y}$. Subspace containing all vectors orthogonal to every vector in $\mathcal{X}$ is denoted with $\mathcal{X}^\perp$.

Matrices and matrix norms. We will denote with $\mathbb{R}^{m\times n}$ (resp. $\mathbb{C}^{m\times n}$) the set of all real (resp. complex) matrices with m rows and n columns; matrices themselves will be denoted with uppercase Latin letters ($A, B, C, \ldots$). Element in i-th row and j-th column is $A_{i,j}$ or $A(i,j)$, with exceptions denoted as in $A = [\alpha_{i,j}]_{i,j}$. We will use MATLAB notation for submatrices: $A(r_1 : r_2, s_1 : s_2)$ is the submatrix of A determined by rows $r_1, r_1 + 1, \ldots, r_2$ and columns $s_1, s_1 + 1, \ldots, s_2$.

If $a_1, a_2, \ldots, a_n$ are columns of matrix A, we will write $A = [a_1\ a_2\ \ldots\ a_n]$. The zero matrix and the identity matrix will be represented by 0 and I, respectively; an index may be used in order to emphasize the dimension: $0_{m,n}$, I_n. Diagonal matrix with elements $\delta_1, \delta_2, \ldots, \delta_n$ on the diagonal will be denoted with $\operatorname{diag}(\delta_1, \delta_2, \ldots, \delta_n)$.

Matrix transposed to A is denoted with $A^\star$ if A is defined over the field $\mathbb{R}$; if the field $\mathbb{C}$ is concerned, the same notation is used for a matrix that is both conjugated and transposed to A. Kernel and image of matrix $A \in \mathbb{K}^{m\times n}$ are subspaces respectively defined by

$$\begin{aligned} \operatorname{Ker}(A) &:= \{x \in \mathbb{K}^n \ : \ Ax = 0\} \leqslant \mathbb{K}^n \quad \text{and} \\ \operatorname{Im}(A) &:= \{Ax \ : \ x \in \mathbb{K}^n\} \leqslant \mathbb{K}^m \ . \end{aligned}$$

Square matrix $A \in \mathbb{K}^{n\times n}$ is normal if $AA^\star = A^\star A$, unitary if $AA^\star = A^\star A = I$, Hermitian if $A^\star = A$, and antihermitian if $A^\star = -A$. Real unitary matrices are also called orthogonal, and real (anti)Hermitian are also called (anti)symmetric. Hermitian matrix is positive definite if for all $x \in \mathbb{K} \setminus \{0\}$ it holds that $x^\star Ax > 0$; if only weaker inequality $x^\star Ax \geq 0$ is valid then A is positive semidefinite. Definition of negative (semi)definite matrices is analogous. Hermitian matrices not belonging to any of these categories are called indefinite. Matrix $A \in \mathbb{K}^{n\times k}$ is orthonormal if $A^\star A = I_k$.

A projector is a linear transformation from $\mathbb{K}^n$ to itself which is idempotent: if P is its matrix notation, then $P^2 = P$. Every element x of $\mathbb{K}^n$ can be uniquely decomposed as a sum of an element belonging to $\operatorname{Im}(P)$ and an element belonging to $\operatorname{Ker}(P)$: $x = Px + (I - P)x$. Thus $\mathbb{K}^n$ is a direct sum of $\operatorname{Im}(P)$ and $\operatorname{Ker}(P)$, which is denoted by

$$\mathbb{K}^n = \operatorname{Im}(P) \oplus \operatorname{Ker}(P) \ .$$

Conversely, if $\mathcal{X}$ and $\mathcal{Y}$ are subspaces of $\mathbb{K}^n$ such that $\mathbb{K}^n = \mathcal{X} \oplus \mathcal{Y}$, then there is a unique projector P such that $\operatorname{Im}(P) = \mathcal{X}$ and $\operatorname{Ker}(P) = \mathcal{Y}$.

If a projector P is such that $\operatorname{Im}(P) \perp \operatorname{Ker}(P)$, then the matrix P is Hermitian; the converse also holds. Such projectors are called orthogonal and can be written in form $P = QQ^\star$, where Q is any (orthonormal) matrix whose columns form an orthonormal

basis for $\mathrm{Im}(P)$. Note that $P_\perp = I - QQ^\star$ is also an orthogonal projector whose image is $\mathrm{Ker}(P)$; thus any x is uniquely represented with $x = Px + P_\perp x$.

Matrix determinant is denoted by $\det(A)$, and matrix trace is defined by

$$\mathrm{tr}(A) := \sum_{i=1}^{n} A_{i,\,i} \ .$$

On the vector space of matrices we consider two special classes of norms: a norm $|||\cdot|||$ defined on $\mathbb{K}^{m\times n}$ is an operator norm if there exist vector norms $\|\cdot\|_p$ on $\mathbb{K}^m$ and $\|\cdot\|_q$ on $\mathbb{K}^n$ such that

$$|||A||| = \max\{\|Ax\|_p \ : \ x \in \mathbb{K}^n, \ \|x\|_q = 1\} \ .$$

Specially,

$$\|A\| := \max\{\|Ax\| \ : \ x \in \mathbb{K}^n, \ \|x\| = 1\}$$

defines what is called a spectral or a 2-norm. This norm belongs to the class of consistent (sub-multiplicative) norms, i.e. for all matrices A and B it holds that $\|AB\| \leq \|A\| \cdot \|B\|$. The Frobenius norm

$$\|A\|_F := \sqrt{\mathrm{tr}(A^\star A)} = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} |A_{i,\,j}|^2}$$

is an example of a consistent non-operator norm. Both spectral and Frobenius norms are unitarily invariant, meaning that

$$\|UAV\| = \|A\|, \quad \|UAV\|_F = \|A\|_F \ ,$$

for all unitary matrices $U \in \mathbb{K}^m$ i $V \in \mathbb{K}^n$.

1.2 Eigenvalues, eigenvectors and eigenspaces

Study of the eigenvalue decomposition problem is the central topic to this work. This section gives an overview of the basic well-known properties that make our toolbox for analysis of the more complex algorithms and their features. For simplicity of exposition, we consider only complex matrices.

Eigenvalues and eigenvectors. A complex number λ is an eigenvalue of the matrix $A \in \mathbb{C}^{n\times n}$ if there exists a non-zero vector $u \in \mathbb{C}^n$ such that $Au = \lambda u$; the vector u is a (right) eigenvector associated to the eigenvalue λ and (λ, u) is called an eigenpair of A. Equivalently, λ is an eigenvalue if and only if the characteristic polynomial of the matrix A

$$\kappa_A(\xi) := \det(A - \xi I)$$

has λ as its root. Thus every matrix in $\mathbb{C}^{n\times n}$ has n eigenvalues.

Algebraic multiplicity of λ is defined as multiplicity of the root λ in polynomial κ_A, while the geometric multiplicity of λ is the dimension of its eigenspace

$$\mathcal{P}_\lambda := \{u \in \mathbb{C}^n \ : \ Au = \lambda u\} = \mathrm{Ker}(A - \lambda I) \ .$$

Algebraic multiplicity of any eigenvalue is greater than or equal to its geometric multiplicity. If equality holds for some eigenvalue λ, such λ is said to be semi-simple. Otherwise, λ is called defective. An eigenvalue of algebraic multiplicity equal to one is referred to as a simple eigenvalue.

Multiset containing all eigenvalues of the matrix A is called the spectrum of A and will be denoted with $\Lambda(A)$. We say that the matrix B is similar to the matrix A if there exists a non-singular matrix X such that $B = XAX^{-1}$. Spectra of two similar matrices coincide.

If $v \in \mathbb{C}^n$ is a non-zero vector such that $v^\star A = \mu v^\star$, then v is called a left eigenvector associated to μ; it is easy to see that μ has to be an eigenvalue of A. Left eigenvector of an eigenvalue is perpendicular to the right eigenvector of any other eigenvalue.

Invariant subspaces. A subspace $\mathcal{X}$ of $\mathbb{C}^n$ is A-invariant if $A\mathcal{X} \subseteq \mathcal{X}$. All of the eigenspaces $\mathcal{P}_\lambda$ are A-invariant. Conversely, if X is a basis of an A-invariant subspace $\mathcal{X}$, then there is a unique matrix L_1 such that $AX = XL_1$. Due to this similarity with one-dimensional case, we sometimes refer to (L_1, X) as a (right) eigenpair of A. Furthermore, for each eigenpair (λ, v) of L_1, (λ, Xv) is an eigenpair of A – thus there is a unique mapping which maps invariant subspaces to subsets of $\Lambda(A)$. When X is orthonormal and Y chosen so that $[X \ \ Y]$ is unitary, then there are matrices H and L_2 such that

$$A = [X \ \ Y] \begin{bmatrix} L_1 & H \\ 0 & L_2 \end{bmatrix} [X \ \ Y]^\star.$$

Subspace $\mathcal{Y}$ spanned by columns of Y is $A^\star$-invariant and is called a left A-invariant subspace; it holds that $Y^\star A = L_2 Y^\star$. Thus (L_2, Y) is a generalization of a left eigenpair. Note that $\Lambda(L_1) \cup \Lambda(L_2) = \Lambda(A)$; if $\Lambda(L_1) \cap \Lambda(L_2) = \emptyset$ then $\mathcal{X}$ is referred to as a simple A-invariant subspace. For each simple invariant subspace the is a corresponding left invariant subspace, as the following theorem states.

Theorem 1.1. *[70, Theorem V.1.5] Let X_1 be an orthogonal basis of a simple invariant subspace $\mathcal{X}_1$ and let $[X_1 \ \ Y_2]$ be unitary. Then there are matrices X_2 and Y_1 such that $[X_1 \ \ X_2]^{-1} = [Y_1 \ \ Y_2]^\star$ and*

$$A = [X_1 \ \ X_2] \begin{bmatrix} L_1 & 0 \\ 0 & L_2 \end{bmatrix} [Y_1 \ \ Y_2]^\star.$$

Thus $\mathcal{X}_2 = \mathrm{Im}(X_2)$ is a complementary A-invariant subspace of $\mathcal{X}_1$. Subspaces $\mathcal{Y}_1 = \mathrm{Im}(Y_1) = \mathcal{X}_2^\perp$ and $\mathcal{Y}_2 = \mathrm{Im}(Y_2) = \mathcal{X}_1^\perp$ are left invariant subspaces corresponding to $\mathcal{X}_1$ and $\mathcal{X}_2$ respectively.

An inductive argument shows that if $\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(p)}$ are all distinct elements of $\Lambda(A)$, then there are matrices $X = [X_1 \quad X_2 \quad \ldots \quad X_p]$, $Y = [Y_1 \quad Y_2 \quad \ldots \quad Y_p]$ and $L_1, \ldots, L_p$ such that $X^{-1} = Y^\star$ and

$$A = X_1 L_1 Y_1^\star + X_2 L_2 Y_2^\star + \ldots + X_p L_p Y_p^\star. \tag{1.1}$$

The spectrum of each L_i consists only of the eigenvalue $\lambda^{(i)}$.

Eigenvalue decomposition. The eigenvalue decomposition of the matrix A is its factorization in the form

$$A = S\Lambda S^{-1}\,, \tag{1.2}$$

where $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ has all elements of the spectrum on its diagonal and S is a non-singular matrix whose columns are eigenvectors associated with $\lambda_1, \lambda_2, \ldots, \lambda_n$, respectfully. Likewise, every row of S^{-1} contains coordinates of a single left eigenvector.

There are matrices that have no eigenvalue decomposition; those that do have are said to be diagonalizable. Two important properties of diagonalizable matrices are stated in the following Theorem.

Theorem 1.2. *The matrix A is diagonalizable if and only if all eigenvalues of A are semi-simple. The set of all diagonalizable matrices is a dense subset in the set of all matrices (using e.g. the spectral norm as a metric).*

If A is a normal matrix, then A is diagonalizable and its S can be chosen as a unitary matrix. The opposite claim is also valid: if S in (1.2) is unitary, then A is normal. In that case,

$$\|A\| = \max_{i=1,2,\ldots,n} |\lambda_i|\,.$$

If A is Hermitian, all of its eigenvalues are real and if A is symmetric, the eigenvectors can be chosen so that their coordinates are real numbers. Eigenvalues of a Hermitian matrix satisfy the Courant–Fischer minimax property.

Theorem 1.3 (Courant–Fischer minimax)**.** *Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_n$. Then*

$$\lambda_j = \max_{\dim \mathcal{X} = j} \min_{\substack{x \in \mathcal{X} \\ \|x\|=1}} x^\star A x, \tag{1.3}$$

where the maximum is taken over all subspaces $\mathcal{X} \leqslant \mathbb{C}^n$ of dimension j.

For positive definite matrices it furthermore holds that $\lambda_i > 0$ for all $i = 1, 2, \ldots, n$. Eigenvalues of positive semidefinite matrices are greater than or equal to zero and for such matrices the square root $A^{1/2} := S\Lambda^{1/2}S^{-1}$ is well defined (note: $A = A^{1/2}A^{1/2}$). Here $\Lambda^{1/2} = \operatorname{diag}(\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_n})$.

If a matrix does not have an eigenvalue decomposition, there are two other matrix factorizations that also reveal the eigenvalues: the Jordan form and the Schur factorization.

Schur factorization. This factorization implies that every matrix is unitarily similar to an upper triangular matrix. Since the unitary similarities on a matrix can be computed in a numerically reliable way, finding the eigenvalues of a small non-symmetric matrix A is usually done by computing its Schur factorization.

Theorem 1.4 (Schur factorization)**.** *Let $A \in \mathbb{C}^{n\times n}$. Then there exists an unitary matrix $Q \in \mathbb{C}^{n\times n}$ and an upper triangular matrix $R \in \mathbb{C}^{n\times n}$ such that $A = QRQ^\star$. This is called the* Schur factorization *of A.*
The diagonal elements of R are the eigenvalues of A. The ordering of the eigenvalues is arbitrary and can be rearranged to fit a given requirement. When A is a real matrix, Q and R can be made real as well if R is allowed to have 2×2 diagonal blocks. Each of these blocks corresponds to a pair of mutually conjugate complex eigenvalues of A.

Note that the first column of Q is an eigenvector of A associated with the eigenvalue $R_{1,1}$ and that for each k, first k columns of Q span a subspace that is A-invariant. When A is a normal matrix, it is easily seen that R is diagonal and in that case the Schur factorization is at the same time the eigenvalue decomposition.

Jordan form. Given a matrix A, all properties of its eigenvalues and eigenspaces can be revealed by computing the Jordan form of A.

Theorem 1.5 (Jordan form)**.** *Let $A \in \mathbb{C}^{n\times n}$. Then there exists a factorization $A = TJT^{-1}$ such that T is a non-singular matrix and J is the* Jordan form *of A. The Jordan form is a is a block-diagonal matrix of Jordan blocks. If we denote with $\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(p)}$ all distinct members of spectrum of A, then*

$$J = \begin{bmatrix} J_1 & & & \\ & J_2 & & \\ & & \ddots & \\ & & & J_p \end{bmatrix}, \; J_i = \begin{bmatrix} J_{i,1} & & & \\ & J_{i,2} & & \\ & & \ddots & \\ & & & J_{i,p_i} \end{bmatrix}, \; J_{i,j} = \begin{bmatrix} \lambda^{(i)} & 1 & & \\ & \lambda^{(i)} & 1 & \\ & & \ddots & \ddots \\ & & & \lambda^{(i)} \end{bmatrix},$$

where blocks $J_{i,j}$ may have various sizes for the same value of i; the largest one is called the index of eigenvalue $\lambda^{(i)}$ and is denoted with $\iota(\lambda^{(i)})$. The matrix J is unique up to ordering of its blocks. The number p_i of blocks in which the eigenvalue $\lambda^{(i)}$ occurs is equal to the geometric multiplicity of $\lambda^{(i)}$. The total number of appearances of $\lambda^{(i)}$ as the diagonal element of J is equal to the algebraic multiplicity of $\lambda^{(i)}$ – thus, if all of the eigenvalues are semi-simple, then the Jordan form of such a matrix is its eigenvalue decomposition as well and the matrix is diagonalizable.

Unlike the Schur form, the usage of Jordan form is limited to matrix theory since the factorization of non-diagonalizable matrices is very sensitive to perturbations. For an example, see [68, Example 1.7], and for an in-depth study see [31] and the thesis of Demmel [21]. The Jordan form will be used here to define a spectral projector.

Spectral projector. Let us accentuate the block column structure of factor T in the Jordan form $A = TJT^{-1}$: $T = [T_1 \ T_2 \ \dots \ T_p]$, where the number of columns in T_i is equal to the size of block J_i in the matrix J. Furthermore, let $J^{(i)}$ denote the matrix obtained from J by replacing J_i with the identity matrix and all the other blocks with zero matrices. Then $P^{(i)} := TJ^{(i)}T^{-1}$ is a projector such that:

- $\mathrm{Im}(P^{(i)})$ is spanned by columns of T_i.
- $P^{(i)}P^{(j)} = P^{(j)}P^{(i)} = 0$, for each $i \neq j$.
- $AP^{(i)} = P^{(i)}A$, for each i.
- Each $x \in \mathbb{C}^n$ can be represented as $x = P^{(1)}x + P^{(2)}x + \ldots + P^{(p)}x$.

We call $P^{(i)}$ a spectral projector associated with eigenvalue $\lambda^{(i)}$. If A is diagonalizable, then each $P^{(i)}x$ is an eigenvector of A associated with $\lambda^{(i)}$ and thus every vector in $\mathbb{C}^n$ is uniquely represented as a sum of eigenvectors of A. Diagonalizable matrix A itself admits this decomposition:

$$A = \lambda^{(1)}P^{(1)} + \lambda^{(2)}P^{(2)} + \ldots + \lambda^{(p)}P^{(p)} \ .$$

Using notation of (1.1), it can be shown that $P^{(i)} = X_i Y_i^\star$ holds even in the general case. Another formula for the spectral projector is given by the integral of the resolvent $R(\xi) = (\xi I - A)^{-1}$ of the matrix A:

$$P^{(i)} = \frac{1}{2\pi i}\int_{\Gamma_i} (\xi I - A)^{-1} d\xi,$$

where Γ_i is a closed Jordan curve in the complex plane that encloses $\lambda^{(i)}$ and no other eigenvalue of A.

Singular value decomposition. The eigenvalue decomposition exists only for certain square matrices. There is an important related decomposition that exists for all matrices.

Theorem 1.6 (Singular value decomposition – SVD)**.** *Let $A \in \mathbb{C}^{m\times n}$, $m \geq n$. Then A can be factorized as*

$$A = U \begin{bmatrix} \Sigma \\ 0_{m-n,\,n} \end{bmatrix} V^\star,$$

where $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$ is a diagonal matrix with real diagonal elements such that $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$, and matrices $U \in \mathbb{C}^{m\times m}$ and $V \in \mathbb{C}^{n\times n}$ are unitary.

The diagonal elements of Σ are called the *singular values* of A, the columns of V are the *right singular vectors*, and the columns of U are the *left singular vectors*. When A is a real matrix, the matrices U and V are orthogonal; when A is square and Hermitian positive definite, the SVD of A coincides with the eigenvalue decomposition. Singular values of the normal matrices are equal to the absolute values of the eigenvalues: $\sigma_i = |\lambda_i|$.

Many matrix properties can be expressed using the singular values, for example:

$$\|A\| = \sigma_1, \quad \|A\|_F = \sqrt{\sigma_1^2 + \sigma_2^2 + \ldots + \sigma_n^2}.$$

The rank r of the matrix A is equal to the number of positive singular values. $\mathrm{Im}(A)$ is spanned by the first r columns of the matrix U, and $\mathrm{Ker}(A)$ is spanned by the last $n - r$ columns of the matrix V. Every matrix is a sum of rank-one matrices:

$$A = \sum_{i=1}^{r} \sigma_i \cdot u_i v_i^\star,$$

where u_i and v_i are respectively the left and the right singular vector associated with σ_i (i.e. the i-th column of U and V). The singular value decomposition is also used for computing the closest approximation of rank k to the matrix A: setting $A_k = \sum_{i=1}^{k} \sigma_i \cdot u_i v_i^\star$ for some $k < r$, we have

$$\|A - A_k\| = \min\{\|A - B\| \ : \ \mathrm{rank}(B) = k\}.$$

1.3 Applications of the eigenvalue problem

Computing eigenvalues of matrices arises in wide variety of real world applications, in particular in modeling physical phenomena. Vibration analysis, computing molecular orbitals, quantum chemistry, Markov chain techniques and control theory are only some of the application areas. To motivate the study of the large scale eigenvalue problem, we illustrate how two very different scientific problems reduce to computing the eigenvalues and eigenvectors of a large matrix.

1.3.1 Electromagnetic fields in cavities

We are given a region Ω, which is a closed cavity with a perfectly conducting boundary $\partial\Omega$. The inside volume of the cavity is filled with a linear, isotropic and non-dispersive material. An example of such a region may be the interior of a particle accelerator, in which standing waves are used to produce the high voltage radio–frequency fields for acceleration of the particles [1]. The electromagnetic field inside the cavity is described by the Maxwell equations:

$$\begin{aligned} \nabla \times H(x,t) &= \frac{\epsilon}{c} \cdot \partial_t E(x,t) \\ \nabla \times E(x,t) &= -\frac{\mu}{c} \cdot \partial_t H(x,t) \\ \nabla \cdot E(x,t) &= 0 \\ \nabla \cdot H(x,t) &= 0. \end{aligned}$$

Here H is the magnetic field, E is the electric field, c is the speed of light, ϵ and μ are respectively the permittivity and the permeability of the material inside the cavity, while x and t denote the space and the time variable. Under the assumption of the harmonic

time dependence, $E(x,t) = \hat{E}(x)e^{i\omega t}$ and $H(x,t) = i\hat{H}(x)e^{i\omega t}$, and by eliminating $\hat{H}$, we obtain the following equations:

$$\left.\begin{array}{rcl} \nabla \times (\nabla \times \hat{E}(x)) & = & \omega^2 \frac{\epsilon\mu}{c^2} \hat{E}(x) \\ \nabla \cdot \hat{E}(x) & = & 0 \end{array}\right\} \quad \text{for } x \in \Omega. \tag{1.4}$$

The boundary conditions for the perfectly conducting material imply that the tangential component of the electric field and the normal component of the magnetic field are equal to zero, leading to

$$n \times \hat{E} = 0, \quad \text{for } x \in \partial\Omega. \tag{1.5}$$

We have denoted by n the outward normal vector to the boundary. Equations (1.4) have a solution only for certain values of ω, which are called the eigenfrequencies or the electromagnetic oscillations of the resonator. Associated solutions $\hat{E}$ are called the eigenmodes. In the design of the particle accelerator, of interest are several (ten to twenty) of the smallest eigenfrequencies and eigenmodes, since they are used as the accelerating fields.

The exact analytical solution to (1.4), (1.5) can be computed only for very few domains Ω, such as a rectangular box. In the general case, a numerical algorithm must be used in order to find the approximate solution. Several approaches can be taken here; we follow the construction of the finite elements method as in [1].

First, a variational formulation of the system is made. Let $L_2(\Omega)^3$ denote the space of square integrable functions $u : \Omega \mapsto \mathbb{R}^3$, and let

$$\langle u \,|\, v \rangle := \int_\Omega u(x) \cdot v(x) dx = \sum_{i=1}^{3} \int_\Omega u_i(x) v_i(x) dx$$

denote the standard inner product of $u(x) = (u_1(x), u_2(x), u_3(x))$ and $v(x) = (v_1(x), v_2(x), v_3(x))$. The Green's formula

$$\langle \nabla \times u \,|\, v \rangle - \langle v \,|\, \nabla \times u \rangle = \int_{\partial\Omega} (v \times n) \cdot u ds$$

holds for all $u \in H(\mathrm{curl}, \Omega)$ and all $v \in \mathcal{D}(\bar{\Omega})^3$. The space $\mathcal{D}(\bar{\Omega})^3$ contains all infinitely differentiable functions $u : \mathbb{R}^3 \to \mathbb{R}^3$ with compact support in Ω, restricted to Ω, and

$$H(\mathrm{curl}, \Omega) = \{u \in L_2(\Omega)^3 \ : \ \nabla \times u \in L_2(\Omega)^3\}.$$

Next, multiply the first equation of (1.4) by some test-function Ψ, and apply the Green's formula to obtain

$$\begin{array}{rcl} \lambda \int_\Omega \hat{E} \cdot \Psi dx & = & \int_\Omega (\nabla \times (\nabla \times \hat{E})) \cdot \Psi dx \\ & = & \int_\Omega (\nabla \times \hat{E}) \cdot (\nabla \times \Psi) dx + \int_{\partial\Omega} (\nabla \times \Psi) \cdot (n \times \Psi) ds. \end{array} \tag{1.6}$$

To implement the second equation and the boundary conditions, we restrict the function spaces for $\hat{E}$ and Ψ. First, let the space $H_0(\mathrm{curl}, \Omega)$ denote the closure of $\mathcal{D}(\Omega)^3$ in

$H(\mathrm{curl}, \Omega)$. It can be shown that functions $u \in H_0(\mathrm{curl}, \Omega)$ satisfy the boundary condition $n \times u = 0$ on $\partial\Omega$. Using test–functions $\Psi \in H_0(\mathrm{curl}, \Omega)$ makes the second term on the right–hand side of (1.6) vanish. Furthermore, let

$$W_0(\Omega) = \{u \in H_0(\mathrm{curl}, \Omega) \ : \ \nabla \cdot u = 0\}.$$

Imposing conditions $u \in W_0(\Omega)$, $\Psi \in H_0(\mathrm{curl}, \Omega)$ on (1.6), we obtain the variational formulation of the initial system: find $\lambda \in \mathbb{R}$ and $\hat{E} \in W_0(\Omega)$ such that

$$\langle \nabla \times \hat{E} \,|\, \nabla \times \Psi \rangle = \lambda \langle \hat{E} \,|\, \Psi \rangle, \qquad \text{for all } \Psi \in H_0(\mathrm{curl}, \Omega). \tag{1.7}$$

The solution $\hat{E}$ satisfying (1.7) is called *the weak solution*. In order to compute the approximate solution to this equation, finite dimensional subspaces of $W_0(\Omega)$ and $H_0(\mathrm{curl}, \Omega)$ have to be constructed. It turns out that it is difficult to implement the divergence–free condition $\nabla \cdot u = 0$; an alternative (but equivalent) variational formulation is:

$$\left.\begin{aligned} \langle \nabla \times \hat{E} \,|\, \nabla \times \Psi \rangle + \langle \nabla p \,|\, \Psi \rangle = \lambda \langle \hat{E} \,|\, \Psi \rangle, &\quad \text{for all } \Psi \in H_0(\mathrm{curl}, \Omega); \\ \langle \hat{E} \,|\, \nabla \varphi \rangle = 0, &\quad \text{for all } \varphi \in H_0^1(\Omega). \end{aligned}\right\} \tag{1.8}$$

This is a *mixed formulation* in which the Lagrange multipliers are used to enforce a divergent–free solution $\hat{E}$. We now have to find $\lambda \in \mathbb{R}$, $\hat{E} \in H_0(\mathrm{curl}, \Omega)$ and $p \in H_0^1(\Omega)$ so that (1.8) is satisfied. The space $H_0^1(\Omega)$ consists of scalar functions:

$$H_0^1(\Omega) = \{q \in L_2(\Omega) \ : \ \nabla q \in L_2(\Omega)^3 \text{ and } q = 0 \text{ at } \partial\Omega\}.$$

The finite element method is used to find the approximation for the weak solution: at first, the domain Ω is subdivided into a number of smaller elements by some triangulation method. The triangulation is parametrized by the diameter h of the largest element. To keep the exposition simple, assume that the elements are rectangular boxes with edges aligned to the coordinate axes e_1, e_2 and e_3. For $i = 1, 2, 3$, let S_i denote the set of all edges parallel to e_i. In this particular application, for each edge j of the triangulation, we define a *basis function* $\Psi_j^{(h)}$. If the edge j is parallel to e_i, let

$$\Psi_j^{(h)}(x) = \begin{cases} e_i, & \text{if } x \text{ belongs to edge } j; \\ 0, & \text{if } x \text{ belongs to edge } S_i \setminus j. \end{cases}$$

The function $\Psi_j^{(h)}$ is expanded by linearity to $x \in \Omega$. The approximate solution $\tilde{E}$ is found in the span of the basis functions:

$$\tilde{E}(x) = \sum_{j=1}^{n_e} E_j \Psi_j^{(h)}(x), \tag{1.9}$$

where n_e is the total number of edges in the triangulation and E_j is the tangential electric field along the j-th edge. We say that the method uses the *edge elements*. Note that the functions $\Psi_j^{(h)}$ are constructed so that their tangential components are continuous across

element borders. This provides that $\Psi_j^{(h)}, \tilde{E} \in H_0(\mathrm{curl}, \Omega)$. Our goal is to compute the coefficients E_j.

For the second equation in (1.8), we use the more common *face elements*. To each vertex j, except for those on $\partial\Omega$, assign a basis function

$$\varphi_j^{(h)}(x) = \begin{cases} 1, & \text{if } x \text{ is the vertex } j; \\ 0, & \text{if } x \text{ is any other vertex;} \end{cases}$$

and expand it by linearity to all $x \in \Omega$. The approximate solution $\tilde{p}$ is once again found in the span of the basis functions:

$$\tilde{p}(x) = \sum_{j=1}^{n_v} p_j \varphi_j^{(h)}(x), \tag{1.10}$$

where n_v is the total number of vertices that do not belong to $\partial\Omega$. Obviously, $\varphi_j^{(h)}, \tilde{p} \in H_0^1(\Omega)$. We need to compute the coefficients p_j.

Insert the expressions (1.9) and (1.10) into (1.8). The requirement that (1.8) has to be fulfilled for all test–functions $\Psi \in H_0(\mathrm{curl}, \Omega)$ and $\varphi \in H_0^1(\Omega)$ is weakened; we ask that it has to hold only for the finite dimensional subsets $\Psi \in \{\Psi_1^{(h)}, \ldots, \Psi_{n_e}^{(h)}\}$ and $\varphi \in \{\varphi_1^{(h)}, \ldots, \varphi_{n_v}^{(h)}\}$. This leads to

$$\sum_{j=1}^{n_e} \langle \nabla \times \Psi_i^{(h)} \,|\, \nabla \times \Psi_j^{(h)} \rangle E_j + \sum_{j=1}^{n_v} \langle \Psi_i^{(h)} \,|\, \nabla \varphi_j^{(h)} \rangle p_j = \lambda \sum_{j=1}^{n_e} \langle \Psi_i^{(h)} \,|\, \Psi_j^{(h)} \rangle E_j$$

and

$$\sum_{j=1}^{n_e} \langle \nabla \varphi_k^{(h)} \,|\, \Psi_j^{(h)} \rangle E_j = 0,$$

for all $i = 1, \ldots, n_e$ and all $k = 1, \ldots, n_v$. Setting

$$\begin{aligned} A_{i,j} &= \langle \nabla \times \Psi_i^{(h)} \,|\, \nabla \times \Psi_j^{(h)} \rangle, \\ C_{i,j} &= \langle \Psi_i^{(h)} \,|\, \nabla \varphi_j^{(h)} \rangle, \\ M_{i,j} &= \langle \Psi_i^{(h)} \,|\, \Psi_j^{(h)} \rangle, \end{aligned}$$

we get the matrix formulation of the problem:

$$\begin{bmatrix} A & C \\ C^\star & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

Here $x = (E_1, \ldots, E_{n_e})$ and $y = (p_1, \ldots, p_{n_v})$. We are interested to compute x and the positive eigenvalues λ only; using the properties of the edge elements, it can be shown that all such can be found by solving the generalized eigenvalue problem

$$Ax = \lambda M x.$$

The matrices A and M are symmetric and positive definite, so this problem is equivalent to solving the eigenvalue problem $M^{-1}Ax = \lambda x$ or $M^{-1/2}AM^{-1/2}x = \lambda x$. The eigenvectors x

automatically satisfy $C^\star x = 0$; however, this condition has to be imposed in the numerical algorithms to ensure convergence. As we have mentioned before, only the smallest few eigenvalues are of interest. The order of A and M is equal to $n_e = \mathcal{O}(h^3)$, which grows rapidly when finer approximations for the electromagnetic field are computed.

1.3.2 The PageRank algorithm

For an average query to a web–search engine, typically millions and millions of web–pages exist containing the associated terms. The search engine needs to select only a few the most relevant of all these pages, and present it to the user. PageRank is an algorithm for measuring the relevance of web–pages. It was introduced by Google [14], and was a decisive element that provided the superior quality of the search results. The source for our description is [27].

The PageRank algorithm is based on the assumption that the importance of a web–page is related to the number of links pointing to the page. Suppose that the web–pages are indexed from 1 to n; let IN_i denote the set of all pages having links to the page i, and let OUT_i denote the set of all pages j such that the page i contains a link to j. Then the rank r_i of the page i is computed according to the following formula:

$$r_i = \sum_{j \in IN_i} \frac{r_j}{|OUT_j|}. \tag{1.11}$$

We have denoted with $|OUT_j|$ the number of elements in the set OUT_j. The reasoning behind (1.11) is the following: page i is important if another important page j has a link toward the page i. Thus the rank r_i is related with the rank of pages from IN_i. On the other hand, if there are many links on the page j, the link towards i is less likely to be followed by the visitors of the page j. Thus the contribution of the page j to the rank of the page i is weighted by the number of outgoing links $|OUT_j|$.

Our goal is to find a vector $r = (r_1, \ldots, r_n)$, where $r_i > 0$ satisfy (1.11). To obtain a matrix formulation of the problem, define $Q \in \mathbb{R}^{n \times n}$ with

$$Q_{i,j} = \begin{cases} \frac{1}{|OUT_j|}, & \text{if } i \in OUT_j; \\ 0, & \text{otherwise.} \end{cases}$$

Then we want to find the eigenvector r associated with the eigenvalue $\lambda = 1$ for the problem

$$Qr = \lambda r.$$

It is not clear that 1 is an eigenvalue of Q; additional steps have to be taken to ensure this.

Eigenvector r can be viewed as a stationary probability distribution for a certain Markov chain. If a web–surfer arrives at the page j, suppose that the probability of following a link to the page $i \in OUT_j$ is equal to $Q_{i,j}$. Let $p^{(k)} = (p_1^{(k)}, \ldots, p_n^{(k)})$ be the vector such that $p_i^{(k)}$ is the probability that page i is the k-th page the surfer visits. Then

$$p^{(k+1)} = Qp^{(k)}.$$

The vector $p^{(0)}$ has a single entry equal to 1, and all the others equal to zero. If a stationary distribution $p = \lim_k p^{(k)}$ exists, it is once again an eigenvector of the matrix Q associated with the eigenvalue 1. An unrealistic element in our random walk model is that the user stops surfing when a web–page j with no outgoing links is reached; column j of the matrix Q associated with this page contains only zeros. In such an event, we let the user visit any random page, with equal probability. Thus the updated transition matrix is

$$T = Q + \frac{1}{n} e d^\star,$$

where $e = (1, 1, \ldots, 1)$, and $d = (d_1, \ldots, d_n)$ is defined with

$$d_j = \begin{cases} 1, & \text{if } |OUT_j| = 0; \\ 0, & \text{otherwise.} \end{cases}$$

All entries of the matrix T are non-negative, and the sum of the entries in each column is equal to 1. We call such matrices *column–stochastic*. A column–stochastic matrix has to be irreducible in order to have a simple eigenvalue 1 with a non-negative eigenvector. This is a well-known fact from the Perron–Frobenius theory.

Definition 1.7. *A square matrix A is called reducible if there exists a permutation matrix P such that*

$$PAP^\star = \begin{bmatrix} K & L \\ 0 & M \end{bmatrix},$$

for some square matrices K and L. Otherwise, A is called irreducible.

Theorem 1.8. *Let $A \in \mathbb{R}^{n \times n}$ be an irreducible column–stochastic matrix. The largest eigenvalue of A is equal to 1, and the corresponding eigenvector $r = (r_1, \ldots, r_n)$, $\|r\| = 1$ is such that $r_i > 0$ for all $i = 1, \ldots, n$. If all entries of A are strictly positive, then all eigenvalues λ other than 1 satisfy $|\lambda| < 1$.*

The irreducibility implies that, starting from any page i, there exists a series of links that eventually lead to any other page j. Given the topology of links on the entire Internet, this is not likely to be true. To fix the issue, we add artificial links from every page to all the others. This is implemented as a convex combination of the matrix T and a matrix of rank 1:

$$A = \alpha T + (1 - \alpha) \frac{1}{n} e e^\star.$$

Here α is any number such that $0 < \alpha < 1$; the value $(1 - \alpha)$ is interpreted as the probability that the surfer will not follow any of the links found on the current page, but jump to any random page instead (this is called "teleportation"). The value $\alpha = 0.85$ provides good results in practice.

The matrix A is a column–stochastic matrix which is obviously irreducible. The Theorem states that it has a single dominant eigenvalue $\lambda_1 = 1$, and that the associated eigenvector r has positive entries. The problem of determining the relevance of web–pages

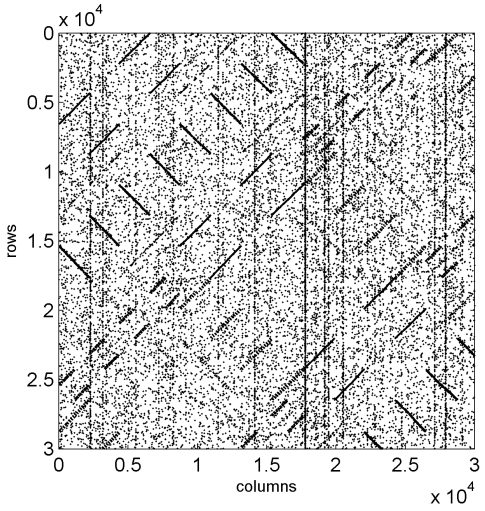


Figure 1.1: Sparsity pattern of the matrix Q in computing the PageRank for the Stanford University web–site (source: `SNAP/web-Stanford` [20]). Only the top-left 30000×30000 submatrix is shown.

using the PageRank algorithm is thus formulated as the problem of computing the dominant eigenvector of a large matrix A. The rank of a web–page i is then defined as the i-th entry of the computed eigenvector.

Given that the order of A, which is equal to the number of web–pages, is measured in billions, the only applicable algorithm for the solution of this problem is the power method (see Section 2.1). This method does not require the matrix A to be explicitly allocated in memory, but only a way of computing $x \mapsto Ax$ for a given vector x. Since

$$Ax = \left(\alpha\left(Q + \frac{1}{n}ed^\star\right) + (1-\alpha)\frac{1}{n}ee^\star\right)x = \alpha Qx + \beta\frac{1}{n}e,$$

where

$$\beta = \alpha d^\star x + (1-\alpha)e^\star x,$$

only the matrix Q has to be stored. This matrix is very sparse, and specialized packed formats are used to reduce the amount of memory needed for its storage. The sparsity pattern for a top-left 30000×30000 submatrix of a sample matrix Q is shown on Figure 1.1. This sample matrix (`SNAP/web-Stanford` from the sparse matrix collection [20]) describes the link topology of the web–pages at Stanford University. The matrix is of order 281903 and has only 2312497 non-zero entries.

Computing or storing the vector d can be avoided as well:

$$\begin{aligned}\beta &= \beta\frac{1}{n}e^\star e = e^\star\left(Ax - \alpha Qx\right)\\ &= e^\star Ax - e^\star(\alpha Qx) = e^\star x - e^\star(\alpha Qx)\\ &= 1 - \left\|\alpha Qx\right\|_1.\end{aligned}$$

We have used the fact that $e^\star A = e^\star$ since A is column–stochastic and assumed that $\|x\|_1 = e^\star x = 1$.

It can be shown that the second largest eigenvalue of the Google matrix A is equal to $\lambda_2 = \alpha$. Since the power methods converges linearly with the ratio equal to $\lambda_2/\lambda_1 = \alpha$, setting $\alpha = 0.85$ will provide sufficiently fast convergence.

1.4 An overview of the perturbation theory

As we have seen, the eigenvalue problems arise from physical models. Entries of matrices are obtained by taking measurements by imprecise instruments or, for example, by computing integrals using approximate numerical formulae. Storing these entries as a collection of floating point numbers in a computer results in further truncation and rounding. These effects imply that instead of computing the eigenvalues of the desired matrix A, we start the computation with some perturbed matrix $A + E$. To correctly interpret the obtained result, we need to relate the computed eigenvalues, eigenvectors and invariant subspaces of $A + E$ to those of A. Perturbation theory will show how this error E in the input data is transferred to the output when the computation is done in the exact arithmetic. It will provide answers to whether the matrix A is so sensitive that a tiny modification to its entries drastically changes its eigenvalues, or whether the eigenvalues are perfectly conditioned and the size of change in the eigenvalues amounts to the size of the perturbation E. This is demonstrated in the following classic example.

Example 1.9. Let $A = A(0)$ and $A + E = A(\epsilon)$, where

$$A(\epsilon) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \epsilon & 0 & 0 & 0 \end{bmatrix}$$

for $\epsilon > 0$. All eigenvalues of A are zero, while eigenvalues of $A + E$ are the fourth roots of ϵ. Thus the perturbation of A of size $\|E\| = \epsilon$ induces a perturbation of size $\epsilon^{1/4}$ in eigenvalues (i.e. $\epsilon^{1/n}$ where n is the dimension of A). For example, if $\epsilon = 10^{-8}$, then eigenvalues of $A + E$ are shifted by 10^{-2} compared to those of A, which is six orders of magnitude more that the input error.

On the other hand, if we set $B = B(0)$ and $B + E = B(10^{-8})$ where

$$B(\epsilon) = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ \epsilon & 0 & 0 & 2 \end{bmatrix},$$

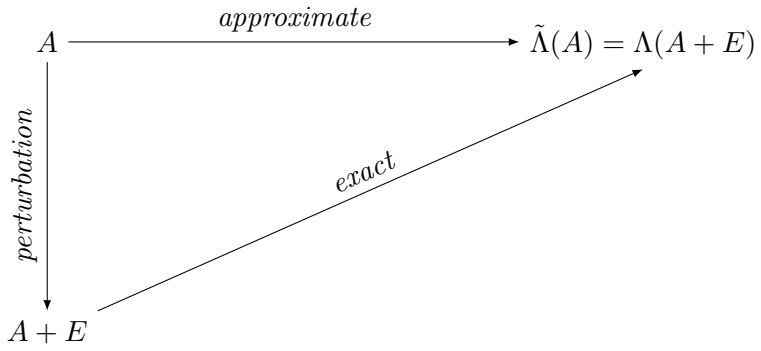


Figure 1.2: The concept of a backward error: the approximate spectrum $\tilde{\Lambda}(A)$ is viewed as an exact spectrum for a perturbed matrix $A + E$.

we can show that

$$\begin{aligned} \Lambda(B) &= \{-2, -1, 1, 2\} \\ \Lambda(B+E) &= \{-2 - \frac{1}{3}\epsilon + \mathcal{O}(\epsilon^2), -1 + \frac{1}{6}\epsilon + \mathcal{O}(\epsilon^2), \\ &\quad 1 - \frac{1}{6}\epsilon + \mathcal{O}(\epsilon^2), 2 + \frac{1}{3}\epsilon + \mathcal{O}(\epsilon^2)\}. \end{aligned}$$

Thus the eigenvalues of $B+E$ are shifted by $\mathcal{O}(\epsilon)$ compared to those of B – the output error is of the same magnitude as the input error.

To emphasize the need for developing the perturbation theory, note that even if the computation is started with the exact matrix A, rounding errors are introduced during the computation due to the usage of the finite precision arithmetic. Also, if the order of the matrix is 5 or greater, its eigenvalues cannot be represented with a finite formula that uses the basic arithmetic operators on the matrix entries, since the eigenvalues are roots of the characteristic polynomial. Therefore, the obtained values $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_k$ are once again only approximations of the exact eigenvalues $\lambda_1, \ldots, \lambda_k$. To determine the quality of the computed values, it is common to express them as the exact eigenvalues of some matrix $A + E$. Numerical analysis of the algorithm used for computation will determine a bound on the size of E, which is in this context called the backward error. The perturbation theory can then be deployed to estimate how far the computed results are from the exact ones.

Proofs for most of perturbation theorems presented here can be found in [70].

1.4.1 Perturbations of eigenvalues

To begin with, we state the basic result on the continuity of matrix eigenvalues. For multisets $S = \{\sigma_1, \ldots, \sigma_n\}$ and $T = \{\tau_1, \ldots, \tau_n\}$ of the same cardinality, we define the

optimum matching distance:

$$md(S,T) = \min_{\pi \in S(n)} \max_{i=1,\ldots,n} |\sigma_i - \tau_{\pi(i)}|,$$

where $S(n)$ is the set of all permutations of $\{1, 2, \ldots, n\}$. Let $\mathbb{C}^n_{sym}$ be the set of all multisets consisting of n complex numbers.

Theorem 1.10 (continuity of eigenvalues)**.** *Denote with $\Lambda : \mathbb{C}^{n\times n} \to \mathbb{C}^n_{sym}$ a transformation that maps a matrix to the multiset of its eigenvalues. With topology induced by the optimum matching distance on $\mathbb{C}^n_{sym}$, Λ is a continuous function.*

This is a qualitative result, which only states that one can choose a region around A so that the eigenvalues of all matrices in the region are as close as desired to the eigenvalues of A. Perturbation theory provides quantitative estimates on how the size of the region effects the distance to the eigenvalues of A. The first result locates the eigenvalues of a perturbed diagonal matrix.

Theorem 1.11 (Gerschgorin)**.** *Let $A \in \mathbb{C}^{n\times n}$ and let $D_i \subseteq \mathbb{C}$ denote a disc of radius $\sum_{j\neq i} |A_{i,j}|$ centered at $A_{i,i}$. The following statements hold:*

(a) $\Lambda(A) \subseteq \bigcup_{i=1}^n D_i$ *;*

(b) if the union of discs $D_{i_1}, D_{i_2}, \ldots, D_{i_q}$ is disjoint with the union of the remaining $n - q$ discs, then $D_{i_1} \cup D_{i_2} \cup \ldots \cup D_{i_q}$ contains exactly q eigenvalues of A.

To expose the hidden power of this theorem, consider matrices B and $B(\epsilon)$ from Example 1.9. If we set

$$X_1 = \begin{bmatrix} 3\epsilon & & & \\ & 1/3 & & \\ & & 1 & \\ & & & 1 \end{bmatrix}, \; X_2 = \begin{bmatrix} \epsilon & & & \\ & 3\epsilon & & \\ & & 1 & \\ & & & 1 \end{bmatrix},$$

then matrices

$$X_1 B(\epsilon) X_1^{-1} = \begin{bmatrix} -2 & 9\epsilon & 0 & 0 \\ 0 & -1 & 1/3 & 0 \\ 0 & 0 & 1 & 1 \\ 1/3 & 0 & 0 & 2 \end{bmatrix}, \; X_2 B(\epsilon) X_2^{-1} = \begin{bmatrix} -2 & 1/3 & 0 & 0 \\ 0 & -1 & 3\epsilon & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 2 \end{bmatrix},$$

are both similar to $B(\epsilon)$ and using Gerschgorin's theorem we see that there are eigenvalues of $B(\epsilon)$ located inside discs of radius 9ϵ and 3ϵ with centers at -2 and -1, respectively.

The following theorem shows that a perturbation of scale $\mathcal{O}(\|E\|^{1/n})$ in eigenvalues of $A + E$ that appeared in the same Example is the largest possible. Thus results for general matrices necessarily have a rather pessimistic bound.

Theorem 1.12 (Ostrowski-Elsner)**.** *For all $A, E \in \mathbb{C}^{n\times n}$ it holds that*

$$md(\Lambda(A), \Lambda(A+E)) \leq (2n-1)\left(\|A\| + \|A+E\|\right)^{1-\frac{1}{n}} \|E\|^{\frac{1}{n}}.$$

Better bounds can be obtained if additional matrix structure is assumed. Note that the Bauer-Fike theorem below uses a different way of measuring distance between spectra of A and $A+E$.

Theorem 1.13 (Bauer-Fike)**.** *Assume that $A \in \mathbb{C}^{n\times n}$ is diagonalizable with spectral decomposition $A = S\Lambda S^{-1}$ and eigenvalues $\lambda_1, \ldots, \lambda_n$. Then for each $\tilde{\lambda} \in \Lambda(A+E)$:*

$$\min_i |\tilde{\lambda} - \lambda_i| \leq \kappa(S) \cdot \|E\|,$$

where $\kappa(S) := \|S\| \cdot \left\|S^{-1}\right\|$ is the condition of the matrix S. Furthermore, if all eigenvalues of A are simple, then

$$\Lambda(A+E) \subseteq \bigcup_{i=1}^{n} D(\lambda_i, n \cdot \frac{\|E\|}{|v_i^\star u_i|}).$$

Here u_i is a right eigenvector and v_i is a left eigenvector associated with λ_i, both having unitary norm. With $D(\xi, r)$ we have denoted the disc of radius r with center ξ.

Previous theorems have considered the sensitivity of the entire spectrum under perturbation of the matrix. However, some eigenvalues may behave more stably than the others, which motivates the development of finer theory demonstrating sensitivity of a single eigenvalue.

Theorem 1.14. *Suppose A is a diagonalizable matrix with a simple eigenvalue λ and corresponding right and left eigenvectors u and v. Then for a sufficiently small perturbation E there exists an eigenvalue $\tilde{\lambda}$ of $A+E$ such that*

$$\tilde{\lambda} = \lambda + \frac{v^\star E u}{v^\star u} + \mathcal{O}(\|E\|^2). \tag{1.12}$$

Note that this theorem also states that simple eigenvalues are differentiable functions of matrix elements. Also, suppose u and v are unit vectors. Then we can rewrite (1.12) as

$$|\tilde{\lambda} - \lambda| \lesssim \frac{\|v\| \cdot \|u\|}{|v^\star u|} \cdot \|E\| = \frac{1}{|v^\star u|} \cdot \|E\|,$$

which clearly exposes the fraction $\frac{1}{|v^\star u|}$ as the condition number of the eigenvalue λ – it is the ratio by which the error in the input ($\|E\|$) is enlarged to produce the error in the output ($|\tilde{\lambda} - \lambda|$). Matrices whose left and right eigenvectors coincide (such as normal matrices) have all of their simple eigenvalues perfectly conditioned. On the other hand, we can construct examples where $v^\star u \to 0$ and the associated eigenvalue being very sensitive to perturbations.

Another useful remark is that

$$\frac{1}{|v^\star u|} = \frac{1}{\cos \measuredangle(u, v)} = \frac{1}{\sin \measuredangle(u, \mathcal{U})},$$

where $\mathcal{U}$ is the complementary invariant subspace of u (see Theorem 1.1). The larger the angle between u and the subspace spanned by the rest of the eigenvectors, the better the condition of associated eigenvalue.

As mentioned before, normal matrices have their eigenvalues perfectly conditioned. We state here another result for such matrices, this time using the Frobenius norm.

Theorem 1.15 (Wielandt-Hoffman)**.** *Suppose A and $A+E$ are both normal matrices with eigenvalues $\lambda_1, \ldots, \lambda_n$ and $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n$ respectively. Then there exists a permutation π such that*

$$\sqrt{\sum_{i=1}^{n} (\tilde{\lambda}_i - \lambda_{\pi(i)})^2} \leq \|E\|_F \,.$$

The proof of this theorem does not suggest how to construct permutation π, which is, on the other hand, obvious for the Hermitian matrices. We will bring some further results from perturbation theory for the Hermitian matrices in the section on approximations from a subspace.

1.4.2 Perturbations of invariant spaces

As we have seen, condition number of a simple eigenvalue depends on the angle between its left and right eigenvectors. On the other hand, perturbation results for eigenvectors and eigenspaces will involve a more complex function of gaps between the eigenvalues and angles among invariant subspaces.

For illustration, suppose that A is a diagonalizable matrix whose eigenvalue λ_1 is simple. Then its eigenvalue decomposition can be written as

$$A = U\Lambda V^\star = [u_1 \;\; u_2 \;\; \ldots \;\; u_n] \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix} [v_1 \;\; v_2 \;\; \ldots \;\; v_n]^\star,$$

where columns of U are the right eigenvectors of unit length, columns of V are the left eigenvectors and it holds that $V^\star U = I$. Suppose that E is a perturbation and

$$(A+E)(u_1+h) = (\lambda_1+\delta)(u_1+h), \tag{1.13}$$

that is, $(\lambda_1+\delta, u_1+h)$ is an eigenvector of $A+E$ expressed as a perturbation of eigenpair (λ_1, u_1) of A. Suppose that the perturbed eigenvector is scaled so that h doesn't have a component in direction of u_1:

$$h = \sum_{i=2}^{n} \xi_i u_i.$$

Then (1.13) can be rewritten as

$$A\sum_{i=2}^{n} \xi_i u_i - \lambda_1 \sum_{i=2}^{n} \xi_i u_i \approx \delta u_1 - E u_1;$$

we have neglected the second order terms Eh and δh. Since $Au_i = \lambda_i u_i$ and $v_j^\star u_i = \delta_{ij}$, premultiplying with $v_j^\star$ yields

$$\xi_j = \frac{1}{\lambda_1 - \lambda_j} v_j^\star E u_1.$$

Thus

$$\tilde{u}_1 = u_1 + h = u_1 + \sum_{i=2}^{n} \frac{1}{\lambda_1 - \lambda_i} u_i v_i^\star E u_1.$$

Finally,

$$\begin{aligned} \sin \measuredangle(\tilde{u}_1, u_1) &\leq \|\tilde{u}_1 - u_1\| = \left\| \sum_{i=2}^{n} \frac{1}{\lambda_1 - \lambda_i} u_i v_i^\star E u_1 \right\| \\ &\leq \sum_{i=2}^{n} \frac{1}{|\lambda_1 - \lambda_i|} \|v_i\| \, \|E\| = \sum_{i=2}^{n} \frac{\|E\|}{|\lambda_1 - \lambda_i| \cos \measuredangle(u_i, v_i)} \\ &\leq \frac{n \, \|E\|}{\min_{i=2\ldots n} |\lambda_1 - \lambda_i| \cos \measuredangle(u_i, v_i)}. \end{aligned} \tag{1.14}$$

We have used the fact that $\|v_i\| = \frac{1}{\cos \measuredangle(u_i, v_i)}$. Note how the perturbation of the eigenvector associated with λ_1 depends on the gap (the distance of λ_1 to the other nearest eigenvalue), but also on the angle that left and right eigenvectors not associated with λ_1 close.

If we started with the spectral resolution of Theorem 1.1 instead of the eigenvalue decomposition, we would have ended up with the following result.

Theorem 1.16. *Suppose $A \in \mathbb{C}^{n \times n}$ has a simple eigenpair (λ_1, u_1). Let*

$$A = [u_1 \;\; U_2] \begin{bmatrix} \lambda_1 & \\ & L_2 \end{bmatrix} [v_1 \;\; V_2]^\star$$

be such that $[u_1 \;\; U_2]^{-1} = [v_1 \;\; V_2]^\star$ with V_2 orthonormal. For a sufficiently small perturbation E, there is an eigenpair $(\tilde{\lambda}_1, \tilde{u}_1)$ of $A + E$ having property

$$\sin \measuredangle(\tilde{u}_1, u_1) \leq \frac{\|E\|}{\|(\tilde{\lambda}_1 I - L_2)^{-1}\|}.$$

The value

$$\operatorname{sep}(\tilde{\lambda}_1, L_2) := \frac{1}{\|(\tilde{\lambda}_1 I - L_2)^{-1}\|} = \sigma_{min}(\tilde{\lambda}_1 I - L_2)$$

is a compact way of describing complex entanglement of both eigenvalues and eigenspaces such as the one in (1.14). Here $\sigma_{min}(\tilde{\lambda}_1 I - L_2)$ denotes the smallest singular value of the matrix $\tilde{\lambda}_1 I - L_2$.

The perturbation theory for eigenspaces is somewhat more involved due to the nature of quantity sep contained in most results. Its definition in this more general setting is related to the Sylvester operator.

Definition 1.17. *For matrices $L_1 \in \mathbb{C}^{n_1 \times n_1}$ and $L_2 \in \mathbb{C}^{n_1 \times n_1}$, define*

$$\operatorname{sep}(L_1, L_2) = \inf_{\|Q\|=1} \|\mathcal{S}(Q)\|,$$

where $\mathcal{S} : \mathbb{C}^{n_2 \times n_1} \to \mathbb{C}^{n_2 \times n_1}$ is the Sylvester operator given by

$$\mathcal{S}(X) = XL_1 - L_2X.$$

Here $\|\cdot\|$ is any consistent norm; when Frobenius norm is used we will write $\operatorname{sep}_F(L_1, L_2)$.

Properties of the sep operator are summarized in the next Proposition.

Proposition 1.18.

(a) $\operatorname{sep}(L_1, L_2) \leq \min\{|\lambda_1 - \lambda_2| \ : \ \lambda_1 \in \Lambda(L_1), \lambda_2 \in \Lambda(L_2)\}$.

(b) If L_1 and L_2 are Hermitian, then $\operatorname{sep}_F(L_1, L_2) = \min\{|\lambda_1 - \lambda_2| \ : \ \lambda_1 \in \Lambda(L_1), \lambda_2 \in \Lambda(L_2)\}$.

(c) If $\operatorname{sep}(L_1, L_2) > 0$, *then* $\operatorname{sep}(L_1, L_2) = \frac{1}{\|\mathcal{S}^{-1}\|}$.

(d) If norm $\|\cdot\|$ is absolute (that is, $\|A\| = \||A|\|$), and $L_1 = \operatorname{diag}(L_1{}^{(1)}, \ldots, L_1{}^{(k_1)})$ and $L_2 = \operatorname{diag}(L_2{}^{(1)}, \ldots, L_2{}^{(k_2)})$, then

$$\operatorname{sep}(L_1, L_2) = \min_{i,j} \operatorname{sep}(L_1{}^{(i)}, L_2{}^{(j)}).$$

(f) $|\operatorname{sep}(L_1 + E_1, L_2 + E_2) - \operatorname{sep}(L_1, L_2)| \leq \|E_1\| + \|E_2\|$.

(g) $\operatorname{sep}(X_1^{-1}L_1X_1, X_2^{-1}L_2X_2) \geq \frac{\operatorname{sep}(L_1,L_2)}{\kappa(X_1)\kappa(X_2)}$, *where* $\kappa(X) = \|X\| \, \|X^{-1}\|$.

We state two theorems on the perturbations of invariant spaces.

Theorem 1.19. *Let $\mathcal{X}_1$ be a simple invariant subspace of A and let*

$$A = [X_1 \ \ X_2] \begin{bmatrix} L_1 & \\ & L_2 \end{bmatrix} [Y_1 \ \ Y_2]^\star$$

be the spectral resolution of Theorem 1.1. *Let E be a perturbation of A and let F_{11}, F_{12}, F_{21}, F_{22} denote matrices such that*

$$[Y_1 \ \ Y_2]^\star (A + E)[X_1 \ \ X_2] = \begin{bmatrix} L_1 + F_{11} & F_{12} \\ F_{21} & L_2 + F_{22} \end{bmatrix}.$$

Let $\delta = \operatorname{sep}(L_1, L_2) - \|F_{11}\| - \|F_{22}\|$. If $4 \|F_{12}\| \, \|F_{21}\| \leq \delta^2$, then there exists a matrix P satisfying $\|P\| < \frac{2\|F_{21}\|}{\delta}$ such that

$$(\tilde{L}_1, \tilde{X}_1) = (L_1 + F_{11} + F_{12}P, X_1 + X_2P)$$

and

$$(\tilde{L}_2, \tilde{Y}_2) = (L_2 + F_{22} - PF_{12}, Y_2 - Y_1P)$$

are complementary, simple, right and left eigenpairs of $A+E$ (that is, $(A+E)\tilde{X}_1 = \tilde{X}_1\tilde{L}_1$ and $\tilde{Y}_2^\star(A + E) = \tilde{L}_2\tilde{Y}_2^\star$).

We can also estimate the angle between invariant subspaces of the original and the perturbed matrix.

Theorem 1.20. *Using the notation and assumptions of Theorem* 1.19*,*

$$\tan \measuredangle(\tilde{\mathcal{X}}_1, \mathcal{X}_1) < \frac{\|E\|}{\beta - 2\|E\| \tan \measuredangle(\mathcal{X}_1, \mathcal{Y}_1)}. \tag{1.15}$$

Here $\tilde{\mathcal{X}}_1 = \operatorname{Im} \tilde{X}_1$ *and* $\beta = \operatorname{sep}(L_1, L_2) - 2\frac{\|E\|}{\cos \measuredangle(\mathcal{X}_1, \mathcal{Y}_1)}$*. Furthermore,*

$$\sin \measuredangle(\tilde{\mathcal{X}}_1, \mathcal{X}_1) < \frac{\|E\|}{\operatorname{sep}(\tilde{L}_1, L_2)}.$$

Note that as $\|E\| \to 0$, the denominator in (1.15) converges to $\operatorname{sep}(L_1, L_2)$, having the effect of

$$\tan \measuredangle(\tilde{\mathcal{X}}_1, \mathcal{X}_1) \lesssim \frac{\|E\|}{\operatorname{sep}(L_1, L_2)}.$$

Thus $1/\operatorname{sep}(L_1, L_2)$ can be considered as a condition number of a simple invariant subspace $\mathcal{X}_1$. For Hermitian matrices, this reads

$$\tan \measuredangle(\tilde{\mathcal{X}}_1, \mathcal{X}_1) \lesssim \frac{\|E\|_F}{\min\{|\lambda_1 - \lambda_2| \ : \ \lambda_1 \in \Lambda(L_1), \lambda_2 \in \Lambda(L_2)\}},$$

meaning that invariant subspaces will be less sensitive to perturbations when associated eigenvalues are further split apart from the rest of the spectrum.

1.5 Approximations from a subspace

Most of the algorithms for acquiring eigenpairs of a large matrix A construct a sequence of subspaces $\mathcal{X}_1, \mathcal{X}_2, \ldots$ using some algorithm-specific method. At iteration k, some elements of subspace $\mathcal{X}_k$ are selected as approximations for eigenvectors of A. In this section we analyze two important questions. First, how to evaluate the quality of a given subspace $\mathcal{X}$ – when does it contain an eigenvector or what angle does it close with some invariant subspace? When this is established, we will have ideas on how to answer the second question: choosing the best possible approximations for eigenpairs from $\mathcal{X}$. Methods for constructing the sequence of subspaces are topic of Chapter 2.

1.5.1 Evaluating quality of a subspace

One-dimensional case. Suppose we are given a matrix A and a subspace $\mathcal{X} = \operatorname{span}\{x\}$. Using x as the approximation for some eigenvector of A, we need to construct an approximation μ of the associated eigenvalue and estimate how close (μ, x) is to an exact eigenpair. Since the exact eigenpair would satisfy $Ax - \mu x = 0$, we feel that a good measurement of quality might be the norm of vector $Ax - \mu x$. This is indeed the case, as the following Theorem suggests.

Theorem 1.21. *Suppose $A = TJT^{-1}$ is the Jordan form of matrix $A \in \mathbb{C}^{n\times n}$. Let (μ, x) be an approximate eigenpair of A, with $\|x\| = 1$. Then there exists an eigenvalue λ of A such that*

$$\frac{|\mu - \lambda|^{\iota(\lambda)}}{1 + |\mu - \lambda| + |\mu - \lambda|^2 + \ldots + |\mu - \lambda|^{\iota(\lambda)-1}} \leq \kappa(T) \|Ax - \mu x\|,$$

where $\kappa(T) = \|T\| \left\|T^{-1}\right\|$ and $\iota(\lambda)$ is the index of the eigenvalue λ. Specially, if A is diagonalizable, this reduces to

$$|\mu - \lambda| \leq \kappa(S) \|Ax - \mu x\|,$$

where $A = S\Lambda S^{-1}$ is the eigenvalue decomposition of A. The vector $Ax - \mu x$ is called the residual of approximate eigenpair (μ, x) of the matrix A.

As we can see, if the eigenvector matrix S is badly conditioned, then a small residual does not necessarily mean that μ is an excellent approximation for some $\lambda \in \Lambda(A)$. Note that the eigenvector matrix is not unique; for example, its columns may be arbitrarily scaled. A theorem of van der Sluis [74] states that, for all matrices $X \in \mathbb{C}^{n\times n}$,

$$\kappa(X\tilde{D}) \leq \sqrt{n} \min\{\kappa(XD) \ : \ D \text{ diagonal}\},$$

where $\tilde{D}$ is such that $X\tilde{D}$ has all columns of equal norm. Thus the eigenvector matrix S is up to the factor of $\sqrt{n}$ optimal (in terms of minimizing the condition) when all of its columns have unit norm. However, since S is unknown during the computation, all we can do is to consider the residual as a good indicator of convergence progress.

There is also a way to bound the sine of the angle between x and some exact eigenvector using the residual. This result is also not very practical since the right hand side includes values we wish to compute.

Theorem 1.22. *Let the spectral resolution of $A \in \mathbb{C}^{n\times n}$ (see Theorem 1.1) be*

$$A = [u \ U] \begin{bmatrix} \lambda & \\ & L \end{bmatrix} [v \ V]^\star$$

and let (μ, x) denote an approximate eigenpair. Then

$$\sin \measuredangle(x, u) \leq \frac{\|Ax - \mu x\|}{\operatorname{sep}(\mu, L)} \leq \frac{\|Ax - \mu x\|}{\operatorname{sep}(\lambda, L) - |\mu - \lambda|}.$$

Furthermore, norm of the residual is the size of the smallest perturbation E of matrix A that makes an approximate eigenpair (μ, x) of A become an exact eigenpair of $A + E$.

Theorem 1.23 (Kahan-Jiang-Parlett)**.** *Let $A \in \mathbb{C}^{n\times n}$. Denote with $\mathcal{E}$ set of all matrices E such that (μ, x) is an exact eigenpair of $A + E$, with $\|x\| = 1$. Then*

$$\min_{E\in\mathcal{E}} \|E\| = \|Ax - \mu x\|.$$

The minimum is obtained for $E = -rx^\star$, with $r = Ax - \mu x$.

In other words, if the residual is small, then only a small perturbation of the matrix A will make (μ, x) exact. This is another argument for using the size of the residual as a quality measurement.

Given x, we can easily compute the best possible μ in terms of the norm of the residual. That value will provide the best bounds in the last two Theorems.

Theorem 1.24. *Let $A \in \mathbb{C}^{n\times n}$ and $x \in \mathbb{C}^n$. Then*

$$\min_{\mu\in\mathbb{C}} \|Ax - \mu x\| = \|Ax - \rho(x)x\|,$$

where $\rho(x) = \frac{x^\star Ax}{x^\star x}$ is the Rayleigh quotient of vector x.

When Rayleigh quotient is used as an eigenvalue approximation, we can show even sharper bounds if the matrix is Hermitian; note that Rayleigh quotients are real numbers for such matrices. Proof of the following Theorem uses the fact that the eigenvectors of A make an orthonormal basis of $\mathbb{C}^n$ and that its eigenvalues can be sorted along the real axis. The error in eigenvalue is now bounded by $\mathcal{O}(\|Ax - \rho(x)x\|^2)$, opposed to only linear bound in Theorem 1.21.

Theorem 1.25. *Suppose $A \in \mathbb{C}^{n\times n}$ is a Hermitian matrix. Let $x \in \mathbb{C}^n$ and $\|x\| = 1$; denote with λ the eigenvalue of A that is closest to $\rho(x)$. Let u be the unit eigenvector obtained by orthogonal projection of x to eigenspace $\mathcal{U}_\lambda$ associated with λ (or any unit eigenvector of λ if x is perpendicular to $\mathcal{U}_\lambda$). Furthermore, let*

$$\gamma = \min\{|\rho(x) - \lambda_i| \ : \ \lambda_i \in \Lambda(A) \setminus \{\lambda\}\}$$

denote the distance of $\rho(x)$ to its second closest eigenvalue of A. Then the following bounds hold:

$$|\rho(x) - \lambda| \le \frac{\|Ax - \rho(x)x\|^2}{\gamma}, \quad \sin\measuredangle(x, u) \le \frac{\|Ax - \rho(x)x\|}{\gamma}.$$

Higher dimensional case. Next, consider the general case where $\mathcal{X}$ is a k-dimensional subspace of $\mathbb{C}^n$. Let X denote a matrix whose columns make an orthogonal basis for $\mathcal{X}$. Once again, if $\mathcal{X}$ was an invariant subspace of A, there would exist a matrix L such that $AX - XL = 0$ and spectrum of L would be a subset of spectrum of A. Thus, for some matrix L we can consider norm of the residual $AX - XL$ as a measurement of how close is $\mathcal{X}$ to being an invariant subspace of A and how close the eigenvalues of L are to the eigenvalues of A.

We can now show the generalizations of Theorems 1.23 and 1.24.

Theorem 1.26. *Let $A \in \mathbb{C}^{n\times n}$, $L \in \mathbb{C}^{k\times k}$ and let $\mathcal{X}$ be a k-dimensional subspace of $\mathbb{C}^n$ with orthonormal basis X. For perturbation $E = -RX^\star$, where $R = AX - XL$ it holds that*

$$(A + E)X = XL, \quad \|E\| = \|R\|.$$

Thus $\mathcal{X}$ is an invariant subspace of a perturbed matrix $A + E$. Note that there is no mentioning of optimality of perturbation E here; see [70, Exercise IV.1.12] for such a result. Minimizing $\|R\|$ leads to generalization of the Rayleigh quotient.

Theorem 1.27. *Let $A \in \mathbb{C}^{n\times n}$ and let $\mathcal{X}$ be a k-dimensional subspace of $\mathbb{C}^n$ with orthonormal basis X. Then*

$$\min_{L\in\mathbb{C}^{k\times k}} \|AX - XL\| = \|AX - X\rho(X)\|,$$

where $\rho(X) = X^\star AX$ is called the Rayleigh quotient of X.

Note that $\|AX - X\rho(X)\|$ is a property of subspace $\mathcal{X}$ – it does not depend on the choice of the orthonormal basis X. Angle between $\mathcal{X}$ and the exact invariant subspace can be bounded when we use the Rayleigh quotient.

Theorem 1.28. *Let $A \in \mathbb{C}^{n\times n}$ and let $\mathcal{X}$ be an approximate subspace with orthonormal basis X. Extend X with $X_\perp$ so that $[X \;\; X_\perp]$ is unitary and let $L = \rho(X)$, $L_\perp = \rho(X_\perp)$,*

$$R = AX - XL, \quad S^\star = X^\star A - LX^\star.$$

If $4\|R\|\,\|S\| \le \operatorname{sep}^2(L, L_\perp)$, then there exists a simple A-invariant subspace $\mathcal{U}$ with associated eigenpair (Λ, U) satisfying

$$\|L - \Lambda\| < \frac{2\|R\|\,\|S\|}{\operatorname{sep}(L, L_\perp)}, \quad \|\tan\measuredangle(\mathcal{X},\mathcal{U})\| < \frac{2\|R\|}{\operatorname{sep}(L, L_\perp)}.$$

When A is Hermitian, then $S = R^\star$ and the bound on $\|L - \Lambda\|$ is quadratic in $\|AX - X\rho(X)\|$, just as in Theorem 1.25. Also, similar bound for the sine of angles can be proved.

Theorem 1.29. *Let the Hermitian matrix $A \in \mathbb{C}^{n\times n}$ have a spectral representation*

$$A = [U_1 \;\; U_2] \begin{bmatrix} L_1 & \\ & L_2 \end{bmatrix} [U_1 \;\; U_2]^\star,$$

where $[U_1 \;\; U_2]$ is unitary and $U_1 \in \mathbb{C}^{n\times k}$ is basis for A-invariant subspace $\mathcal{U}_1$. Let $\mathcal{X}$ be a k-dimensional subspace of $\mathbb{C}^n$ with an orthonormal basis X and let $L \in \mathbb{C}^{k\times k}$ be any Hermitian matrix. If

$$\gamma := \min\{|\lambda - \lambda_2| \;:\; \lambda \in \Lambda(L), \lambda_2 \in \Lambda(L_2)\} > 0,$$

then

$$\|\sin\measuredangle(\mathcal{X},\mathcal{U}_1)\|_F \le \frac{\|AX - XL\|_F}{\gamma}.$$

1.5.2 Choosing approximations for eigenpairs

In the last section we have seen how to evaluate whether given subspace $\mathcal{X}$ is a good approximation of some A-invariant subspace. Now we review several options on how to choose particular vectors from $\mathcal{X}$ so that they constitute good approximates for eigenvectors of A. Ideally, if our goal is to approximate some eigenvector u of A, the optimal vector $x \in \mathcal{X}$ that we can pick would be the one satisfying $\measuredangle(x, u) = \measuredangle(\mathcal{X}, u)$. Since u is unknown, this is not feasible in practice. The best we can do is to somehow relate the quality of chosen vector to the quality of subspace $\mathcal{X}$, and this way have some guarantee that we do not miss a good approximate of u should $\mathcal{X}$ contain one.

Rayleigh-Ritz approximations. As we have seen, when $AX - XL = 0$, then the eigenvalues of L are also eigenvalues of A, and if y is an eigenvector of L, then Xy is an eigenvector of A. On the other hand, when $\mathcal{X}$ is not A-invariant, then $\|AX - XL\|$ is minimized for $L = \rho(X)$. This fact indicates that for an eigenpair (θ, y) of $\rho(X)$, the pair (θ, Xy) should be a good approximation of some eigenpair of A when subspace $\mathcal{X}$ is close to being invariant.

Definition 1.30. *Let $A \in \mathbb{C}^{n\times n}$ and let $\mathcal{X}$ be a k-dimensional subspace of $\mathbb{C}^n$ with orthonormal basis X. Suppose (θ, y) is an eigenpair of $\rho(X)$. Then θ is called a Ritz value, y is a primitive Ritz vector, and Xy is a Ritz vector of the matrix A from subspace $\mathcal{X}$. Also, (θ, y) is called a primitive Ritz pair and (θ, Xy) is called a Ritz pair.*

Another motivation to introduce Ritz values is the Galerkin principle: if vector $Ax-\mu x$ is perpendicular to every vector in $\mathbb{C}^n$, then it must be zero, and (μ, x) is an exact eigenpair of A. Ensuring that $Ax-\mu x$ is perpendicular at least to every vector from $\mathcal{X}$ should make (μ, x) optimal in that sense. This condition is equivalent to

$$X^\star(Ax - \mu x) = 0.$$

Setting $x = Xy$ for some $y \in \mathbb{C}^k$, we see that $(X^\star AX)y = \mu y$ and (μ, x) is a Ritz pair.

Note that each Ritz value is a Rayleigh quotient of associated Ritz vector:

$$\theta = y^\star \rho(X) y = y^\star X^\star AXy = \rho(Xy).$$

We can thus use results from one-dimensional part of Section 1.5.1 to evaluate quality of a single Ritz pair as an eigenpair approximation. On the other hand, we can link the quality of subspace $\mathcal{X}$ to the quality of Ritz approximations from it. The following results show that if $\mathcal{X}$ contains a good approximation of an eigenvector, then some Ritz vector will be one as well. First we state a backward error result.

Theorem 1.31. *Let (λ, u) be a simple eigenpair of $A \in \mathbb{C}^{n\times n}$ and let X be an orthonormal basis of $\mathcal{X} \leqslant \mathbb{C}^n$. There there exists a matrix E such that λ is an eigenvalue of $\rho(X) + E$ and*

$$\|E\| \leq \|A\| \tan \measuredangle(\mathcal{X}, u).$$

Using the Ostrowski-Elsner Theorem 1.12 and the fact that X is orthonormal, if follows that there is a Ritz value θ such that

$$|\theta - \lambda| \leq (2k-1)\,\|A\|\,(2+\tau)^{1-\frac{1}{k}}\,\tau^{\frac{1}{k}}, \tag{1.16}$$

where $\tau = \tan\measuredangle(\mathcal{X}, u)$. The angle between a Ritz vector and an exact eigenvector can also be bounded.

Theorem 1.32. *Let (θ, y) be a primitive Ritz vector of the matrix A from subspace $\mathcal{X}$ with orthogonal basis X and and let $[y\ \ Y]$ be unitary. Furthermore, let*

$$[y\ \ Y]^\star \rho(X)[y\ \ Y] = \begin{bmatrix} \theta & h^\star \\ 0 & M \end{bmatrix}.$$

Then for an eigenpair (λ, u) of A it holds that

$$\sin\measuredangle(Xy, u) \leq \sin\measuredangle(\mathcal{X}, u)\sqrt{1 + \frac{\|X^\star A - \rho(X)X^\star\|^2}{\operatorname{sep}^2(\lambda, M)}}.$$

Suppose we have a sequence of subspaces $\mathcal{X}_1, \mathcal{X}_2, \ldots$ so that $\measuredangle(\mathcal{X}_j, u) \xrightarrow{j} 0$, where (λ, u) is a simple eigenpair of A. Then (1.16) says that there is a sequence of Ritz values from these spaces which will converge to λ. However, that might not be the case with Ritz vectors – additional condition of $\operatorname{sep}(\lambda, M)$ being bounded from below is required. Since

$$\operatorname{sep}(\lambda, M) \geq \operatorname{sep}(\mu, M) - |\mu - \lambda|$$

and μ's are converging to λ, it is required that μ's are uniformly separated from the rest of the spectrum of appropriate Rayleigh quotients. This can be checked during the computation. If uniform separation condition fails, one can expect *spurious* Ritz pairs – the value is close to the exact eigenvalue, but the vector is nowhere near the exact eigenvector. Nevertheless, if the residuals of Ritz pairs converge to zero, Theorem 1.22 guarantees the convergence of Ritz vectors as well. If, on the other hand, the uniform separation condition is satisfied, then the bound (1.16) is too pessimistic: let $x = Xy = \gamma u + \sigma u_\perp$, where $u_\perp$ is a unit vector perpendicular to u, $|\gamma| = \cos\measuredangle(x, u)$ and $|\sigma| = \sin\measuredangle(x, u)$. Theorem 1.32 states that with the uniform separation $\sin\measuredangle(x, u) = \mathcal{O}(\sin\measuredangle(\mathcal{X}, u))$ holds since $\|X^\star A - \rho(X)X^\star\| \leq \|A\| + \|\rho(X)\| \leq 2\,\|A\|$. Then we have

$$\begin{aligned} |\theta - \lambda| &= |x^\star A x - \lambda| = \left||\gamma|^2\lambda + \overline{\gamma}\sigma u^\star A u_\perp + |\sigma|^2 u_\perp^\star A u_\perp - \lambda\right| \\ &\leq ((1-|\gamma|^2) + |\gamma||\sigma| + |\sigma|^2)\,\|A\| \leq (|\sigma|^2 + |\sigma| + |\sigma|^2)\,\|A\| \\ &= \mathcal{O}(\sin\measuredangle(\mathcal{X}, u))\,\|A\|\,, \end{aligned} \tag{1.17}$$

meaning that the approximation by a Ritz value will be no worse than the best approximation of an eigenvector in entire subspace $\mathcal{X}$.

As usual, additional results and better bounds can be achieved for Hermitian matrices. Recall the minimax property:

$$\lambda_j = \max_{\dim \mathcal{W}=j} \min_{\substack{x\in\mathcal{W} \\ \|x\|=1}} x^\star A x,$$

where λ_j is the j-th largest eigenvalue of the Hermitian matrix A and maximum is taken over all subspaces $\mathcal{W}$ of dimension j. Similar equality holds for the j-th largest Ritz value from any subspace $\mathcal{X}$:

$$\theta_j = \max_{\substack{\dim \mathcal{W}=j \\ \mathcal{W}\leqslant\mathcal{X}}} \min_{\substack{x\in\mathcal{W} \\ \|x\|=1}} x^\star A x,$$

but now $\mathcal{W}$'s have to be subspaces of $\mathcal{X}$. One of the main tools for exploring properties of Rayleigh quotients for Hermitian matrices is a simple consequence of these minimax properties.

Theorem 1.33 (Cauchy interlacing theorem)**.** *Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix and let $\mathcal{X}$ be a k-dimensional subspace of $\mathbb{C}^n$ with orthonormal basis X. Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ denote eigenvalues of A and let $\theta_1 \geq \theta_2 \geq \ldots \geq \theta_k$ denote eigenvalues of $\rho(X) = X^\star A X$. For all $j = 1, 2, \ldots, k$ it holds that*

$$\lambda_{j+(n-k)} \leq \theta_j \leq \lambda_j.$$

Next we state an improved version of bound (1.17) and a specialization of Theorem 1.32 to Hermitian matrices.

Theorem 1.34. *Let the assumptions of Theorem* 1.33 *hold. Then the following statements are true.*

(a) There exist distinct indices $i_1, i_2, \ldots, i_k$ such that

$$|\theta_j - \lambda_{i_j}| \leq \|AX - X\rho(X)\|,$$

for all $j = 1, 2, \ldots, k$.

(b) There exist distinct indices $i_1, i_2, \ldots, i_k$ such that

$$\sqrt{\sum_{j=1}^{k} (\theta_j - \lambda_{i_j})^2} \leq \sqrt{2}\,\|AX - X\rho(X)\|_F.$$

(c) Let λ_{i_j} be an eigenvalue of A that is closest to θ_j, let u_{i_j} be the associated eigenvector and let x_j be the associated Ritz vector. Then

$$\sin\measuredangle(x_j, u_{i_j}) \leq \sin\measuredangle(\mathcal{X}, u_{i_j}) \sqrt{1 + \frac{\|AX - X\rho(X)\|^2}{\delta_j^2}},$$

where $\delta_j = \min_{i\neq j} |\theta_i - \lambda_{i_j}|$.

There is a subtle, but important difference in statement between parts (a) and (b) of the previous Theorem as opposed to part (c) and Theorem 1.25. The first two results state that each Ritz value approximates it own, unique eigenvalue of A. On the other hand, in part (c) and Theorem 1.25 it can happen that the same eigenvalue is associated with two distinct Ritz values.

In the case of a positive definite matrix A, one can obtain bounds on the relative error. Such bounds guarantee good approximations for the eigenvalues which are small in magnitude. We state a result from [25]; even the quadratic bounds can be obtained [13].

Theorem 1.35. *[25] Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian positive definite matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n > 0$, and let $A = LL^\star$ denote its Cholesky factorization. Suppose that the columns of $X \in \mathbb{C}^{n\times k}$ span an orthonormal basis of a k-dimensional subspace $\mathcal{X}$. Let $R = AX - X\rho(X)$ and $\tilde{A} = A - (RX^\star + XR^\star)$. Then the set of Ritz values for A from $\mathcal{X}$ is a subset of the spectrum of $\tilde{A}$. The matrix $\tilde{A}$ is positive definite, and if its eigenvalues are denoted with $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \ldots \geq \tilde{\lambda}_n > 0$, then it holds that*

$$\begin{aligned} \max_{j=1,\ldots,n} \frac{|\lambda_j - \tilde{\lambda}_j|}{\lambda_j} &\leq \frac{\sin \measuredangle(L^\star\mathcal{X}, L^{-1}\mathcal{X})}{1 - \sin \measuredangle(L^\star\mathcal{X}, L^{-1}\mathcal{X})}; \\ \max_{j=1,\ldots,n} \frac{|\lambda_j - \tilde{\lambda}_j|}{\tilde{\lambda}_j} &\leq \sin \measuredangle(L^\star\mathcal{X}, L^{-1}\mathcal{X}). \end{aligned}$$

Note how $\measuredangle(L^\star\mathcal{X}, L^{-1}\mathcal{X})$ measures how far is the subspace $\mathcal{X}$ from being A-invariant; this angle equals zero when $\mathcal{X}$ is exactly A-invariant.

Refined (Ritz) approximations. Theorem 1.24 states that given a vector $x \in \mathbb{C}^n$, the value of μ at which minimum of $\|Ax - \mu x\|$ is attained is the Rayleigh quotient, $\mu = \rho(x)$. Since Ritz values from subspace $\mathcal{X}$ are Rayleigh quotients of Ritz vectors, they are optimal from this viewpoint. On the other hand, suppose we are given a complex number μ and we want to determine

$$\min_{\substack{x\in\mathcal{X} \\ \|x\|=1}} \|Ax - \mu x\|\,. \tag{1.18}$$

In general, if $\mu = \rho(x)$, then it is not true that x minimizes (1.18); neither are the Ritz vectors minimizers when Ritz values are inserted as μ's. This effect motivates the following definition.

Definition 1.36. *For $A \in \mathbb{C}^{n\times n}$, subspace $\mathcal{X} \leqslant \mathbb{C}^n$ and $\mu \in \mathbb{C}$, a unit vector x for which (1.18) is attained is called a refined vector for μ from $\mathcal{X}$ and (μ, x) is called a refined pair. When μ is a Ritz value from $\mathcal{X}$, then x is called a refined Ritz vector and (μ, x) is a refined Ritz pair.*

To compute a refined vector for μ, suppose X is an orthonormal basis for a k-dimensional subspace $\mathcal{X}$. Then

$$\begin{aligned} \min_{\substack{x\in\mathcal{X} \\ \|x\|=1}} \|Ax - \mu x\| &= \min_{\|y\|=1} \|AXy - \mu Xy\| = \min_{\|y\|=1} \|(AX - \mu X)y\| \\ &= \sigma_{min}(AX - \mu X), \end{aligned}$$

where $\sigma_{min}(AX - \mu X)$ is the smallest singular value of the matrix $AX - \mu X$ and the minimum is achieved for y equal to the associated right singular vector, i.e. $x = Xy$.

Computation of a refined vector for a single value μ requires forming the matrix AX and computing a singular value decomposition of $AX - \mu X$, which is an $n \times k$ matrix. This is a $\mathcal{O}(nk^2)$ operation, and the same complexity is required to compute all Ritz pairs from $\mathcal{X}$. If one needs to compute several (say, k) approximations, then this approach is much too expensive. However, one can use the fact that right singular vectors of $AX - \mu X$ are the same as the eigenvectors of

$$(AX - \mu X)^\star (AX - \mu X) = (AX)^\star (AX) - \mu (X^\star AX + (X^\star AX)^\star) + \mu^2 I, \tag{1.19}$$

so it suffices to compute AX and $X^\star(AX)$ just once. After that refined vectors for various μ are computed as eigenvectors of $k \times k$ matrices (1.19). For special subspaces $\mathcal{X}$ such as Krylov, there are even more efficient methods.

To appreciate the effort in computing refined vectors, consider the following result.

Theorem 1.37. *[68, Theorem IV.4.10] Let*

$$A = [u \;\; U] \begin{bmatrix} \lambda & \\ & \Lambda \end{bmatrix} [v \;\; V]^\star$$

be the spectral resolution of Theorem 1.1 *and let θ be a Ritz value from subspace $\mathcal{X}$. Then for a refined Ritz vector x associated with θ it holds that*

$$\sin \measuredangle(x, u) \leq \frac{\|A - \theta I\| \sin \measuredangle(\mathcal{X}, u) + |\theta - \lambda|}{\cos \measuredangle(\mathcal{X}, u) \, (\operatorname{sep}(\lambda, \Lambda) - |\theta - \lambda|)},$$

under condition that $\operatorname{sep}(\lambda, \Lambda) - |\theta - \lambda| > 0$ *is satisfied.*

As we have seen in the discussion after Theorem 1.32, if $\mathcal{X}_1, \mathcal{X}_2, \ldots$ is a sequence of subspaces such that $\measuredangle(\mathcal{X}_i, u) \longrightarrow 0$, then there is a sequence of Ritz values from these subspaces that converge to λ. This is not always true for associated Ritz vectors. On the other hand, Theorem 1.37 guarantees that if λ is a simple eigenvalue, then the sequence of refined Ritz vectors will converge to the eigenvector; there is no need for a uniform separation condition.

Harmonic Ritz approximations. Suppose we are given an increasing sequence of subspaces $\mathcal{X}_1 \leqslant \mathcal{X}_2 \leqslant \ldots$ as a source of Ritz approximations. Then Ritz values exhibit a steady convergence to the external eigenvalues of A; by external eigenvalues we consider those on or close to the edge of convex hull spanned by all eigenvalues of A. This is best demonstrated when A is Hermitian: if $\theta_j^{(i)}$ is the j-th largest Ritz value from $\mathcal{X}_i$, then for each j it holds that

$$\theta_j^{(i)} \leq \theta_j^{(i+1)} \leq \lambda_j,$$

where λ_j is the j-th largest eigenvalue of A. Thus convergence of Ritz values to the largest eigenvalues is monotone; the same is true for the smallest eigenvalues and for the external eigenvalues of normal matrices since the minimax theorems generalize to those as well.

ALGORITHM 1: Harmonic Ritz pairs from subspace $\mathcal{X}$ (w.r.t. $(A-\tau I)\mathcal{X}$)

Input: Orthonormal basis X for subspace $\mathcal{X}$, target τ
Output: Harmonic Ritz pairs (θ_h, x_h) from subspace $\mathcal{X}$

1 Compute $B = X^\star(A-\tau I)^\star X$;
2 Compute $C = X^\star(A-\tau I)^\star(A-\tau I)X$;
3 Compute eigenpairs (ξ_h, y_h) of $B^{-1}C$;
4 Set $\theta_h = \xi_h + \tau$ and $x_h = Xy_h$;

On the other hand, consider the case when we have to compute the eigenvalues closest to some target $\tau \in \mathbb{C}$ deeper inside the convex hull. It is easily seen that setting $\theta_\tau^{(i)}$ as the Ritz value from $\mathcal{X}_i$ that is closest to τ does not provide a steady convergence. A Ritz value which is very close to some interior eigenvalue λ is often a Rayleigh quotient of a (Ritz) vector which is not near to the eigenvector associated with λ.

A simple remedy to this problem would be to compute Ritz approximations using the matrix $(A-\tau I)^{-1}$ instead of A. For each eigenpair (λ, u) of A there is an eigenpair $(\frac{1}{\lambda-\tau}, u)$ of $(A-\tau I)^{-1}$, so if (θ, x) is a Ritz pair approximating the latter, then $(\frac{1}{\theta}+\tau, x)$ approximates the former. Eigenvalues closest to τ are mapped to the external eigenvalues of $(A-\tau I)^{-1}$ and Ritz values should provide monotone convergence to those.

Definition 1.38. *[63] Complex number θ_h is a harmonic Ritz value of A with respect to subspace $\mathcal{X}$ and target $\tau \in \mathbb{C}$ if $\frac{1}{\theta_h-\tau}$ is a Ritz value of $(A-\tau I)^{-1}$ with respect to $\mathcal{X}$.*

Computing harmonic Ritz values involves computing the Rayleigh quotient of $(A-\tau I)^{-1}$, which in turn involves solving linear systems of form $(A-\tau I)^{-1}b$. In many cases this is very time consuming unless subspace $\mathcal{X}$ has a special structure.

Theorem 1.39. *[63] Let $A \in \mathbb{C}^{n\times n}$ and let $\mathcal{X}$ be a subspace of $\mathbb{C}^n$ with basis X. Then θ_h is a harmonic Ritz value of A with respect to $\tilde{\mathcal{X}} = (A-\tau I)\mathcal{X}$ and target τ if and only if for some non-zero vector $x_h \in \mathcal{X}$ it holds that*

$$Ax_h - \theta_h x_h \perp (A-\tau I)\mathcal{X}. \tag{1.20}$$

Let $\tilde{X}$ be the basis for $\tilde{\mathcal{X}}$ and let $\rho_h(X) = (\tilde{X}^\star X)^{-1}\tilde{X}^\star(A-\tau I)X$. Then (1.20) is equivalent to $x_h = Xy_h$ where $(\theta_h - \tau, y_h)$ is an eigenpair of $\rho_h(X)$. Vector x_h is called the harmonic Ritz vector and (θ_h, x_h) is a harmonic Ritz pair.

Using the Theorem, it is easily shown that for an orthonormal X the following holds:

$$\|Ax_h - \theta_h x_h\| \leq 2|\theta_h - \tau|.$$

Therefore, if a harmonic Ritz value is near the target τ, then the residual of the harmonic pair is small and the harmonic Ritz vector should be a good approximation of the eigenvector; this is not guaranteed for Ritz pairs.

Note that harmonic Ritz vectors are elements of $\mathcal{X}$ although the values are with respect to subspace $\tilde{\mathcal{X}} = (A-\tau I)\mathcal{X}$. Harmonic pairs are always used in this context and $\tilde{\mathcal{X}}$ is

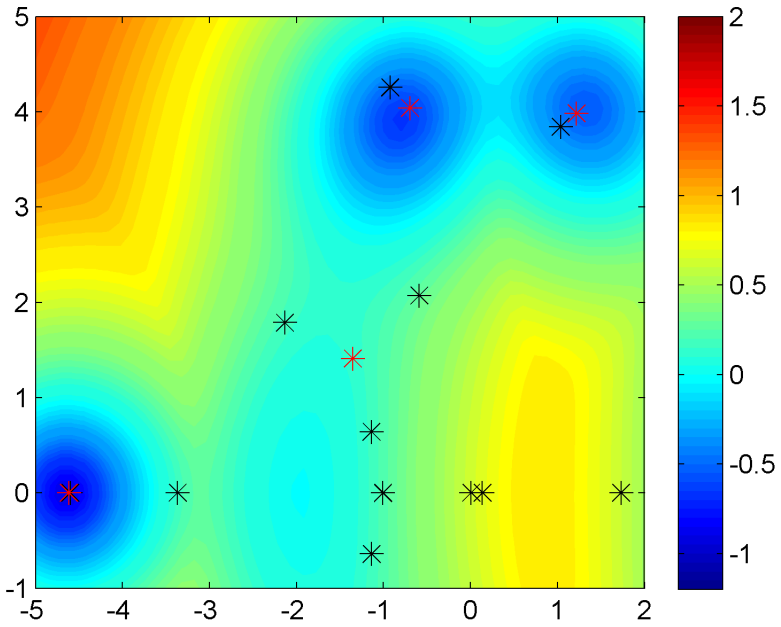


Figure 1.3: Choosing eigenvalue approximations. Eigenvalues of a random 20×20 matrix are black asterisks. Ritz value from a 9-dimensional Krylov subspaces are red asterisks. Each point of the complex plane is colored according to the logarithm of the residuals of refined pairs.

just an intermediary in the computation: we are originally given a subspace $\mathcal{X}$ and we need an approximation from it. Also, we find the elements of $\mathcal{X}$ in general to be better approximations of internal eigenvectors. Should $\tilde{x}_h \in \tilde{\mathcal{X}}$ be an approximation for such an eigenvector, then $(A - \tau I)^{-1}\tilde{x}_h \in \mathcal{X}$ is a better approximation since it is obtained using one step of inverse iteration algorithm on $\tilde{x}_h$; see Section 2.1. Algorithm 1 shows how to compute approximations to internal eigenvalues from subspace $\mathcal{X}$ using $(A - \tau I)X$ as a basis for $\tilde{\mathcal{X}}$. Once again, when $\mathcal{X}$ is a Krylov subspace, an algorithm of smaller complexity can be designed.

Furthermore, some authors suggest that the Rayleigh quotient $\rho(x_h)$ should be taken as an approximate eigenvalue instead of θ_h or that the refined vector of θ_h should be taken as an approximate eigenvector. For more on this topic and convergence results, see [34]. The optimality of Ritz and harmonic Ritz values in the Hermitian case is further discussed in [8, 62], along with the connection to the eigenvalue inclusion regions (*Lehmann's optimal intervals*).

Concluding remarks. For a given subspace $\mathcal{X}$ it may be difficult to declare an approximate eigenpair as a good or even best possible approximation from $\mathcal{X}$. Methods for extracting approximations from a given subspace are heuristics and to understand the difficulty, even when we know the eigenvalues in advance, consider Figure 1.3. For a random matrix A of order 20, we have computed a Krylov subspace $\mathcal{X}$ of dimension 9.

Black asterisks are the eigenvalues of A and red asterisks are the Ritz values from $\mathcal{X}$. The complex plane is colored in the following way: for each complex number μ, a refined vector x for μ from $\mathcal{X}$ is computed and μ is colored according to the logarithm of the (minimal) residual $\|Ax - \mu x\|$. For some eigenvalues, like the one near $(-4.6, 0)$, there is little doubt on what should be considered the optimal approximation from $\mathcal{X}$: the eigenvalue almost coincides with the Ritz value and the local minimum of the refined residuals. For the other eigenvalues, like the one near $(1, 4)$, it is unclear how to pick the best approximation. Ritz value is relatively close to the extreme of the refined residuals, but on the other hand, the eigenvalue itself is not and is even further from the extreme than the Ritz value. The theorems we have shown state that when a subspace contains a good approximation with low residual, then both the Ritz and the refined Ritz method will locate it. However, it is often the case that during initial phases of algorithms for large scale eigenvalue extraction, the subspaces will not necessarily contain such approximations. A lot of caution is needed in declaring a part of the subspace as "good" or "bad" resulting in keeping it or purging from further consideration.

Chapter 2

Arnoldi-type Algorithms

During the last 170 years, a number of various techniques for the computation of matrix's eigenpairs was developed. Modern algorithms are designed in order to target matrices of certain sizes. Matrix dimension limits the arsenal of tools that can be used to obtain the eigenpairs; it also sets the limit on what can be calculated at all. Roughly speaking, we separate eigenvalue algorithms into these categories:

- Algorithms that target "small" matrices. These algorithms can compute the entire eigenvalue decomposition, using orthogonal similarity transformations of matrices in order to convert them to diagonal form. Usually the target matrices are also full and non-structured, so these algorithms do not pay attention to keeping the sparsity pattern of the matrix.

  The most notable algorithms belonging to this group are the QR algorithm [29, 37] for general and Hermitian matrices and the Jacobi algorithm [33] as well as the divide-and-conquer approach [16] in the Hermitian case. Key issues here are the speed and accuracy of the algorithms; see e. g. [22, 23, 26].

- Algorithms that target "medium" size matrices. These matrices can be characterized as too large to be diagonalized by applying the similarity transformations; thus only a small part of their spectrum can be obtained. Matrices have a sparsity pattern that allows efficient calculation of the matrix-times-vector operation Ax. On the other hand, these matrices are small enough to enable the usage of using approximate solutions of the linear systems such as $(A-\xi I)x = b$ occasionally during the process of calculating the eigenpairs. Incomplete LU-factorization is also at disposition.

  Algorithms in this group include e. g. Jacobi-Davidson [63] or various methods based on the shift-and-invert principle. Also, algorithms from the next category can be used as well.

- Algorithms that target "huge" matrices. By huge matrices we consider those that make the auxiliary usage of linear systems' solvers inefficient. The only tool avail-

able to the algorithms in this group is the application of the matrix-times-vector operation.

These algorithms are variants of the Arnoldi algorithm and the Davidson algorithm, both equipped with some sort of restarting technique. For the positive definite Hermitian case, the LOBPCG algorithm of Knyazev [35] is applicable as well.

In this work we consider "large" matrices – those that belong to the last two categories. All known algorithms that can be applied to such matrices calculate the approximate eigenpairs according to the following general algorithm sketch:

Algorithm 2: General Subspace Iteration Algorithm

Input: Initial subspace $\mathcal{X}^{(1)}$, integers p and $maxiter$
Output: p eigenpair approximations

```
1 for k = 1, 2, ..., maxiter do
2 |   Select p eigenpair approximations from X^(k);
3 |   if all p approximations are converged then
4 |   |   Output the approximations and exit;
5 |   else
6 |   |   Build a new subspace X^(k+1) using information from X^(k);
7 |   end
8 end
9 Report misconvergence;
```

Techniques for extracting eigenpair approximations for a given subspace have been studied in Section 1.5. The procedure of choosing the next subspace in Line 6 is what makes the difference between various algorithms for large matrices – Table 9 shows how some well-known algorithms select this subspace. This chapter is devoted to studying algorithms based on so called Krylov subspaces

$$\mathcal{X}^{(k)} = \operatorname{span}\{v, Av, A^2v, \ldots, A^{k-1}v\},$$

which lead to the Arnoldi algorithm. However, for full understanding it is necessary to first consider the simplest possible case: $p = 1$ and all $\mathcal{X}^{(k)}$ are one-dimensional.

2.1 The power method

The basic algorithm for the large scale eigenvalue calculation is the power iteration method. It arises from the following observation: given a matrix $A \in \mathbb{C}^{n\times n}$ and an initial vector $x^{(1)}$, the sequence

$$x^{(k+1)} = \frac{Ax^{(k)}}{\left\|Ax^{(k)}\right\|}, \quad k = 1, 2, \ldots$$

Table 2.1: Some popular choices for approximate subspaces

Algorithm	$\mathcal{X}^{(k+1)}$	$\dim(\mathcal{X}^{(k)})$
power method	$\mathcal{X}^{(1)} = \text{span}\{x\}$, $\mathcal{X}^{(k+1)} = \text{span}\{A^k x\} = A\mathcal{X}^{(k)}$	1
Arnoldi, Lanczos	$\mathcal{X}^{(1)} = \text{span}\{x\}$, $\mathcal{X}^{(k+1)} = \text{span}\{\mathcal{X}^{(k)} \cup \{A^k x\}\}$	k
Davidson	$\mathcal{X}^{(1)} = \text{span}\{x\}$, $\mathcal{X}^{(k+1)} = \text{span}\{\mathcal{X}^{(k)} \cup \{t^{(k)}\}\}$, where: $M^{(k)} t^{(k)} = r^{(k)}$, $(\tilde{\lambda}^{(k)}, \tilde{u}^{(k)})$ is approximate eigenpair from $\mathcal{X}^{(k)}$, $M^{(k)} \approx A - \tilde{\lambda}^{(k)} I$, $r^{(k)} = A\tilde{u}^{(k)} - \tilde{\lambda}^{(k)}$	k
Jacobi-Davidson	$\mathcal{X}^{(1)} = \text{span}\{x\}$, $\mathcal{X}^{(k+1)} = \text{span}\{\mathcal{X}^{(k)} \cup \{t^{(k)}\}\}$, where: $P^{(k)}(A - \tilde{\lambda}^{(k)} I)P^{(k)} t^{(k)} = r^{(k)}$, $(\tilde{\lambda}^{(k)}, \tilde{u}^{(k)})$ is approximate eigenpair from $\mathcal{X}^{(k)}$, $P^{(k)} = I - \tilde{u}^{(k)}(\tilde{u}^{(k)})^\star$, $r^{(k)} = A\tilde{u}^{(k)} - \tilde{\lambda}^{(k)}\tilde{u}^{(k)}$	k
LOBPCG	$\mathcal{X}^{(0)} = \text{span}\{\tilde{u}_1^{(0)}, \ldots, \tilde{u}_p^{(0)}\}$, $\mathcal{X}^{(k+1)} = \text{span}\{\tilde{u}_1^{(k)}, \ldots, \tilde{u}_p^{(k)}, r_1{}^{(k)}, \ldots, r_p{}^{(k)}, z_1{}^{(k)}, \ldots, z_p{}^{(k)}\}$, where: $(\tilde{\lambda}_j^{(k)}, \tilde{u}_j^{(k)})$ are approximate eigenpairs from $\mathcal{X}^{(k)}$, $r_j{}^{(k)} = A\tilde{u}_j^{(k)} - \tilde{\lambda}_j^{(k)}\tilde{u}_j^{(k)}$, $z_j{}^{(k)} = \tilde{u}_j^{(k)} - \tilde{u}_j^{(k-1)}$	$3p$

converges to the eigenvector of A which is associated with the eigenvalue of the largest module (the *dominant* eigenvector/eigenpair). To see why this is the case, suppose A is diagonalizable with its distinct eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_p$ ordered in the following way:

$$|\lambda_1| > |\lambda_2| > \ldots > |\lambda_p|.$$

Note that eigenvalues of A still may be multiple. Let the initial vector have the following representation in the eigenvector basis:

$$x^{(1)} = \xi_1 u_1 + \xi_2 u_2 + \ldots + \xi_p u_p, \quad \xi_1 \neq 0,$$

Here

$$u_i = \frac{P_{\lambda_i} x^{(1)}}{\left\| P_{\lambda_i} x^{(1)} \right\|}$$

Algorithm 3: Power Method

Input: Initial unit vector $x^{(1)}$, integer *maxiter*
Output: Approximation for the dominant eigenpair

$t = Ax^{(1)}$;
for $k = 1, 2, \ldots, maxiter$ **do**
  Compute Rayleigh quotient $\theta = \left(x^{(k)}\right)^\star Ax^{(k)} = \left(x^{(k)}\right)^\star t$;
  if *the residual* $\left\|Ax^{(k)} - \theta x^{(k)}\right\| = \left\|t - \theta x^{(k)}\right\|$ *is small* **then**
    Output $(\theta, x^{(k)})$ as approximate eigenpair and exit;
  else
    $x^{(k+1)} = t/\left\|t\right\|$; $t = Ax^{(k+1)}$;
  end
end

are the eigenvectors and P_{λ_i} is the spectral projector associated with the eigenvalue λ_i. An important assumption is that $P_{\lambda_1} x^{(1)} \neq 0$.

Using this representation, we see that the $(k+1)$-th iteration $x^{(k+1)}$ is a vector having the same direction as

$$\begin{aligned}\frac{1}{\xi_1 \lambda_1^k} A^k x^{(1)} &= \frac{1}{\xi_1 \lambda_1^k}\left(\xi_1 \lambda_1^k u_1 + \xi_2 \lambda_2^k u_2 + \ldots + \xi_n \lambda_p^k u_p\right) \\ &= u_1 + \left(\frac{\xi_2}{\xi_1}\left(\frac{\lambda_2}{\lambda_1}\right)^k u_2 + \ldots + \frac{\xi_p}{\xi_1}\left(\frac{\lambda_p}{\lambda_1}\right)^k u_p\right) \\ &\xrightarrow{k} u_1.\end{aligned}$$

Thus, $x^{(k)} \xrightarrow{k} u_1$ and the rate of convergence is $\frac{|\lambda_2|}{|\lambda_1|}$. The same can be shown for defective matrices (see e.g. [59, page 111]), but λ_1 is still required to be semi-simple. The pseudocode for the power method is shown in Algorithm 3.

We point out several key issues with this algorithm, some of which will be transferred over to the more sophisticated algorithms later.

Use deflation to reach other dominant eigenpairs. If more eigenpairs are sought, one must still first calculate the approximation for the dominant eigenpair and then *deflate* it out of the problem once it is converged. There are several ways to achieve the deflation: let $(\tilde{\lambda}_1, \tilde{u}_1)$ be the converged approximation.

First, let

$$\hat{x}^{(1)} = x^{(1)} - \xi_1 u_1.$$

Suppose the power iterations are restarted with $\hat{x}^{(1)}$ as the initial vector. The component of this vector in the direction of u_1 is zero and thus the new sequence of power iterations converges to the next dominant eigenvector u_2. Nevertheless, in practice one can only use

$$\tilde{x}^{(1)} = x^{(1)} - \xi_1 \tilde{u}_1$$

instead of $\hat{x}^{(1)}$, for some $\tilde{u}_1 \approx u_1$. However tiny the component in the direction of u_1 initially is in $\tilde{x}^{(1)}$, it would build up eventually to regain domination and prevent convergence to u_2. The roundoff errors during calculation additionally emphasize this effect. This can be circumvented by annihilating the component in the direction of $\tilde{u}_1$ in each iteration. For example, if A is normal, one could project each iteration onto the orthogonal complement of $\text{span}\{\tilde{u}_1\}$, generating sequence

$$\tilde{x}^{(k+1)} = (I - \tilde{u}_1\tilde{u}_1^\star)A\tilde{x}^{(k)},$$

which may be viewed as the power iterations for the matrix $(I - \tilde{u}_1\tilde{u}_1^\star)A(I - \tilde{u}_1\tilde{u}_1^\star)$ – the compression of A on $\text{span}\{\tilde{u}_1\}^\perp$.

An alternative method to perform deflation is to alter the matrix A instead of the initial vector. Let z be any vector such that $z^\star u_1 = 1$ (typically $z = u_1$, but other choices are also viable). By setting

$$\hat{A} = A - \lambda_1 u_1 z^\star \tag{2.1}$$

we have erased the eigenpair (λ_1, u_1) and added the non-dominant eigenpair $(0, u_1)$ into the matrix $\hat{A}$:

$$\hat{A}u_1 = Au_1 - \lambda_1 u_1 z^\star u_1 = \lambda_1 u_1 - \lambda_1 u_1 = 0u_1.$$

For $j > 1$, the eigenvector u_j has changed into

$$\hat{u}_j = u_j - \left(\frac{\lambda_1}{\lambda_j} z^\star u_j\right) u_1,$$

leaving the associated eigenvalue λ_j intact:

$$\begin{aligned}
\hat{A}\hat{u}_j &= (A - \lambda_1 u_1 z^\star)\left(u_j - \left(\frac{\lambda_1}{\lambda_j} z^\star u_j\right) u_1\right) \\
&= Au_j - \left(\frac{\lambda_1}{\lambda_j} z^\star u_j\right) Au_1 - \lambda_1 u_1 z^\star u_j + \left(\frac{\lambda_1}{\lambda_j} z^\star u_j\right) \lambda_1 u_1 z^\star u_1 \\
&= \lambda_j u_j - \left(\frac{\lambda_1}{\lambda_j} z^\star u_j\right) \lambda_1 u_1 - \lambda_1 u_1 z^\star u_j + \left(\frac{\lambda_1}{\lambda_j} z^\star u_j\right) \lambda_1 u_1 \\
&= \lambda_j \left(u_j - \left(\frac{\lambda_1}{\lambda_j} z^\star u_j\right) u_1\right) \\
&= \lambda_j \hat{u}_j.
\end{aligned}$$

Thus, the dominant eigenpair of $\hat{A}$ is $(\lambda_2, \hat{u}_2)$ and restarting the power iteration method with $\hat{A}$ will give $\hat{u}_2$ from which u_2 can be reconstructed. This method is called the Wielandt deflation.

Use *shift-and-invert* to reach interior eigenvalues. If one seeks the interior eigenvalues or those close to some prescribed $\tau \in \mathbb{C}$, repeated deflation may take too long to reach the desired part of the spectrum. In such case it is better to resort to the *shift-and-invert* technique. If (λ, u) is an eigenpair of A, then $(\frac{1}{\lambda - \tau}, u)$ is an eigenpair of $(A - \tau I)^{-1}$. Thus, eigenvectors of $(A - \tau I)^{-1}$ are the same as those of A, but the eigenvalue

of A closest to τ is converted into the dominant one of $(A-\tau I)^{-1}$ and running the power method on $(A-\tau I)^{-1}$ will retrieve the desired eigenvector associated to it.

Note that in each iteration one has to calculate

$$x^{(k+1)} = \frac{(A-\tau I)^{-1}x^{(k)}}{\left\|(A-\tau I)^{-1}x^{(k)}\right\|} .$$

This is done by solving the linear system

$$(A-\tau I)y = x^{(k)},$$

by using an iterative sparse solver such as GMRES and then normalizing the solution y into $x^{(k+1)}$. The linear solver can be accelerated with preconditioning, e.g. the one obtained by the incomplete LU factorization of $A-\tau I$.

Multiple eigenvalues. If λ_1 is a multiple eigenvalue, then the Wielandt method of deflation erases only one instance of λ_1 and the matrix $\hat{A}$ in (2.1) still has λ_1 as its dominant eigenvalue. However, if the same vector $x^{(1)}$ is used as initial vector in power iterations with $\hat{A}$, the spectral projection $\hat{P}_{\lambda_1}x^{(1)}$ will be equal to zero. To see that, suppose that $\lambda_1,\ldots,\lambda_p$ are all distinct eigenvalues of a diagonalizable matrix A. The vector $x^{(1)}$ can be uniquely written as a sum of (not necessarily unit) eigenvectors corresponding to these eigenvalues:

$$x = u_1 + \sum_{j=2}^{p} u_j.$$

(Suppose for simplicity that $u_j \neq 0$ for all j.) Then $P_{\lambda_j}x = u_j$. We can rewrite x as the sum of eigenvectors for $\hat{A}$:

$$x = \left(1 + \sum_{j=2}^{p} \frac{\lambda_1}{\lambda_j} z^\star u_j\right) u_1 + \sum_{j=2}^{p} \hat{u}_j.$$

No eigenvector associated with the eigenvalue λ_1 of $\hat{A}$ participates in the sum: for $j = 2,\ldots,p$, the eigenvector $\hat{u}_j$ is associated with λ_j, and u_1 is associated with the eigenvalue 0. Therefore, $\hat{P}_{\lambda_1}x = 0$. This means that none of the vectors $\hat{A}^k x^{(1)}$ will have a component in eigenspace associated with λ_1 and thus the sequence will converge to an eigenvector of λ_2. One has to use some other initial vector if the multiplicity of λ_1 is of interest. The same conclusion is established when compression is used for the deflation.

Some distributions of eigenvalues may cause slow convergence. Since the algorithm converges with the rate of $\frac{|\lambda_2|}{|\lambda_1|}$, it is clear that the closer this ratio is to 1, the slower the convergence will be. Typically, if the absolute value of many eigenvalues is clustered close to $|\lambda_1|$, the convergence rate will deteriorate significantly. In the extreme case, all of the eigenvalues are on the same circle centered in origin and the algorithm does not converge at all. Since the eigenvalues are not known in advance, it is important to embed the algorithm with mechanisms that recognize occurrence of such unfavorable

situations and resort to some other tool that will restore convergence (such as the shift or shift-and-invert technique). In some cases, it is possible to tune the matrix on which the power iterations are run so that the convergence is faster. For example, in the PageRank algorithm described in the Section 1.3, we are free to set the parameter $\alpha \in \langle 0, 1\rangle$. This parameter is exactly the convergence rate of the power iterations. (Note that setting α to a very low value has negative consequences on the quality of computed page ranks.)

2.2 Simultaneous iteration

The idea of the power method can be generalized in order to obtain more than one eigenpair in the following natural way. Instead of iterating with a single vector, a subspace $\mathcal{X}^{(1)}$ of dimension m is chosen and the following sequence is considered:

$$\mathcal{X}^{(k+1)} = A\mathcal{X}^{(k)}. \tag{2.2}$$

Let the eigenvalues of A, ordered by decreasing module, be:

$$|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_m| > |\lambda_{m+1}| \geq \ldots \geq |\lambda_n|.$$

Similarly as in the power method, we expect that the components in directions of eigenvectors associated with the non-dominating eigenvalues $\lambda_{m+1}, \lambda_{m+2}, \ldots, \lambda_n$ decay as k increases, forcing the $\mathcal{X}^{(k)}$ to converge to the eigenspace associated with m dominating eigenvalues. The expected convergence rate is

$$\frac{|\lambda_{m+1}|}{|\lambda_m|}.$$

This statement is made precise in Theorem 2.1 within a more general setting.

From a practical point of view, $\mathcal{X}^{(k)}$ is represented with its basis stored as columns of some matrix $X^{(k)} \in \mathbb{C}^{n\times m}$. If (2.2) is simply translated as $X^{(k+1)} = AX^{(k)}$, then the j-th column of $X^{(k)}$ is a result of k steps of the power method run on j-th column of $X^{(1)}$ and thus converges to the dominant eigenvector associated with λ_1. This means that all columns of $X^{(k)}$ converge to the same vector, making the basis for $\mathcal{X}^{(k)}$ consist of vectors that become more and more colinear. In conditions of the finite arithmetic, the span of such a basis is not well defined. To resolve this issue, basis of $\mathcal{X}^{(k)}$ is kept orthogonal in each iteration by using the QR factorization. With this observation, we obtain the Algorithm 4: the simultaneous iteration.

If eigenpairs other than the dominant ones are sought, or in order to speed up the convergence, we can alter the matrix used in the iteration. Consider a polynomial $\pi \in \mathbb{P}_s$. For each eigenpair (λ, u) of A, there is an eigenpair $(\pi(\lambda), u)$ of $\pi(A)$. Let us order the eigenvalues $\pi(\lambda_1), \pi(\lambda_2), \ldots, \pi(\lambda_n)$ of $\pi(A)$ so that

$$|\pi(\lambda_1)| \geq |\pi(\lambda_2)| \geq \ldots \geq |\pi(\lambda_m)| > |\pi(\lambda_{m+1})| \geq \ldots \geq |\pi(\lambda_n)|.$$

If we run (2.2) with $\pi(A)$ instead of A:

$$\mathcal{X}^{(k+1)} = \pi(A)\mathcal{X}^{(k)}, \tag{2.3}$$

Algorithm 4: Simultaneous Iteration

Input: Initial basis $X^{(1)} \in \mathbb{C}^{n\times m}$, integer *maxiter*
Output: Approximation for the m dominant eigenpairs

$T = AX^{(1)}$;
for $k = 1, 2, \ldots, maxiter$ **do**
  Compute Rayleigh quotient $\Theta = \left(X^{(k)}\right)^\star AX^{(k)} = \left(X^{(k)}\right)^\star T$;
  Compute the eigenpairs $(\theta_1, y_1), (\theta_2, y_2), \ldots, (\theta_m, y_m)$ of Θ;
  if *residuals of all pairs* (θ_i, y_i) *are small enough* **then**
    report convergence and exit.
  else
    Compute the QR factorization of $AX^{(k)}$: $T = QR$;
    $X^{(k+1)} = Q$;
    $T = AX^{(k+1)}$;
  end
end

then the simultaneous iteration will find m dominant eigenpairs of $\pi(A)$. The expected ratio of convergence is

$$\frac{|\pi(\lambda_{m+1})|}{|\pi(\lambda_m)|},$$

which might be made very small by the appropriate choice of π.

When the polynomial π is represented as

$$\pi(\zeta) = (\zeta - \sigma_1)(\zeta - \sigma_2)\ldots(\zeta - \sigma_s),$$

we refer to the zeros $\sigma_1, \sigma_2, \ldots, \sigma_s$ as the *shifts*. Suppose that there is a shift close to some eigenvalue λ of A. Then $\pi(\lambda)$ is small, making $\pi(\lambda)$ non-dominant eigenvalue of $\pi(A)$ and the simultaneous iteration purging the associated eigenvector out of $\mathcal{X}^{(k)}$. Thus, if we have initial approximations for the some of the "unwanted" eigenvalues, we can use these approximations as shifts which would help remove the corresponding directions from $\mathcal{X}^{(k)}$. Even more: if during the process we somehow obtain better and better approximations for these unwanted eigenvalues, we might use a different polynomial $\pi^{(k)}$ in each iteration, hoping that the improved shifts would also improve the convergence. This results in a non-stationary iteration

$$\mathcal{X}^{(k+1)} = \pi^{(k)}(A)\mathcal{X}^{(k)}. \tag{2.4}$$

The convergence ratio for this final general form of the simultaneous iteration is given in the following Theorem.

Theorem 2.1. *[78, Theorem 5.1.3.] Let A be any complex matrix of order n and let $\pi \in \mathbb{P}_s$ be a polynomial of degree $s < n$. Denote with $\lambda_1, \lambda_2, \ldots, \lambda_n$ the eigenvalues of A ordered so that*

$$|\pi(\lambda_1)| \geq |\pi(\lambda_2)| \geq \ldots \geq |\pi(\lambda_n)|,$$

and let $|\pi(\lambda_m)| > |\pi(\lambda_{m+1})|$ *for some* $m < n$. *Suppose* $\mathcal{U}$ *and* $\mathcal{V}$ *are the eigenspaces of* A *associated with* $\{\lambda_1, \ldots, \lambda_m\}$ *and* $\{\lambda_{m+1}, \ldots, \lambda_n\}$ *respectively. Consider a sequence of polynomials* $\pi^{(k)} \in \mathbb{P}_s$ *such that*

$$\lim_{k\to\infty} \pi^{(k)} = \pi$$

and $\pi^{(k)}(\lambda_j) \neq 0$ *for all* $j \leq m$. *If* $\mathcal{X}^{(1)}$ *is an* m*-dimensional subspace of* $\mathbb{C}^n$ *such that* $\mathcal{X}^{(1)} \cap \mathcal{V} = \{0\}$ *then there exists a constant* γ *so that the elements of the sequence*

$$\mathcal{X}^{(k+1)} = \pi^{(k)}(A)\mathcal{X}^{(k)}$$

satisfy

$$\sin \measuredangle(\mathcal{U}, \mathcal{X}^{(k+1)}) \leq \gamma \rho^k,$$

where $\rho = \frac{|\pi(\lambda_{m+1})|}{|\pi(\lambda_m)|}$.

Proof. To keep the exposition simple, we prove only the stationary case $\pi^{(k)} = \pi$ for normal matrices A. Let $A = Q\Lambda Q^\star$ be the spectral decomposition of A and let $Q = [Q_1 \; Q_2]$ where $Q_1 \in \mathbb{C}^{n\times m}$ contains the orthonormal basis for the eigenspace $\mathcal{U}$ associated with $\{\lambda_1, \lambda_2, \ldots, \lambda_m\}$. Then

$$\begin{aligned}
\measuredangle(\mathcal{X}^{(k+1)}(A),\, \mathcal{U}) &= \measuredangle(\pi^k(A)\mathcal{X}^{(1)},\, \mathcal{U}) \\
&= \measuredangle(Q\pi^k(\Lambda)Q^\star\mathcal{X}^{(1)},\, \mathcal{U}) \\
&= \measuredangle(\pi^k(\Lambda)Q^\star\mathcal{X}^{(1)},\, Q^\star\mathcal{U}).
\end{aligned}$$

To compute this angle, we need bases for subspaces $\pi^k(\Lambda)Q^\star\mathcal{X}^{(1)}$ and $Q^\star\mathcal{U}$. The basis for $Q^\star\mathcal{U}$ is simply $\left[\begin{smallmatrix} I_m \\ 0 \end{smallmatrix}\right]$, where I_m is the identity matrix of order m. Let $\left[\begin{smallmatrix} X_1 \\ X_2 \end{smallmatrix}\right] \in \mathbb{C}^{n\times m}$ be a basis for $Q^\star\mathcal{X}^{(1)}$ partitioned so that X_1 has m rows. Note that X_1 is regular, since otherwise there would exist a non-trivial vector in $\mathcal{X}^{(1)} \cap \mathcal{V}$. The columns of

$$\begin{bmatrix} I_m \\ \pi^k(\Lambda_2)(X_2X_1^{-1})\pi^{-k}(\Lambda_1) \end{bmatrix}$$

make the basis for the subspace $\pi^k(\Lambda)Q^\star\mathcal{X}^{(1)}$, since

$$\begin{aligned}
\pi^k(\Lambda)\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} &= \begin{bmatrix} \pi^k(\Lambda_1) & 0 \\ 0 & \pi^k(\Lambda_2) \end{bmatrix}\begin{bmatrix} I_m \\ X_2X_1^{-1} \end{bmatrix} X_1 \\
&= \begin{bmatrix} I_m \\ \pi^k(\Lambda_2)(X_2X_1^{-1})\pi^{-k}(\Lambda_1) \end{bmatrix} \pi^k(\Lambda_1)X_1.
\end{aligned}$$

Here $\Lambda_1 = \operatorname{diag}(\lambda_1, \ldots, \lambda_m)$ and $\Lambda_2 = \operatorname{diag}(\lambda_{m+1}, \ldots, \lambda_n)$. Finally, it is easy to show that

$$\begin{aligned}
\sin \measuredangle(\mathcal{X}^{(k+1)}(A),\, \mathcal{U}) &\leq \tan \measuredangle(\mathcal{X}^{(k+1)}(A),\, \mathcal{U}) \\
&= \left\|\pi^k(\Lambda_2)(X_2X_1^{-1})\pi^{-k}(\Lambda_1)\right\| \\
&\leq \left\|X_2X_1^{-1}\right\| \cdot \left\|\pi^{-k}(\Lambda_1)\right\| \cdot \left\|\pi^k(\Lambda_2)\right\| \\
&= \left\|X_2X_1^{-1}\right\| \left(\frac{|\pi(\lambda_{m+1})|}{|\pi(\lambda_m)|}\right)^k,
\end{aligned}$$

which completes the proof. □

2.2.1 The QR algorithm

Although the simultaneous iteration is an important algorithm itself in the theory of large scale eigenvalue problems, the main reason we introduced it was to study its special case when the dimension of approximate subspaces $\mathcal{X}^{(k)}$ is equal to the dimension of the matrix A, i.e. the case $m = n$. This will ultimately lead to the implicitly shifted QR algorithm, one of the most used methods for the small scale problems. In perspective, the QR algorithm will serve us as an essential tool for restarting the Arnoldi method studied in depth later in the text. The matrix A in this section is considered to be "small".

Let us rewrite the simultaneous iteration once again, this time with matrices $X^{(k)}$ being unitary of order n:

$$\pi(A)X^{(k)} = X^{(k+1)}R^{(k)}, \quad \text{for } k = 1, 2, \ldots \tag{2.5}$$

We take $X^{(1)} = I_n$. From the property of the QR factorization it follows that the first j columns of the matrix $X^{(k+1)}$ span the same subspace as the first j columns of $\pi(A)X^{(k)}$, for each j. In the limit, the first j columns of $X^{(k)}$ will span the eigenspace associated with the eigenvalues $\{\lambda_1, \lambda_2, \ldots, \lambda_j\}$ if $\pi(\lambda_j) \neq \pi(\lambda_{j+1})$. This is because $X^{(k)}(:, 1 : j)$ is equal to the matrix obtained by the simultaneous iteration started with $X^{(1)}(:, 1 : j)$. Thus the Rayleigh quotient matrix

$$A^{(k)} = (X^{(k)})^\star A X^{(k)}$$

will converge to the block upper triangular form: let $X^{(k)} = [X_1^{(k)}\ X_2^{(k)}]$ with $X_1^{(k)}$ having j columns. Then $AX_1^{(k)} \approx X_1^{(k)}T^{(k)}$ for some matrix $T^{(k)}$ and

$$\begin{aligned} A^{(k)} &= \begin{bmatrix} (X_1^{(k)})^\star \\ (X_2^{(k)})^\star \end{bmatrix} \left(A[X_1^{(k)}\ X_2^{(k)}] \right) \\ &\approx \begin{bmatrix} (X_1^{(k)})^\star \\ (X_2^{(k)})^\star \end{bmatrix} \left[X_1^{(k)}T^{(k)}\ AX_2^{(k)} \right] \\ &= \begin{bmatrix} T^{(k)} & (X_1^{(k)})^\star A X_2^{(k)} \\ 0 & (X_2^{(k)})^\star A X_2^{(k)} \end{bmatrix}. \end{aligned}$$

This effect occurs for each j such that $\pi(\lambda_j) \neq \pi(\lambda_{j+1})$, so the matrix $A^{(k)}$ will ultimately have a block triangular form. Each block will correspond to a single eigenvalue of A; its size will be equal to the algebraic multiplicity of the eigenvalue. If A has only simple eigenvalues, then $A^{(k)}$ will have a triangular form in the limit.

Also note that $A^{(k)}$ is unitarily similar to A and thus has the same eigenvalues as A. In the limit, these eigenvalues will occur on the diagonal of $A^{(k)}$. Thus, in order to calculate the eigenvalues of A, it suffices to track the matrices $A^{(k)}$. There is a simple reformulation of (2.5) that shows how $A^{(k)}$ changes to $A^{(k+1)}$. First, observe that

$$\begin{aligned} \pi(A^{(k)}) &= \pi((X^{(k)})^\star A X^{(k)}) = (X^{(k)})^\star \pi(A) X^{(k)} \\ &= (X^{(k)})^\star X^{(k+1)} R^{(k)} = Q^{(k)} R^{(k)} \end{aligned}$$

ALGORITHM 5: Explicitly Shifted QR Algorithm

Input: integer *maxiter*
Output: Approximation for the all eigenpairs of matrix A

$X^{(1)} = I_n$; $A^{(1)} = A$;
for $k = 1, 2, \ldots, maxiter$ **do**
  if *all elements under the diagonal of* $A^{(k)}$ *are small enough* **then**
    report convergence and exit with diagonal elements of $A^{(k)}$ as the eigenvalues and $X^{(k)}(:, 1:j)$ as bases for eigenspaces;
  end
  Select s shifts $\sigma_1, \sigma_2, \ldots, \sigma_s$;
  Set $\pi(A^{(k)}) = (A^{(k)} - \sigma_1 I)(A^{(k)} - \sigma_2 I) \ldots (A^{(k)} - \sigma_s I)$;
  Compute the QR factorization $\pi(A^{(k)}) = Q^{(k)} R^{(k)}$;
  $A^{(k+1)} = (Q^{(k)})^\star A^{(k)} Q^{(k)}$;
  Accumulate $X^{(k+1)} = X^{(k)} Q^{(k)}$;
end

is a QR factorization with $Q^{(k)} = (X^{(k)})^\star X^{(k+1)}$ and that

$$\begin{aligned} A^{(k+1)} &= (X^{(k+1)})^\star A X^{(k+1)} = (X^{(k+1)})^\star X^{(k)} A^{(k)} (X^{(k)})^\star X^{(k+1)} \\ &= (Q^{(k)})^\star A^{(k)} Q^{(k)}. \end{aligned}$$

Now (2.5) can be rewritten as

$$\left.\begin{aligned} \pi(A^{(k)}) &= Q^{(k)} R^{(k)} \\ A^{(k+1)} &= (Q^{(k)})^\star A^{(k)} Q^{(k)} \end{aligned}\right\} \quad \text{for } k = 1, 2, \ldots, \tag{2.6}$$

where $A^{(1)} = A$. Using the process (2.6) we can compute the eigenvalues of A as the diagonal elements of $A^{(k)}$ in the limit. The eigenvectors can be reconstructed from the product $X^{(k)} = Q^{(1)} Q^{(2)} \ldots Q^{(k-1)}$. This is the *explicitly shifted* QR algorithm with pseudocode written as Algorithm 5; note once again that we can make a different choice of shifts in each step.

To track progression of the convergence, it is useful to monitor sizes of elements under the diagonal of $A^{(k)}$. This task is particularly easy if $A^{(k)}$ is in the upper Hessenberg form: it that case, only elements of the first subdiagonal have to be monitored. One very important property of (2.6) is the invariance of the upper Hessenberg form – it is a simple exercise to show that if $\pi(A)$ is non-singular and $A^{(k)}$ has this form, then so will $A^{(k+1)} = R^{(k)} A^{(k)} (R^{(k)})^{-1}$. Before starting the process, any initial matrix $A^{(1)} = A$ can be made upper Hessenberg in $\mathcal{O}(n^3)$ time by using a series of unitary similarities with Householder reflectors. Then, during the whole process, all matrices $A^{(k)}$ will be upper Hessenberg.

Efficiency of the QR algorithm is another particularly important reason for keeping matrices $A^{(k)}$ in the upper Hessenberg form. Suppose for a moment that π is a polynomial

of degree one. A single step of the process (2.6) takes $\mathcal{O}(n^3)$ flops: this is the complexity of both the QR factorization and the similarity transformation. However, when $A^{(k)}$ is upper Hessenberg, this single step can be done with only $\mathcal{O}(n^2)$ flops! To achieve this effect, we make use of the following theorem.

Theorem 2.2 (Implicit-Q theorem)**.** *Let $A \in \mathbb{C}^{n\times n}$. Suppose Q is a unitary matrix such that $H = Q^\star AQ$ is an unreduced upper Hessenberg matrix (i.e. $H_{j+1,j} \neq 0$ for all j). Then the matrices H and Q are uniquely determined by the first column of Q, up to the scaling of columns of Q by unitary complex numbers.*

Proof. Suppose Q_1 and Q_2 are unitary matrices such that $Q_1^\star AQ_1 = H_1$ and $Q_2^\star AQ_2 = H_2$ are both unreduced upper Hessenberg and that $Q_1e_1 = Q_2e_1 = q$. We need to prove that columns of Q_1 and Q_2 are the same up to the unitary scaling, i.e. that the matrix $W = Q_1^\star Q_2$ is diagonal with unitary complex numbers on the diagonal. It holds that:

$$WH_2 = Q_1^\star Q_2 H_2 = Q_1^\star AQ_2 = H_1 Q_1^\star Q_2 = H_1 W.$$

Consider the i-th column of H_1W, for $i < n$:

$$H_1(We_i) = WH_2e_i = W\sum_{j=1}^{i+1}(H_2)_{j,i}e_j = (H_2)_{i+1,i}We_{i+1} + \sum_{j=1}^{i}(H_2)_{j,i}We_j,$$

that is,

$$(H_2)_{i+1,i}We_{i+1} = H_1(We_i) - \sum_{j=1}^{i}(H_2)_{j,i}We_j\ . \tag{2.7}$$

Since $(H_2)_{i+1,i} \neq 0$ and H_1 is upper Hessenberg and $We_1 = Q_1^\star Q_2 e_1 = Q_1^\star q = e_1$, by easy induction from (2.7) it follows that only first $i+1$ components of vector We_{i+1} can be non-zero. This means W is an upper triangular matrix; since it is also unitary, it has to hold $W = \operatorname{diag}(e^{i\varphi_1}, e^{i\varphi_2}, \ldots, e^{i\varphi_n})$, for some $\varphi_1, \ldots, \varphi_n \in [0, 2\pi\rangle$. □

The Implicit-Q theorem implies that in order to compute $A^{(k+1)}$ it suffices to find a sequence of unitary matrices $W_1, W_2, \ldots, W_{n-1}$ such that

$$W_{n-1}^\star \ldots W_1^\star A^{(k)} W_1 \ldots W_{n-1}$$

has the upper Hessenberg form while the first column of $W_1W_2\ldots W_{n-1}$ is the same as the first column of $Q^{(k)}$. We will track the procedure for selecting W_j on a 5×5 example with $\pi(A) = A - \sigma I$ and comment on the general case where π is of degree s. The procedure can be done entirely in real arithmetic if $A^{(k)}$ and π are real.

Let $A^{(k)} = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} & \alpha_{14} & \alpha_{15} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} & \alpha_{24} & \alpha_{25} \\ & \alpha_{32} & \alpha_{33} & \alpha_{34} & \alpha_{35} \\ & & \alpha_{43} & \alpha_{44} & \alpha_{45} \\ & & & \alpha_{54} & \alpha_{55} \end{bmatrix}$; then the first column of the matrix $Q^{(k)}$ is a unit vector in direction $\begin{bmatrix} \alpha_{11}-\sigma \\ \alpha_{21} \\ 0 \\ 0 \\ 0 \end{bmatrix}$. Let

$$\xi = \frac{\alpha_{11}-\sigma}{|\alpha_{11}-\sigma|},\ c_1 = \frac{|\alpha_{11}-\sigma|}{\sqrt{|\alpha_{11}-\sigma|^2 + |\alpha_{21}|^2}},\ s_1 = \frac{\overline{\xi}\cdot\alpha_{21}}{\sqrt{|\alpha_{11}-\sigma|^2 + |\alpha_{21}|^2}}.$$

The vector $[c_1\ s_1\ 0\ 0\ 0]^\tau$ is valid as the first column of the matrix $Q^{(k)}$. Put

$$W_1 = \begin{bmatrix} c_1 & -\overline{s_1} & & & \\ s_1 & c_1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix},\ \text{then } W_1^\star A^{(k)} W_1 = \begin{bmatrix} \alpha_{11}^{(1)} & \alpha_{12}^{(1)} & \alpha_{13}^{(1)} & \alpha_{14}^{(1)} & \alpha_{15}^{(1)} \\ \alpha_{21}^{(1)} & \alpha_{22}^{(1)} & \alpha_{23}^{(1)} & \alpha_{24}^{(1)} & \alpha_{25}^{(1)} \\ \gamma_{31} & \alpha_{32}^{(1)} & \alpha_{33} & \alpha_{34} & \alpha_{35} \\ & & \alpha_{43} & \alpha_{44} & \alpha_{45} \\ & & & \alpha_{54} & \alpha_{55} \end{bmatrix}.$$

(In the general case, only the first $s+1$ components of $q_1 = \pi(A^{(k)})e_1$ can be non-zero, since $A^{(k)}$ is upper Hessenberg. We choose any $(s+1)\times(s+1)$ unitary matrix $\hat{W}_1$ whose first column is a scalar multiple of q_1 and set $W_1 = \hat{W}_1 \oplus I_{n-s-1}$.)

The only goal of other unitary transformations $W_2, \ldots, W_{n-1}$ is to restore the upper Hessenberg form. At first, we choose W_2 to remove the "bulge" γ_{31}.

Let $\hat{W}_2^\star$ be a complex 2×2 Householder reflector that transforms vector $x = \begin{bmatrix} \alpha_{21}^{(1)} \\ \gamma_{31} \end{bmatrix}$ into a multiple of e_1:

$$z = x \pm \|x\|\, e_1; \quad \hat{W}_2^\star = I_2 - \left(1 + \frac{x^\star z}{z^\star x}\right) \cdot \frac{z z^\star}{z^\star z}.$$

Now we have:

$$W_2 = \begin{bmatrix} 1 & & & \\ & \hat{W}_2 & & \\ & & 1 & \\ & & & 1 \end{bmatrix},\ \text{thus } W_2^\star W_1^\star A^{(k)} W_1 W_2 = \begin{bmatrix} \alpha_{11}^{(1)} & \alpha_{12}^{(2)} & \alpha_{13}^{(2)} & \alpha_{14}^{(1)} & \alpha_{15}^{(1)} \\ \alpha_{21}^{(2)} & \alpha_{22}^{(2)} & \alpha_{23}^{(2)} & \alpha_{24}^{(2)} & \alpha_{25}^{(2)} \\ & \alpha_{32}^{(2)} & \alpha_{33}^{(2)} & \alpha_{34}^{(2)} & \alpha_{35}^{(2)} \\ & \gamma_{42} & \alpha_{43}^{(2)} & \alpha_{44} & \alpha_{45} \\ & & & \alpha_{54} & \alpha_{55} \end{bmatrix}.$$

Note how unitary similarity with W_2 affects only the second and the third row and column. The bulge has moved to element at $(4, 2)$.

(In the general case, the lower triangle of $(W_1^\star A^{(k)} W_1)(2 : s+2,\ 1 : s+1)$ is a "bulge" of size $s \times s$ that disturbs it from being upper Hessenberg. In the first column there are s elements that "stick" under the first subdiagonal. We construct $\hat{W}_2^\star$ as the $(s+1)\times(s+1)$ Householder reflector that transforms $(W_1^\star A^{(k)} W_1)(2 : s+2,\ 1)$ into a multiple of e_1 and set $W_2 = I_1 \oplus \hat{W}_2 \oplus I_{n-s-2}$. This clears the bulge from the first column, but now there are s elements that "stick" under the first subdiagonal in the second column.)

The bulge γ_{42} from the second column is removed in the same manner: $\hat{W}_3^\star$ is a Householder reflector that maps $\begin{bmatrix} \alpha_{32}^{(2)} \\ \gamma_{42} \end{bmatrix}$ into a multiple of e_1. Then we have

$$W_3 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \hat{W}_3 & \\ & & & 1 \end{bmatrix};\ \text{thus } W_3^\star W_2^\star W_1^\star A^{(k)} W_1 W_2 W_3 = \begin{bmatrix} \alpha_{11}^{(1)} & \alpha_{12}^{(2)} & \alpha_{13}^{(3)} & \alpha_{14}^{(3)} & \alpha_{15}^{(1)} \\ \alpha_{21}^{(2)} & \alpha_{22}^{(2)} & \alpha_{23}^{(3)} & \alpha_{24}^{(3)} & \alpha_{25}^{(2)} \\ & \alpha_{32}^{(3)} & \alpha_{33}^{(3)} & \alpha_{34}^{(3)} & \alpha_{35}^{(3)} \\ & & \alpha_{43}^{(3)} & \alpha_{44}^{(3)} & \alpha_{45}^{(3)} \\ & & \gamma_{53} & \alpha_{54}^{(3)} & \alpha_{55} \end{bmatrix}.$$

Finally, by choosing a Householder reflector $\hat{W}_4^\star$ that transforms $\begin{bmatrix} \alpha_{43}^{(3)} \\ \gamma_{53} \end{bmatrix}$ into a multiple of

Algorithm 6: Implicitly Shifted QR Algorithm

Input: upper Hessenberg matrix A, integer *maxiter*
Output: Schur factorization of matrix A

```
k = 0; Q = I_n;
while k < n do
    Select a shift σ ∈ ℂ;
    Transform A and Q by calling Implicit Shift(A, σ, Q);
    if |A_{n,n-1}| is sufficiently small then
        k = k + 1;
        λ_k = A_{n,n-1};
        From now on act on submatrices A = A(1 : n − 1, 1 : n − 1) and
        Q = Q(:, 1 : n − 1);
        n = n − 1;
    end
end
Schur form: A and Q contain the triangular and the unitary factor, resp.
```

e_1 we end up with an upper Hessenberg matrix:

$$W_4 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & \\ & & \hat{W}_4 & \end{bmatrix}, \text{ and thus}$$

$$W_4^\star W_3^\star W_2^\star W_1^\star A^{(k)} W_1 W_2 W_3 W_4 = \begin{bmatrix} \alpha_{11}^{(1)} & \alpha_{12}^{(2)} & \alpha_{13}^{(3)} & \alpha_{14}^{(4)} & \alpha_{15}^{(4)} \\ \alpha_{21}^{(2)} & \alpha_{22}^{(2)} & \alpha_{23}^{(3)} & \alpha_{24}^{(4)} & \alpha_{25}^{(4)} \\ & \alpha_{32}^{(3)} & \alpha_{33}^{(3)} & \alpha_{34}^{(4)} & \alpha_{35}^{(4)} \\ & & \alpha_{43}^{(4)} & \alpha_{44}^{(4)} & \alpha_{45}^{(4)} \\ & & & \alpha_{54}^{(4)} & \alpha_{55}^{(4)} \end{bmatrix}.$$

(In the general case, last s Householder reflectors will have dimensions $s+1$, s, ..., 2 respectively.)

Due to the structure of matrices, only transformation W_1 affects the first column of the matrix $W_1 W_2 \dots W_{n-1}$, so the first column of this product is the same as the first column of $Q^{(k)}$. Since $W_{n-1}^\star \dots W_1^\star A^{(k)} W_1 \dots W_{n-1}$ is upper Hessenberg, by the Implicit-Q theorem this is the same matrix as $A^{(k+1)}$.

Each unitary similarity affects only $(s+1)$ rows and columns and can be applied with only $\mathcal{O}(sn)$ flops. Thus the transition from $A^{(k)}$ to $A^{(k+1)}$ can be carried in only $\mathcal{O}(sn^2)$ flops. This *implicitly shifted* QR algorithm for π of degree 1 is summarized in Algorithms 6 and 7.

Algorithm 6 monitors the size of $A^{(k)}(n, n-1)$. If this element is small, it can be deflated to zero and the element $A^{(k)}(n,n)$ considered the eigenvalue of $A^{(k)}$, since

$$A^{(k)} \approx \begin{bmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ 0 & \tilde{\lambda}_n \end{bmatrix},$$

where $A_{11}^{(k)}$ is the leading $(n-1) \times (n-1)$ submatrix. The QR algorithm is then restarted with $A_{11}^{(k)}$, which in turn computes λ_{n-1}, etc.

ALGORITHM 7: Implicit Shift(A, σ, Q)

Input: upper Hessenberg matrix A, shift $\sigma \in \mathbb{C}$, unitary matrix Q
Output: upper Hessenberg $\tilde{A}$, unitary $\tilde{Q}$
`// if` $Q = I$`, then` $\tilde{A} = \tilde{Q}^\star A \tilde{Q}$`, where` $A - \sigma I = \tilde{Q}R$

$\tilde{A} = A$; $\tilde{Q} = Q$;
Generate 2×2 unitary $\hat{W}_1$ whose first column is multiple of $\begin{bmatrix} \tilde{A}_{11} - \sigma \\ \tilde{A}_{21} \end{bmatrix}$;
for $i = 1, 2, \ldots, n-1$ **do**
    `// transform rows of` $\tilde{A}$`:`
    $\tilde{A}(i : i+1, :) = \hat{W}_i^\star \tilde{A}(i : i+1, :)$;
    `// transform columns of` $\tilde{A}$`:`
    $\tilde{A}(:, i : i+1) = \tilde{A}(:, i : i+1)\hat{W}_i$;
    `// accumulate unitary transformation` $\hat{W}_i$ `in` $\tilde{Q}$`:`
    $\tilde{Q}(:, i : i+1) = \tilde{Q}(:, i : i+1)\hat{W}_i$;
    `// next Householder reflector to annihilate bulge` $\tilde{A}_{i+2,i}$`:`
    if $i < n-1$ **then**
        Generate 2×2 unitary $\hat{W}_{i+1}$ such that $\hat{W}_{i+1}^\star \begin{bmatrix} \tilde{A}_{i+1,i} \\ \tilde{A}_{i+2,i} \end{bmatrix}$ is a multiple of e_1;
    end
end

Typically only very few calls to IMPLICIT SHIFT are needed for an eigenvalue to converge. For example, Parlett [52] states that experimental results show that if A is a symmetric matrix then in average 1.7 calls suffice for the convergence of a single eigenvalue, with the appropriate choice of shifts. Thus we can recover all eigenpairs of a matrix using $\mathcal{O}(n^3)$ flops.

When the matrix A is real, it may still be necessary to use the complex shifts. In order to avoid the complex arithmetic, the *double-shifting* technique is deployed: for a given complex number σ, an algorithm analogous to IMPLICIT SHIFT simultaneously applies shifts σ and $\overline{\sigma}$ without ever leaving the real arithmetic. We omit the details on this and many other topics on both theoretical and practical aspects of a modern QR algorithm such as the shift selection, the deflation criteria and the convergence theory. An excellent overview of all this and more can be found in Watkins's book [78].

2.3 Krylov subspaces

The power iteration method described in the last section takes some initial vector $x^{(0)}$ and generates a sequence of vectors having directions $x^{(0)}, Ax^{(0)}, A^2x^{(0)}, \ldots$ In step k of the method, approximation of the dominant eigenvector is a scalar multiple of $A^k x^{(0)}$.

The obvious shortcoming is that the information gathered in all previous steps are thrown away and does not influence the approximation in step k in any way.

The Arnoldi algorithm generates the same sequence of vectors as the power method does, but in each step it considers a subspace $\mathcal{X}$ spanned by all these vectors. Using the Rayleigh-Ritz procedure, approximations for several eigenvectors are drawn from $\mathcal{X}$. Since the most time consuming operation of algorithms for the large scale eigenvalue problem is the matrix-times-vector operation Ax, a more complicated extraction of approximations at a single step is well justified if their quality is significantly better. To demonstrate that this is the case, we need to study the Krylov subspaces.

Definition 2.3. *Let $A \in \mathbb{C}^{n\times n}$ and $v \in \mathbb{C}^n$. For $m = 1, 2, \ldots$, the subspaces*

$$\mathcal{K}_m(v; A) = \operatorname{span}\{v, Av, A^2v, \ldots, A^{m-1}v\}$$

are called the Krylov subspace*. If it is clear from context which matrix or vector is concerned, we write $\mathcal{K}_m(v)$ or $\mathcal{K}_m$.*

From the definition we immediately get that

$$\mathcal{K}_m(v; A) = \{\pi(A)v \ : \ \pi \in \mathbb{P}_{m-1}\}, \tag{2.8}$$

where $\mathbb{P}_{m-1}$ is the set of all polynomials of degree $m-1$ or less. Also note that for a fixed matrix A there exists some $k \leq n$ which depends on starting vector v such that

$$\mathcal{K}_1(v) \subset \mathcal{K}_2(v) \subset \mathcal{K}_3(v) \subset \ldots \subset \mathcal{K}_k(v) = \mathcal{K}_{k+1}(v) = \ldots \tag{2.9}$$

There is an interesting connection between Krylov subspaces and the minimal polynomial in A that annihilates vector v.

Proposition 2.4.

(a) For k defined by (2.9), the subspace $\mathcal{K}_k(v)$ is the smallest A-invariant subspace of $\mathbb{C}^n$ that contains vector v.

(b) Let μ be the polynomial of smallest possible degree such that $\mu(A)v = 0$. Then $\deg(\mu) = k$.

(c) If A is diagonalizable, then

$$\mathcal{K}_k(v) = \operatorname{span}\{P_\lambda v \ : \ P_\lambda v \neq 0, \lambda \in \Lambda(A)\},$$

where P_λ is the spectral projector associated with the eigenvalue λ.

Proof. Clearly $\mathcal{K}_k(v)$ is A-invariant. The smallest A-invariant subspace that contains v must also contain all of the vectors $Av, A^2v, \ldots$ Thus the claim in *(a)* is evident.

To show *(b)*, let m denote the degree of polynomial μ. Then for some coefficients $\xi_0, \xi_1, \ldots, \xi_p$ it holds that

$$\mu(A)v = \xi_0 v + \xi_1 Av + \ldots + \xi_p A^p v = 0,$$

with $\xi_p \neq 0$. We can express

$$A^p v = \frac{-1}{\xi_p} \left(\xi_0 v + \xi_1 A v + \ldots + \xi_{p-1} A^{p-1} v \right),$$

meaning that $A^p v \in \mathcal{K}_p(v)$. It follows that $\mathcal{K}_p(v) = \mathcal{K}_{p+1}(v)$ and thus $p \geq k$. To show the opposite, note that since $\mathcal{K}_k(v) = \mathcal{K}_{k+1}(v)$, the vector $A^k v$ belongs to subspace $\mathcal{K}_k(v)$ and is a linear combination of $v, Av, \ldots, A^{k-1}v$. Then 0 is a linear combination of $v, Av, \ldots, A^{k-1}v, A^k v$ and there is a polynomial of degree k which annihilates v. Thus $k \geq p$.

To show *(c)*, notice first that $AP_\lambda v = \lambda P_\lambda v$ since A is diagonalizable, and that

$$v = \sum_{\substack{\lambda \in \Lambda(A) \\ P_\lambda v \neq 0}} P_\lambda v.$$

Thus $\mathcal{X} := \text{span}\{P_\lambda v \ : \ P_\lambda v \neq 0, \lambda \in \Lambda(A)\}$ is a subspace of $\mathcal{K}_k(v)$. To see that they are equal, it suffices to prove that the dimension if $\mathcal{X}$ is less than or equal to k. For the polynomial μ from *(b)* it holds that

$$0 = \mu(A)v = \sum_{\substack{\lambda \in \Lambda(A) \\ P_\lambda v \neq 0}} \mu(\lambda) P_\lambda v.$$

Since the vectors $P_\lambda v$ are linearly independent, if follows that $\mu(\lambda) = 0$ for all λ such that $P_\lambda v \neq 0$. The polynomial μ has degree k, and thus the number of such λ's and the dimension of $\mathcal{X}$ must be less than of equal to k. □

In Section 1.5.2 we have shown that Ritz vectors from an invariant subspace are eigenvectors of the original matrix. Proposition 2.4 instructs on how to reach the minimal invariant subspace that contains a prescribed vector v. However, in practice v is chosen at random and thus contains components from all eigenspaces, so it is unlikely to expect k to be small compared to the dimension of the matrix A. The computation time to reach the invariant subspace is much too high, so approximations from a lower dimensional Krylov subspace will have to suffice.

The *(c)* part of the Proposition puts one limitation on any algorithm approximating eigenvectors from a Krylov subspace: for every vector $x \in \mathcal{K}_k(v)$ and every eigenvalue $\lambda \in \Lambda(A)$, the projection $P_\lambda x$ is a scalar multiple of $P_\lambda v$. This means that if λ is not a simple eigenvalue, such an algorithm would not be capable of getting any eigenvector associated with λ other than the one in the direction of $P_\lambda v$. Only the roundoff errors during the calculation might remedy the situation by introducing components of other eigenvectors associated with λ into the Krylov subspace. The proper way of handling with such cases is the use of deflation or restarting, more of which will be discussed later in the text. Also, algorithms using *block Krylov subspaces* in which vector v is substituted by a full-column-rank matrix V are also able to retrieve multiple eigenvalues.

Next, we list some simple invariance properties of the Krylov subspaces.

Proposition 2.5. *For arbitrary $\tau, \sigma \in \mathbb{C}$, a non-singular matrix W and all $j = 1, 2, \ldots$ the following holds:*

(a) $\mathcal{K}_j(\tau v; \sigma A) = \mathcal{K}_j(v; A)$, *where* $\tau, \sigma \neq 0$;

(b) $\mathcal{K}_j(v; A - \sigma I) = \mathcal{K}_j(v; A)$;

(c) $\mathcal{K}_j(Wv; WAW^{-1}) = W\mathcal{K}_j(v; A)$.

This Proposition shows that we may somewhat restrict the class of matrices in order to study the theory of Krylov subspaces and the convergence of arising algorithms. The first two properties allow us to select any circle in the complex plane and place all eigenvalues of A inside it by shifting and scaling the matrix – these actions will keep the Krylov subspaces intact, as if we were working with the original matrix from the start. The third property shows that changing the basis does not affect relations between vectors in $\mathbb{C}^n$ and those from $\mathcal{K}_j$. For example, if W is unitary, angles and norms will be preserved. Thus if A is diagonalizable, we may analyze all the effects in $\mathcal{K}_j(\cdot; A)$ by analyzing $\mathcal{K}_j(\cdot; \Lambda)$, where Λ is the diagonal matrix containing the eigenvalues of A.

2.3.1 Ritz approximations from Krylov subspaces

To compute Ritz approximations from $\mathcal{K}_m$, one has to construct an orthonormal basis for this subspace. One particularly convenient basis is constructed iteratively by adding the component of $A^{j-1}v$ which is orthogonal to $\mathcal{K}_{j-1}$ into the basis for $\mathcal{K}_{j-1}$, for each $j = 2, 3, \ldots, m$. The following Theorem states important properties of the basis obtained.

Theorem 2.6. *Let* $\dim(\mathcal{K}_m(v; A)) = m$ *and let* $v_1, v_2, \ldots, v_m$ *denote the sequence of vectors generated by the Gram-Schmidt procedure when run on vectors* v, Av, A^2v, ..., $A^{m-1}v$ *respectively. For* $j \leq m$, *let* $V_j = [v_1\ v_2\ \ldots\ v_j]$ *and let* $H_j = V_j^\star A V_j$ *denote the Rayleigh quotient. Then:*

(a) For $j < m$, $Av_j \in \mathcal{K}_{j+1}(v; A)$ *and* $Av_j \notin \mathcal{K}_j(v; A)$.

(b) H_m *is an unreduced* $m \times m$ *upper Hessenberg matrix, i.e. none of the elements* $\beta_1, \beta_2, \ldots, \beta_{m-1}$ *on its first subdiagonal is equal to zero.*

(c) The following equality holds for all $j < m$*:*

$$AV_j = V_j H_j + \beta_j v_{j+1} e_j^\star. \tag{2.10}$$

In particular, the last column reveals a recursive relation

$$\beta_j v_{j+1} = Av_j - V_j h_j = Av_j - V_j(V_j^\star A v_j) = (I - V_j V_j^\star) A v_j. \tag{2.11}$$

Here h_j *is the last column of* H_j.

(d) Suppose $\tilde{V}_j$ is a $n \times j$ orthonormal matrix, $\tilde{H}_j$ is an unreduced $j \times j$ upper Hessenberg matrix with positive elements of the first subdiagonal and $\tilde{r}_j$ is a vector in $\mathbb{C}^n$ such that $A\tilde{V}_j = \tilde{V}_j\tilde{H}_j + \tilde{r}_j e_j^\star$. If the first column of $\tilde{V}_j$ is equal to v_1, then $\tilde{V}_j = V_j$, $\tilde{H}_j = H_j$ and $\tilde{r}_j = \beta_j v_{j+1}$.

(e) Let (θ, x) denote a Ritz pair from $\mathcal{K}_j(A; v)$; let $x = V_j y$. Then

$$\|Ax - \theta x\| = |\beta_j| \cdot |e_j^\star y|. \tag{2.12}$$

Any equality of the form (2.10) where the matrices V_j and H_j have properties listed in part *(d)* of the Theorem is called the Arnoldi decomposition. This decomposition is unique if the first column of the matrix V_j is prescribed.

Relation (2.11) instructs on how to build an algorithm for the computation of V_j and H_j and will lead the to Arnoldi algorithm described later. Whether a single Ritz pair is converged is usually determined by the size of its residual norm (2.12). To this end, also note that

$$|\beta_j| = \left\|\beta_j v_{j+1} e_j^\star\right\| = \|AV_j - V_j H_j\|$$

is the norm of the residual of the matrix Rayleigh quotient H_j.

Next, we would like to study the convergence properties of Ritz approximations from the Krylov subspaces. The theory is well developed for the Hermitian matrices, but rather difficult and incomplete in the general case.

Let us first justify the choice of Rayleigh-Ritz procedure for obtaining approximations. The representation (2.8) of vectors in $\mathcal{K}_j$ hints that the optimality of the matrix H_j and the Ritz pairs as approximations of the eigenpairs of A can be naturally expressed in terms of polynomials. Let $\mathbb{P}_j^\star$ denote the set of all polynomials of degree j with the leading coefficient equal to 1.

Theorem 2.7. *The following statement holds:*

$$\|\kappa_j(A)v\| = \min_{\pi \in \mathbb{P}_j^\star} \|\pi(A) \cdot v\|.$$

Here $\kappa_j \in \mathbb{P}_j^\star$ is the characteristic polynomial of the matrix H_j.

We have shown previously that the k-dimensional Krylov subspace is A-invariant iff there is polynomial μ of degree k such that $\|\mu(A)v\| = 0$. In that case, zeros of μ are the eigenvalues of A as well. Theorem 2.7 indicates that since κ_j minimizes $\|\pi(A)v\|$, eigenvalues of H_j (i.e. zeros of κ_j) should be the best possible approximations for eigenvalues of A that one can obtain from $\mathcal{K}_j$. We will use this Theorem in Chapter 4 to establish the necessary and sufficient conditions for a tuple of complex numbers to appear as the Ritz values of a given normal matrix.

Unfortunately, it seems that this is the only optimality result for the non-symmetric matrices concerning the Rayleigh quotient. Hermitian matrices offer a variety of optimality results, including the minimax theorems and the Cauchy interlacing theorem, see

Section 1.5.2.

The tools we developed in Chapter 1.5.1 can now be used to evaluate the quality of the Krylov subspaces. To keep the exposition simple, we discuss here only the Hermitian case. Once again, the theory in the non-Hermitian case is rather difficult; we mention some results from [10] in Section 2.5.3, see also [11].

The Hermitian case. In case the matrix A is Hermitian, the question on the proximity of eigenvector approximations from a Krylov subspace can be rephrased in terms of search for a polynomial which is optimal in a certain way.

Theorem 2.8. *[68, page 270] Let A be a Hermitian matrix with eigenpairs (λ_1, u_1), (λ_2, u_2), ..., (λ_n, u_n) and let π be any polynomial of degree m or less such that $\pi(\lambda_i) \neq 0$ for some chosen eigenvalue λ_i. Then*

$$\tan \measuredangle(\mathcal{K}_{m+1}(v; A), u_i) \leq \tan \measuredangle(\pi(A)v, u_i) \leq \max_{j \neq i} \frac{|\pi(\lambda_j)|}{|\pi(\lambda_i)|} \tan \measuredangle(v, u_i).$$

Proof. Without loss of generality, suppose v is a unitary vector and let $v = \xi_1 u_1 + \xi_2 u_2 + \ldots + \xi_n u_n$ be its representation in the eigenvector basis. Then $\cos^2 \measuredangle(v, u_i) = \xi_i^2$ and $\sin^2 \measuredangle(v, u_i) = 1 - \xi_i^2$, so

$$\tan^2 \measuredangle(v, u_i) = \sum_{j \neq i} \frac{\xi_j^2}{\xi_i^2},$$

and similarly, since $\pi(A)v = \xi_1 \pi(\lambda_1) u_1 + \xi_2 \pi(\lambda_2) u_2 + \ldots + \xi_n \pi(\lambda_n) u_n$,

$$\tan^2 \measuredangle(\pi(A)v, u_i) = \sum_{j \neq i} \frac{\xi_j^2 \pi^2(\lambda_j)}{\xi_i^2 \pi^2(\lambda_i)} \leq \left(\max_{j \neq i} \frac{|\pi(\lambda_j)|}{|\pi(\lambda_i)|} \right)^2 \sum_{j \neq i} \frac{\xi_j^2}{\xi_i^2},$$

from which the claim follows directly. □

(Note that the statement of the Theorem holds for the normal matrices as well.) To find the best bound on $\tan \measuredangle(\mathcal{K}_{m+1}(v; A), u_i)$, we seek a polynomial π that minimizes $\max_{j \neq i} |\pi(\lambda_j)|$, while keeping $\pi(\lambda_i)$ fixed at some value. In other words, the goal is to find the value of

$$\min_{\substack{\pi \in \mathbb{P}_m^+ \\ \pi(\lambda_i) = 1}} \max_{j \neq i} |\pi(\lambda_j)|, \tag{2.13}$$

where $\mathbb{P}_m^+$ is the set of all polynomials of degree m or less. This task is too general the be dealt with; we introduce further assumptions in order to approximately solve (2.13). Let A be a Hermitian matrix with eigenvalues ordered so that $\lambda_1 > \lambda_2 \geq \lambda_3 \geq \ldots \geq \lambda_n$; consider the case $i = 1$. We simplify the task even more: instead of solving the discrete problem of finding the minimax value at points $\lambda_2, \lambda_3, \ldots, \lambda_n$, (2.13) is approximated by the continuous problem

$$\min_{\substack{\pi \in \mathbb{P}_m^+ \\ \pi(\lambda_i) = 1}} \max_{\xi \in [\lambda_n, \lambda_2]} |\pi(\lambda_j)|. \tag{2.14}$$

Note that the (2.14) gives an upper bound on (2.13), i.e.

$$\min_{\substack{\pi\in\mathbb{P}_m^+\\ \pi(\lambda_i)=1}} \max_{j\neq i} |\pi(\lambda_j)| \leq \min_{\substack{\pi\in\mathbb{P}_m^+\\ \pi(\lambda_i)=1}} \max_{\xi\in[\lambda_n,\lambda_2]} |\pi(\lambda_j)|.$$

The problem (2.14) is well known in the theory of polynomial optimization. The optimal polynomial is given by

$$\pi(\xi) = \frac{T_m\left(1+2\frac{\xi-\lambda_2}{\lambda_2-\lambda_n}\right)}{T_m\left(1+2\frac{\lambda_1-\lambda_2}{\lambda_2-\lambda_n}\right)},$$

where T_m is the Chebyshev polynomial, see Appendix A. The optimal value of (2.14) is

$$\min_{\substack{\pi\in\mathbb{P}_m^+\\ \pi(\lambda_i)=1}} \max_{\xi\in[\lambda_n,\lambda_2]} |\pi(\lambda_j)| = \frac{1}{T_m\left(1+2\frac{\lambda_1-\lambda_2}{\lambda_2-\lambda_n}\right)}.$$

The discussion above is recapitulated in the following Theorem.

Theorem 2.9. *Let A be a Hermitian matrix with eigenpairs (λ_1, u_1), (λ_2, u_2), …, (λ_n, u_n) such that $\lambda_1 < \lambda_2 \leq \lambda_3 \leq \ldots \leq \lambda_n$. Then*

$$\tan\measuredangle(\mathcal{K}_{m+1}(v;A), u_1) \leq \frac{1}{T_m\left(1+2\frac{\lambda_1-\lambda_2}{\lambda_2-\lambda_n}\right)} \tan\measuredangle(v, u_i).$$

Note how the tangent depends on the *spread*

$$\eta = \frac{\lambda_1-\lambda_2}{\lambda_2-\lambda_n}$$

of the spectrum – the larger the spread, the smaller the angle. Using the invariance properties of Krylov subspaces does not help us get a better bound, since all the matrices appearing in Proposition 2.5 have the same spread.

Theorem 2.9 shows the existence of some vector from $\mathcal{K}_m$ that is close to eigenvector u_1 if the spread and iteration number m are large enough. This still does not guarantee that the Ritz pairs approximating (λ_1, u_1) from the same subspace will be of the same quality. However, this is the case, and not only for the largest eigenvalue, as the following Theorem shows.

Theorem 2.10. *[59, Theorem VI.6.4] Let A be a Hermitian matrix with eigenpairs labeled as in Theorem 2.9. Let $(\theta_1, x_1), (\theta_2, x_2), \ldots, (\theta_m, x_m)$ denote Ritz pairs from $\mathcal{K}_m(v;A)$ ordered so that $\theta_1 \geq \theta_2 \geq \ldots \geq \theta_m$. Then, for all $j = 1, 2, \ldots, m$,*

$$0 \leq \lambda_j - \theta_j \leq (\lambda_1-\lambda_n)\left(\frac{\tau_j \tan\measuredangle(v,u_j)}{T_{m-j}(1+2\eta_j)}\right)^2$$

and

$$\sin\measuredangle(x_j, u_j) \leq \frac{\tau_j\sqrt{1+\beta_m^2/\delta_j^2}}{T_{m-j}(1+2\eta_j)} \tan\measuredangle(v, u_j).$$

Here

$$\tau_1 = 1, \quad \tau_j = \frac{\theta_1 - \lambda_n}{\theta_1 - \lambda_j} \cdot \frac{\theta_2 - \lambda_n}{\theta_2 - \lambda_j} \cdot \ldots \cdot \frac{\theta_{j-1} - \lambda_n}{\theta_{j-1} - \lambda_j};$$
$$\eta_j = \frac{\lambda_j - \lambda_{j+1}}{\lambda_{j+1} - \lambda_n};$$
$$\delta_j = \min_{i \neq j} |\lambda_i - \theta_j|.$$

The value β_m is the as in the Arnoldi decomposition (2.10)*; $|\beta_m| = \|AV_m - V_m H_m\|$.*

To compare bounds from Theorems 2.9 and 2.10 with those for the power method, note that using the same technique as in the proof of Theorem 2.8 we can show that

$$\tan \measuredangle(A^{m-1}v, u_1) \leq \left(\frac{|\lambda_2|}{|\lambda_1|}\right)^{m-1} \tan \measuredangle(v, u_1).$$

Thus it suffices to compare

$$\mathsf{PM} = \left(\frac{\lambda_2}{\lambda_1}\right)^{m-1} \tag{2.15}$$

with

$$\mathsf{KR} = \frac{1}{T_{m-1}(2\frac{\lambda_1}{\lambda_2} - 1)} \leq \frac{1}{T_{m-1}(1 + 2\frac{\lambda_1 - \lambda_2}{\lambda_2 - \lambda_n})}; \tag{2.16}$$

suppose here that A is positive definite. The following Table 2.2 shows the ratio between (2.15) and (2.16) for different values of λ_2/λ_1.

$\frac{\lambda_2}{\lambda_1}$ \ m	5	10	15	20	25
0.5	$\frac{6.3e-002}{1.7e-003}$	$\frac{1.9e-003}{2.6e-007}$	$\frac{6.1e-005}{3.8e-011}$	$\frac{1.9e-006}{5.7e-015}$	$\frac{5.9e-008}{8.5e-019}$
0.9	$\frac{6.6e-001}{1.4e-001}$	$\frac{3.9e-001}{5.5e-003}$	$\frac{2.3e-001}{2.1e-004}$	$\frac{1.4e-001}{7.9e-006}$	$\frac{7.9e-002}{2.9e-007}$
0.99	$\frac{9.6e-001}{7.5e-001}$	$\frac{9.1e-001}{3.2e-001}$	$\frac{8.7e-001}{1.2e-001}$	$\frac{8.3e-001}{4.4e-002}$	$\frac{7.9e-001}{1.6e-002}$

Table 2.2: Ratios $\frac{\mathsf{PM}}{\mathsf{KR}}$ for varoius choices of m and $\frac{\lambda_2}{\lambda_1}$

The advantage of the approximations from Krylov subspaces is obvious. When the ratio λ_2/λ_1 is close to 1, the convergence of both methods is slower. To further compare them for such ratio, let $\xi = \lambda_1/\lambda_2 - 1 \approx 0$. Then

$$\mathsf{PM} = (1 + \xi)^{-(m-1)} \approx e^{-(m-1)\xi}$$

and

$$\mathsf{KR} = \frac{1}{T_{m-1}(1 + 2\xi)} \approx 2e^{-2(m-1)\sqrt{\xi}},$$

where we have used the approximation bound for Chebyshev polynomials, see Appendix A. Exponent in the expression for KR is smaller by the factor of $\sqrt{\xi}$ than the one for PM, giving the Krylov subspace method a great advantage for small ξ.

2.4 The basic Arnoldi algorithm

All the theory developed so far in this chapter leads to the Arnoldi algorithm which is an effective method for extraction of Ritz pairs from the sequence of subspaces $\mathcal{K}_1(v;A)$, $\mathcal{K}_2(v;A)$, ... As mentioned earlier in the text, recursive relation (2.11):

$$\beta_j v_{j+1} = Av_j - V_j h_j = Av_j - V_j(V_j^\star Av_j) = (I - V_j V_j^\star)Av_j$$

from Theorem 2.6 instructs on how to build a matrix

$$V_j = [v_1\ v_2\ \ldots\ v_j]\,,$$

whose columns form an orthonormal basis for $\mathcal{K}_j(v;A)$. The relation shows that in order to form $V_{j+1} = [V_j\ v_{j+1}]$ one has to compute the orthogonal projection of Av_j onto the complement of $\operatorname{Im} V_j$ and then normalize it. To update the Rayleigh quotient matrix $H_j = V_j^\star A V_j$, suppose V_j and H_{j-1} have already been calculated and observe that

$$H_j = \begin{bmatrix} V_{j-1}^\star \\ v_j^\star \end{bmatrix} A \begin{bmatrix} V_{j-1} & v_j \end{bmatrix} = \begin{bmatrix} H_{j-1} & V_{j-1}^\star(Av_j) \\ v_j^\star A V_{j-1} & v_j^\star(Av_j) \end{bmatrix}.$$

The last column of this matrix is $V_j^\star(Av_j)$, which is computed during the projection needed for calculation of v_{j+1}. Also note that by premultiplying (2.10) with $v_{j+1}^\star$ we get $v_j^\star A V_{j-1} = \beta_{j-1} e_{j-1}^\star$ and that β_{j-1} is already available from the previous step. This shows that while computing V_{j+1} we get H_j at no cost in extra floating point operations. The pseudocode for this basic Arnoldi method is given in Algorithm 8.

It is not necessary to compute the product $x_i = V_j y_i$ and then the residual $Ax_i - \theta_i x_i$ in order to monitor convergence of a Ritz pair in Line 16. As Theorem 2.6*(e)* shows,

$$\|Ax_i - \theta_i x_i\| = \beta_j |e_j^\star y_i|,$$

i.e. the residual norm is simply the product of a subdiagonal element β_j of H_{j+1} and the last coordinate of the eigenvector y_i of H_j. No matrix nor vector of dimension n is involved in the calculation. However, computing a tiny component $e_j^\star y_i$ of an eigenvector reliably is numerically quite a difficult task. Before declaring a Ritz pair as converged, the residual should be recomputed explicitly.

For sake of keeping the presentation simple, the Gram-Schmidt procedure of orthogonalizing Av_j against V_j in Algorithm 8 is implemented as the classical Gram-Schmidt, which is known to be numerically unstable. An alternative might be to perform the modified Gram-Schmidt method; however, this method uses only BLAS-1 operations instead of better optimized BLAS-2 operations of classical Gram-Schmidt. Also, note that as $\operatorname{Im} V_j$ approaches A-invariance (which is the goal of the Arnoldi algorithm), both orthogonalization methods tend to produce vectors v_{j+1} that are far from being orthogonal to

Algorithm 8: Basic Arnoldi Algorithm

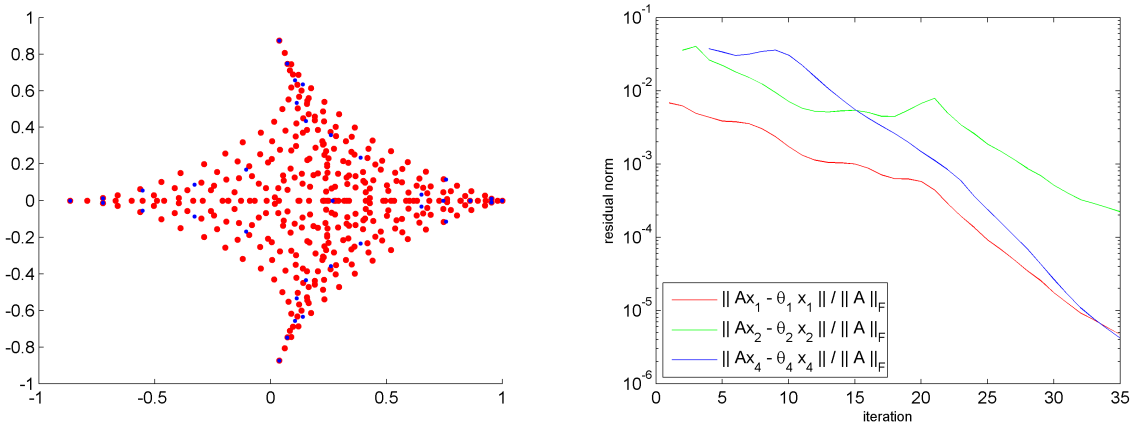
Input: Initial unit vector v_1, integers m and k
Output: Approximation of k dominant eigenpairs from $\mathcal{K}_m(v_1; A)$

1 $V_1 = [v_1]$;;
2 **for** $j = 1, \ldots, m$ **do**
  `// Compute the last column` h_j `of` H_j`:`
3 $t = Av_j$;
4 $h_j = V_j^\star(Av_j) = V_j^\star t$;
  `// Form the Rayleigh quotient matrix:`
5 **if** $j > 1$ **then**
6 $H_j = \begin{bmatrix} H_{j-1} & h_j(1:j-1) \\ \beta_{j-1}e_j^\star & h_j(j) \end{bmatrix}$;
7 **else**
8 $H_1 = h_1 = v_1^\star A v_1$;
9 **end**
  `// Compute the orthogonal projection onto` $\operatorname{Im} V_j{}^\perp$`:`
10 $r_j = (I - V_j V_j^\star)(Av_j) = t - V_j h_j$;
11 $\beta_j = \|r_j\|$;
12 **if** $\beta_j = 0$ **then** V_j is A-invariant; report convergence and exit.
13 $v_{j+1} = r_j/\beta_j$;
14 $V_{j+1} = [V_j\ v_{j+1}]$;
  `// Evaluate Ritz pairs from` $\mathcal{K}_j(v_1; A)$
15 Compute the dominant eigenvectors $(\theta_1, y_1), (\theta_2, y_2), \ldots, (\theta_k, y_k)$ of H_j;
16 **if** $j \geq k$ *and residuals of Ritz pairs* $(\theta_i, V_j y_i)$ *are small enough* **then** report convergence and exit.
17 **end**

$\operatorname{Im} V_j$. The proper BLAS-2 method of orthogonalization is achieved by replacing Line 10 in Algorithm 8 with the following (see [17]):

  `// Compute the orthogonal projection onto` $\operatorname{Im} V_j{}^\perp$`:`
10.1 $r_j = t - V_j h_j$;
10.2 **if** $\|h_j\|^2 > \eta(\|h_j\|^2 + \|r_j\|^2)$ **then**
10.3 $z = V_j^\star r_j$;
10.4 $r_j = r_j - V_j z$;
10.5 $h_j = h_j + z$;
  end

This includes a safety option of running an extra orthogonalization once again in case of $\beta_j = \|r_j\|$ being very small compared to $\|Av_j\|$, making sure that v_{j+1} and V_j close an angle greater than $\arccos(\eta)$, for a chosen parameter $0 < \eta < 1$.



(a) Locations of eigenvalues (red) and the Ritz values (blue)

(b) Residuals of three Ritz pairs having Ritz values of the largest module

Figure 2.1: Convergence of the basic Arnoldi algorithm for the matrix `gre343`.

Example 2.11. Figure 2.1 shows a typical convergence scenario for the Arnoldi algorithm. We have set A to be the matrix `gre343` from the MatrixMarket collection [44]; this matrix belongs to the "Grenoble" set generated in a simulation of computer systems. The matrix A is a real non-symmetric matrix of order 343, with a moderate condition number $\kappa(A) \approx 2.5 \cdot 10^2$. The starting vector v was set to $v = (1, 1, \ldots, 1)$ and 35 iterations of the basic Arnoldi algorithm were run. On the left figure, we have plotted the locations of the eigenvalues (red) and the Ritz values (blue) computed from the subspace $\mathcal{K}_{35}(v; A)$. As the theory predicts, the approximations for the outer eigenvalues are far better than the approximations for the inner ones. For example, we see that the leftmost and the topmost Ritz values almost coincide with the eigenvalues.

On the right figure, we have plotted the history of the residuals for three Ritz pairs. In iteration j, we have computed Ritz pairs $(\theta_1, x_1), (\theta_2, x_2), \ldots, (\theta_j, x_j)$ so that $|\theta_1| \geq |\theta_2| \geq \ldots \geq |\theta_j|$. The red line shows the residual for the Ritz pair (θ_1, x_1), the green one for the pair (θ_2, x_2) and the blue one for the pair (θ_4, x_4). (The value θ_3 is the complex conjugate of θ_2 so the Ritz pairs have the same residual.)

2.4.1 The Hermitian case: the Lanczos algorithm

If the matrix A is Hermitian, the Arnoldi algorithm reduces to an especially simple form called the Lanczos algorithm. In order to devise this method, first note that the matrix

$H_j = V_j^\star A V_j$ is tridiagonal since it is both Hermitian and upper Hessenberg. Let

$$H_j = \begin{bmatrix} \alpha_1 & \beta_1 & & & & \\ \beta_1 & \alpha_2 & \beta_2 & & & \\ & \beta_2 & \alpha_3 & \beta_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \beta_{j-2} & \alpha_{j-1} & \beta_{j-1} \\ & & & & \beta_{j-1} & \alpha_j \end{bmatrix};$$

all β_i are real and positive since they are equal to the norms of certain vectors. The last column of the Arnoldi decomposition

$$AV_j = V_j H_j + \beta_j v_{j+1} e_j^\star$$

now reads

$$\beta_j v_{j+1} = Av_j - \beta_{j-1} v_{j-1} - \alpha_j v_j. \tag{2.17}$$

This relation shows that we only need to orthogonalize Av_j against v_{j-1} and v_j in order to get v_{j+1}. No other column of V_j takes part in the computation, so we can store $v_1, v_2, \ldots, v_{j-2}$ in the secondary storage (or even drop them if only eigenvalue approximations are of interest) and work only with 4 vectors of dimension n at a time: Av_j, v_{j-1}, v_j and r_j. The basic version of the Lanczos method is implemented as Algorithm 9.

Using the equation (2.17) to obtain v_{j+1} introduces problems with keeping the orthogonality of V_j that are far more severe than those in the Arnoldi algorithm. Vector v_{j+1} will be numerically orthogonal to v_{j-1} and v_j, but the orthogonality to $v_1, v_2, \ldots, v_{j-1}$, while being guaranteed in theory and the exact arithmetic, may be completely lost in the finite precision arithmetic. A lot of research effort was invested into resolving this issue and several solutions have risen:

- Complete reorthogonalization. Vector Av_j is projected on $\operatorname{Im} V_j^\perp$ as in the Arnoldi algorithm, using all of $v_1, v_2, \ldots, v_j$ for the job. This surely solves the problem of orthogonality, but the simplicity and the low flop count of the basic Lanczos algorithm is lost.

- No additional reorthogonalization. By simply pursuing the basic Lanczos algorithm, after a number of iterations there will inevitably come to some side-effects. These side-effects appear as the *spurious* eigenvalues: H_j will seem to have a multiple eigenvalue θ, although there may be only a simple eigenvalue of A approximated by θ. Cullum and Willoughby [15] have devised a method for recognizing and eliminating these spurious eigenvalues.

- Partial or selective reorthogonalization. Theory of Paige [51] shows that the orthogonality of V_j (and then also of Ritz vectors) may deteriorate only when there is a converged Ritz vector in the basis. In such case and under the influence of finite arithmetic, v_{j+1} may have a stronger component in the direction of this converged

ALGORITHM 9: Basic Lanczos Algorithm

Input: Initial unit vector v_1, integers m and k
Output: Approximation of k dominant eigenpairs from $\mathcal{K}_m(v_1; A)$

$\beta_0 = 0$;
for $j = 1, \ldots, m$ **do**
  `// Compute the element in the last column in` H_j`:`
  $t = Av_j$;
  $\alpha_j = v_j^\star(Av_j) = v_j^\star t$;
  `// Form the Rayleigh quotient matrix:`
  if $j > 1$ **then**
    $H_j = \begin{bmatrix} H_{j-1} & \beta_{j-1}e_j \\ \beta_{j-1}e_j^\star & \alpha_j \end{bmatrix}$;
  else
    $H_1 = \alpha_1$;
  end
  `// Compute the orthogonal projection onto` $\operatorname{Im} V_j^\perp$`:`
  $r_j = (I - V_jV_j^\star)(Av_j) = t - \beta_{j-1}v_{j-1} - \alpha_j v_j$;
  $\beta_j = \|r_j\|$;
  if $\beta_j = 0$ **then**
    $V_j = [v_1\ v_2\ \ldots\ v_j]$ is A-invariant; report convergence and exit.
  end
  $v_{j+1} = r_j/\beta_j$;
  `// Evaluate Ritz pairs from` $\mathcal{K}_j(v_1; A)$
  Compute the dominant eigenvectors $(\theta_1, y_1), (\theta_2, y_2), \ldots, (\theta_k, y_k)$ of H_j;
  if $j \geq k$ *and residuals of Ritz pairs* (θ_i, V_jy_i) *are small enough* **then** report convergence and exit.
end

Ritz vector. Grcar, and Parlett and Scott [54] introduced methods of partial and selective reorthogonalization respectively; these methods additionally orthogonalize r_j against either specific columns of V_j or specific Ritz vectors if certain conditions arising from Paige's theory are met. The additional orthogonalizations are only sporadic and do not severely effect the flop count of the Lanczos algorithm. Simon [61] has shown that the selective reorthogonalization of Parlett guarantees the numerical orthogonality of V_j, making it the method of choice for the practical implementation of Lanczos algorithm.

2.5 Restarting the Arnoldi algorithm

We mentioned earlier that the goal of the large scale eigenvalue algorithms is to determine only a few (say g) eigenvalues that correspond to a certain part of the complex plane (e.g. those with largest or smallest moduli or real parts). It is difficult to predict in advance

the number of steps that the Arnoldi algorithm will take for computing desired eigenpairs. If the matrix A is very large, the amount of memory necessary for storing the basis for $\mathcal{K}_k$ may surpass the computer's capacity. Since the dimension of the projected matrix H_j increases with j, the time spent for computing its eigenvalue decomposition will eventually start to dominate other operations. Moreover, the number of Ritz approximations that are not of our interest gets larger and larger – as we would like to compute exactly g eigenpairs with certain properties, we find the Ritz pairs approximating any of the others useless for our purpose.

To avoid these unwanted effects, we resort to the idea of restarting the Arnoldi algorithm. Suppose that m steps of the basic algorithm are completed and that some approximations for the eigenpairs are known. Based on these approximations, we choose a new and improved initial vector $v^{(1)}$ and run another m Arnoldi steps, now starting with $v^{(1)}$. If even then the approximations for the wanted eigenvalues are not satisfactory, we repeat the procedure. The advantage of this approach is obvious: the algorithm now requires a fixed amount of memory known in advance. At each point the basis for a Krylov subspace consists of at most m vectors of length n and time needed for computing the eigenvalue decomposition of H_j is also limited. On the other hand, it is not clear how the restart procedure affects the convergence. Practice shows that in many cases careful selection of $v^{(1)}$ preserves convergence of the algorithm. However, we will see that there are examples where the restarted algorithm may completely fail to compute the desired eigenpairs.

If the subspace $\mathcal{K}_m(v; A)$ is invariant, the Ritz values will exactly coincide with the eigenvalues of A. This is equivalent to the initial vector v belonging to a subspace spanned by m eigenvectors of the matrix A. Denote with $(\lambda_1^{\checkmark}, u_1^{\checkmark})$, $(\lambda_2^{\checkmark}, u_2^{\checkmark})$, ..., $(\lambda_g^{\checkmark}, u_g^{\checkmark})$ the eigenpairs of A that we would like to compute and let $(\lambda_1^{\times}, u_1^{\times})$, $(\lambda_2^{\times}, u_2^{\times})$, ..., $(\lambda_{n-g}^{\times}, u_{n-g}^{\times})$ be the other eigenpairs (here $g \leq m$). The ideal starting vector which would require $\leq m$ basic Arnoldi steps for computing the wanted eigenpairs is

$$v^{\checkmark} = \sum_{j=1}^{g} \xi_j u_j^{\checkmark},$$

where $0 \neq \xi_j \in \mathbb{C}$ are arbitrary. Suppose the Arnoldi algorithm was originally started with $v = \sum_{j=1}^{g} \zeta_j^{\checkmark} u_j^{\checkmark} + \sum_{j=1}^{n-g} \zeta_j^{\times} u_j^{\times}$. One logical choice for $v^{(1)} \approx v^{\checkmark}$ would be

$$v^{(1)} = \pi(A)v = \sum_{j=1}^{g} \zeta_j^{\checkmark} \pi(\lambda_j^{\checkmark}) u_j^{\checkmark} + \sum_{j=1}^{n-g} \zeta_j^{\times} \pi(\lambda_j^{\times}) u_j^{\times}, \quad |\pi(\lambda_j^{\checkmark})| \gg |\pi(\lambda_k^{\times})|, \tag{2.18}$$

where $\pi \in \mathbb{P}_p$ is called a *polynomial filter*, for some integer $p \geq 1$. If in advance or during the algorithm we obtain subsets $\Omega^{\checkmark}$ and $\Omega^{\times}$ of the complex plane so that the wanted (the "good") eigenvalues are members of $\Omega^{\checkmark}$ and the unwanted (the "bad") eigenvalues belong to $\Omega^{\times}$, theory of polynomial optimization instructs on how to construct π so that $|\pi(\xi)| \ll 1$ for $\xi \in \Omega^{\times}$ and $|\pi(\xi)| \gg 1$ for $\xi \in \Omega^{\checkmark}$. For example, if A is Hermitian and the largest eigenvalues are sought, one may order the Ritz values in descending order $\theta_1^{\checkmark} \geq \ldots \geq \theta_g^{\checkmark} \geq \theta_1^{\times} \geq \ldots \geq \theta_{m-g}^{\times}$ and set $\theta_j^{\checkmark} \approx \lambda_j^{\checkmark}$. Then $\Omega^{\times} = [\theta_{m-g}^{\times} - \beta_m, \theta_1^{\times}]$,

$\Omega^{\checkmark} = [\theta_g^{\checkmark}, \theta_1^{\checkmark} + \beta_m]$ and the Chebyshev polynomial or order p for the segment $\Omega^{\times}$ may be used as a polynomial filter that will purge components of the unwanted eigenvectors out of the new initial vector $v^{(1)}$.

After m steps of the basic Arnoldi algorithm, approximations for the eigenvectors are also at disposal: $x_1^{\checkmark} \approx u_1^{\checkmark}, \ldots, x_g^{\checkmark} \approx u_g^{\checkmark}$. This leads to another possible choice for the new initial vector:

$$v^{(1)} = \sum_{j=1}^{g} \zeta_j x_j^{\checkmark}. \tag{2.19}$$

Since $x_j^{\checkmark} \in \mathcal{K}_m(v; A)$, it can be represented as $x_j^{\checkmark} = \pi_j(A)v$ for some polynomial $\pi_j \in \mathbb{P}_m$ and thus $v^{(1)}$ from (2.19) is obtained using a polynomial filter as well.

A polynomial filter may also be given by a list of its roots. One obvious choice for the roots of π are the unwanted Ritz values $\theta_1^{\times}, \ldots, \theta_{m-g}^{\times}$:

$$\pi(\xi) = \prod_{j=1}^{m-g} (\xi - \theta_j^{\times}).$$

It is easy to show that (e.g. using Lemma 3.1)

$$x_j^{\checkmark} = \left(\prod_{\substack{i=1 \\ i \neq j}}^{g} (A - \theta_i^{\checkmark} I) \right) \cdot \pi(A)v,$$

and thus

$$A^l x_j^{\checkmark} = A^l \left(\prod_{\substack{i=1 \\ i \neq j}}^{g} (A - \theta_i^{\checkmark} I) \right) \cdot v^{(1)} \in \mathcal{K}_m(v^{(1)}; A), \quad \text{for all } l = 0, \ldots m - g.$$

It follows that $\mathcal{K}_{m-g+1}(x_j^{\checkmark}; A) \subseteq \mathcal{K}_m(v^{(1)}; A)$, for all j. This choice of $v^{(1)}$ will thus lead to the Krylov subspace containing all vectors (2.19) along with their Krylov subspaces of dimension $m-g+1$. Within this context, the unwanted Ritz values are also referred to as *the exact shifts*, since there is a strong analogy with the shifts used in the QR algorithm for accelerating the convergence. Exact shifts are very successfully used in practice and are incorporated as the restart technique of the ARPACK software.

So far we have developed the *explicit restart* of the Arnoldi algorithm; see Algorithm 10.

2.5.1 The implicit restart

One pass through the loop in Algorithm 10 applies $m+p-1$ matrix–vector multiplications $x \mapsto Ax$: the first $m-1$ multiplications are done during the m Arnoldi steps that build up the Krylov subspace. Then, p additional multiplications are performed during the restart phase with the polynomial of degree p. After each loop, the new Krylov subspace contains only the vector $v^{(i)}$ and has dimension 1.

Algorithm 10: Arnoldi Algorithm with Explicit Restart

Input: initial vector $v^{(0)}$, integers g, m, p
Output: approximations for g wanted eigenpairs

$i = 0$;
while *true* **do**
  Run m steps of the Arnoldi algorithm with initial vector $v^{(i)}$:
  $AV_m^{(i)} - H_m^{(i)} V_m^{(i)} = \beta_m^{(i)} v_{m+1}^{(i)} e_m^\star$;
  Compute Ritz pairs $(\theta_{i,j}^\vee, x_{i,j}^\vee)$ and $(\theta_{i,j}^\times, x_{i,j}^\times)$;
  if *all g Ritz pairs $(\theta_{i,j}^\vee, x_{i,j}^\vee)$ are accurate approximations of eigenpairs of A* **then**
    Exit the loop and report Ritz pairs as eigenpair approximations;
  else
    Choose a polynomial $\pi \in \mathbb{P}_p$ using the computed Ritz pairs;
    $v^{(i+1)} = \pi(A)v^{(i)}$; $i = i + 1$;
  end
end

Sorensen [65] devised a clever way of restarting so that only p multiplications are needed in each pass. Moreover, the restarted Krylov subspace will have dimension $m - p$. To see how this can be done, suppose that we have the Arnoldi decomposition

$$AV_m = V_m H_m + r_m e_m^\star \tag{2.20}$$

with v being the first column of V_m and now need to apply the filter polynomial π with roots (shifts) $\sigma_1, \sigma_2, \ldots, \sigma_p$. Our goal is to transform (2.20) into another Arnoldi decomposition

$$A\tilde{V}_{m-p} = \tilde{V}_{m-p}\tilde{H}_{m-p} + \tilde{r}_{m-p} e_{m-p}^\star, \tag{2.21}$$

so that the first column of $\tilde{V}_{m-p}$ is a scalar multiple of $\pi(A)v$. We will achieve this goal by introducing the shifts one by one into (2.20).

First we introduce σ_1:

$$(A - \sigma_1 I)V_m = V_m(H_m - \sigma_1 I) + r_m e_m^\star. \tag{2.22}$$

Let $H_m - \sigma_1 I = Q^{(1)} R^{(1)}$ be the QR factorization and let $\tilde{H}_m^{(1)} = (Q^{(1)})^\star H_m Q^{(1)}$. We claim that the first column of $V_m Q^{(1)}$ is a scalar multiple of $(A - \sigma_1 I)v$: since $R^{(1)}$ is upper triangular, the first column of (2.22) is

$$(A - \sigma_1 I)v = V_m Q^{(1)} R^{(1)} e_1 = \rho_1 (V_m Q^{(1)}) e_1,$$

where $\rho_1 = R_{1,1}^{(1)}$. Note that we have already encountered a very similar situation with the QR algorithm, cf. (2.6).

Next, multiply (2.20) with $Q^{(1)}$ from the right:

$$A(V_m Q^{(1)}) = (V_m Q^{(1)}) \left((Q^{(1)})^\star H_m Q^{(1)} \right) + r_m e_m^\star Q^{(1)},$$

or if we put $\tilde{V}_m^{(1)} = V_m Q^{(1)}$,

$$A\tilde{V}_m^{(1)} = \tilde{V}_m^{(1)}\tilde{H}_m^{(1)} + r_m e_m^\star Q^{(1)}. \tag{2.23}$$

The matrix $\tilde{V}_m^{(1)}$ is orthonormal and $\tilde{H}_m^{(1)}$ is upper Hessenberg: $\tilde{H}_m^{(1)} = R^{(1)}H(R^{(1)})^{-1}$ with H upper Hessenberg and $R^{(1)}$ upper triangular, so only the last term in (2.23) prevents it from being an Arnoldi decomposition with initial vector in direction of $(A - \sigma_1 I)v$.

Using the same technique we introduce the shifts σ_2, σ_3, $\ldots$, σ_p, respectively, into (2.23). Let

$$\begin{aligned}
\tilde{H}_m^{(j-1)} - \sigma_j I &= Q^{(j)}R^{(j)} \quad \text{(a QR factorization)},\\
\tilde{H}_m^{(j)} &= (Q^{(j)})^\star \tilde{H}_m^{(j-1)} Q^{(j)},\\
\tilde{V}_m^{(j)} &= V_m \cdot Q^{(1)}Q^{(2)}\ldots Q^{(j)},\\
\tilde{v}^{(j)} &= \tilde{V}_m^{(j)} e_1,\\
\rho_j &= R_{1,1}^{(j)},
\end{aligned}$$

for $j = 2, \ldots, p$, and let $Q = Q^{(1)}\ldots Q^{(p)}$. Straightforward induction shows that

$$\tilde{v}^{(j)} = \frac{1}{\rho_j}(A - \sigma_j I)\tilde{v}^{(j-1)} = \rho(A - \sigma_j I)(A - \sigma_{j-1}I)\ldots(A - \sigma_1 I)v,$$

for some $\rho \in \mathbb{C}$. Analogously as (2.23) we obtain

$$A\tilde{V}_m^{(p)} = \tilde{V}_m^{(p)}\tilde{H}_m^{(p)} + r_m e_m^\star Q. \tag{2.24}$$

All matrices $\tilde{H}_m^{(p)}$ are upper Hessenberg and so are their unitary factors $Q^{(j)}$ from the QR factorization. Thus the matrix Q has zeros under the p-th subdiagonal, since it is a product of p upper Hessenberg matrices. Its last row equals to

$$e_m^\star Q = [\underbrace{0\ 0\ \ldots\ 0\ \hat{\beta}}_{m-p}\ \underbrace{b^\star}_{p}], \tag{2.25}$$

for some $\hat{\beta} \in \mathbb{C}$ and some $b \in \mathbb{C}^p$.

Equality (2.24) can be truncated to the Arnoldi decomposition. Denote with $\tilde{V}_{m-p}$ the first $m-p$ columns of $\tilde{V}_m^{(p)}$ and with $\tilde{H}_{m-p}$ the principal $(m-p)\times(m-p)$ submatrix of $\tilde{H}_m^{(p)}$:

$$\tilde{V}_m^{(p)} = [\tilde{V}_{m-p}\ \hat{V}_p]; \quad \tilde{H}_m^{(p)} = \begin{bmatrix} \tilde{H}_{m-p} & \hat{H}_{m-p,p} \\ \tilde{\beta}e_1 e_{m-p}^\star & \hat{H}_{p,p} \end{bmatrix}. \tag{2.26}$$

Insert (2.25) and (2.26) into (2.24):

$$A[\tilde{V}_{m-p}\ \hat{V}_p] = [\tilde{V}_{m-p}\ \hat{V}_p]\begin{bmatrix} \tilde{H}_{m-p} & \hat{H}_{m-p,p} \\ \tilde{\beta}e_1 e_{m-p}^\star & \hat{H}_{p,p} \end{bmatrix} + [\underbrace{0\ 0\ \ldots\ 0\ \hat{\beta}r_m}_{m-p}\ \underbrace{r_m b^\star}_{p}].$$

Algorithm 11: Arnoldi Algorithm with Implicit Restart

Input: initial vector $v^{(0)}$, integers g, m, p
Output: approximations for g wanted eigenpairs

$i = 0$;
Compute m steps of Arnoldi algorithm with initial vector $v^{(0)}$:
$AV_m^{(0)} = H_m^{(0)} V_m^{(0)} + r_m^{(0)} e_m^\star$;
Compute Ritz pairs $(\theta_{0,j}^{\checkmark}, x_{0,j}^{\checkmark})$ and $(\theta_{0,j}^{\times}, x_{0,j}^{\times})$;
while *not all g Ritz pairs $(\theta_{i,j}^{\checkmark}, x_{i,j}^{\checkmark})$ are accurate approximations of eigenpairs of A* **do**
  Choose shifts $\sigma_1, \ldots, \sigma_p \in \mathbb{C}$ using the computed Ritz pairs;
  $Q = I_m$;
  for $j = 1, 2, \ldots, p$ **do**
    Modify $H_m^{(i)}$ and Q by calling Implicit Shift$(H_m^{(i)}, \sigma_j, Q)$;
  end
  $\hat{\beta} = Q_{m,m-p}$; $\tilde{\beta} = (H_m^{(i)})_{m-p+1,m-p}$;
  $\tilde{r}_{m-p} = \hat{\beta} r_m^{(i)} + \tilde{\beta} V_m^{(i)} Q e_{m-p+1}$;
  $\tilde{V}_{m-p} = (V_m^{(i)} \cdot Q)(:, 1 : m-p)$;
  $\tilde{H}_{m-p} = H_m^{(i)}(1 : m-p, 1 : m-p)$;
  $i = i + 1$;
  Compute p more steps of the Arnoldi algorithm
    started with $A\tilde{V}_{m-p} = \tilde{V}_{m-p}\tilde{H}_{m-p} + \tilde{r}_{m-p} e_{m-p}^\star$;
    denote obtained decomposition with $AV_m^{(i)} = H_m^{(i)} V_m^{(i)} + r_m^{(i)} e_m^\star$;
  Compute Ritz pairs $(\theta_{i,j}^{\checkmark}, x_{i,j}^{\checkmark})$ and $(\theta_{i,j}^{\times}, x_{i,j}^{\times})$;
end

When the last p columns are deleted from the equality above, we end up with

$$A\tilde{V}_{m-p} = \tilde{V}_{m-p}\tilde{H}_{m-p} + \tilde{r}_{m-p} e_{m-p}^\star, \tag{2.27}$$

where $\tilde{r}_{m-p} = \tilde{\beta}\hat{V}_p e_1 + \hat{\beta} r_m$. Since the first column of the orthonormal matrix $\tilde{V}_{m-p}$ is a multiple of $\pi(A)v$ and $\tilde{H}_{m-p}$ is upper Hessenberg, by Theorem 2.6 it follows that (2.27) is an Arnoldi decomposition with a desired initial vector.

It is important to note that this entire transition from the subspace $\mathcal{K}_m(v; A)$ to the subspace $\mathcal{K}_{m-p}(\pi(A)v; A)$ was done without any matrix–vector multiplications. All that is required are p calls to Implicit Shift algorithm from Section 2.2 which construct $\tilde{H}^{(j+1)}$ from $\tilde{H}^{(j)}$ and compute $Q^{(j)}$. Only $m - p$ additional Arnoldi steps are necessary in order to return to the approximation from the m-dimensional Krylov space $\mathcal{K}_m(\pi(A)v; A)$. This *implicitly shifted* Arnoldi algorithm is shown in Algorithm 11.

Deflation: locking and purging. The implicitly shifted Arnoldi method (IRAM) is most commonly used with the exact shifts: the eigenvalues of the matrix H_m which are the unwanted Ritz values. If σ_1 is an eigenvalue of H_m, then $H_m - \sigma_1 I$ is a singular matrix, and the bottom right element of the triangular factor $R^{(1)}$ in the QR–factorization $H_m - \sigma_1 I = Q^{(1)} R^{(1)}$ is equal to zero. The implication is that $\tilde{H}_m^{(1)} = (Q^{(1)})^\star H_m Q^{(1)}$ has a zero as the last subdiagonal element. Similar argument shows that shifting with the other unwanted Ritz values produces zeros at subdiagonal elements of $\hat{H}_{p,p}$ from the equation (2.26), as well as $\tilde{\beta} = 0$.

Running the implicit shifting in the finite arithmetic can result in these elements not being far from zeros. This effect is known as the forward instability of the QR–algorithm [53], and can result in vectors other than the wanted Ritz vectors being preserved after the restart. This was recognized by Lehoucq and Sorensen [42, 66], who suggested methods of deflating the Ritz values which help in controlling the issue and provide the stability of the IRAM method. As the restarted Arnoldi algorithm progresses, the Ritz pairs start to converge to the eigenpairs. Depending on whether the converged pair is wanted or unwanted, the appropriate form of deflation is performed.

If a wanted Ritz pair converges, the Arnoldi decompositions undergoes a *locking*. The pair is decoupled from the Arnoldi decomposition and stored in a way that it is subject to no further change. To be more precise, consider the following unitary transformation.

Proposition 2.12. *Let $y = (\eta_1, \ldots, \eta_m) \in \mathbb{C}^m$ be a unit vector, and let $y_j = (\eta_1, \eta_2, \ldots, \eta_j)$ for each $j = 1, \ldots, j$. Let*

$$q_j = \begin{bmatrix} \gamma_j y_{j-1} \\ \rho_j \\ 0 \end{bmatrix}, \quad \gamma_j = \frac{-\eta_j}{\|y_{j-1}\| \, \|y_j\|}, \quad \rho_j = \frac{\|y_{j-1}\|}{\|y_j\|}, \quad j = 2, \ldots, m,$$

and let $Q_L = [y \;\; q_2 \;\; q_3 \;\; \ldots \;\; q_m]$. Then the matrix Q_L is unitary, with $Q_L e_1 = y$ and $e_m^\star Q_L = \eta_m e_1^\star + \|y_{m-1}\| \, e_m^\star$.

Suppose that $(\theta, x) = (\theta, V_m y)$ is a converged wanted Ritz pair in the Arnoldi decomposition

$$AV_m = V_m H_m + r_m e_m^\star. \tag{2.28}$$

Multiplying by Q_L from the right-hand side, we get

$$\begin{aligned} A(V_m Q_L) &= (V_m Q_L)(Q_L^\star H_m Q_L) + r_m e_m^\star Q_L \\ &= [x \;\; \hat{V}_{m-1}] \begin{bmatrix} \theta & h^\star \\ 0 & \hat{H}_{m-1} \end{bmatrix} + r_m \|y_{m-1}\| \, e_m^\star + (e_m^\star y) r_m e_1. \end{aligned}$$

Theorem 2.6(e) states that $|e_m^\star y| \cdot \|r_m\|$ is small since the Ritz pair is converged, so the last term in the equality can be neglected:

$$A[x \;\; \hat{V}_{m-1}] = [x \;\; \hat{V}_{m-1}] \begin{bmatrix} \theta & h^\star \\ 0 & \hat{H}_{m-1} \end{bmatrix} + r_m \|y_{m-1}\| \, e_m^\star. \tag{2.29}$$

The projected matrix is split into the converged part in the top left corner, and the active part in the bottom right. The restart of the Arnoldi algorithm will be performed on the active part once it is transformed back into the upper Hessenberg form. Such transformation is implemented as a unitary similarity W which is a product of Householder reflectors, providing that $W^\star \hat{H}_{m-1} W = \tilde{H}_{m-1}$ is upper Hessenberg and that $e_{m-1}^\star W = e_{m-1}^\star$. We discuss algorithms for constructing W in Chapter 5. By multiplying (2.29) with $\left[\begin{smallmatrix} 1 & 0 \\ 0 & W \end{smallmatrix}\right]$, we obtain

$$A[x \;\; \hat{V}_{m-1}W] = [x \;\; \underbrace{\hat{V}_{m-1}W}_{\tilde{V}_{m-1}}] \begin{bmatrix} \theta & h^\star W \\ 0 & \tilde{H}_{m-1} \end{bmatrix} + \underbrace{r_m \, \|y_{m-1}\|}_{\tilde{r}_{m-1}} e_m^\star.$$

We have decoupled the converged Ritz pair from the Arnoldi decomposition: the first column of the equality reads $Ax = \theta x$, while the other $m-1$ columns are

$$A\tilde{V}_{m-1} = [x \;\; \tilde{V}_{m-1}] \begin{bmatrix} h^\star W \\ \tilde{H}_{m-1} \end{bmatrix} + \tilde{r}_{m-1} e_{m-1}^\star.$$

All subsequent actions such as restarting and expanding the Krylov subspace basis modify $\tilde{V}_{m-1}$, leaving the converged pair (θ, x) intact. The locking transformation we described easily applies when further Ritz values converge: the Arnoldi decomposition generally has the form

$$A[V^{(c)} \;\; V^{(a)}] = [V^{(c)} \;\; V^{(a)}] \begin{bmatrix} H^{(c)} & H_{12} \\ 0 & H^{(a)} \end{bmatrix} + \check{r}_m e_m^\star. \tag{2.30}$$

Here $V^{(c)}$ contains an orthonormal basis for the subspace of the converged and locked Ritz vectors, and $V^{(a)}$ is the active part of the Krylov subspace which is still subject to transformations. The matrix $H^{(c)}$ is upper triangular having the converged Ritz values on the diagonal, and the matrix $H^{(a)}$ is upper Hessenberg. The new Ritz approximations are computed using the eigenpairs of $H^{(a)}$; both $V^{(c)}$ and $V^{(a)}$ are used when orthogonalizing the vector Av_m against the Krylov subspace in the expansion phase. Note that the first block column of (2.30) reads

$$AV^{(c)} = V^{(c)} H^{(c)},$$

which is a partial Schur factorization of the matrix A.

In the case when the converged Ritz pair (θ, x) is unwanted, a deflation method called *purging* is used. Purging removes x from the Krylov subspace using a similar technique as locking. Once again, suppose that we are given the Arnoldi decomposition (2.28). Let P be a permutation matrix such that for all $\xi_1, \ldots, \xi_m \in \mathbb{C}$ it holds

$$[\xi_1 \;\; \xi_2 \;\; \ldots \;\; \xi_m] P = [\xi_2 \;\; \xi_3 \;\; \ldots \;\; \xi_m \;\; \xi_1],$$

and let $Q_P = Q_L P$. Suppose that y now denotes a left eigenvector of the matrix H_m associated with θ: $y^\star H_m = \theta y^\star$. Using notation of Proposition 2.12, we have

$$y^\star Q_P = e_m^\star, \quad e_m^\star Q_P = \|y_{m-1}\| \, e_{m-1}^\star + \eta_m e_m^\star.$$

It is easy to see that multiplying (2.28) by Q_P from the right produces

$$A[\check{V}_{m-1} \;\; V_m Q_P e_m] = [\check{V}_{m-1} \;\; V_m Q_p e_m] \begin{bmatrix} \check{H}_{m-1} & \check{h} \\ 0 & \theta \end{bmatrix} + r_m [0 \;\; 0 \;\; \dots \;\; 0 \;\; \|y_{m-1}\| \;\; \eta_m].$$

Erasing the last column, we obtain the Arnoldi factorization

$$A\check{V}_{m-1} = \check{V}_{m-1}\check{H}_{m-1} + r_m \|y_{m-1}\| \, e^\star_{m-1}.$$

One can show that $\check{H}_{m-1}$ is already upper Hessenberg. Since $Q_P e_m = y$ is the left eigenvector of H_m, the first $m-1$ columns of Q_P span the same subspace as the right eigenvectors of H_m that are not associated with θ do. Thus the columns of $\check{V}_{m-1}$ span the subspace of all Ritz vectors except the one we wanted to purge. Note that this method of deflation does not require that (θ, x) is converged.

It can be shown that using locking and purging circumvents the effects of the forward instability of the implicit shifting. When A is a real matrix, the whole process of deflation can be carried in the real arithmetic by simultaneous deflation of Ritz values θ and $\overline{\theta}$.

2.5.2 The Krylov-Schur algorithm

Stewart [67] introduced the *Krylov–Schur* method, which restarts the Arnoldi algorithm in a different way. This method does not use implicit shifting, so it does not have the problem with the forward instability of the Hessenberg factor. Let

$$AV_m = V_m H_m + r_m e^\star_m \tag{2.31}$$

denote the Arnoldi decomposition we wish to restart using the p unwanted Ritz values as shifts. Consider the Schur decomposition $H_m = QSQ^\star$. The Ritz values appear in some order on the diagonal of the matrix S. There is an algorithm for reordering the diagonal so that the wanted Ritz values are moved to the top left corner, and the unwanted Ritz values to the bottom right. It suffices to show that we can swap any two neighbouring diagonal elements of S, so that a bubble–sort algorithm can be used to reorder the diagonal to our liking.

Suppose that

$$S = \begin{bmatrix} S_{11} & S_{12} & S_{13} & S_{14} \\ & \theta_1 & S_{23} & S_{24} \\ & & \theta_2 & S_{34} \\ & & & S_{44} \end{bmatrix},$$

where θ_1 and θ_2 are the eigenvalues we wish to exchange and $S_{11} \in \mathbb{C}^{k_1 \times k_1}$, $S_{22} \in \mathbb{C}^{k_2 \times k_2}$. Let $Q = [x_2 \;\; x_\perp]$ denote a unitary 2×2 matrix such that x_2 is an eigenvector of

$$\begin{bmatrix} \theta_1 & S_{23} \\ & \theta_2 \end{bmatrix}$$

corresponding to the eigenvalue θ_2. Then it is easy to see that

$$Q^\star \begin{bmatrix} \theta_1 & S_{23} \\ & \theta_2 \end{bmatrix} Q = \begin{bmatrix} \theta_2 & \hat{S}_{23} \\ & \theta_1 \end{bmatrix},$$

for some $\hat{S}_{23} \in \mathbb{C}$. Let $\hat{S} = \hat{Q}^\star S \hat{Q}$, where $\hat{Q} = I_{k_1} \otimes Q \otimes I_{k_2}$ is a unitary matrix. The matrix $\hat{S}$ is upper-triangular, and the diagonal elements are the same as in S, except that θ_1 and θ_2 have exchanged their positions.

If we wish to use real arithmetic, then a slightly more complex algorithm for exchanging 2×2 blocks in real Schur form can be devised [7].

Let $\theta_1^{\checkmark}, \ldots, \theta_{m-p}^{\checkmark}$ denote the wanted, and let $\theta_1^{\times}, \ldots, \theta_p^{\times}$ denote the unwanted Ritz values. Using the technique just described, we can assume that the initially computed Schur factorization $H_m = QSQ^\star$ is already such that each wanted Ritz values comes before any of the unwanted along the diagonal. Multiplying (2.31) by Q from the right, we have

$$A \underbrace{[\tilde{V}_{m-p} \;\; \tilde{V}_p]}_{V_m Q} = [\tilde{V}_{m-p} \;\; \tilde{V}_p] \underbrace{\begin{bmatrix} S_{11} & S_{12} \\ & S_{22} \end{bmatrix}}_{Q^\star H_m Q} + r_m \underbrace{[b_1^\star \;\; b_2^\star]}_{e_m^\star Q}, \tag{2.32}$$

where $\tilde{V}_{m-p}$ and $b_1^\star$ each have $m-p$ columns, and $\Lambda(S_{11}) = \{\theta_1^{\checkmark}, \ldots, \theta_{m-p}^{\checkmark}\}$ and $\Lambda(S_{22}) = \{\theta_1^{\times}, \ldots, \theta_p^{\times}\}$. Also note that columns of $\tilde{V}_{m-p}$ span the same subspace as the wanted Ritz vectors do. By removing the last p columns of (2.32), we have

$$A\tilde{V}_{m-p} = \tilde{V}_{m-p} S_{11} + r_m b_1^\star. \tag{2.33}$$

This is not an Arnoldi decomposition; S_{11} is not in the upper Hessenberg form and b_1 is not necessarily equal to e_{m-p}. There are two options available now. The first one is to construct a unitary transformation W such that $W^\star S_{11} W$ is upper Hessenberg and $b_1^\star W = e_{m-p}^\star$. Such a transformation exists, and is studied in detail in Chapter 5. Multiplying (2.33) by W from the right transforms it into a regular Arnoldi decomposition for the subspace $\mathcal{K}_{m-p}(\pi(A)v; A)$ with the desired polynomial filter $\pi(\xi) = (\xi - \theta_1^{\times}) \ldots (\xi - \theta_p^{\times})$.

The other option is simply to proceed with p Arnoldi steps in (2.33) just as if it were an Arnoldi decomposition: by normalizing the residual vector r_m, we compute the next vector v_{m-p+1} and add it to the Krylov basis $\tilde{V}_{m-p}$. Then Av_{m-p+1} is computed and orthogonalized against the basis; the coefficients in the Gram–Schmidt process are appended after the last column in S_{11}. The component of Av_{m-p+1} orthogonal to the basis is the next residual vector. After p such steps we reach

$$A\tilde{V}_m = \tilde{V}_m \tilde{H}_m + \tilde{r}_m e_m^\star, \tag{2.34}$$

with

$$\tilde{H}_m = \begin{bmatrix} S_{11} & \tilde{H}_{12} \\ e_1 b_1^\star & \tilde{H}_{22} \end{bmatrix},$$

and $\tilde{H}_{22}$ is an upper Hessenberg matrix. The eigenvalues of $\tilde{H}_m$ are the Ritz values from the Krylov subspace of dimension m spanned by the columns of $\tilde{V}_m$. If the Ritz

approximations for the wanted part of the spectrum are not satisfactory, we can repeat the entire Krylov–Schur procedure, using (2.34) as the starting point.

The analysis in [67] showed that the Krylov–Schur algorithm for restarting the Arnoldi method is numerically stable.

2.5.3 Convergence of the restarted Arnoldi method

The convergence theory of the restarted Arnoldi algorithm has been a very active field of research in recent years. Various polynomial filters are used with success in practise and the vast numerical experience has been backed up with several important results which are, unfortunately, far from being exhaustive.

Consider a sequence of Arnoldi decompositions

$$AV_m^{(i)} = V_m^{(i)} H_m^{(i)} + \beta_m^{(i)} v_{m+1}^{(i)} e_m^\star, \tag{2.35}$$

where $V_m^{(i)}$ contains an orthonormal basis for an m-dimensional Krylov subspace $\mathcal{K}_m(v^{(i)}; A)$. The starting vectors $v^{(i)}$ are related with

$$v^{(i+1)} = \pi_i(A) v^{(i)}, \tag{2.36}$$

where π_i is a polynomial filter applied at the i-th restart.

If the filter does not change through the restarts, $\pi_i = \pi$, for all i and some polynomial π of degree p, then the convergence theory is rather simple. Suppose that the matrix A is diagonalizable and let the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ of A be ordered so that

$$|\pi(\lambda_1)| \geq |\pi(\lambda_2)| \geq \ldots \geq |\pi(\lambda_{m-p})| > |\pi(\lambda_{m-p+1})| \geq |\pi(\lambda_{m-p+2})| \geq \ldots \geq |\pi(\lambda_n)|.$$

Let the initial vector $v^{(1)}$ have a non-zero component in the direction of each eigenvector associated with $\lambda_1, \lambda_2, \ldots, \lambda_{m-p}$:

$$v^{(1)} = \sum_{j=1}^{n} \xi_j u_j, \quad \xi_1, \ldots, \xi_{m-p} \neq 0.$$

The restarted initial vectors then satisfy

$$v^{(i)} = \alpha_i \sum_{j=1}^{n} \xi_j \left(\frac{|\pi(\lambda_j)|}{|\pi(\lambda_{m-p})|} \right)^i u_j,$$

with some normalizing factor α_i. Obviously, the components of the vectors $u_1, \ldots, u_{m-p}$ begin to dominate, forcing $v^{(i)}$ ultimately into an invariant subspace. Thus the restarted Arnoldi method will converge, meaning that the angle between the Krylov subspace and the eigenspace associated with $\lambda_1, \ldots, \lambda_{m-p}$ converges to zero.

The more interesting case is when the polynomial filter changes dynamically according to the information gained from the Krylov subspace just before the restart. In particular, the most widely used polynomial filters are the ones whose roots are the unwanted Ritz values (the exact shifts). Sorensen has shown that the restarted algorithm with such filters converges when the matrix A is Hermitian.

Theorem 2.13 (Sorensen [65]). *Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ be the eigenvalues of the Hermitian matrix A and $u_1, u_2, \ldots, u_n$ the associated eigenvectors. Consider a sequence of Arnoldi decompositions* (2.35), (2.36) *such that the polynomial filters π_i have degree p and roots*

$$\theta_{m-p+1}^{(i)}, \theta_{m-p+2}^{(i)}, \ldots, \theta_n^{(i)}.$$

Here $\theta_1^{(i)} \geq \theta_2^{(i)} \geq \ldots \geq \theta_m^{(i)}$ are the Ritz values computed from the Arnoldi decomposition (2.35). *Suppose that the starting vector $v^{(1)}$ has a non-zero component in the direction of each of u_1, u_2, …, u_{m-p}. Furthermore, suppose that $H_{m-p}^{(i)}(j+1, j) > \epsilon > 0$ for all i, j. Then*

$$\theta_j^{(i)} \xrightarrow{i} \lambda_j, \qquad \text{for all } j = 1, 2, \ldots, m-p.$$

For a long time it was unknown whether this result can be generalized to non-Hermitian matrices. Finally, Embree [28] constructed an counterexample for the convergence. He has shown that there exists a matrix A and an initial vector $v^{(1)}$ such that the unwanted Ritz value is precisely equal to the eigenvalue we strive to compute. More precisely, let

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 6 & -2 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad v^{(1)} = \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

Suppose that we want to compute the eigenvalue of A of the largest magnitude, that is $\lambda_1 = 1$. The Krylov subspace of dimension 2 has the associated projected matrix

$$H_2^{(1)} = \begin{bmatrix} 7/4 & 3\sqrt{35}/140 \\ \sqrt{35}/4 & 5/4 \end{bmatrix}$$

with Ritz values $\theta_1^{(1)} = 2$, $\theta_2^{(1)} = 1$. If we restart this subspace to the subspace of the dimension 1, then the unwanted Ritz value is the smaller one: $\theta_2^{(1)} = 1$. This is exactly the eigenvalue we wish to compute, and the new initial vector is

$$v^{(2)} = \begin{bmatrix} 0 \\ 3 \\ 1 \\ -1 \end{bmatrix}.$$

This vector has no component in the direction of the eigenvector $u_1 = e_1$ and the eigenvector cannot be retrieved by any further Arnoldi iteration.

Embree constructs a whole class of similar matrices for which this effect occurs. These matrices are all non-normal, so that a Krylov subspace can be constructed with some of Ritz values outside of the convex hull spanned by A's eigenvalues, and some (the unwanted ones) equaling the wanted eigenvalues. In Chapter 4 we will discuss a possibility of a similar example in the case of normal matrices.

Finally, a very general theory exists in case we can bound the unwanted eigenvalues in some compact subset $\Omega^{\times}$ of the complex plane, which does not contain any of the wanted eigenvalues of A, and the wanted eigenvalues in some compact subset $\Omega^{\surd}$. We also suppose that the shifts used in all of the restart polynomials fall outside of $\Omega^{\surd}$. Beattie et al. [10] show that

$$\begin{aligned}\sin \measuredangle(\tilde{\mathcal{U}}^{\surd}, \mathcal{K}_m(v^{(i)}; A)) &\leq \left(\max_{\psi \in \mathbb{P}_{w-1}} \frac{\left\| \psi(A)\mathcal{P}^{\times} v^{(1)} \right\|}{\left\| \psi(A)\mathcal{P}^{\surd} v^{(1)} \right\|} \right) \kappa(\Omega^{\times}) \max_{\xi \in \Omega^{\times}} |1 - \Psi(\xi)\alpha^{\surd}(\xi)| \\ &= C_0 C_1 \max_{\xi \in \Omega^{\times}} |1 - \Psi(\xi)\alpha^{\surd}(\xi)|.\end{aligned}$$

Here $\mathcal{P}^{\surd}$ is a spectral projector onto the eigenspace $\mathcal{U}^{\surd}$ of the dimension w associated with the wanted eigenvalues, and $\mathcal{P}^{\times}$ is a spectral projector onto the eigenspace $\mathcal{U}^{\times}$ associated with the unwanted eigenvalues. With $\tilde{\mathcal{U}}^{\surd} = \mathcal{K}_n(\mathcal{P}^{\surd} v^{(1)}; A)$ we have denoted a subspace of $\mathcal{U}^{\surd}$ which can be reached using the starting vector $v^{(1)}$. We suppose that the restarts reduce the dimension to $m = 2w$.

The polynomial $\alpha^{\surd}$ is the minimal polynomial of A with respect to $\mathcal{P}^{\surd} v^{(1)}$, that is, the monic polynomial of smallest degree for which $\alpha^{\surd}(A)\mathcal{P}^{\surd} v^{(1)} = 0$. It is easy to see that the roots of this polynomial are the wanted eigenvalues of A that can be reached using the starting vector $v^{(1)}$. If the roots of all filter polynomial applied in restarts 1, 2, ..., $i - 1$ are $\sigma_1, \ldots, \sigma_M$, then Ψ is the unique polynomial of degree M that interpolates the function $1/\alpha^{\surd}$ at $\sigma_1, \ldots, \sigma_p$. We thus expect $1 - \Psi(\xi)\alpha^{\surd}(\xi) \approx 0$.

The constant $\kappa(\Omega^{\times})$ measures the non-normality of A: it is the smallest constant for which

$$\|f(A)\Pi^{\times}\| \leq \kappa(\Omega^{\times}) \max_{\xi \in \Omega^{\times}} |f(\xi)|$$

holds for all functions f analytic on $\Omega^{\times}$; here $\Pi^{\times}$ is an orthogonal projector onto the maximum reachable subspace $\tilde{\mathcal{U}}^{\times} = \mathcal{K}_n(\mathcal{P}^{\times} v^{(1)}; A)$ of $\mathcal{U}^{\times}$. Note that $\kappa(\Omega^{\times}) \geq 1$, with equality holding for normal matrices.

The bound shows that the distance of the restarted subspace to the invariant subspace depends on the affinity of the initial vector to that invariant subspace (constant C_0), the non-normality of part of the spectrum of A associated with the unwanted eigenvalues (constant C_1), and the selection of shifts. The shifts should be chosen so that the expression

$$\max_{\xi \in \Omega^{\times}} |1 - \Psi(\xi)\alpha^{\surd}(\xi)|$$

is minimized. This is a problem in polynomial optimization and leads to asymptotic convergence bounds when shifts are chosen as e.g. Leja or Fejer points for the subset $\Omega^{\times}$.

Chapter 3

Using Krylov–Schur algorithm with arbitrary shifts

As we have seen, the implicitly restarted Arnoldi method (IRAM) contains an algorithm for the efficient reduction of the orthogonal basis for the Krylov subspace $\mathcal{K}_m(A, v)$ to the orthogonal basis for the subspace $\mathcal{K}_{m-p}(A, \pi(A)v)$. Here $\pi(\xi)$ is a monic polynomial of degree p defined by its roots, which are called shifts in this setting. Most commonly, the IRAM is used with the exact shifts – the Ritz values from $\mathcal{K}_m(A, v)$ discarded as the approximations of the unwanted part of the spectrum of the matrix A. Application of such shifts has proved to be very successful for the task of computing the exterior eigenvalues. However, if another part of the spectrum is of interest, the exact shifts do not always provide satisfactory results. Several other techniques for shift selection have therefore been taken into consideration in recent literature. For computing the eigenvalues closest to some prescribed target $\tau \in \mathbb{C}$, it is more common to use the harmonic Ritz values as shifts [47, 80]. Various other choices have also been applied in specific situations: Leja points [6], roots of Chebyshev [58] or other orthogonal polynomials such as Fekete, Faber or Fejer points [10].

On the other hand, when the exact shifts are used, the Krylov–Schur algorithm is another option for restarting the Arnoldi algorithm. In the addendum [69] of the article [67], Stewart notes that translation can be used to apply the Krylov–Schur algorithm for restarting with some other shifts, specifically the harmonic Ritz values.

In this chapter, we explore the possibility of using the Krylov–Schur algorithm with arbitrary shifts and show how this algorithm belongs to a more general restarting scheme. We note that using shifts other than the exact ones is closely related to the partial pole placement problem.

3.1 A more general restarting scheme

Let $A \in \mathbb{C}^{n\times n}$ and $v \in \mathbb{C}^n$. Consider a general *Krylov decomposition* of the form

$$AX = XB + xb^\star, \tag{3.1}$$

such that the columns of $[X\ x]$ form a basis for $\mathcal{K}_{m+1}(A, v)$ for some m. Here $X \in \mathbb{C}^{n\times m}$, $x \in \mathbb{C}^m$, $B \in \mathbb{C}^{m\times m}$ and $b \in \mathbb{C}^m$. Any decomposition of the form (3.1) has an equivalent Arnoldi decomposition

$$AV = VH + (\beta u)e_m^\star, \tag{3.2}$$

such that $[V\ u]$ is an orthonormal basis of the same Krylov subspace and H is an upper Hessenberg matrix. To see that, first let $X = UR$ be a thin QR factorization of X:

$$A(XR^{-1}) = (XR^{-1}) \cdot (RBR^{-1}) + x(b^\star R^{-1}),$$

or, setting $\tilde{B} = RBR^{-1}$, $\tilde{b} = R^{-\star}b$,

$$AU = U \cdot \tilde{B} + x\tilde{b}^\star.$$

Now U is orthonormal; let $x = Uy + \tilde{u}$ so that the vector $\tilde{u}$ is perpendicular to $\operatorname{Im} U$. Rescaling $\tilde{b}$ by $\|\tilde{u}\|$, we arrive at

$$AU = U \cdot (\tilde{B} + y\tilde{b}^\star) + \tilde{u}\tilde{b}^\star = U\hat{B} + u\hat{b}^\star, \tag{3.3}$$

where u is a unit vector. This is an *orthogonal Krylov decomposition*: $[U\ u]$ now forms an orthonormal basis for $\mathcal{K}_{m+1}(A, v)$. To obtain an Arnoldi decomposition, we need to simultaneously transform the matrix $\hat{B}$ to the upper Hessenberg form and the vector $\hat{b}$ to equal e_m. This can be achieved by constructing an orthogonal transformation $Q \in \mathbb{C}^{m\times m}$ such that

$$\begin{bmatrix} Q^\star & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{B} \\ \hat{b}^\star \end{bmatrix} Q = \begin{bmatrix} H \\ \beta e_m^\star \end{bmatrix}.$$

Here the matrix H is upper Hessenberg and β is a scalar. The matrix Q is a product of a sequence of row-oriented Householder reflectors that introduce zeros at appropriate entries starting from the last row; see Chapter 5 for details. Post-multiplying (3.3) by Q, we finally get an equivalent Arnoldi decomposition

$$AV = VH + (\beta u)e_m^\star,$$

with $V = UQ$ and $H = Q^\star \hat{B} Q$.

As we will see, the general Krylov decomposition (3.1) will play an important role when restarting the Arnoldi decomposition with arbitrary shifts. We need the following technical Lemma to obtain that result.

Lemma 3.1. *Let $AX = XB + xb^\star$ be a Krylov decomposition such that the columns of $[X\ x]$ form a basis for the subspace $\mathcal{K}_{m+1}(A, v)$. Let $q \in \mathbb{C}^m$ denote the representation of the initial vector v in basis X: $v = Xq$. Then for all $j = 0, 1, \ldots, m-1$ it holds that*

$$A^j v = XB^j q.$$

Proof. We use induction on j to prove the claim. When $j = 0$, the statement is obvious. Suppose that the statement holds for some $j < m - 1$. Then

$$A^{j+1}v = A(XB^jq) = (AX)B^jq = (XB + xb^\star)B^jq = XB^{j+1}q + xb^\star B^jq.$$

It suffices to show that $xb^\star B^jq = 0$. Using the notation from the discussion above the Lemma, we know that $AV = VH + (\beta u)e_m^\star$ is an equivalent Arnoldi decomposition with $H = Q^\star(RBR^{-1} + yb^\star R^{-1})Q$ and $\beta e_m^\star = b^\star R^{-1}Q$. Also, note that

$$v = Ve_1 = UQe_1 = XR^{-1}Qe_1,$$

and thus $q = R^{-1}Qe_1$. Using all these facts, we have

$$\begin{aligned} x(b^\star)\cdot(B^j)\cdot(q) &= x(\beta e_m^\star Q^\star R)\cdot(R^{-1}(QHQ^\star - yb^\star R^{-1})R)^j\cdot(R^{-1}Qe_1) \\ &= x(\beta e_m^\star Q^\star R)\cdot R^{-1}(QHQ^\star - yb^\star R^{-1})^j R\cdot(R^{-1}Qe_1) \\ &= x(\beta e_m^\star Q^\star)\cdot Q(H - Q^\star yb^\star R^{-1}Q)^j Q^\star\cdot(Qe_1) \\ &= x\beta\cdot e_m^\star(H - (\beta Q^\star y)e_m^\star)^j e_1. \end{aligned}$$

Now, H is an upper Hessenberg matrix and so is $\tilde{H} := H - (\beta Q^\star y)e_m^\star$. Taking a j-th power of an upper Hessenberg matrix produces a matrix with at most j non-zeros subdiagonals. Since $j < m - 1$, all elements on the $(m-1)$-th subdiagonal of the matrix $\tilde{H}^j$ will still be zero. This includes

$$(\tilde{H}^j)_{m,1} = e_m^\star(H - (\beta Q^\star y)e_m^\star)^j e_1 = 0.$$

□

Theorem 2.7 claims that using the Ritz pairs as the approximations of the eigenpairs is optimal in the following way: should there exist a monic polynomial μ of degree m such that $\mu(A)v = 0$, then $\mathcal{K}_m(A, v)$ would be A-invariant and the roots of μ would be the eigenvalues of A. When $\mathcal{K}_m(A, v)$ is not invariant, we expect that roots of the polynomial π that minimizes $\|\pi(A)v\|$ are good approximations for the eigenvalues. Theorem 2.7 states that the minimizing polynomial π is precisely the polynomial whose roots are the Ritz values, i.e. the characteristic polynomial κ_H of the Hessenberg matrix H in (3.2). We now show that eigenvalues of B in every Krylov decomposition (3.1) can also be considered as optimal in a similar way.

Theorem 3.2. *Let $AX = XB + xb^\star$ be a Krylov decomposition such that the columns of $[X\ x]$ span the subspace $\mathcal{K}_{m+1}(A, v)$. Then*

$$\min_{\pi\in\mathbb{P}_m^\star} \|\mathcal{P}_{x_\perp}\pi(A)v\| = 0,$$

and the minimum is attained for the characteristic polynomial κ_B of the matrix B. Here $\mathbb{P}_m^\star$ is the set of all monic polynomials of degree m, and $\mathcal{P}_{x_\perp} = I - \tilde{x}\tilde{x}^\star$ is the orthogonal projector along the vector $\tilde{x} = x/\|x\|$.

Proof. Using Lemma 3.1, we see that

$$\begin{aligned} A^j v &= XB^j q, \quad j = 0, 1, \ldots, m-1 \\ A^m v &= XB^m q + xb^\star B^{m-1} q, \end{aligned}$$

for some vector $q \in \mathbb{C}^m$. Thus every $\pi \in \mathbb{P}_m^\star$ satisfies $\pi(A)v = X\pi(B)q + xb^\star B^{m-1}q$ and

$$\|\mathcal{P}_{x_\perp} \pi(A)v\| = \|\mathcal{P}_{x_\perp} X\pi(B)q\|.$$

This is minimized for $\pi = \kappa_B$, since every matrix annihilates its characteristic polynomial, and the minimum is equal to 0. □

When this Theorem is used with an orthogonal Krylov decomposition $AU = UB + ub^\star$, we have

$$\|\pi(A)v\|^2 = \left\|U\pi(B)q + ub^\star B^{m-1}q\right\|^2 = \|U\pi(B)q\|^2 + \left\|ub^\star B^{m-1}q\right\|^2.$$

The second term does not depend on the polynomial π and we have the claim of Theorem 2.7:

$$\operatorname*{argmin}_{\pi \in \mathbb{P}_m^\star} \|\pi(A)v\| = \operatorname*{argmin}_{\pi \in \mathbb{P}_m^\star} \|U\pi(B)q\| = \operatorname*{argmin}_{\pi \in \mathbb{P}_m^\star} \|\mathcal{P}_{u_\perp} \pi(A)v\| = \kappa_B.$$

Next, we consider a scheme that generalizes restarting of the Krylov–Schur type.

Theorem 3.3. *Suppose that*

$$A \underbrace{[X_1\ X_2]}_{X} = [X_1\ X_2] \cdot \underbrace{\begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix}}_{B} + x \underbrace{[b_1^\star\ b_2^\star]}_{b^\star}$$

is a Krylov decomposition such that columns of X span the subspace $\mathcal{K}_m(A, v)$ which is not A-invariant. Let X, B and $b^\star$ be split so that X_2, B_{22} and $b_2^\star$ have p columns each. The following statements are equivalent:

(a) Eigenvalues of B_{22} are $\sigma_1, \ldots, \sigma_p$.

(b) Columns of X_1 form a basis of $\mathcal{K}_{m-p}(A, \pi(A)v)$, where π is a monic polynomial with roots $\sigma_1, \ldots, \sigma_p$.

Proof. Assume first that (a) holds. Using Lemma 3.1, for all $j = 0, \ldots, m-1$ we have

$$A^j v = [X_1\ X_2] \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix}^j q = [X_1\ X_2] \begin{bmatrix} B_{11}^j & B_{12}^{(j)} \\ 0 & B_{22}^j \end{bmatrix} q,$$

for some matrix $B_{12}^{(j)}$ and some vector q. The characteristic polynomial $\kappa_{B_{22}}$ has degree k and thus for all $j = 0, \ldots, m-p-1$,

$$\begin{aligned} A^j \kappa_{B_{22}}(A) v &= [X_1\ X_2] \begin{bmatrix} B_{11}^j \kappa_{B_{22}}(B_{11}) & \tilde{B}_{12}^{(j)} \\ 0 & B_{22}^j \kappa_{B_{22}}(B_{22}) \end{bmatrix} q \\ &= [X_1\ X_2] \begin{bmatrix} B_{11}^j \kappa_{B_{22}}(B_{11}) & \tilde{B}_{12}^{(j)} \\ 0 & 0 \end{bmatrix} q \\ &= X_1 \begin{bmatrix} B_{11}^j \kappa_{B_{22}}(B_{11}) & \tilde{B}_{12}^{(j)} \end{bmatrix} q. \end{aligned}$$

This means that $A^j \kappa_{B_{22}}(A) v \in \operatorname{Im} X_1$, implying that $\mathcal{K}_{m-p}(A, \kappa_{B_{22}}(A)v) \subseteq \operatorname{Im} X_1$. The equality here holds, since the subspaces are of equal dimension (the original Krylov subspace is not A-invariant).

To show the other implication, assume that $\operatorname{Im} X_1 = \mathcal{K}_{m-p}(A, \pi(A)v)$, for some monic polynomial π of degree p. We have already shown that $\operatorname{Im} X_1 = \mathcal{K}_{m-p}(A, \kappa_{B_{22}}(A)v)$. Therefore, $\pi(A)v \in \mathcal{K}_{m-p}(A, \kappa_{B_{22}}(A)v)$ and $\pi(A)v = \xi(A) \cdot \kappa_{B_{22}}(A)v$ for some polynomial ξ of degree $m-p-1$ or less. If $\xi \neq 1$, then the polynomial $\tilde{\pi} = \pi - \xi \kappa_{B_{22}}$ is a non-trivial polynomial of degree less than m for which $\tilde{\pi}(A)v = 0$. This contradicts the fact that $\mathcal{K}_m(A, v)$ is not A-invariant. Therefore, $\pi(A)v = \kappa_{B_{22}}(A)v$. Using the same argument, it follows that $\pi = \kappa_{B_{22}}$. □

Suppose we are given the Krylov decomposition for $\mathcal{K}_m(v; A)$

$$AX_m = X_m B_m + x_m b_m^\star = [X_m\ \ x_m] \begin{bmatrix} B_m \\ b_m^\star \end{bmatrix}, \tag{3.4}$$

which we would like to restart to the Krylov decomposition

$$A\tilde{X}_{m-p} = \tilde{X}_{m-p}\tilde{B}_{m-p} + \tilde{x}_{m-p}\tilde{b}_{m-p}^\star.$$

We want the starting vector for the associated Krylov subspace $\operatorname{Im} \tilde{X}_{m-p} = \mathcal{K}_{m-p}(\tilde{v}; A)$ to equal

$$\tilde{v} = (A - \sigma_1 I) \cdot \ldots \cdot (A - \sigma_p I) v,$$

where $\sigma_1, \ldots, \sigma_p \in \mathbb{C}$ are some prescribed shifts. To do so, transform (3.4) to fit the form given in Theorem 3.3, and erase the last p columns from the decomposition. Note that the matrix B_{11} and its spectrum are irrelevant for the restart – only the spectrum of the matrix B_{22} has an effect on the restarted Krylov subspace.

3.2 Restarting and the pole placement problem

Our next goal is to explore how the Krylov decomposition (3.4) is transformed into the form suitable for the restart. What are the permissible transformations that maintain the Krylov decomposition? Clearly, the columns of the matrix X_m have to span the Krylov

subspace $\mathcal{K}_m(v; A)$, so we can only change the basis for this subspace: this is equivalent to multiplying (3.4) by some regular $m \times m$ matrix T from the right:

$$A(X_m T) = [X_m \;\; x_m] \begin{bmatrix} B_m T \\ b_m^\star T \end{bmatrix}. \tag{3.5}$$

This is not the Krylov decomposition yet; instead of just X_m, we must have $X_m T$ on the right hand side as well:

$$[X_m \;\; x_m] = [X_m T \;\; \tilde{x}] \begin{bmatrix} T^{-1} & a \\ 0 & \beta \end{bmatrix},$$

for any $a \in \mathbb{C}^m$, $\beta \in \mathbb{C} \setminus \{0\}$. Here $\tilde{x} \in \mathbb{C}^n$ equals $\tilde{x} = 1/\beta(X_m T a - x_m)$. Combining with (3.5), it follows that

$$\begin{aligned} A\underbrace{(X_m T)}_{\hat{X}_m} &= [(X_m T) \;\; \tilde{x}] \begin{bmatrix} T^{-1} B_m T + a b_m^\star T \\ \beta b_m^\star T \end{bmatrix} \\ &= \underbrace{(X_m T)}_{\hat{X}_m} \underbrace{\left(T^{-1} B_m T + a b_m^\star T\right)}_{\hat{B}_m} + \underbrace{(\beta \tilde{x})}_{\hat{x}_m} \cdot \underbrace{(b_m^\star T)}_{\hat{b}_m^\star}. \end{aligned}$$

Therefore, if we wish to restart using shifts $\sigma_1, \ldots, \sigma_p$, we must find T, a and β such that the matrix $\hat{B}_m$ is block upper–triangular, and that the eigenvalues of the $(2,2)$–block are exactly the shifts.

Let us take a look at the transformation that B_m undergoes to fulfill such a request. Assume that the initial Krylov decomposition was orthogonal. Then the eigenvalues of B_m are the Ritz values $\theta_1, \ldots, \theta_m$. The matrix $\hat{B}_m = T^{-1}(B_m + T a b_m^\star)T$ is similar to the matrix

$$B_m + g b_m^\star,$$

with $g = Ta$. Therefore, in order to execute the restart of the Arnoldi algorithm, one has to (implicitly or explicitly) find a vector g such that $B_m + g b_m^\star$ has the eigenvalues $\sigma_1, \ldots, \sigma_p$ (and any other $m - p$ complex numbers) and explicitly form a matrix that is similar to it.

This is the (partial) pole placement problem, well known in the control theory (see e.g. [18] or [19] for an overview of algorithms) and notorious for its ill conditioning.

Definition 3.4. *Let $B \in \mathbb{C}^{m \times m}$, and $b \in \mathbb{C}^n$, and $\sigma_1, \ldots, \sigma_p$ be given. If $p = m$, then the problem of finding a vector $g \in \mathbb{C}^m$ such that the eigenvalues of $B + g b^\star$ are equal to $\sigma_1, \ldots, \sigma_p$ is called the pole placement problem. If $p < m$, the same problem is called the partial pole placement problem.*

The pole placement problem is solvable for any choice of shifts if and only if

$$\operatorname{rank}[\lambda I - B^\star \;\; b] = m,$$

for all complex numbers λ, or, equivalently, if no eigenvector of B is perpendicular to b. While computing the vector g may be numerically reliable, it is very difficult to expect that the eigenvalues of $B + gb^\star$ will be good approximations of the shifts. The analysis in [45] and the vast numerical experience shows that the eigenvalues of $B + gb^\star$ are in general very sensitive to tiny to perturbations of B or b, even when g is computed in the exact arithmetic.

Theorem 3.5. *[45] Consider the pole placement problem with the input B, b, $\sigma_1, \ldots, \sigma_m$, and a perturbed problem with the input $\tilde{B}$, $\tilde{b}$, $\tilde{\sigma}_1, \ldots, \tilde{\sigma}_m$ such that*

$$\max\{\left\|B - \tilde{B}\right\|, \left\|b - \tilde{b}\right\|, |\sigma_j - \tilde{\sigma}_j|\} = \epsilon$$

is sufficiently small. Suppose that g and $\tilde{g}$ are the solutions to the original and the perturbed problem, respectively. Let $\hat{\sigma}_1, \ldots, \hat{\sigma}_m$ denote the eigenvalues of the matrix $B + \tilde{g}b^\star$. Then

$$|\sigma_j - \hat{\sigma}_j| \leq \epsilon(1 + (1 + \|\tilde{g}\|)\kappa),$$

where κ is the spectral condition number of the matrix $\tilde{B} + \tilde{g}\tilde{b}^\star$.

We see that if the solution vector $\tilde{g}$ has a large norm, or if the matrix $\tilde{B} + \tilde{g}\tilde{b}^\star$ has a large condition number, then we cannot guarantee that the eigenvalues of $B + \tilde{g}b^\star$ are near $\sigma_1, \ldots, \sigma_m$. The solution of a slightly perturbed problem may be completely useless as the solution of the original problem – this is a prototype of an ill conditioned task. In general, we are not able to predict the sizes of these quantities on which the forward stability of the algorithm relies on. Experience shows that, if the target eigenvalues are relatively close to each other, or placed in some relatively rigid formation (e.g. they are colinear), then solving the pole placement can dramatically fail.

While there may exist different algorithms which transform the Krylov decomposition into the form of the Theorem 3.3, we have seen that all of them ultimately have to solve a pole placement problem. We explore two such algorithms; the first one is well-known and widely used, but, to our knowledge, it has never been linked to the pole placement problem.

Restarting with the harmonic Ritz values. Stewart [69] noted that the Krylov–Schur restarting in the Arnoldi algorithm can be used so that the chosen Ritz vectors of any (potentially skew) Rayleigh quotient are preserved. Let

$$AU_m = U_m B_m + u_m b_m^\star \tag{3.6}$$

denote an orthogonal Krylov decomposition; let $V_m \in \mathbb{C}^{n\times m}$ be any matrix such that $V_m^\star U_m$ is non-singular. Consider a *skew Rayleigh quotient* with respect to U_m and V_m:

$$\hat{B}_m = \rho(U_m, V_m) = (V_m^\star U_m)^{-1}(V_m^\star A U_m).$$

If $(\lambda, U_m y)$ is an eigenpair of A, then (λ, y) is an eigenpair of $\hat{B}_m$. Thus, we expect that for an eigenpair (θ_{UV}, y_{UV}) of $\hat{B}_m$ to generate a good *skew Ritz pair* approximation

(θ_{UV}, Uy_{UV}) to an eigenpair of A. As we have seen in Section 1.5.2, the harmonic Ritz pairs are the skew Ritz pairs with $V_m = (A - \tau I)U_m$. Stewart shows how to transform (3.6) into the one which uses $\hat{B}_m$ instead of B_m: let $g = (V_m^\star U_m)^{-1} V_m^\star u_m$. Then

$$AU_m = U_m \underbrace{(B_m + g b_m^\star)}_{\hat{B}_m} + (u_m - U_m g) b_m^\star. \tag{3.7}$$

The process of changing u_m to $u_m - U_m x$ and B_m to $B_m + x b_m^\star$ for some vector x is called a *translation*. Now we can proceed as in the ordinary Krylov–Schur restarting, computing the sorted Schur form of the matrix $\hat{B}_m$:

$$\hat{B}_m = [Q_1 \;\; Q_2] \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix} [Q_1 \;\; Q_2]^\star, \tag{3.8}$$

so that the $m - p$ wanted skew Ritz values appear on the diagonal of S_{11}. Multiplying (3.7) by Q_1 on the right-hand side results in

$$A \underbrace{(U_m Q_1)}_{\tilde{U}_{m-p}} = (U_m Q_1) S_{11} + \underbrace{(u_m - U_m g)}_{\hat{u}_{m-p}} b_m^\star Q_1.$$

To restore the orthogonal Krylov decomposition, another translation is used: let $\hat{u}_{m-p} = \tilde{U}_{m-p} g_1 + \tilde{u}_{m-p}$ be the result of the Gram–Schmidt orthogonalization of the vector $\hat{u}_{m-p}$ against the subspace $\tilde{U}_{m-p}$. Then

$$A\tilde{U}_{m-p} = \tilde{U}_{m-p}(S_{11} + g_1 \tilde{b}_{m-p}^\star) + \tilde{u}_{m-p} \tilde{b}_{m-p}^\star \tag{3.9}$$

is a restarted Krylov decomposition such that the skew Ritz pairs are the same as the wanted skew Ritz pairs before the restart.

The described method of restarting is almost exclusively used with the harmonic Ritz vectors. This is particularly convenient since the vector g in the translation (3.7) is explicitly given by the formula

$$g = (B_m - \tau I)^{-\star} b_m, \tag{3.10}$$

which easily follows from $V_m = (A - \tau I)U_m$ and

$$\begin{aligned} \hat{B}_m &= (V_m^\star U_m)^{-1} \cdot V_m^\star A U_m \\ &= ((U_m(B_m - \tau I) + u_m b_m^\star)^\star U_m)^{-1} \cdot ((U_m(B_m - \tau I) + u_m b_m^\star)^\star (U_m B_m + u_m b_m^\star) \\ &= (B_m - \tau I)^{-\star} \cdot ((B_m - \tau I)^\star B_m + b_m b_m^\star) \\ &= B_m + g b_m^\star. \end{aligned}$$

Restarting with arbitrary shifts. With the help of Theorem 3.3, there is a simple way to restart the Krylov decomposition using any given shifts $\sigma_1, \ldots, \sigma_p$, just as in the implicitly restarted Arnoldi method. Given (3.6), apply any pole placement algorithm to

compute the vector g such that $B_m + gb_m^\star$ has eigenvalues $\sigma_1, \ldots, \sigma_p$, and any other $m-p$ complex numbers:

$$AU_m = U_m(B_m + gb_m^\star) + (u_m - U_m g)b_m^\star. \tag{3.11}$$

Then, compute a sorted Schur decomposition of $B_m + gb_m^\star$, so that the shifts appear as the eigenvalues of the S_{22} block:

$$B_m + gb_m^\star = [Q_1 \;\; Q_2] \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix} [Q_1 \;\; Q_2]^\star. \tag{3.12}$$

We are now in the situation described by the Theorem 3.3. Multiplying (3.11) by Q_1 and reorthogonalizing produces a Krylov decomposition

$$A\tilde{U}_{m-p} = \tilde{U}_{m-p}\tilde{B}_{m-p} + \tilde{u}_{m-p}\tilde{b}_{m-p}^\star$$

such that the starting vector is $(A - \sigma_1 I) \cdot \ldots \cdot (A - \sigma_p I)v$.

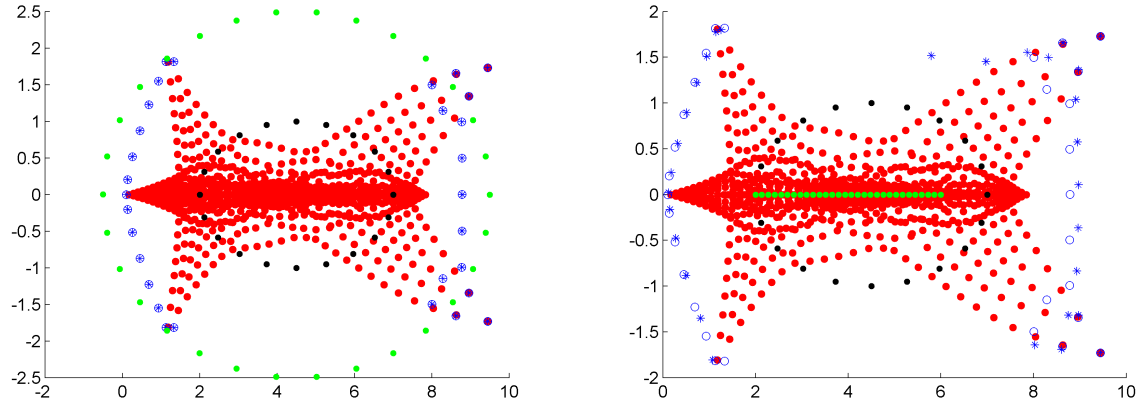
Both of the described restarting techniques explicitly solve a pole placement problem. Matrices B_m or S with eigenvalues equal to the (ordinary) Ritz values are modified by a rank-one matrix $gb_m^\star$ in order to make the modified matrix have some other prescribed set of eigenvalues. The following Example shows how this way of computing does not always produce the desired Krylov subspace after the restart.

Example 3.6. Consider the matrix `PDE900` from the MatrixMarket [44] collection. This is a 900×900 matrix with a moderate norm $\|A\|_F \approx 1.5 \cdot 10^2$ and condition $\kappa(A) \approx 2.9 \cdot 10^2$. We ran the Arnoldi algorithm, producing a subspace of dimension $m = 50$. Then a Krylov–Schur algorithm was used to restart to the dimension $m - p = 30$. We used 20 shifts that were uniformly distributed along the perimeter of the ellipse with center at 4.5 and semi-axes 2.5 and 1. The shifts are shown as the black dots in Figures 3.1(a) and 3.1(b).

To restart the Krylov–Schur algorithm with arbitrary shifts, we need to select the eigenvalues of $B + gb^\star$, which is equivalent to setting the eigenvalues of S_{11} and S_{22} in (3.12). The eigenvalues of S_{22} are the shifts. For the eigenvalues of S_{11}, we consider two options. First, we choose these eigenvalues uniformly distributed along the perimeter of another ellipse with the same center and semi-axes 5 and 2.5. These eigenvalues are shown as the green dots in the left figure. The alternative is to put the eigenvalues of S_{11} to uniformly cover the segment $[2, 6]$ on the real axis. These eigenvalues are the green dots in the right figure.

Once the eigenvalues of $B + gb^\star$ are chosen, the vector g is computed using a pole placement algorithm and the Arnoldi decomposition is restarted to the subspace of dimension 30. We have computed the Ritz values from the restarted subspace; these are shown as the blue asterisks in both the left and the right Figure. Red dots are put at the locations of the eigenvalues of A.

To verify the Ritz values computed by the Krylov–Schur algorithm, we ran the Implicitly restarted Arnoldi algorithm using the same input. Note that IRAM does not require



(a) Case 1: Eigenvalues of S_{11} are distributed along the perimeter of an ellipse

(b) Case 2: Eigenvalues of S_{11} are distributed along the segment

Figure 3.1: Restarting the Krylov–Schur algorithm with arbitrary shifts. Eigenvalues of A are red dots, the shifts used for the restart are the black dots. Eigenvalues of S_{11} in (3.12) are green dots. The Ritz values computed after restarting with Krylov–Schur are blue asterisks, and the ones obtained by the IRAM are blue circles.

the additional data about the eigenvalues of S_{11}. The Ritz values computed from the subspace restarted using the IRAM are shown as the blue circles.

The Krylov–Schur algorithm and the IRAM are two equivalent algorithms when the exact arithmetic is used. Thus, we expect that all of the blue asterisks fit perfectly into the blue circles. This is indeed the case for the left figure. However, the Ritz values computed by these algorithms are very different in the right figure.

Such a discrepancy is due to the ill-conditioning of the pole placement problem used by the Krylov–Schur algorithm. The pole placement algorithm has failed to put the eigenvalues of $B + gb^\star$ to the exact prescribed values. The initial vector for the restarted Krylov subspace is, therefore, far from the expected vector $\pi(A)v$, where π is a polynomial with roots $\sigma_1, \ldots, \sigma_p$.

This effect does not show up for the left figure. The pole placement algorithm can be relatively stable when the target eigenvalues are well spread in the complex plane and not too far from the eigenvalues of B. On the other hand, placing the eigenvalues rigidly along the segment is known to be a hard test-case for the pole placement.

The Example warns for caution when using the Krylov–Schur algorithm with any shifts other than the exact ones. As we have mentioned earlier, the interior eigenvalues of A are often approximated by the harmonic Ritz values, and the Krylov–Schur algorithm is restarted with the unwanted harmonic values as shifts using the explicit formula (3.10) for the solution of the pole placement problem. While a pole placement algorithm may issue a warning when the eigenvalues of $B + gb^\star$ differ severely from the expected, there is no such mechanism with (3.10). On the other hand, the harmonic Ritz values do not tend

to concentrate around a single point and are quite well distributed. Although this may suggest that the pole placement will succeed, the nature of the pole placement problem is such that there may be cases in which the restart with such shifts fails as well.

To conclude this discussion, we note that the implicitly restarted Arnoldi method does not implicitly nor explicitly solve a pole placement problem. Just before the restart, this method departs the Krylov decomposition, producing a rank-two residual:

$$A[\tilde{V}_{m-p}\ \hat{V}_p] = [\tilde{V}_{m-p}\ \hat{V}_p] \begin{bmatrix} \tilde{H}_{m-p} & \hat{H}_{m-p,p} \\ \tilde{\beta} e_1 e_{m-p}^\star & \hat{H}_{p,p} \end{bmatrix} + [\underbrace{0\ 0\ \ldots\ 0\ \hat{\beta} r_m}_{m-p}\ \underbrace{r_m b^\star}_{p}]$$

is first transformed into

$$\begin{aligned} A[\tilde{V}_{m-p}\ \hat{V}_p] &= [\tilde{V}_{m-p}\ \hat{V}_p] \begin{bmatrix} \tilde{H}_{m-p} & \hat{H}_{m-p,p} \\ 0 & \hat{H}_{p,p} \end{bmatrix} + \tilde{\beta}\hat{V}_p e_1 e_{m-p}^\star + [\underbrace{0\ 0\ \ldots\ 0\ \hat{\beta} r_m}_{m-p}\ \underbrace{r_m b^\star}_{p}] \\ &= [\tilde{V}_{m-p}\ \hat{V}_p] \begin{bmatrix} \tilde{H}_{m-p} & \hat{H}_{m-p,p} \\ 0 & \hat{H}_{p,p} \end{bmatrix} + x_1 b_1^\star + x_2 b_2^\star, \end{aligned}$$

and only then the last p columns are removed. With restarting of such kind, Theorem 3.3 cannot be applied.

3.3 Matrix balancing and the computation of the Ritz values

In this section we report an interesting effect that arises when computing the Ritz or the harmonic Ritz approximations using eigenvalue algorithms that incorporate the balancing preconditioning on the projection matrix B in the Krylov or the Arnoldi decomposition $AU = UB + ub^\star$. We have observed the effect in the various versions of the Arnoldi algorithm, but it is best explained with the harmonic Ritz approximations and the Krylov–Schur algorithm. This effect is not related to the pole placement problem discussed in the last section.

The usual approach when computing eigenvalues and eigenvectors of a small dense non-Hermitian matrix B is to reduce B to an upper Hessenberg matrix and then run a QR algorithm. Most algorithms, such as `DGEEV` from LAPACK and `eig` from MATLAB have an additional balancing step prior to the Hessenberg reduction. The goal of balancing is to find a permutation D of a diagonal matrix such that

$$B = D^{-1}\hat{B}D$$

and that norms of the rows and the columns of $\hat{B}$ do not widely vary in size. (To avoid roundoff errors, elements of D are usually chosen as powers of 2.) Suppose that the eigenvalue decompositions of B and $\hat{B}$ are given with $B = Y\Theta Y^{-1}$ and $\hat{B} = \hat{Y}\Theta\hat{Y}^{-1}$, respectively. The goal of balancing is to, hopefully, improve the condition number of the eigenvector matrix $\hat{Y}$ of $\hat{B}$ compared to the condition number of the eigenvector matrix

Y of B. If the improvement is achieved, then by the perturbation theory (see the Bauer–Fike Theorem 1.13), the computation of eigenvalues of $\hat{B}$ produces smaller errors than the computation of B's eigenvalues.

While balancing appears to be a non-invasive maneuver from which the computation can only benefit, there seem to exist situations in which the computed eigenpairs of B do not comply with what is expected when balancing is involved. In such cases, the matrix B is well conditioned for the eigenvalue computation, but the obtained result is of much lower quality than expected. Watkins [77] reports such occurrences when B is an upper Hessenberg matrix already. We discuss a case when B is a projection matrix within the Krylov–Schur algorithm with harmonic Ritz extraction.

Example 3.7. Let A be the matrix `nnc666` from the MatrixMarket [44] collection. This real non-Hermitian matrix has order 666 and its eigenvalues are either real numbers within the segment $[-800, 800]$, either purely imaginary within $[-2i, 2i]$.

Suppose that we want to compute 3 eigenvalues closest to $\tau = 650$ using the harmonic Ritz approximation. A random initial vector is used. The maximum dimension of the Krylov subspace is set to 40, after which it is restarted to dimension 20. For the restart, we use the technique of Stewart described in Section 3.2: the Schur form (3.8) is ordered so that 20 of its eigenvalues (harmonic Ritz values) which are closest to target τ appear in the top left block S_{11} and the others are chosen as shifts and put in the bottom right block S_{22}.

To track the convergence, at each Arnoldi step and after each restart we compute harmonic Ritz pairs (θ_h, x_h) of the orthogonal Krylov decomposition

$$AU_m = U_m B_m + u_m b_m^\star$$

along with their residuals $\|Ax_h - \theta_h x_h\|$. This is done using the following MATLAB pseudocode:

```
1  g = (B − τI)^⋆ \ b ;
2  [Y_h, Θ_h] = eig(B + g b^⋆);
3  X_h = U Y_h ;
4  (θ_{h,i}, x_{h,i}) = (Θ_h(i, i), X_h(:, i)), for i = 1, 2, . . .;
```

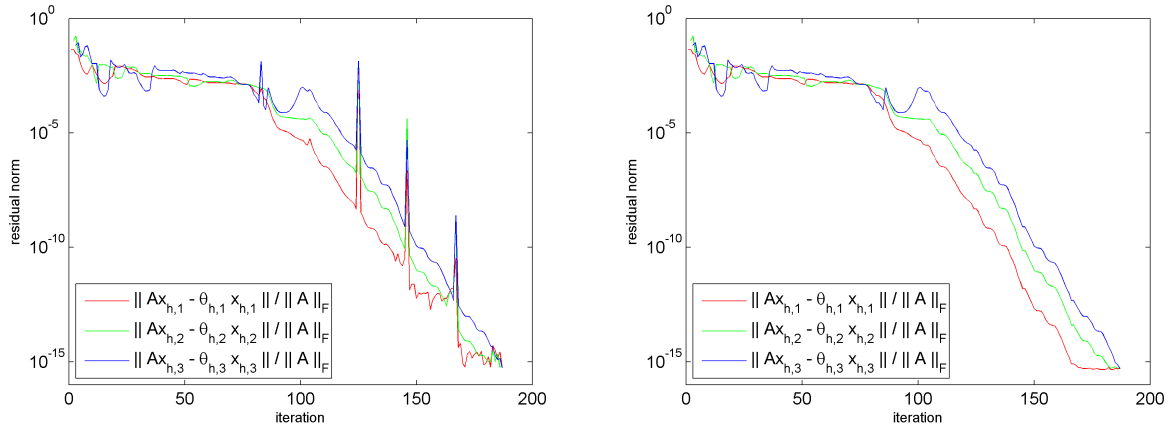
The residual is computed explicitly using the matrix–vector multiplication.

Figure 3.2(a) shows the residual history for the 3 pairs whose value was closest to the target. Strange spikes appear in the graph, especially just after several restarts, but the convergence seems to continue smoothly in the very next iteration.

Figure 3.2(b) shows the situation when Line 2 above is replaced with

```
2'  [Y_h, Θ_h] = eig(B + g b^⋆, 'nobalancing');
```

When no balancing of the matrix $B + gb^\star$ is used to extract the harmonic Ritz pairs, there



(a) Case 1: Eigenvalues of $B + gb^\star$ are computed using `eig(`$B + gb^\star$`)`

(b) Case 2: Eigenvalues of $B + gb^\star$ are computed using `eig(`$B + gb^\star$`, 'nobalance')`

Figure 3.2: Computed residual norms for top 3 harmonic Ritz values pairs closest to target 650 of matrix `nnc666`.

are no spikes in the graph. Also, the residuals of the first pair differ quite drastically in the later iterations.

Note that tracking the progress of the convergence and the computation of the harmonic Ritz values is usually done in a different way. The Example only warns of another case in which the successful extraction of the eigenpairs depends on whether balancing is used or not.

The matrix $\tilde{B} = B + gb^\star$ for which the balancing produces an erroneous result in the Example is a small perturbation of an upper triangular matrix. During the restart, the Schur factor S_{11}, whose eigenvalues are the harmonic Ritz values, is first modified by a translation (3.9) which maintains orthogonality of the Krylov decomposition: $B = S_{11} + g_1 b^\star$. Then, another translation with $gb^\star$ is made to retrieve the matrix S_{11}. The roundoff errors provide that $\tilde{B} = S_{11} + E$ for some matrix E such that $\|E\|$ is small.

To understand how balancing harms the eigenvalue extraction for $\tilde{B}$, let us construct a smaller example. Set

$$\tilde{B} = \begin{bmatrix} 0.6 & 0.3 & 1 & -0.3 & 1.0 \\ 10^{-13} & -1.3 & 1.4 & 0.3 & 0.7 \\ 10^{-15} & 9\cdot 10^{-14} & 0.4 & -0.04 & -1.5 \\ 10^{-17} & 10^{-15} & -5\cdot 10^{-13} & 0.5 & -0.3 \\ -10^{-19} & -5\cdot 10^{-17} & 2\cdot 10^{-14} & 3\cdot 10^{-14} & 2.2 \end{bmatrix}.$$

(We have used an optimization method in MATLAB to obtain this matrix.) The matrix $\tilde{B}$ has a condition number $\kappa(\tilde{B}) \approx 26.2$ and its eigenpairs should be computed accurately. We expect that MATLAB 's routine $[Y, \Theta]$ `= eig(`$\tilde{B}$`)` produces $\tilde{B}Y \approx Y\Theta$. However,

$$\left\| \tilde{B}Y - Y\Theta \right\| \approx 0.35,$$

which is a huge error. In particular, for the "eigenpair" (θ, y), $\theta \approx 2.2$, it holds that

$$\left\| \tilde{B}y - \theta y \right\| \approx 0.35.$$

The balancing is the cause of this error: we see that by calling $[D, T]$ `= balance(`$\tilde{B}$`)`. Here $T = D^{-1}\tilde{B}D$ and the eigenvalue algorithm in `eig` continues with computing the eigenvalues of T. Inspecting the scaling matrix D reveals that its diagonal elements vary widely in size:

$$D \approx \mathrm{diag}(4 \cdot 10^9, 10^6, 3 \cdot 10^1, 3 \cdot 10^{-2}, 7 \cdot 10^{-6}).$$

On the other hand, the matrix T is almost diagonal:

$$T \approx \begin{bmatrix} 0.6 & 7 \cdot 10^{-5} & 7 \cdot 10^{-9} & -2 \cdot 10^{-12} & 2 \cdot 10^{-15} \\ 4 \cdot 10^{-10} & -1.3 & 4 \cdot 10^{-5} & 9 \cdot 10^{-9} & 5 \cdot 10^{-12} \\ -10^{-7} & -3 \cdot 10^{-9} & 0.4 & -4 \cdot 10^{-5} & -4 \cdot 10^{-7} \\ 10^{-6} & 3 \cdot 10^{-7} & -5 \cdot 10^{-10} & 0.5 & -7 \cdot 10^{-5} \\ -6 \cdot 10^{-5} & -7 \cdot 10^{-6} & 8 \cdot 10^{-7} & 10^{-10} & 2.2 \end{bmatrix},$$

so its eigenvector $\tilde{y}$ associated with the eigenvalue $\theta \approx 2.2$ is very close to e_5. Suppose that $\tilde{y}$ is computed to high accuracy, so that

$$\|T\tilde{y} - \theta\tilde{y}\| \approx \epsilon \approx 2 \cdot 10^{-16},$$

but also that there exists a tiny component of the same magnitude of the vector $T\tilde{y} - \theta\tilde{y}$ in the direction e_1. Now, $D\tilde{y}$ is an approximation of a eigenvector of the matrix $\tilde{B}$. Using $\tilde{B} = DTD^{-1}$, we have:

$$\begin{aligned} \left\| \tilde{B}(D\tilde{y}) - \theta(D\tilde{y}) \right\| &= \|D(T\tilde{y} - \theta\tilde{y})\| \geq |D(1,1)| \cdot |e_1^\star(T\tilde{y} - \theta\tilde{y})| \\ &\approx 4 \cdot 10^9 \cdot 2 \cdot 10^{-16} \approx 8 \cdot 10^{-7}. \end{aligned}$$

However, $D\tilde{y}$ is not a unit vector:

$$\|D\tilde{y}\| \approx \|De_5\| = |D(5,5)| \approx 7 \cdot 10^{-6}.$$

Finally, setting $y = D\tilde{y}/\|D\tilde{y}\|$, we see that the residual of the computed eigenpair (θ, y) has a large norm:

$$\left\| \tilde{B}y - \theta y \right\| \approx \left\| \tilde{B}(D\tilde{y}) - \theta(D\tilde{y}) \right\| / \|D\tilde{y}\| \approx 8 \cdot 10^{-7}/7 \cdot 10^{-6} \approx 10^{-1}.$$

This phenomenon does not occur when balancing is not used. When the eigenvalue decomposition is computed using $[Y, \Theta]$ `= eig(`$\tilde{B}$`, 'nobalance')`, it holds that

$$\left\| \tilde{B}Y - Y\Theta \right\| \approx 3.5 \cdot 10^{-15}.$$

We have experienced similar issues with balancing when the Ritz or the harmonic Ritz values were extracted using either the Krylov–Schur or the implicitly restarted Arnoldi algorithm. In all cases, not balancing the matrix produced better results than balancing.

We have also found another Arnoldi code that addresses the issue of balancing. In the MATLAB code `ahbeigs` [4], first the default `eig` routine is called to compute the eigenpairs of the (block) Hessenberg matrix $B = Y\Theta Y^{-1}$. Then the residual $\|BY - \Theta Y\|$ is inspected, and if it is too large, `eig` is called once again, this time with the `'nobalance'` parameter. However, the article [3] which describes the `ahbeigs` routine does not mention balancing at all. Other software, such as ARPACK, avoid using the eigenvalue routines for the extraction of the Ritz pairs. The primitive Ritz pairs are computed from the Schur factorization of the Hessenberg matrix B. The Schur factorization does not use balancing as a preprocessing step.

Chapter 4

Ritz values of normal matrices

The restarted Arnoldi method, which uses the unwanted Ritz values as the roots of the filter polynomial, converges and successfully retrieves the extreme eigenpairs of a Hermitian matrix. This fact, proved by Sorensen [65] is based on the important and distinctive property of the Hermitian matrices: the Cauchy interlacing lemma. Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix and let $X \in \mathbb{C}^{n\times k}$ be an orthonormal matrix. If $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ are the eigenvalues of A and $\theta_1 \geq \theta_2 \geq \ldots \geq \theta_k$ are the eigenvalues of $\rho(X) = X^\star AX$, then the Cauchy interlacing lemma states that

$$\lambda_{n-k+j} \leq \theta_j \leq \lambda_j,$$

for all $j = 1, 2, \ldots, k$. In the context of the Arnoldi algorithm, this property provides the monotone growth of the approximations for each eigenvalue λ_j during the expansion phase of the algorithm. The restart with the exact shifts simply preserves the approximations intact, maintaining convergence.

The counterexample for the convergence in the general case [28] shows that there cannot exist a similar monotonicity property for non-normal matrices. The problem in that setting arises not only from the lack of proper ordering of the complex eigenvalues, but also from the non-trivial geometry of the eigenvectors. We believe that more insight in the convergence of the Arnoldi algorithm can be made by considering a simpler case of normal matrices.

In this Chapter, we take a look at a generalization of the minimax theorem for the normal matrices, and discuss preservation of monotonicity during the augmentation and the restart phase of the Arnoldi algorithm. We also construct an example in which a close approximation to the wanted eigenvalue of A is deflated during the restart. Using a Cauchy matrix, we give necessary and sufficient conditions for a tuple of complex numbers to appear as the Ritz values of a given normal matrix. By reproducing some less trivial results for Hermitian matrices, we show that this characterization is a powerful tool in exploring the mutual positions of the Ritz values.

4.1 Minimax theorem and the Cauchy interlacing property

The Cauchy interlacing theorem for Hermitian matrices is a consequence of the minimax characterization of the eigenvalues. As we have already seen in previous chapters, if $A \in \mathbb{C}^{n\times n}$ is a Hermitian matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$, then for all $j = 1, 2, \ldots, n$ we have

$$\lambda_j = \max_{\dim \mathcal{X}=j} \min_{\substack{x\in\mathcal{X} \\ \|x\|=1}} x^\star Ax.$$

Another consequence is the Weyl's theorem: if E is another Hermitian matrix and the eigenvalues of $A + E$ are $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \ldots \geq \tilde{\lambda}_n$, then

$$\max_{j=1,\ldots,n} |\tilde{\lambda}_j - \lambda_j| \leq \|E\|.$$

In order to generalize these facts to non-Hermitian matrices, we first need to introduce an ordering into the set of complex numbers.

Definition 4.1. *Let $a, b \in \mathbb{C}$, and $\varphi \in [0, 2\pi\rangle$. We define a binary relation $\leq^{(\varphi)}$ with*

$$\begin{aligned} a \leq^{(\varphi)} b \quad \Leftrightarrow \quad & \operatorname{Re}\left(e^{-i\varphi}a\right) < \operatorname{Re}\left(e^{-i\varphi}b\right) \quad \text{or} \\ & \operatorname{Re}\left(e^{-i\varphi}a\right) = \operatorname{Re}\left(e^{-i\varphi}b\right) \quad \text{and} \quad \operatorname{Im}\left(e^{-i\varphi}a\right) \leq \operatorname{Im}\left(e^{-i\varphi}b\right). \end{aligned}$$

Here $\operatorname{Re}(z)$ and $\operatorname{Im}(z)$ denote the real and the imaginary part of the complex number z, respectively. Relation $\geq^{(\varphi)}$ is defined analogously.
Furthermore, for a subset $S \subseteq \mathbb{C}$, define $\min^{(\varphi)} S$ and $\max^{(\varphi)} S$ as respectively the smallest and the largest element of S in terms of ordering $\leq^{(\varphi)}$.

The relation $\leq^{(\varphi)}$ defines a total ordering on $\mathbb{C}$. However, $\mathbb{C}$ cannot be arranged into an ordered field, and the ordering $\leq^{(\varphi)}$ is not compatible with the multiplication of complex numbers: from $0 \leq^{(\varphi)} a$ and $0 \leq^{(\varphi)} b$ it does not follow $0 \leq^{(\varphi)} ab$. The relation $\leq^{(\varphi)}$ has the following geometric interpretation: let p_φ be a line through the origin closing an angle φ with the positive direction of the real axis. Project the complex numbers a and b onto p_φ: smaller is the one which is more "down the line"; see Figure 4.1. Note that $e^{-i\varphi}a$ rotates $a = Ae^{i\alpha}$ by the angle φ clockwise:

$$\operatorname{Re}\left(e^{-i\varphi}a\right) = \operatorname{Re}\left(Ae^{i(\alpha-\varphi)}\right) = A\cos(\alpha - \varphi),$$

which is exactly the distance from the origin to the projection of a onto line p_φ. Another way of interpretation is that the $\leq^{(\varphi)}$ is the lexicographical ordering when the coordinate axes are rotated by the angle φ in the positive direction.

Quiero and Duarte [56] used this ordering in spectral analysis of normal matrices and showed the (a) and (b) parts of the following Theorem.

Theorem 4.2. *Let A be a normal matrix with eigenvalues $\lambda_1 \geq^{(\varphi)} \lambda_2 \geq^{(\varphi)} \ldots \geq^{(\varphi)} \lambda_n$.*

(a) The minimax property holds:

$$\lambda_j = \max^{(\varphi)}_{\dim \mathcal{X}=j} \min^{(\varphi)}_{\substack{x\in\mathcal{X} \\ \|x\|=1}} x^\star Ax.$$

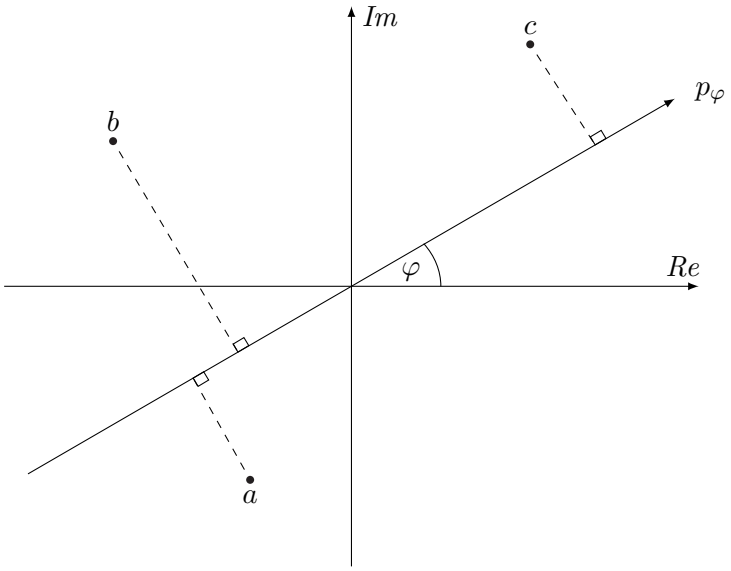


Figure 4.1: The ordering $\leq^{(\varphi)}$: it holds that $a \leq^{(\varphi)} b \leq^{(\varphi)} c$.

(b) The Cauchy interlacing holds: if $X \in \mathbb{C}^{n\times k}$ is an orthonormal matrix such that $X^\star AX$ is normal with eigenvalues $\theta_1 \geq^{(\varphi)} \ldots \geq^{(\varphi)} \theta_k$, then for all $j = 1, 2, \ldots, k$ we have

$$\lambda_{(n-k)+j} \leq^{(\varphi)} \theta_j \leq^{(\varphi)} \lambda_j.$$

(c) Weyl's theorem holds: let E be any matrix such that $A + E$ is normal and has eigenvalues $\tilde{\lambda}_1 \geq^{(\varphi)} \tilde{\lambda}_2 \geq^{(\varphi)} \ldots \geq^{(\varphi)} \tilde{\lambda}_n$. Then

$$-e^{i\varphi}(\|E\| + i\,\|E\|) \leq^{(\varphi)} \lambda_j - \tilde{\lambda}_j \leq^{(\varphi)} e^{i\varphi}(\|E\| + i\,\|E\|).$$

Proof. (a) Let $Au_i = \lambda_i u_i$, $\mathcal{U}_i = \operatorname{span}\{u_1, \ldots, u_i\}$. Then for $x \in \mathcal{X} = \mathcal{U}_j$, $\|x\| = 1$, we have $x = \sum_{i=1}^{j} \alpha_i u_i$, $\sum_{i=1}^{j} |\alpha_i|^2 = 1$, and

$$x^\star Ax = \sum_{i=1}^{j} |\alpha_i|^2 \lambda_i \geq^{(\varphi)} \lambda_j.$$

(Note that last inequality is easily generalized from $\geq$ to $\geq^{(\varphi)}$.) Thus,

$$\min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} x^\star Ax \geq^{(\varphi)} \lambda_j,$$

and

$$\max_{\dim\mathcal{X}=j}^{(\varphi)} \min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} x^\star Ax \geq^{(\varphi)} \lambda_j.$$

To show the opposite, let $\mathcal{X}$ be any j-dimensional subspace. There is a nontrivial unit vector $x \in \mathcal{X} \cap \operatorname{span}\{u_j, u_{j+1}, \ldots, u_n\}$. As before,

$$x^\star A x \leq^{(\varphi)} \lambda_j,$$

so we have

$$\min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} x^\star A x \leq^{(\varphi)} \lambda_j.$$

Since $\mathcal{X}$ was arbitrary,

$$\max_{\dim\mathcal{X}=j}^{(\varphi)} \min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} x^\star A x \leq^{(\varphi)} \lambda_j.$$

(b) The statement trivially follows from (a), noting that

$$\theta_j = \max_{\substack{\dim\mathcal{Y}=j\\ \mathcal{Y}\leqslant\mathcal{X}}}^{(\varphi)} \min_{\substack{y\in\mathcal{Y}\\ \|y\|=1}}^{(\varphi)} y^\star A y,$$

where $\mathcal{X}$ is the subspace spanned by the columns of X.

(c) By the statement in (a), we have

$$\tilde{\lambda}_j = \max_{\dim\mathcal{X}=j}^{(\varphi)} \min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} x^\star (A+E) x.$$

It can be shown that for any functions f and g and for any set $\mathcal{S}$ it holds that

$$\min_{x\in\mathcal{S}}^{(\varphi)} (f(x)+g(x)) \geq^{(\varphi)} \min_{x\in\mathcal{S}}^{(\varphi)} f(x) + \min_{x\in\mathcal{S}}^{(\varphi)} g(x);$$

thus

$$\min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} (x^\star A x + x^\star E x) \geq^{(\varphi)} \min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} (x^\star A x) + \min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} (x^\star E x). \tag{4.1}$$

We claim that

$$e^{i\varphi}(\|E\| + i\,\|E\|) \geq^{(\varphi)} \min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} (x^\star E x) \geq^{(\varphi)} -e^{i\varphi}(\|E\| + i\,\|E\|)$$

for any matrix E: this is equivalent to $\|E\| \geq \operatorname{Re}\left(e^{-i\varphi} x^\star E x\right) \geq -\|E\|$ (and similarly for $\operatorname{Im}()$ if equality holds). Both claims follow from the fact $|e^{-i\varphi} x^\star E x| \leq \|x^\star\|\,\|E\|\,\|x\| = \|E\|$.

Returning to (4.1), it follows that for each j-dimensional $\mathcal{X}$ we have

$$\min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} (x^\star (A+E) x) \geq^{(\varphi)} \min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} (x^\star A x) - e^{i\varphi}(\|E\| + i\,\|E\|).$$

Let $\tilde{\mathcal{X}}$ be the subspace for which the maximum in (a) is obtained (the eigenspace for A). For such subspace it holds that

$$\begin{aligned}
\tilde{\lambda}_j &= \max_{\dim\mathcal{X}=j}^{(\varphi)} \min_{\substack{x\in\mathcal{X}\\ \|x\|=1}}^{(\varphi)} (x^\star (A+E) x) \\
&\geq^{(\varphi)} \min_{\substack{x\in\tilde{\mathcal{X}}\\ \|x\|=1}}^{(\varphi)} (x^\star (A+E) x) \\
&\geq^{(\varphi)} \min_{\substack{x\in\tilde{\mathcal{X}}\\ \|x\|=1}}^{(\varphi)} (x^\star A x) - e^{i\varphi}(\|E\| + i\,\|E\|) \\
&= \lambda_j - e^{i\varphi}(\|E\| + i\,\|E\|).
\end{aligned}$$

The other inequality is proved similarly (or just swap A, $A+E$).

□

Several comments are in order. First, note that the set of Hermitian matrices has the desirable property of being closed under projection: if A is Hermitian, then $X^\star AX$ is Hermitian as well. Unfortunately, the same does not hold for the set of normal matrices, and this fact narrows the use of part (b) of the Theorem. Second, the Weyl inequality does not imply that $|\lambda_j - \tilde{\lambda}_j|$ is small. It only implies that the projection of $\lambda_j - \tilde{\lambda}_j$ onto the line p_φ is small, for every φ. (But note that the ordering of the eigenvalues changes with φ.) The Wielandt–Hoffman theorem states that there exists a permutation π such that

$$\sqrt{\sum_{j=1}^{n} |\lambda_j - \tilde{\lambda}_{\pi(j)}|^2} \leq \|E\|_F \,,$$

and the permutation itself cannot be reconstructed from the proof. A small step in that direction is the following result.

Corollary 4.3. *Let the angles φ and ψ define two orthogonal lines through the origin of the complex plane ($\psi = \varphi + \pi/2$). Assume that λ and $\tilde{\lambda}$ are both j-th largest eigenvalues of A and $A+E$ respectively, using the ordering $\leq^{(\phi)}$, for some j. Also, assume that λ and $\tilde{\lambda}$ are both k-th largest eigenvalues of A and $A+E$ respectively, using the ordering $\leq^{(\psi)}$, for some k. Then*

$$|\lambda - \tilde{\lambda}| \leq \sqrt{2}\,\|E\| \,.$$

Proof. Let $\Delta = \lambda - \tilde{\lambda}$. Then

$$\operatorname{Re}\left(e^{-i\varphi}\Delta\right) = \cos\varphi \cdot \operatorname{Re}(\Delta) + \sin\varphi \cdot \operatorname{Im}(\Delta) \tag{4.2}$$

and

$$\operatorname{Re}\left(e^{-i\psi}\Delta\right) = -\sin\varphi \cdot \operatorname{Re}(\Delta) + \cos\varphi \cdot \operatorname{Im}(\Delta) \,. \tag{4.3}$$

By Weyl's theorem, we have

$$|\operatorname{Re}\left(e^{-i\varphi}\Delta\right)|^2 + |\operatorname{Re}\left(e^{-i\psi}\Delta\right)|^2 \leq 2\,\|E\|^2 \,;$$

on the other hand, by (4.2) and (4.3),

$$|\operatorname{Re}\left(e^{-i\varphi}\Delta\right)|^2 + |\operatorname{Re}\left(e^{-i\psi}\Delta\right)|^2 = \operatorname{Re}(\Delta)^2 + \operatorname{Im}(\Delta)^2 = |\lambda - \tilde{\lambda}|^2.$$

□

The minimax theorem provides the precise location of an eigenvalue in the complex plane, with the price of very restrictive use of the interlacing property. We now take a look at the somewhat weaker approach, which only locates the projection of an eigenvalue to p_φ, ignoring the orthogonal component when the projections are equal. This approach also allows for easier use of the minimax, with no complications involving $\leq^{(\varphi)}$.

Definition 4.4. *Let $\varphi \in [0, 2\pi\rangle$.*
For $a \in \mathbb{C}$, denote $a^{(\varphi)} = \frac{1}{2}\left(e^{-i\varphi}a + e^{i\varphi}\overline{a}\right)$. For any matrix A, denote

$$A^{(\varphi)} = \frac{1}{2}\left(e^{-i\varphi}A + e^{i\varphi}A^\star\right).$$

Our first observation is that

$$\operatorname{Re}\left(e^{-i\varphi}a\right) = \frac{1}{2}\left(e^{-i\varphi}a + e^{i\varphi}\overline{a}\right) = a^{(\varphi)}$$

for every $a \in \mathbb{C}$, which shows that $a^{(\varphi)} \in \mathbb{R}$ is the (oriented) length of the projection of the number a onto the line p_φ. Furthermore, the matrix $A^{(\varphi)}$ is always a Hermitian matrix. We can also easily derive the appropriate formula for a Rayleigh quotient of $A^{(\varphi)}$: if X is orthonormal and $H = X^\star AX$, then

$$\begin{aligned} H^{(\varphi)} &= \frac{1}{2}\left(e^{-i\varphi}H + e^{i\varphi}H^\star\right) \\ &= \frac{1}{2}\left(e^{-i\varphi}X^\star AX + e^{i\varphi}X^\star A^\star X\right) \\ &= X^\star\frac{1}{2}\left(e^{-i\varphi}A + e^{i\varphi}A^\star\right)X \\ &= X^\star A^{(\varphi)}X. \end{aligned}$$

Theorem 4.5. *Suppose A is a normal matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$. Then the eigenvalues of $A^{(\varphi)}$ are $\lambda_1^{(\varphi)}, \ldots, \lambda_n^{(\varphi)}$. Thus, if the eigenvalues of A are ordered so that $\lambda_1^{(\varphi)} \geq \ldots \geq \lambda_n^{(\varphi)}$, then for all $j = 1, \ldots, n$ it holds that*

$$\lambda_j^{(\varphi)} = \max_{\dim\mathcal{X}=j}\ \min_{\substack{x\in\mathcal{X}\\ \|x\|=1}} x^\star A^{(\varphi)}x.$$

Proof. The statement of the theorem follows from the definition of $A^{(\varphi)}$ and the fact that if (λ, u) is an eigenpair of A, then $(\overline{\lambda}, u)$ is an eigenpair of $A^\star$. Note that all three matrices A, $A^\star$, $A^{(\varphi)}$ have the same eigenvectors! □

For general matrices, there seems to be no simple relation between the eigenvalues of $A^{(\varphi)}$ and the projections $\lambda_j^{(\varphi)}$. However, if A is normal and we are approximating its eigenvectors from some subspace $\mathcal{X}$ using the Rayleigh quotient $H = X^\star AX$, then there is a connection between eigenvalues of $H^{(\varphi)}$ and projection of the Ritz values.

Theorem 4.6. *Let $A \in \mathbb{C}^{n\times n}$ be a normal matrix and $H = X^\star AX$ for some orthonormal matrix $X \in \mathbb{C}^{n\times k}$. Let (θ, y) be an eigenpair of H and $r = AXy - \theta Xy$ the residual of a Ritz pair. Then there exists an eigenvalue $\tilde{\theta}$ of $H^{(\varphi)}$ such that*

$$|\tilde{\theta} - \theta^{(\varphi)}| \leq \frac{1}{2}\|r\|.$$

As a consequence, if the residual of a Ritz pair converges to zero, then an eigenvalue of $H^{(\varphi)}$ converges to $\theta^{(\varphi)}$.
(Note that there exists an eigenvalue λ of A such that $|\lambda - \theta| \leq \|r\|$, and thus we trivially have $|\lambda^{(\varphi)} - \theta^{(\varphi)}| \leq \|r\|$.)

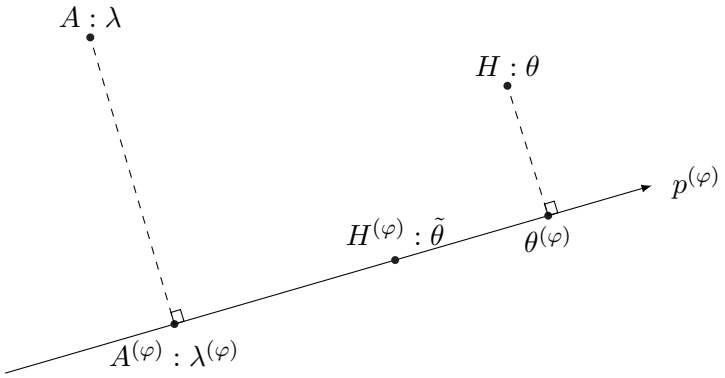


Figure 4.2: Eigenvalues λ and θ of matrices A and H are projected to the line $p^{(\varphi)}$ to obtain the eigenvalue $\lambda^{(\varphi)}$ of $A^{(\varphi)}$ and the number $\theta^{(\varphi)}$, which is generally not equal to the eigenvalue $\tilde{\theta}$ of the matrix $H^{(\varphi)}$.

Proof. Consider the eigenvalue decomposition $A = U\Lambda U^\star$, and let $x = Xy$. Then

$$\|r\| = \|U\Lambda U^\star x - \theta x\| = \|\Lambda U^\star x - \theta U^\star x\| = \|\Lambda z - \theta z\|$$

for $z = U^\star x$. Similarly, with $s = A^\star x - \overline{\theta} x$ we have

$$\|s\|^2 = \left\|\Lambda^\star z - \overline{\theta} z\right\|^2 = \sum_{j=1}^{n} |\overline{\lambda}_j - \overline{\theta}|^2 |z_j|^2 = \sum_{j=1}^{n} |\lambda_j - \theta|^2 |z_j|^2 = \|\Lambda z - \theta z\|^2 = \|r\|^2 .$$

Thus

$$\left\|H^\star y - \overline{\theta} y\right\| = \left\|X^\star A^\star X y - \overline{\theta} y\right\| = \|X^\star s\| \le \|X^\star\| \, \|s\| = \|r\| ,$$

and we can write $H^\star y = \overline{\theta} y + d$, for some vector d such that $\|d\| \le \|r\|$. We use y as the approximation of an eigenvector of $H^{(\varphi)}$:

$$H^{(\varphi)} y = \frac{1}{2}\left(e^{-i\varphi} H y + e^{i\varphi} H^\star y\right) = \frac{1}{2}\left(e^{-i\varphi}\theta + e^{i\varphi}\overline{\theta}\right) y + \frac{1}{2} e^{i\varphi} d = \theta^{(\varphi)} y + \frac{1}{2} e^{i\varphi} d.$$

The matrix $H^{(\varphi)}$ is Hermitian and there exists its eigenvalue $\tilde{\theta}$ such that

$$|\tilde{\theta} - \theta^{(\varphi)}| \le \left\| \frac{1}{2} e^{i\varphi} d \right\| \le \frac{1}{2} \|r\| .$$

□

It is easier to monitor the eigenvalues of Hermitian matrix $H^{(\varphi)}$ than to track projections $\theta^{(\varphi)}$ of the eigenvalues of H. Theorem 4.6 demonstrates that the former are tied to the latter in the same way the Ritz values are tied to the eigenvalues of A.

We now turn our discussion to the Arnoldi algorithm. Using the technique we developed with Hermitian matrices $A^{(\varphi)}$, $H^{(\varphi)}$, it is very simple to show that eigenvalues of $H^{(\varphi)}$ exhibit a monotone growth during the augmentation phase. This is stated in the following result, which holds even for non-normal matrices A.

Theorem 4.7. *Let $A \in \mathbb{C}^{n\times n}$ be an arbitrary matrix; let $X \in \mathbb{C}^{n\times k}$ and $X_+ = [X \;\; x_+] \in \mathbb{C}^{n\times(k+1)}$ be orthonormal matrices and let $H = X^\star AX \in \mathbb{C}^{k\times k}$ and $H_+ = X_+^\star AX_+ \in \mathbb{C}^{(k+1)\times(k+1)}$ be the corresponding Rayleigh quotients. For $\varphi \in [0, 2\pi\rangle$, denote with $\lambda_j^{(\varphi)}$, $\tilde\theta_j$, $\tilde\theta_{j+}$ the j-th largest eigenvalues of $A^{(\varphi)}$, $H^{(\varphi)}$ and ${H_+}^{(\varphi)}$ respectively. The following statements hold for all $j = 1, \ldots, k$:*

(a) $\lambda_{(n-k)+j}^{(\varphi)} \le \tilde\theta_j \le \lambda_j^{(\varphi)}$;

(b) $\tilde\theta_{j+1,+} \le \tilde\theta_j \le \tilde\theta_{j+}$.

Proof. (a) Let $\mathcal{X} = \operatorname{Im} X$.

$$\begin{aligned}
\lambda_j^{(\varphi)} &= \max_{\dim\mathcal{V}=j} \min_{\substack{x\in\mathcal{V}\\ \|x\|=1}} x^\star A^{(\varphi)} x \ge \max_{\substack{\dim\mathcal{V}=j\\ \mathcal{V}\subseteq\mathcal{X}}} \min_{\substack{x\in\mathcal{V}\\ \|x\|=1}} x^\star A^{(\varphi)} x \\
&= \max_{\dim\mathcal{V}=j} \min_{\substack{x=Xv\\ v\in\mathcal{V},\|v\|=1}} x^\star A^{(\varphi)} x = \max_{\dim\mathcal{V}=j} \min_{\substack{v\in\mathcal{V}\\ \|v\|=1}} v^\star X^\star A^{(\varphi)} X v \\
&= \max_{\dim\mathcal{V}=j} \min_{\substack{v\in\mathcal{V}\\ \|v\|=1}} v^\star H^{(\varphi)} v = \tilde\theta_j.
\end{aligned}$$

For the other inequality, insert $-A$ instead of A above.

(b) Note $H = X^\star H_+ X$ for $X = \left[\begin{smallmatrix} I_k \\ 0 \end{smallmatrix}\right]$ and use (a).

□

This theorem, combined with the previous one, strongly supports the well-known fact that the Ritz values converge to the external eigenvalues of A. Let A be a normal matrix and λ an eigenvalue on the convex hull of the spectrum $\Lambda(A)$. Consider any line p_φ passing through λ and through the interior of the convex hull. Without loss of generality, we can assume that the line passes through the origin as well (or consider a line parallel to p_φ that does). Let $\mathcal{X}^{(1)} \leqslant \mathcal{X}^{(2)} \leqslant \ldots$ be an increasing sequence of subspaces and $H^{(1)}, H^{(2)}, \ldots$ the associated Rayleigh quotients. If we view the line p_φ as the real line with origin at λ with the positive direction towards the exterior of the convex hull, then λ coincides with $\lambda_1^{(\varphi)}$. The eigenvalues $\tilde\theta_1^{(1)}, \tilde\theta_1^{(2)}, \ldots$ of matrices $(H^{(1)})^{(\varphi)}, (H^{(2)})^{(\varphi)}, \ldots$ monotonically move towards λ. We expect these values to approximate the projections $(\theta_1^{(1)})^{(\varphi)}, (\theta_1^{(2)})^{(\varphi)}, \ldots$ of the Ritz values on the line p_φ.

The main problem that appears in proving the convergence of the sequence $(\tilde\theta_1^{(i)})_i$ towards $\lambda = \lambda^{(\varphi)}$ is the inability to retain the monotonicity of $\tilde\theta$'s during the restart of the Arnoldi algorithm. Consider a run of the implicitly restarted Arnoldi method

$$AV_j^{(i)} = V_j^{(i)} H_j^{(i)} + r_j^{(i)} e_j^\star,$$

where so far $i-1$ restarts have been taken, $V_j^{(i)}$ is the orthonormal basis for the Krylov space $\mathcal{K}_j(v^{(i)}; A)$. Restarts are done when $j = m$, and after each restart the dimension is equal to $j = m - p$. We explore the behavior of $(H_m^{(i)})^{(\varphi)}$.

During the augmentation phase (for a fixed value of i, and for the dimension j increasing from $m-p$ to m), eigenvalues of $(H_j^{(i)})^{(\varphi)}$ exhibit a monotone growth, as shown in

Theorem 4.7. At the same time, these eigenvalues interlace the eigenvalues of $A^{(\varphi)}$. We would like to construct the shifts for the restart so that this monotonicity is kept.

Note that the exact shifts won't do the trick: let Q be the similarity that is applied to IRAM with these shifts. By the construction we have

$$Q^\star H_m^{(i)} Q = \begin{bmatrix} H_{m-p}^{(i+1)} & \hat{H}^{(i)} \\ 0 & \tilde{H}^{(i)} \end{bmatrix},$$

where $H_{m-p}^{(i+1)}$ contains the wanted eigenvalues (say, the largest ones when sorted by $\leq^{(\varphi)}$) and is precisely the Rayleigh quotient after the restart. The matrix $\tilde{H}^{(i)}$ contains the unwanted eigenvalues of $H_m^{(i)}$. However, the matrix $\left(H_{m-p}^{(i+1)}\right)^{(\varphi)}$ is a submatrix of

$$\left(Q^\star H_m^{(i)} Q\right)^{(\varphi)} = \begin{bmatrix} \left(H_{m-p}^{(i+1)}\right)^{(\varphi)} & \frac{1}{2}e^{-i\varphi}\hat{H}^{(i)} \\ \frac{1}{2}e^{i\varphi}(\hat{H}^{(i)})^\star & (\tilde{H}^{(i)})^{(\varphi)} \end{bmatrix},$$

which is similar to $\left(H_m^{(i)}\right)^{(\varphi)}$. By the Cauchy interlacing theorem for Hermitian matrices, the eigenvalues of $\left(H_{m-p}^{(i+1)}\right)^{(\varphi)}$ interlace those of $\left(H_m^{(i)}\right)^{(\varphi)}$, and monotonicity is not necessarily maintained.

We need some other method for choosing the shifts. Let us consider the case $p = 1$, i. e. only one shift σ has to be chosen. If Q is the transformation matrix, then the above discussion shows that we would like to achieve

$$\left(Q^\star H_m^{(i)} Q\right)^{(\varphi)} = \begin{bmatrix} \left(H_{m-p}^{(i+1)}\right)^{(\varphi)} & 0 \\ 0 & \theta \end{bmatrix}.$$

In this way, $\left(H_{m-p}^{(i+1)}\right)^{(\varphi)}$ would have the same eigenvalues as $\left(H_m^{(i)}\right)^{(\varphi)}$ and the monotonicity would be preserved. Obviously, the last column of Q must be an eigenvector of $\left(H_m^{(i)}\right)^{(\varphi)}$ for its eigenvalue θ:

$$\left(Q^\star H_m^{(i)} Q\right)^{(\varphi)} e_m = Q^\star \left(H_m^{(i)}\right)^{(\varphi)} Q e_m = \theta e_m \quad \Longrightarrow \quad \left(H_m^{(i)}\right)^{(\varphi)} (Q e_m) = \theta (Q e_m).$$

Let (θ, x) be the eigenpair of $\left(H_m^{(i)}\right)^{(\varphi)}$. When is it possible to construct a shift σ for the IRAM such that $Qe_m = x$?

To simplify the notation, let $H = H_m^{(i)}$. The explicit shifting has the form

$$\left.\begin{aligned} H - \sigma I &= QR \\ H_1 = Q^\star H Q &= RQ + \sigma I \end{aligned}\right\}$$

Thus $(H - \sigma I)^{-1} = R^{-1}Q^\star$ and $(H - \sigma I)^{-\star} = QR^{-\star}$. Note that $R^{-\star}$ is lower triangular, so the last column of this equality reads

$$(H - \sigma I)^{-\star} e_m = QR^{-\star} e_m = \rho \cdot Q e_m = \rho x,$$

for some scalar ρ (which depends on the shift σ). We need to select σ so that the last column of $(H - \sigma I)^{-\star}$ is colinear with a prescribed vector x. Equivalently,

$$(H^\star - \overline{\sigma} I)x = \tilde{\rho} e_m. \tag{4.4}$$

Let us rewrite this equation row by row. If $x = [x_1\ x_2\ \ldots\ x_m]^\tau$, and $H = [H_{ij}]_{i,j=1\ldots m}$, we see that σ is determined already by the first row of the equality above:

$$(\overline{H_{11}} - \overline{\sigma})x_1 + \overline{H_{21}} x_2 = 0.$$

It is highly unlikely that this σ would satisfy the remaining $m-1$ equations. For example, when H is normal, we have shown that eigenvectors of H, $H^\star$ and $H^{(\varphi)}$ coincide (Theorem 4.5). The equation (4.4) then transforms into

$$(H^\star - \overline{\sigma} I)x = (\overline{\theta} - \overline{\sigma})x = \tilde{\rho} e_m.$$

Since we have already excluded the case $\sigma = \theta$, it must hold that $x = e_m$, which is obviously not true in general.

Therefore, we have demonstrated that it is not possible to choose shifts in IRAM so that the monotonicity of $\tilde{\theta}$'s is preserved during the restart.

4.2 Characterization using a Cauchy matrix

A rather different approach to the problem of locating the Ritz values of a normal matrix is based on the optimality of the characteristic polynomial for the Rayleigh quotient matrix. We have proved this well-known theorem in a more general setting in Chapter 3 (see Theorem 3.2). Our discussion now focuses on the Ritz values computed from the Krylov subspaces only.

Theorem 4.8. *Let $A \in \mathbb{C}^{n\times n}$, and let $AU = UB + ub^\star$ be an orthogonal Krylov decomposition (the matrix $[U\ \ u]$ is orthonormal) such that $\operatorname{Im} U = \mathcal{K}_k(v; A)$ for some starting vector v. Then the characteristic polynomial κ_B of the matrix B satisfies*

$$\|\kappa_B(A)v\| = \min_{\pi\in\mathbb{P}_k^\star} \|\pi(A)v\|,$$

where the minimum is taken over all monic polynomials π of degree k.

This is the only known optimality property of the Krylov subspace approximations that holds in the general setting. Using this Theorem, we will show that a k-tuple of complex numbers can appear as the Ritz values for a given normal matrix if and only if a certain system of linear equations has a positive solution. Let us first define what we mean by a positive solution.

Definition 4.9. *For a vector $x = (x_1, x_2, \ldots, x_n) \in \mathbb{C}^n$, we write $x > 0$ if the coordinates $x_1, x_2, \ldots, x_n$ are all non-negative real numbers and at least one of them is positive. We say that x is a positive vector.*
For a vector $x = (x_1, x_2, \ldots, x_n) \in \mathbb{C}^n$, we write $x \gg 0$ if the coordinates $x_1, x_2, \ldots, x_n$ are all positive real numbers. We say that x is a strictly positive vector.

The linear system of interest will have a special structure of a Cauchy matrix.

Definition 4.10. *Let $A = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{C}^n$ and $B = (\beta_1, \beta_2, \ldots, \beta_k) \in \mathbb{C}^k$ be such that $\alpha_i \neq \beta_j$ for all $i = 1, \ldots, n$ and $j = 1, \ldots, k$. The Cauchy matrix for tuples A and B is defined as*

$$C(A,B) = \begin{bmatrix} \frac{1}{\alpha_1-\beta_1} & \frac{1}{\alpha_2-\beta_1} & \cdots & \frac{1}{\alpha_n-\beta_1} \\ \frac{1}{\alpha_1-\beta_2} & \frac{1}{\alpha_2-\beta_2} & \cdots & \frac{1}{\alpha_n-\beta_2} \\ & & \vdots & \\ \frac{1}{\alpha_1-\beta_k} & \frac{1}{\alpha_2-\beta_k} & \cdots & \frac{1}{\alpha_n-\beta_k} \end{bmatrix} \in \mathbb{C}^{k\times n}.$$

Cauchy matrices appear often in the matrix theory. Special cases such as the Hilbert matrix ($\alpha_i - \beta_j = i + j - 1$) are of particular interest as well. We list a couple of interesting facts for the square Cauchy matrices, see [73] for details and a proof.

Theorem 4.11. *Let $A = (\alpha_1, \ldots, \alpha_n)$ and $B = (\beta_1, \ldots, \beta_n)$ be n-tuples of complex numbers such that $\alpha_i \neq \beta_j$ for all $i = 1, \ldots, n$ and $j = 1, \ldots, n$.*

(a) The determinant of the Cauchy matrix $C(A,B)$ is given by the formula

$$\det C(A,B) = \frac{\prod_{1\leq i<j\leq n}(\alpha_i - \alpha_j)(\beta_j - \beta_i)}{\prod_{i,j=1}^{n}(\alpha_i - \beta_j)}.$$

(b) The matrix $C(A,B)$ is non-singular and the elements of the inverse $D = C^{-1}(A,B)$ are given by

$$D_{pq} = \frac{\prod_{i=1}^{n}(\alpha_i - \beta_p)(\alpha_q - \beta_i)}{(\alpha_p - \beta_q)\prod_{\substack{i=1\\ i\neq p}}^{n}(\beta_p - \beta_i)\prod_{\substack{i=1\\ i\neq q}}^{n}(\alpha_i - \alpha_q)}.$$

Let us define another matrix $P(\Lambda, \Theta) \in \mathbb{C}^{n\times k}$ given the tuples $\Lambda = (\lambda_1, \ldots, \lambda_n)$ and $\Theta = (\theta_1, \ldots, \theta_k)$:

$$(P(\Lambda,\Theta))_{ij} = (\lambda_i - \theta_j) \cdot \prod_{\substack{q=1\\ q\neq j}}^{k} |\lambda_i - \theta_q|^2.$$

We are now ready to state our main result.

Theorem 4.12. *Let $A \in \mathbb{C}^{n\times n}$ be a normal matrix with eigenvalues $\Lambda = (\lambda_1, \ldots, \lambda_n)$. Let $\Theta = (\theta_1, \ldots, \theta_k)$ be a k-tuple of distinct complex numbers such that $\lambda_i \neq \theta_j$, for all $i = 1, \ldots, n$ and $j = 1, \ldots, k$.*
Then the following statements are equivalent:

(a) There exists a vector $v \in \mathbb{C}^n$ such that $\theta_1, \ldots, \theta_k$ are the Ritz values computed from the subspace $\mathcal{K}_k(v; A)$.

(b) The linear system $P(\Lambda, \Theta)\hat{v} = 0$ has a positive solution $\hat{v} > 0$.

(c) The linear system $C(\Lambda, \Theta)w = 0$ has a positive solution $w > 0$.

Proof. We first show that (a)⇒(c). Let $AU = UB + ub^\star$ denote the orthogonal Krylov decomposition corresponding to the subspace $\mathcal{K}_k(v; A)$, and let

$$\kappa_B(\xi) = (\xi - \theta_1)(\xi - \theta_2)\dots(\xi - \theta_k)$$

denote the characteristic polynomial of the matrix B. Then $\mathrm{Re}\,(\theta_i)$ and $\mathrm{Im}\,(\theta_i)$ are the critical points for the function

$$(\alpha_i, \beta_i)_{i=1}^k \mapsto \|\pi(A)v\|^2,\ \pi(\xi) = (\xi - (\alpha_1 + i\beta_1))(\xi - (\alpha_2 + i\beta_2))\dots(\xi - (\alpha_k + i\beta_k)).$$

We expand $\|\pi(A)v\|^2$, using the eigenvalue decomposition $A = ULU^\star$ where U is unitary and $L = \mathrm{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$:

$$\begin{aligned}
\|\pi(A)v\|^2 &= \|U\pi(L)U^\star v\|^2 = \|\pi(L)(U^\star v)\|^2 \\
&= \|\mathrm{diag}(\pi(\lambda_1), \pi(\lambda_2), \dots, \pi(\lambda_n)) \cdot \hat{v}\|^2 \\
&= \sum_{i=1}^n |\pi(\lambda_i)|^2 \cdot |\hat{v}_i|^2 \\
&= \sum_{i=1}^n |\lambda_i - (\alpha_1 + i\beta_1)|^2 \cdot |\lambda_i - (\alpha_2 + i\beta_2)|^2 \cdots |\lambda_i - (\alpha_k + i\beta_k)|^2 \cdot |\hat{v}_i|^2 \\
&= \sum_{i=1}^n \prod_{j=1}^k \left((\mathrm{Re}\,(\lambda_i) - \alpha_j)^2 + (\mathrm{Im}\,(\lambda_i) - \beta_j)^2\right) \cdot |\hat{v}_i|^2 \\
&\to \min
\end{aligned}$$

This is a smooth function in variables $\alpha_1, \dots, \alpha_k, \beta_1, \dots, \beta_k$ and the minimum is reached only when all partial derivatives are equal to zero:

$$\begin{aligned}
\partial_{\alpha_j} \|\pi(A)\hat{v}\|^2 &= -2\sum_{i=1}^n \left((\mathrm{Re}\,(\lambda_i) - \alpha_j) \cdot \prod_{q \neq j} |\lambda_i - (\alpha_q + i\beta_q)|^2 \cdot |\hat{v}_i|^2 \right) \\
&= -2\sum_{i=1}^n \frac{\mathrm{Re}\,(\lambda_i) - \alpha_j}{|\lambda_i - (\alpha_j + i\beta_j)|^2} \cdot p_i |\hat{v}_i|^2 = 0 \\
\partial_{\beta_j} \|\pi(A)\hat{v}\|^2 &= -2\sum_{i=1}^n \frac{\mathrm{Im}\,(\lambda_i) - \beta_j}{|\lambda_i - (\alpha_j + i\beta_j)|^2} \cdot p_i |\hat{v}_i|^2 = 0.
\end{aligned}$$

Here we have denoted

$$p_i = \prod_{j=1}^k |\lambda_i - (\alpha_j + i\beta_j)|^2.$$

If the minimum of $\|\pi(A)v\|^2$ is reached for $\theta_j = \alpha_j + i\beta_j$, then the following holds:

$$\begin{aligned}
\sum_{i=1}^n \frac{\mathrm{Re}\,(\lambda_i - \theta_j)}{|\lambda_i - \theta_j|^2} \cdot p_i |\hat{v}_i|^2 &= 0, \\
\sum_{i=1}^n \frac{\mathrm{Im}\,(\lambda_i - \theta_j)}{|\lambda_i - \theta_j|^2} \cdot p_i |\hat{v}_i|^2 &= 0.
\end{aligned}$$

We can write these equations more compactly as

$$\sum_{i=1}^{n} \frac{\lambda_i - \theta_j}{|\lambda_i - \theta_j|^2} \cdot p_i |\hat{v}_i|^2 = 0,$$

or

$$\sum_{i=1}^{n} \frac{1}{\lambda_i - \theta_j} \cdot p_i |\hat{v}_i|^2 = 0, \tag{4.5}$$

when the conjugate is taken. Setting $w_i = p_i|\hat{v}_i|^2 \geq 0$, and $w = (w_1, \ldots, w_n)$, we see that (4.5) is precisely the j-th equation of the linear system $C(\Lambda, \Theta)w = 0$. This completes the proof for (c). Obviously, the statements (b) and (c) are equivalent, since (4.5) demonstrates that $\hat{v} = (\hat{v}_1, \ldots, \hat{v}_n)$ is a positive solution to $P(\Lambda, \Theta)\hat{v} = 0$.

To show the implication (c)⇒(a), suppose that the system $C(\Lambda, \Theta)w = 0$ has a positive solution $w = (|w_1|^2, |w_2|^2, \ldots, |w_n|^2) > 0$ for some complex numbers $w_1, \ldots, w_n$. For $i = 1, \ldots, n$, let

$$P_i = \prod_{j=1}^{k} |\lambda_i - \theta_j|.$$

Define a vector $v = (v_1, v_2, \ldots, v_n)$ with $v_i = w_i/P_i$, and consider a subspace $\mathcal{X} = \mathcal{K}_k(v; A)$. Without loss of generality, we can assume that the matrix A is diagonal, $A = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$; see Proposition 2.5. For $j = 1, \ldots, k$, define the polynomials

$$\begin{aligned} \pi(\xi) &= \prod_{l=1}^{k} (\xi - \theta_l), \\ \pi_j(\xi) &= \prod_{\substack{l=1 \\ l \neq j}}^{k} (\xi - \theta_l) = \frac{\pi(\xi)}{\xi - \theta_j}. \end{aligned}$$

The set consisting of polynomials $\pi_1, \ldots, \pi_k$ forms a basis for $\mathbb{P}_{k-1}$, the space of all polynomials of degree $k-1$ or less. Thus the Krylov subspace

$$\mathcal{X} = \mathrm{span}\{\pi(A)v \ : \ \pi \in \mathbb{P}_{k-1}\}$$

is spanned by the columns of the matrix

$$X = [\pi_1(A)v \ \ \pi_2(A)v \ \ \ldots \ \ \pi_k(A)v].$$

The Ritz values from the subspace $\mathcal{X}$ are the eigenvalues of the generalized eigenvalue problem

$$X^\star A X y = \theta X^\star X y.$$

Let us show that these are precisely $\theta_1, \ldots, \theta_k$. It suffices to prove that the matrix $Z = X^\star(A - \theta I)X$ is singular for $\theta \in \{\theta_1, \ldots, \theta_k\}$. Using the expansion $v = \sum_{l=1}^{n} v_l e_l$, we first

compute the entry $Z_{i,j}$ of this matrix:

$$\begin{aligned} Z_{i,j} &= e_i^\star X^\star (A-\theta I) X e_j = (\pi_i(A)v)^\star (A-\theta I)\pi_j(A)v \\ &= \left(\sum_{l=1}^{n} \overline{\pi_i(\lambda_l)}\cdot \overline{v}_l e_l^\star\right)\left(\sum_{l=1}^{n}(\lambda_l-\theta)\pi_j(\lambda_l)\cdot v_l e_l\right) \\ &= \sum_{l=1}^{n} \overline{\pi_i(\lambda_l)}(\lambda_l-\theta)\pi_j(\lambda_l)\cdot |v_l|^2 \\ &= \sum_{l=1}^{n} \frac{1}{\overline{\lambda_l-\theta_i}}\left(\overline{\pi(\lambda_l)}(\lambda_l-\theta)\pi_j(\lambda_l)\cdot |v_l|^2\right). \end{aligned}$$

Consider the j-th column of this matrix. When $\theta=\theta_j$, then

$$\overline{\pi(\lambda_l)}(\lambda_l-\theta_j)\pi_j(\lambda_l)\cdot|v_l|^2 = \overline{\pi(\lambda_l)}\pi(\lambda_l)\cdot|v_l|^2 = P_l^2\cdot|v_l|^2 = |w_l|^2,$$

and

$$Z_{i,j} = \sum_{l=1}^{n}\frac{1}{\overline{\lambda_l-\theta_i}}|w_l|^2 = 0,$$

since this is exactly the conjugate of the i-th equation of the linear system $C(\Lambda,\Theta)w=0$. Therefore, the j-th column of the matrix $X^\star(A-\theta_j I)X$ is zero, implying that this matrix is singular and that θ_j is a Ritz value from $\mathcal{K}_k(v;A)$. □

This Theorem shows that, in order to check whether some complex values $\theta_1,\ldots,\theta_k$ can simultaneously be Ritz values for some starting vector v of the Krylov subspace $\mathcal{K}_k(v;A)$, one needs to verify if the linear system $C(\Lambda,\Theta)w=0$ has a positive solution. This is a well-known *feasibility problem* found in linear programming (LP). Unfortunately, this problem is equally as difficult as the minimization problem of the LP (for general matrices – it is not known to us whether specialized feasibility results for Cauchy matrices exist). However, some results from the theory of the linear programming prove to be useful.

Theorem 4.13 (a special case of the Farkas Lemma, see e.g. [57])**.**
Let $M\in\mathbb{R}^{n\times k}$. Then the following statements hold:

(a) Either $Mv=0$ for some $v\gg 0$, or $u^\star M>0$ for some $u\in\mathbb{R}^k$.

(b) Either $Mv=0$ for some $v>0$, or $u^\star M\gg 0$ for some $u\in\mathbb{R}^k$.

To apply this result in our setting, we first convert the system to fit the real arithmetic of the Farkas Lemma. If the linear system $C(\Lambda,\Theta)w=0$ has a solution $w>0$, then the same vector is a solution to the system

$$\begin{bmatrix} \mathrm{Re}\,(C(\Lambda,\Theta)) \\ \mathrm{Im}\,(C(\Lambda,\Theta)) \end{bmatrix} w = 0.$$

The Farkas Lemma states that then for all $\tilde{u}_j, \hat{u}_j \in \mathbb{R}$, $j = 1, \ldots, k$ there must exist some index $i \in \{1, \ldots, n\}$ for which

$$\sum_{j=1}^{k} \mathrm{Re}\left(\frac{1}{\lambda_i - \theta_j}\right) \cdot \tilde{u}_j + \sum_{j=1}^{k} \mathrm{Im}\left(\frac{1}{\lambda_i - \theta_j}\right) \cdot \hat{u}_j \leq 0. \tag{4.6}$$

Returning to the complex arithmetic, we set $u_j = \tilde{u}_j - i\hat{u}_j$, and the equation (4.6) is equivalent to

$$\mathrm{Re}\left(\sum_{j=1}^{k} \frac{u_j}{\lambda_i - \theta_j}\right) \leq 0.$$

We summarize this result in the following Proposition.

Proposition 4.14. *Under the assumptions of Theorem* 4.12*, the statements (a), (b) and (c) are equivalent to:*

(d) For all $u \in \mathbb{C}^k$ there exists an eigenvalue λ_i of A such that

$$\mathrm{Re}\left(\sum_{j=1}^{k} \frac{u_j}{\lambda_i - \theta_j}\right) \leq 0. \tag{4.7}$$

We can now use the developed theory to explore mutual positions of the Ritz values in the complex plane.

Example 4.15. Suppose that a normal matrix A and its eigenvalues $\Lambda = (\lambda_1, \ldots, \lambda_n)$ are given, as well as some $(k-1)$-tuple of complex numbers $(\theta_1, \ldots, \theta_{k-1})$. We would like to explore all possible locations for θ_k in the complex plane so that the k-tuple $\Theta = (\theta_1, \ldots, \theta_k)$ is a valid set of Ritz values for the matrix A, using some Krylov subspace $\mathcal{K}_k(v; A)$. Consider the following algorithm:

```
Select λ_1, ..., λ_n and θ_1, ..., θ_{k-1};
for each point θ_k of the complex plane do
    Form the matrix C(Λ, Θ);
    Use a linear programming feasibility solver to check whether
        C(Λ, Θ)w = 0 has a solution w > 0;
    if w > 0 exists then
        Paint the point θ_k in grey;
    end
end
```

We have implemented this algorithm in MATLAB (using the MPT solver [38] for determining feasibility of the linear programming problem) and run it for various matrices A. (Note that, when using general–purpose algorithms, this sort of experimentation is

limited to small examples only due to the numerical instabilities in computations involving Cauchy matrices. Specialized, highly accurate algorithms for Cauchy matrices do exist.)

In our first example, we set the eigenvalues of A to the vertices of regular 12–gon inscribed in the unit circle. Also, we fix $\theta_1 = 0.7+0.4i$, $\theta_2 = -0.8-0.2i$. In Figure 4.3(a), the eigenvalues are marked as the red dots, and the convex hull closing the spectrum is bounded by the red line. Fixed Ritz values are marked as blue dots. The gray area in the plot represents the set of all possible values for θ_3 such that $(\theta_1, \theta_2, \theta_3)$ can appear as the Ritz values from some 3-dimensional Krylov subspace. Note that the fixed θ's near the boundary of the convex hull disallow the value of θ_3 of doing the same. This effect is more notable for θ_2 which is close to the side of the polygon than for θ_1, which is close to the eigenvalue.

In the second example, shown in Figure 4.3(b), we set $\Lambda = (-5, -3+2i, -3-2i, 4+i, 4-i, 6)$ and a fixed Ritz value $\theta_1 = -4$. Locations of all possible θ_2's computed from a 2-dimensional Krylov subspace are shown in gray. Note the intriguing geometry of the boundary of this area.

A simpler case is the one in Figure 4.3(c) in which the eigenvalues are colinear. This means that the matrix A is a Hermitian times a unit complex scalar $e^{i\varphi}$ – the real eigenvalues of the Hermitian matrix are rotated by the angle φ. We have set $\Lambda = (0.1+0.1i, 0.2+0.2i, 0.5+0.5i, 0.7+0.7i, 0.9+0.9i, 1.2+1.2i)$; the fixed Ritz values are now $\theta_1 = 0.3+0.3i$ and $\theta_2 = 0.8+0.8i$. We plot the permissible locations for the third Ritz value. It appears as though two Ritz values may not appear between consecutive eigenvalues of A – we show that this is indeed the case in Theorem 4.16.

Finally, we explore the effect of adding a single non-real eigenvalue to a Hermitian matrix. We set $\Lambda = (-2, -1, 1, 3, 6, 8, 12, 0.3+0.6i)$ and $\theta_1 = 2.2+0.1i$. A rather complicated geometry arises once again in the Figure 4.3(d).

Theoretical support for the effects observed by the examples is still the subject of our research. However, we are able to prove some well-known results using a very different technique. We first give a simple proof for a result similar to the Corollary 10.6.2 of [52], explaining the Figure 4.3(c).

Theorem 4.16. *Suppose A is a Hermitian matrix. Let $\theta_1 \geq \ldots \geq \theta_k$ denote the Ritz values from a k-dimensional Krylov subspace; suppose that no θ_j is an eigenvalue of A. Then for each $j = 1, \ldots, k-1$ there exists an eigenvalue λ of A such that*

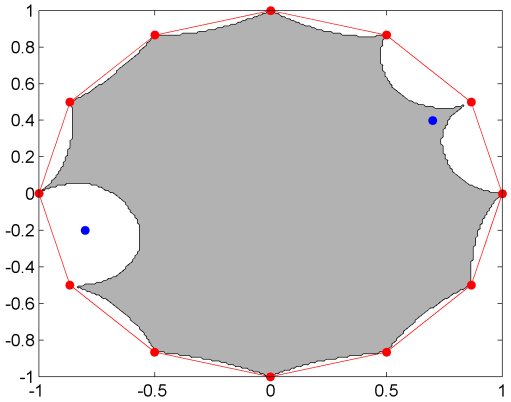
$$\theta_{j+1} < \lambda < \theta_j.$$

Proof. Assume the contrary: there are indices j and p such that

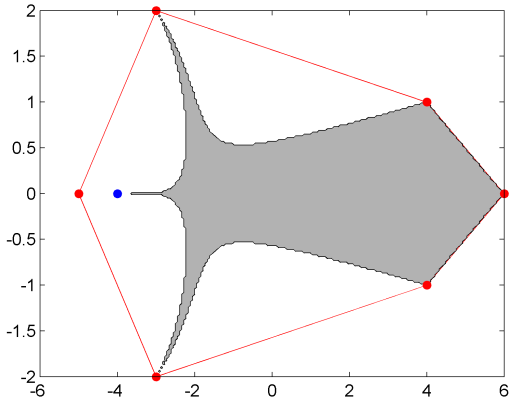
$$\lambda_n \leq \ldots \leq \lambda_{p+1} < \theta_{j+1} < \theta_j < \lambda_p \leq \ldots \leq \lambda_1,$$

where $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of A. Thus

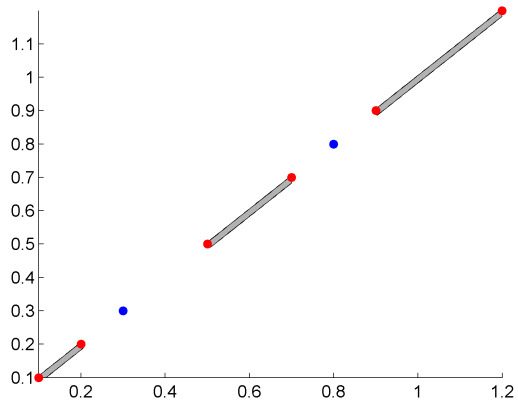
$$\begin{aligned} &0 < \lambda_i - \theta_j < \lambda_i - \theta_{j+1}, && \text{for all } i = 1, \ldots, p; \\ &0 < \theta_{j+1} - \lambda_i < \theta_j - \lambda_i, && \text{for all } i = p+1, \ldots, n, \end{aligned}$$



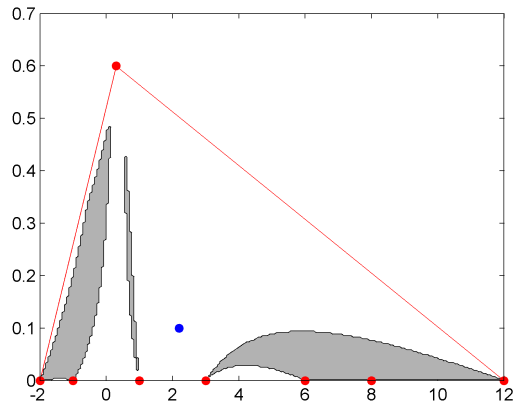
(a) A unitary matrix, two Ritz values fixed



(b) A 6×6 matrix, one Ritz value fixed



(c) Matrix of the form $e^{i\varphi}A$, where A is Hermitian



(d) The effect of a single complex eigenvalue added to spectrum of a Hermitian matrix

Figure 4.3: Permissible Ritz locations. The red dots are the eigenvalues of A, with the convex hull of the spectrum bounded by the red line. Blue dots mark the fixed Ritz values $\theta_1, \ldots, \theta_{k-1}$. The gray area shows the permissible positions for another Ritz value θ_k.

or equivalently

$$0 < \frac{1}{\lambda_i - \theta_{j+1}} < \frac{1}{\lambda_i - \theta_j}, \qquad \text{for all } i = 1, \ldots, p;$$
$$\frac{1}{\lambda_i - \theta_{j+1}} < \frac{1}{\lambda_i - \theta_j} < 0, \qquad \text{for all } i = p+1, \ldots, n,$$

This implies

$$\sum_{i=1}^{n} \frac{w_i}{\lambda_i - \theta_{j+1}} < \sum_{i=1}^{n} \frac{w_i}{\lambda_i - \theta_j},$$

for all $w = (w_1, \ldots, w_n) > 0$. But both sides of the inequality have to be equal to zero when we insert the solution of the system $C(\Lambda, \Theta)w = 0$: the left hand side is the $(j+1)$-the equation of the system, and the right hand side is the j-th equation. We have reached a contradiction. □

The Cauchy interlacing property for Hermitian matrices easily follows from this Theorem. First, there has to be an eigenvalue that is larger than any of the Ritz values: if all $\frac{1}{\lambda_i - \theta_1}$ were negative, then the property (d) of Proposition 4.14 would fail for $u = (-1, 0, \ldots, 0)$. Thus, $\lambda_1 > \theta_1$. The Theorem then provides that $\theta_1 > \lambda_i > \theta_2$, for some i. Thus $\lambda_2 > \theta_2$. Inductively, we get $\lambda_i > \theta_i$, for all i, and this is precisely the Cauchy interlacing lemma.

Proving that the Ritz values fall inside the convex hull spanned by the eigenvalues of A is very simple using any of the equivalent properties. For example, we can use the dual formulation (d): suppose that a Ritz value θ is not inside the convex hull. Then there exists a line π through θ such that all λ_i are on the same side of the line. Let z be a complex number such that $u = z - \theta$ is perpendicular to that line and let z be on the same side of the line as the eigenvalues are. Then for all $i = 1, \ldots, n$ we have

$$\mathrm{Re}\left(\frac{u}{\lambda_i - \theta}\right) = \mathrm{Re}\left(\frac{z - \theta}{\lambda_i - \theta}\right) = \frac{|z - \theta|}{|\lambda_i - \theta|} \cdot \cos \measuredangle(\lambda_i \theta z) > 0,$$

which contradicts (d). Here $\measuredangle(\lambda_i \theta z)$ is the angle with vertex in θ and rays passing through λ_i and θ. This angle is acute since λ_i and z are on the same side of the line π.

The statement Theorem 4.12(b) provides even more: it gives a representation of a Ritz value as a convex combination of eigenvalues of A. To see that, rewrite the j-th row of the system $P(\Lambda, \Theta)\hat{v} = 0$:

$$\sum_{i=1}^{n} (\lambda_i - \theta_j) P_{i,j} = 0, \quad P_{i,j} = \prod_{\substack{l=1 \\ l \neq j}}^{k} |\lambda_i - \theta_l|^2 \cdot |\hat{v}_i|^2.$$

Then

$$\theta_j = \sum_{i=1}^{n} \lambda_i \cdot \frac{P_{i,j}}{P_{1,j} + P_{2,j} + \ldots + P_{n,j}}$$

is obviously a convex combination of the eigenvalues.

Using the experience accumulated in generating images we construct an example that shows how the restarted Arnoldi method may fail even for the case of the normal matrices.

Example 4.17. As we mentioned in Section 2.5.3, Embree [28] has constructed a matrix $A \in \mathbb{C}^{4\times 4}$ and the starting vector v for the IRAM with the following property: if λ_1 is the largest eigenvalue of A by module, then the Ritz values of $\mathcal{K}_2(v; A)$ are θ_1 and $\theta_2 = \lambda_1$ such that $|\theta_1| > |\theta_2|$. Restarting the Arnoldi method to dimension 1 will then use λ_1 as a shift, removing any component of the associated eigenvector from the restarted Krylov subspace. This results in the inability of the IRAM to compute any approximation to λ_1.

When the matrix A is normal, it is not possible to construct an analogous example. The largest eigenvalue λ_1 of A is placed on boundary of the convex hull, and all of the Ritz values are inside the convex hull. Therefore, should a Ritz value equal λ_1, there could not exist another Ritz value with a greater absolute value.

On the other hand, we can construct an example in which the second largest eigenvalue of A is purged. Consider the matrix

$$A = \begin{bmatrix} 4 & & & & \\ & 3 & 2 & & \\ & -2 & 3 & & \\ & & & 3.9 & \\ & & & & 8 \end{bmatrix}.$$

Its eigenvalues are $\Lambda = (8, 4, 3.9, 3 + 2i, 3 - 2i)$. The two largest eigenvalues (by both the real part and by the module) are $\lambda_1 = 8$ and $\lambda_2 = 4$.

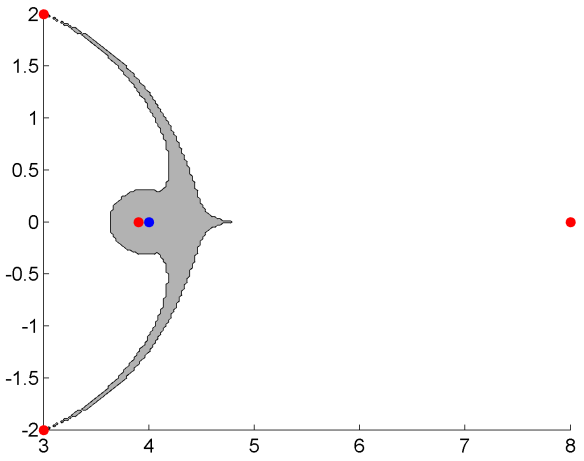
We want to construct a starting vector v such that the smallest Ritz value θ_3 computed using $\mathcal{K}_3(v; A)$ is equal or very close to λ_2. Using the algorithm described in the previous Example, we can determine the location of the additional Ritz values if one is set to $\lambda_2 + \epsilon$, for some small ϵ (but not too small due to the bad conditioning of the linear system). Figure 4.4(a) shows the image when $\epsilon = 10^{-6}$. The image suggests that is should be possible to construct a Krylov subspace with the real starting vector so that there are two mutually conjugated Ritz values larger than λ_2. Set

$$v = \begin{bmatrix} -0.775693250142234 \\ 0.028238213050217 \\ 0.028273977339263 \\ 0.629795237727870 \\ -0.007818295736434 \end{bmatrix}.$$

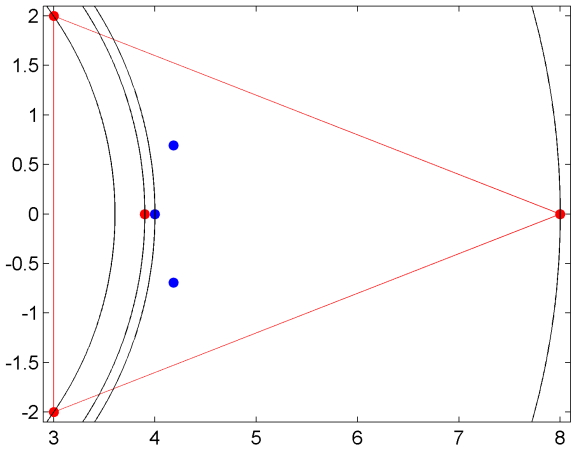
Then the Ritz values from $\mathcal{K}_3(v; A)$ are (see Figure 4.4(b))

$$\begin{aligned} \theta_1 &= 4.183227620474041 + 0.692098306609705i, \\ \theta_2 &= 4.183227620474041 - 0.692098306609705i, \\ \theta_3 &= 4.000000000000762. \end{aligned}$$

(We have used optimization routines to push θ_3 as close as possible to λ_2 and to keep θ_1 and θ_2 as large as possible.) None of the Ritz values of $\mathcal{K}_j(v; A)$, $j = 1, 2, 3$ have the



(a) Permissible Ritz values when $\theta = \lambda_2 + \epsilon$, for $\epsilon = 10^{-6}$. (In the figure, the eigenvalue λ_2 is overlapped by the blue dot at θ.)



(b) Eigenvalues of A and the Ritz values computed from $\mathcal{K}_3(v; A)$. Two of the Ritz values are larger by both module and the real part than $\theta_3 \approx \lambda_2 + 10^{-12}$.

Figure 4.4: A very close approximation to the second largest eigenvalue is used as a shift for the restarted Arnoldi method in Example 4.17.

residual small enough to be deflated. Therefore, the restart of the IRAM to the subspace of dimension 2 will declare θ_3 unwanted and remove a very close approximation to the eigenvalue λ_2 from the restarted subspace. This will leave only a tiny component in the direction of the associated eigenvector, almost indistinguishable to the roundoff error, delaying or even preventing the convergence.

To conclude this Chapter, we remind that our techniques for locating the Ritz values apply for Krylov subspaces only. In the case of the extraction from general subspaces,

some substantially more complex results are known in the literature. We state a theorem from [43] that clearly demonstrates such complexity.

Definition 4.18. *For a vector $a = (\alpha_1, \ldots, \alpha_n)$ and for $j \in \{1, \ldots, n\}$, let $C_j(a)$ denote a vector whose entries are all possible products of some j entries in the vector a:*

$$C_j(a) := (\alpha_{i_1} \cdot \alpha_{i_2} \cdot \ldots \cdot \alpha_{i_j})_{1 \leq i_1 < i_2 < \ldots < i_j \leq n} \in \mathbb{C}^{\binom{n}{j}}.$$

Definition 4.19. *We say that the matrix $S \in \mathbb{C}^{n \times n}$ is unitary stochastic if there exists a unitary matrix $U \in \mathbb{C}^{n \times n}$ such that $S_{ij} = |U_{ij}|^2$ for all $i, j = 1, \ldots, n$.*
For vectors $x \in \mathbb{C}^k$ and $y \in \mathbb{C}^n$, $k \leq n$, we write $x \prec_{uds} y$ if there exist $x_{k+1}, \ldots, x_n \in \mathbb{C}$ and a unitary stochastic matrix S such that $\tilde{x} = Sy$. Here $\tilde{x} = (x, x_{k+1}, \ldots, x_n)$.

Theorem 4.20. *[43] Suppose that $\mathcal{X}$ is a k-dimensional subspace and that $\theta_1, \ldots \theta_k$ are the Ritz values from $\mathcal{X}$ for a normal matrix $A \in \mathbb{C}^{n \times n}$. Let $\Lambda = (\lambda_1, \ldots, \lambda_n)$ denote the vector of all eigenvalues of A, and let $\Theta = (\theta_1, \ldots, \theta_k)$ denote the vector of all Ritz values. Then for all $\alpha \in \mathbb{C}$, and for all $j = 1, \ldots, k$ it holds that*

$$C_j(\Theta - \alpha) \prec_{uds} C_j(\Lambda - \alpha). \tag{4.8}$$

Here $\Theta - \alpha = (\theta_1 - \alpha, \ldots, \theta_k - \alpha)$, and $\Lambda - \alpha = (\lambda_1 - \alpha, \ldots, \lambda_n - \alpha)$.

Conversely, let $\Theta = (\theta_1, \ldots, \theta_{n-1})$ be any complex vector. If (4.8) holds for $j = n - 1$ and for all $\alpha = \lambda_1, \ldots, \lambda_n$, then there exists a $(n-1)$-dimensional subspace $\mathcal{X}$ such that $\theta_1, \ldots, \theta_{n-1}$ are the Ritz values for the matrix A from $\mathcal{X}$.

Unlike the Theorem 4.12, it is very difficult to use this result for verification that a set of k complex numbers appears as the set of Ritz values for a given normal matrix A. Also, Theorem 4.20 gives both the necessary and the sufficient conditions only in the case when $k = n - 1$.

Chapter 5

Reduction to the m-Hessenberg form

As we have seen in Chapter 2, Hessenberg matrices play a special role in construction and implementation of Krylov subspace based algorithms. In particular, for a given matrix A and an initial vector v, after k steps of the Arnoldi algorithm we have the decomposition

$$AV_k = V_k H_k + r_k e_k^\star, \tag{5.1}$$

where the columns of V_k make an orthonormal basis for the Krylov subspace $\mathcal{K}_k(v; A)$, and H_k is an upper Hessenberg matrix. We use the eigenvalues of H_k as approximations for certain eigenvalues of A. As the algorithm progresses, several scenarios may occur in which the Arnoldi decomposition (5.1) is compromised. When an eigenvalue of H_k converges, we lock it or deflate the corresponding eigenvector from the Krylov subspace, during which the matrix H_k may depart from its Hessenberg form. Also, some restarting techniques, such as Krylov–Schur, can perturb the Hessenberg form as well. In order for the algorithm to continue, one has to restore the Arnoldi decomposition, which usually reduces to restoring the Hessenberg form of the projected matrix H_k.

In this chapter we develop a blocked algorithm for reduction of a general, full matrix to a special form we call the *m-Hessenberg form* (also known as the *banded Hessenberg form*):

Definition 5.1. *We say that $H \in \mathbb{C}^{n\times n}$ is in the m-Hessenberg form if $H_{ij} = 0$ for all indices i, j such that $i > j + m$ and $j \in \{1, 2, \ldots, n - m\}$.*

Reducing the matrix $A \in \mathbb{C}^{n\times n}$ to the m-Hessenberg form is an operation of finding a unitary (orthogonal for real matrices A) matrix Q such that $Q^\star AQ$ is m-Hessenberg. These matrices generalize the notion of (1-)Hessenberg matrices; some properties are easily translated into the new setting. An m-Hessenberg matrix has a natural block partition

that reveals a block–Hessenberg structure:

$$H = \begin{bmatrix} H_{[1,1]} & H_{[1,2]} & H_{[1,3]} & H_{[1,4]} & \dots & H_{[1,k-1]} & H_{[1,k]} \\ H_{[2,1]} & H_{[2,2]} & H_{[2,3]} & H_{[2,4]} & \dots & H_{[2,k-1]} & H_{[2,k]} \\ 0 & H_{[3,2]} & H_{[3,3]} & H_{[3,4]} & \dots & H_{[3,k-1]} & H_{[3,k]} \\ 0 & 0 & H_{[4,3]} & H_{[4,4]} & \dots & H_{[4,k-1]} & H_{[4,k]} \\ 0 & 0 & 0 & H_{[5,4]} & \dots & H_{[5,k-1]} & H_{[5,k]} \\ & & & & \vdots & & \\ 0 & 0 & 0 & 0 & \dots & H_{[k,k-1]} & H_{[k,k]} \end{bmatrix}, \tag{5.2}$$

where $k = \lceil n/m \rceil$. The blocks $H_{[i,j]}$ are all $m \times m$ matrices, except eventually for $i = k$ and $j = k$. All subdiagonal blocks $H_{[i,i+1]}$ are upper triangular.

Definition 5.2. *An m-Hessenberg matrix H is unreduced if*

$$\prod_{i=1}^{k-1} \det(H_{[i+1,i]}) \equiv \prod_{i=1}^{n-m} H_{i+m,i} \neq 0.$$

As it is well known, the Implicit-Q theorem (Theorem 2.2) naturally extends to m-Hessenberg matrices. We state it here as follows:

Theorem 5.3. *Let $H = Q^\star AQ$ and $\tilde{H} = \tilde{Q}^\star A\tilde{Q}$ be two m-Hessenberg reductions of the matrix A, where at least one of the matrices H, $\tilde{H}$ is unreduced. Suppose that the columns of Q are split into blocks, accordingly as in* (5.2)*:*

$$Q = [Q_{[1]} \;\; Q_{[2]} \;\; \dots \;\; Q_{[k]}],$$

and similarly for $\tilde{Q}$. Suppose that $\tilde{Q}_{[1]} = Q_{[1]}\Phi_1$ for some unitary matrix Φ_1. Then there exist unitary matrices Φ_i such that $\tilde{Q}_{[i]} = Q_{[i]}\Phi_i$, $i = 2, \dots, k$, that is, $\tilde{Q} = Q\Phi$, where $\Phi = \oplus_{i=1}^{k}\Phi_i$. In particular, $\tilde{H} = \Phi^\star H\Phi$. If Φ_1 is diagonal unitary, then all Φ_i are diagonal unitary matrices. If Φ_1 is the identity, the same holds for all Φ_i.

The m-Hessenberg matrices occur in blocked Arnoldi algorithms, as we will see in the next section. Another important application comes from the control theory, which is discussed in Section 5.2, along with the default implementation of the reduction as found in the control-theory software library SLICOT [64].

We develop a blocked algorithm for the reduction in Section 5.3. This algorithm, while based on the LAPACK implementation of the standard Hessenberg reduction [2, see routine `DGEHRD`], introduces several speedup techniques enabled by a larger band. Section 5.4 suggests on how to implement the hybrid algorithm that uses both the central processing unit (CPU) and the graphics processing unit (GPU). Modern video cards have an incredible computing power and a specially crafted algorithm that uses the best properties of both CPU and GPU may benefit dramatically when compared to the one using only the CPU. Numerical experiments for both the CPU-only and the hybrid algorithm are presented in the last section of this chapter. We compare the performances of newly developed algorithms to the existing non-blocked implementation from SLICOT and the 1-Hessenberg reduction routine `DGEHRD` from LAPACK .

5.1 Block Arnoldi algorithm and the m-Hessenberg form

Extracting eigenvalues using the decomposition

$$AV_k = V_k H_k + r_k e_k^\star \tag{5.3}$$

has several issues which sometimes have great influence on the performance of the Arnoldi algorithm.

We have already noted that the algorithm cannot detect the multiplicities of the eigenvalues of A. During the expansion phase of the decomposition, the (unreduced) matrix H_k can only have simple eigenvalues. Only after deflation or locking, another copy of multiple eigenvalue may appear and that is only due to rounding errors. To be more precise, suppose that $\mathcal{X}$ is an eigenspace of dimension $d > 1$ corresponding to some multiple eigenvalue λ, and that $\mathcal{Y}$ is a complementary invariant subspace. The starting vector v can be written as $v = v_\mathcal{X} + v_\mathcal{Y}$ where $v_\mathcal{X} \in \mathcal{X}$ and $v_\mathcal{Y} \in \mathcal{Y}$. After a number of Arnoldi steps, we may declare an approximate eigenvector for λ from $\mathcal{V}_k = \operatorname{Im} V_k$ as converged. Since $A^j v_\mathcal{X} = \lambda^j v_\mathcal{X}$ and $A^j v_\mathcal{Y} \in \mathcal{Y}$, all vectors from $\mathcal{V}_k$ are linear combinations of $v_\mathcal{X}$ and some vector from $\mathcal{Y}$. Thus the approximate eigenvector satisfies $\tilde{v}_\mathcal{X} \approx v_\mathcal{X}$ and no other eigenvector for λ can be revealed. However, the usual approach in practice is to store the converged eigenvector and then remove it out of the basis for Krylov subspace: the effect is the same as if the initial vector was orthogonalized against the converged eigenvector. Should the equality $\tilde{v}_\mathcal{X} = v_\mathcal{X}$ hold, the component of $v_\mathcal{X}$ would be purged from the initial vector. Thus the following Krylov subspaces would be subspaces of $\mathcal{Y}$ and the multiplicity of λ not recognized at all. Deflation of $\tilde{v}_\mathcal{X}$ purges out most of $v_\mathcal{X}$ and only a tiny component from $\mathcal{X}$ still remains in $\mathcal{V}_k$. Subsequent steps of Arnoldi algorithm would very slowly build up this component, and eventually a new instance of λ might be found as the eigenvalue. Thus, although the Arnoldi algorithm can find multiple eigenvalues in practice, it does so very slowly.

Another issue is the optimal software implementation of the atomic operation $x \mapsto Ax$, performed at each step. This is a matrix-times-vector (BLAS 2) operation; the Arnoldi algorithm cannot take advantage of matrix-times-matrix (BLAS 3) operations which are much better fit for the memory hierarchy of the modern computer architectures. The operation $x \mapsto Ax$ is usually a custom made function that exploits sparsity of A, and the performance is improved if the computation is done for several vectors x at once, instead of just one by one.

To resolve these issues, a block Arnoldi algorithm is introduced. Let $A \in \mathbb{C}^{n\times n}$. Instead of only one, several starting vectors $v_1, v_2, \ldots, v_b \in \mathbb{C}^n$ are chosen and put into the orthonormal matrix $V_1 = V_1^{[b]}$. At step k, we have the following decomposition

$$AV_k^{[b]} = V_k^{[b]} H_k^{[b]} + R_k^{[b]} (E_k^{[b]})^\star, \tag{5.4}$$

which we call *the block Arnoldi decomposition.* It has the following properties: the matrix $V_k^{[b]}$ is an $n \times kb$ orthonormal matrix such that its columns span a *block Krylov subspace*

$$\mathcal{K}_k(V_1^{[b]}; A) = \operatorname{span}\{V_1^{[b]}, AV_1^{[b]}, A^2V_1^{[b]}, \ldots, A^{k-1}V_1^{[b]}\}.$$

$H_k^{[b]}$ is a $kb \times kb$ matrix in a b-Hessenberg form. The matrix $R_k^{[b]}$ is an $n \times b$ residual matrix satisfying $(V_k^{[b]})^\star R_k^{[b]} = 0$, and $E_k^{[b]}$ consists of the last b columns of the $kb \times kb$ identity matrix. To proceed to the next step, we first do a QR factorization $R_k^{[b]} = V_{k+1} H_{[k+1,k]}$. The basis matrix is appended with b columns:

$$V_{k+1}^{[b]} = [V_k^{[b]} \;\; V_{k+1}],$$

and the product AV_{k+1} is computed (a BLAS 3 operation!). Each column of AV_{k+1} is orthogonalized against $V_{k+1}^{[b]}$:

$$AV_{k+1} = V_1 H_{[1,k+1]} + \ldots + V_{k+1} H_{[k+1,k+1]} + R_{k+1}^{[b]} = V_{k+1}^{[b]} H_{[1:k+1,k+1]} + R_{k+1}^{[b]},$$

and the coefficients $H_{[1:k+1,k+1]}$ are appended to the matrix $H_k^{[b]}$. This gives the block Arnoldi decomposition at step $(k+1)$:

$$AV_{k+1}^{[b]} = V_{k+1}^{[b]} \underbrace{\begin{bmatrix} H_{[1,1]} & H_{[1,2]} & H_{[1,3]} & H_{[1,4]} & \ldots & H_{[1,k]} & H_{[1,k+1]} \\ H_{[2,1]} & H_{[2,2]} & H_{[2,3]} & H_{[2,4]} & \ldots & H_{[2,k]} & H_{[2,k+1]} \\ 0 & H_{[3,2]} & H_{[3,3]} & H_{[3,4]} & \ldots & H_{[3,k]} & H_{[3,k+1]} \\ 0 & 0 & H_{[4,3]} & H_{[4,4]} & \ldots & H_{[4,k]} & H_{[4,k+1]} \\ 0 & 0 & 0 & H_{[5,4]} & \ldots & H_{[5,k]} & H_{[5,k+1]} \\ & & & & \vdots & & \\ 0 & 0 & 0 & 0 & \ldots & H_{[k+1,k]} & H_{[k+1,k+1]} \end{bmatrix}}_{H_{k+1}^{[b]}} + R_{k+1}^{[b]} (E_{k+1}^{[b]})^\star.$$

Note that all $H_{[j+1,j]}$ are upper triangular and thus $H_{k+1}^{[b]}$ is b-Hessenberg.

Since each starting vector contributes its own component from a given eigenspace $\mathcal{X}$, similar analysis as above shows that the block Arnoldi algorithm can compute multiplicities of eigenvalues up to b without the need for deflation. That is, up to b mutually orthogonal vectors from $V_k^{[b]}$ can approximate eigenvectors of the same multiple eigenvalue λ.

Of course, since

$$\mathcal{K}_k(V_1^{[b]}; A) = \operatorname{span}\{\pi_1(A)v_1 + \ldots + \pi_b(A)v_b \;:\; \deg(\pi_j) \leq k-1\},$$

we expect that the convergence rate is slower when compared to a single vector Krylov subspace of the same dimension kb:

$$\mathcal{K}_{kb}(v; A) = \operatorname{span}\{\pi(A)v \;:\; \deg(\pi) \leq kb-1\}.$$

Several methods for restarting the block Arnoldi algorithm exist, see e.g. [3, 46]; some of them require recovery of b-Hessenberg form for the projected matrix $H_k^{[b]}$. To demonstrate the difference, we briefly discuss the implicit restarting method of Lehoucq and Maschhoff [40], and the Krylov-Schur approach as presented in [46].

Suppose we want to restart (5.4) and reduce its dimension by pb. In the implicitly restarted method, we cleverly choose p numbers $\sigma_1, \sigma_2, \ldots, \sigma_p$ as shifts. For the shift σ_1,

we proceed as in the single vector Arnoldi method:

$$\begin{aligned} H_k^{[b]} - \sigma_1 I &= Q_1 R; \\ \check{H}_k^{[b]} &= Q_1^\star H_k^{[b]} Q_1; \\ A(V_k^{[b]} Q_1) &= (V_k^{[b]} Q_1)\check{H}_k^{[b]} + R_k^{[b]} (E_k^{[b]})^\star Q_1. \end{aligned}$$

Since $H_k^{[b]}$ is b-Hessenberg, so are the unitary matrix Q_1 and the shifted projected matrix $\check{H}_k^{[b]}$. Non-zero entries may appear in up to $2b$ last columns of $(E_k^{[b]})^\star Q_1$. Repeating the procedure for each of $\sigma_2, \ldots, \sigma_p$, we arrive at

$$A(V_k^{[b]} Q_p) = (V_k^{[b]} Q_p)\tilde{H}_k^{[b]} + R_k^{[b]} (E_k^{[b]})^\star Q_p,$$

where Q_p is the product of p b-Hessenberg matrices, and thus a bp-Hessenberg matrix itself. Non-zero entries may appear in up to pb last columns of $(E_k^{[b]})^\star Q_p$. Thus $(E_k^{[b]})^\star Q_p = [\tilde{Q}_1 (E_{k-p}^{[b]})^\star \;\; \tilde{Q}_p]$, with $\tilde{Q}_1$ being a $b \times b$ matrix and $\tilde{Q}_p$ being a $b \times pb$ matrix.

We split $V_k^{[b]} Q_p$ into blocks of $(k-p)b$ and pb columns as well:

$$A[\tilde{V}_{k-p}^{[b]} \;\; \hat{V}_p^{[b]}] = [\tilde{V}_{k-p}^{[b]} \;\; \hat{V}_p^{[b]}] \begin{bmatrix} \tilde{H}_{k-p}^{[b]} & \hat{H}_{k-p,p}^{[b]} \\ \tilde{H}_{k-p+1,k-p} (E_{k-p}^{[b]})^\star & \hat{H}_{p,p}^{[b]} \end{bmatrix} + R_k^{[b]} [\tilde{Q}_1 (E_{k-p}^{[b]})^\star \;\; \tilde{Q}_p],$$

and then rewrite only the first $(k-p)b$ columns of this equality:

$$A\tilde{V}_{k-p}^{[b]} = \tilde{V}_{k-p}^{[b]} \tilde{H}_{k-p}^{[b]} + (\hat{V}_p^{[b]} \tilde{H}_{k-p+1,k-p} + R_k^{[b]} \tilde{Q}_1)(E_{k-p}^{[b]})^\star.$$

This is once again a block Arnoldi decomposition associated with block Krylov subspace $\mathcal{K}_{k-p}(\pi(A)V_1^{[b]}; A)$, where $\pi(\xi) = (\xi - \sigma_1) \cdot \ldots \cdot (\xi - \sigma_p)$.

Before the restart, the matrix $H_k^{[b]}$ has kb eigenvalues – the Ritz values. If the exact shift strategy is used, we have to mark only p of these eigenvalues as unwanted and use them as shifts; the other $kb - p$ eigenvalues are marked as wanted. After the restart, the dimension is reduced by pb and $\tilde{H}_{k-p}^{[b]}$ has $(k-p)b$ eigenvalues. Unlike the ordinary non-blocked restarting where Ritz values after the restart are exactly the wanted Ritz values before the restart, it is unclear what the eigenvalues of $\tilde{H}_{k-p}^{[b]}$ are. On the other hand, it is easy to show that the restarted Krylov subspace is spanned by Ritz vectors associated with wanted Ritz values. Also, the b-Hessenberg form of the projected matrix is kept and there is not need to recover it at any stage of the restart.

If we would like to keep the wanted Ritz values after the restart, we may resort to a Krylov-Schur type of restarting. Let

$$H_k^{[b]} = [Q_1 \;\; Q_2] \begin{bmatrix} S_{11} & S_{12} \\ & S_{22} \end{bmatrix} [Q_1 \;\; Q_2]^\star$$

be the Schur factorization such that $(k-p)b$ wanted Ritz values appear as diagonal elements of S_{11} and the pb unwanted Ritz values as diagonal elements of S_{22}. Postmultiply

(5.4) by $Q = [Q_1 \;\; Q_2]$:

$$\begin{aligned} A[V_k^{[b]}Q_1 \;\; V_k^{[b]}Q_2] &= [V_k^{[b]}Q_1 \;\; V_k^{[b]}Q_2] \begin{bmatrix} S_{11} & S_{12} \\ & S_{22} \end{bmatrix} + R_k^{[b]}(E_k^{[b]})^\star Q \\ &= [V_k^{[b]}Q_1 \;\; V_k^{[b]}Q_2 \;\; R_k^{[b]}] \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \\ (E_k^{[b]})^\star Q_1 & (E_k^{[b]})^\star Q_2 \end{bmatrix}. \end{aligned}$$

Removing the last pb columns leads to

$$A(V_k^{[b]}Q_1) = [V_k^{[b]}Q_1 \;\; R_k^{[b]}] \begin{bmatrix} S_{11} \\ (E_k^{[b]})^\star Q_1 \end{bmatrix}; \tag{5.5}$$

we have to restore zeros in first $(k-p-1)b$ columns of $(E_k^{[b]})^\star Q_1$ in order to have the Arnoldi decomposition once again. Since $(E_k^{[b]})^\star Q_1$ is a $b \times (k-p)$ matrix, we can use a product $\hat{Q}$ of b row-wise Householder reflectors to introduce the desired zeros:

$$((E_k^{[b]})^\star Q_1)\hat{Q} = RE_{k-p}^{[b]},$$

for some $b \times b$ upper triangular matrix R. Postmultiply (5.5) by $\hat{Q}$:

$$A(V_k^{[b]}Q_1\hat{Q}) = [V_k^{[b]}Q_1\hat{Q} \;\; R_k^{[b]}] \begin{bmatrix} \hat{Q}^\star S_{11}\hat{Q} \\ RE_{k-p}^{[b]} \end{bmatrix}, \tag{5.6}$$

but now $\hat{Q}^\star S_{11}\hat{Q}$ is not b-Hessenberg any more. Algorithm that this chapter is devoted to can now be used in order to find a unitary matrix $\tilde{Q}$ such that $\tilde{H}_{k-p}^{[b]} = \tilde{Q}^\star(\hat{Q}^\star S_{11}\hat{Q})\tilde{Q}$ is b-Hessenberg and that the transformation $x^\star \mapsto x^\star\tilde{Q}$ does not modify the last b components of vector x. Thus, after applying $\tilde{Q}$ on (5.6) from right, we finally have

$$A\underbrace{(V_k^{[b]}Q_1\hat{Q}\tilde{Q})}_{\tilde{V}_{k-p}^{[b]}} = [V_k^{[b]}Q_1\hat{Q}\tilde{Q} \;\; R_k^{[b]}] \begin{bmatrix} \tilde{H}_{k-p}^{[b]} \\ R(E_{k-p}^{[b]})^\star \end{bmatrix} = \tilde{V}_{k-p}^{[b]}\tilde{H}_{k-p}^{[b]} + (R_k^{[b]}R)(E_{k-p}^{[b]})^\star,$$

a restarted block Arnoldi decomposition which preserved all of the wanted Ritz values.

Recovering the b-Hessenberg form in the block Arnoldi decomposition is an important step that also appears in other situations such as deflation and locking. Although it does not involve the large matrix A, this operation is potentially time-consuming and should be implemented to perform optimally.

5.2 The controller Hessenberg form

Hessenberg matrices are extensively used in computational control theory, see e.g. [18, 39, 75], or just browse any state of the art control theory software package. For instance, to reveal structural properties of a linear time invariant system (LTI)

$$\left.\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}\right\} \text{ where } \left\{\begin{aligned} A &\in \mathbb{R}^{n\times n}, \quad B \in \mathbb{R}^{n\times m}, \\ C &\in \mathbb{R}^{p\times n}, \quad D \in \mathbb{R}^{p\times m}, \end{aligned}\right. \tag{5.7}$$

one usually transforms it to a simpler canonical form using an orthogonal coordinate transformation $x = Q\tilde{x}$ in the state space. The transformed system reads

$$\left.\begin{array}{rcl} \dot{\tilde{x}}(t) &=& \tilde{A}\tilde{x}(t) + \tilde{B}u(t) \\ y(t) &=& \tilde{C}\tilde{x}(t) + Du(t) \end{array}\right\} \text{ where } \left\{\begin{array}{l} \tilde{A} = Q^\star AQ, \quad \tilde{B} = Q^\star B, \\ \tilde{C} = CQ. \end{array}\right. \tag{5.8}$$

In the single input case ($m = 1$), the orthogonal matrix Q can be constructed so that $\tilde{A}$ is upper Hessenberg and $\tilde{B}$ is collinear with the first column of the $n \times n$ identity matrix I_n. This is the *controller Hessenberg form* of a linear time invariant system. For a general m, the matrix A is m-Hessenberg and B is upper triangular; that is,

$$\tilde{A} = \left[\begin{array}{ccc|ccc|ccc} \star & \star & \star & \star & \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star & \star & \star & \star & \star \\ \hline \star & \star & \star & \star & \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star & \star & \star & \star & \star \\ 0 & 0 & \star & \star & \star & \star & \star & \star & \star \\ \hline 0 & 0 & 0 & \star & \star & \star & \star & \star & \star \\ 0 & 0 & 0 & 0 & \star & \star & \star & \star & \star \\ 0 & 0 & 0 & 0 & 0 & \star & \star & \star & \star \end{array}\right], \quad \tilde{B} = \left[\begin{array}{ccc} \star & \star & \star \\ 0 & \star & \star \\ 0 & 0 & \star \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right],$$

where we have set $m = 3$, $n = 9$.

To reduce a LTI (A, B, C, D) to a controller Hessenberg form, the state of the art computational control library SLICOT [64] contains the subroutine `TB01MD`. This subroutine also includes code that reduces a general dense matrix to the m-Hessenberg form. However, `TB01MD` does not use aggregated transformations (block reflectors) which are optimized for the modern memory architectures as they use BLAS 3 operations. As a consequence, its low flop-to-memory-reference ratio cannot provide optimal efficiency. This is an unsatisfactory situation, because the controller Hessenberg reduction is one of the basic tools in many computational control tasks.

In this section, we review the routine `TB01MD` to present a basic algorithm for reduction to the m-Hessenberg form. `TB01MD` implements the following computation: first, the matrix B is reduced to upper triangular form using the QR factorization:

$$B = Q_B \left[\begin{array}{c} R \\ 0 \end{array}\right] =: Q_B \hat{B}.$$

If we let $\hat{A} = Q_B^\star A Q_B$ and $\hat{C} = CQ_B$ and $\hat{x} = Q_B^\star x$, then (5.7) is equivalent to the system

$$\left.\begin{array}{rcl} \dot{\hat{x}}(t) &=& \hat{A}\hat{x}(t) + \hat{B}u(t) \\ y(t) &=& \hat{C}\hat{x}(t) + Du(t) \end{array}\right\} \tag{5.9}$$

It is easy to see that, using Householder reflectors, one can construct an orthogonal matrix Q_A of the form $Q_A = I_m \oplus \hat{Q}_A$ such that $\tilde{B} = Q_A^\star \hat{B} = \hat{B}$ and $\tilde{A} = Q_A^\star \hat{A} Q_A$ has the form (5.8). The transformation $Q = Q_B Q_A$ then reduces the initial system (5.7) to the controller Hessenberg form (5.8).

Algorithm 12 provides all details of the process; we give only a few comments:

Algorithm 12: Non-blocked implementation of the controller Hessenberg reduction (`TB01MD`)

Input: (A, B)
Output: $(\tilde{A}, \tilde{B})$ in controller Hessenberg form

1 tB = zero-vector of dimension m ;
2 **for** $i = 1, 2, \ldots, m$ **do**
3 | Let $H = I - \tau vv^\star$ be the Householder reflector for $B(i:n, i)$;
4 | Store v in $B(i+1:n, i)$ and τ in $tB(i)$ (note $v(1) = 1$) ;
5 | Transform $A(i:n, :) \leftarrow HA(i:n, :)$;
6 | Transform $A(:, i:n) \leftarrow A(:, i:n)H$;
7 | Transform $B(i:n, i:m) \leftarrow HB(i:n, i:m)$;
8 **end**

9 tA = zero-vector of dimension n ;
10 **for** $i = m, m+1, \ldots, n$ **do**
11 | $j = i - m + 1$;
12 | Let $H = I - \tau vv^\star$ be the Householder reflector for $A(i+1:n, j)$;
13 | Store v in $A(i+1:n, j)$ and τ in $tA(i)$;
14 | Transform $A(i+1:n, i:n) \leftarrow HA(i+1:n, i:n)$;
15 | Transform $A(:, i+1:n) \leftarrow A(:, i+1:n)H$;
16 **end**

17 $\tilde{A} \leftarrow A$; $\tilde{B} \leftarrow B$;

- The lines 1–8 compute the QR factorization of B, transform A into $\hat{A}$ and B into $\hat{B}$. The matrix Q_B is implicitly stored in the lower trapezoidal part of $\hat{B}$ and in the auxiliary vector tB.

- The lines 9–16 transform $\hat{A}$ into the m-Hessenberg form $\tilde{A}$. The matrix Q_A is implicitly stored under the m-th diagonal of $\tilde{A}$ and in the auxiliary vector tA.

- The Householder reflectors in the lines 3 and 12 are computed using the LAPACK routine `xLARFG` and applied in lines 5–7 and 14–15 using `xLATZM`.

The main performance bottleneck in the above algorithm is the lack of blocking. The Householder reflectors are applied to A and B immediately after they have been constructed, instead of just being stored while waiting for application at a later, better suited moment. This prevents the routine from taking advantage of a modern machine's memory hierarchy and from usage of efficient cache-aware (or cache-oblivious) BLAS 3 algorithms.

It is easy to modify the QR-part of the algorithm to use blocked code. LAPACK contains blocked routines `xGEQRF` and `xORMQR` which, respectively, calculate the QR factorization and apply a series of Householder reflectors to a given matrix; see Algorithm 13.

If $m = 1$, the computation of Algorithm 13 completes by calling the LAPACK 's subroutine `xGEHRD` to transform the matrix $\hat{A}$ to the upper 1-Hessenberg form. This sub-

ALGORITHM 13: Blocked algorithm for transforming system (5.7) into (5.9)

Input: (A, B)
Output: $(\hat{A}, \hat{B})$

1 tB = zero-vector of dimension m ;
2 Call `DGEQRF` on B to transform it into upper triangular with Householder vectors stored under its main diagonal and τs stored in vector tB ;
3 Call `DORMQR` on A to multiply it from the right by Householder reflectors stored in B and tB;
4 Call `DORMQR` on A to multiply it from the left by Householder reflectors stored in B and tB;

routine uses a highly optimized BLAS 3 algorithm with block reflectors. As demonstrated in [9], such an algorithm achieves speedup up to the factor of four over `TB01MD`. However, there is currently no routine in LAPACK that would help us in circumventing the similar problem with the m-Hessenberg part of the algorithm when $m > 1$. In the next section we describe adaptation of `TB01MD` to work with $m > 1$ and we analyze the runtime performance of the resulting algorithm.

5.3 A blocked algorithm

As with the QR and the 1-Hessenberg LAPACK routines, the main idea for a faster m-Hessenberg algorithm is to partition the $n \times n$ matrix A column-wise into blocks of b consecutive columns, $A = [A^{(1)} \quad A^{(2)} \quad \dots \quad A^{(l)}]$, $l = \lceil n/b \rceil$. Each of the leading $l-1$ blocks carries b columns, while the last one contains the remaining $n - (l-1)b$ columns. After determining suitable block size b, the blocks are processed one at a time, from left to right, and the computation is organized to enhance temporal and spatial data locality. Many details and tricks are involved, and this section contains the blueprint of the new subroutine.

5.3.1 Processing a single block

Processing of a block consists of constructing b Householder reflectors that annihilate appropriate elements in each of the block's columns. More precisely, consider the i-th block

$$A^{(i)} = [a_1^{(i)} \quad a_2^{(i)} \quad \dots \quad a_b^{(i)}].$$

Let us recursively define a sequence of Householder reflectors: for $j = 1, 2, \dots, b$, denote with $H_j = I - \tau_j v_j v_j^\star$ the Householder reflector such that the vector $H_j H_{j-1} \cdots H_1 a_j^{(i)}$ has zeros as its $(k+j+1)$-th, $(k+j+2)$-th, …, n-th element (with an offset k that will be a function of i and b). Here τ_j is a scalar and v_j is a vector such that $v_j(1 : k+j-1) = 0$ and $v_j(k+j) = 1$. We accumulate these vectors in the matrices $V_j = [v_1 \quad v_2 \quad \dots \quad v_j]$, $V = V_b$. Here the introduction of the V_j matrices is for explanatory purposes only. As

in `xGEQRF` or `xGEHRD`, the non-trivial elements of the vectors v_j are actually stored in the matrix A in places of entries that have been zeroed by H_j's. In our description of the algorithm, we overwrite the initial data, thus there will be no "time stepping" index in our notation.

To set the stage for the new algorithm and to introduce necessary notation, we briefly describe block reflectors. For more details we refer the reader to [60].

Proposition 5.4. *Let $Q_j = H_1 H_2 \dots H_j$. The matrices Q_j can be represented in a special form of the block reflector: $Q_j = I - V_j T_j V_j^\star$, where T_j is an upper triangular $j \times j$ matrix.*

Proof. The construction of T_j is inductive: we first set $V_1 = [v_1]$, $T_1 = \tau_1$, and then we easily verify that, for $t_+ = -\tau_j T_{j-1} V_{j-1}^\star v_j$, it holds

$$\begin{aligned} Q_j &= Q_{j-1} H_j \\ &= (I - V_{j-1} T_{j-1} V_{j-1}^\star) \cdot (I - \tau_j v_j v_j^\star) \\ &= I - [V_{j-1} \;\; v_j] \cdot \begin{bmatrix} T_{j-1} & t_+ \\ 0 & \tau_j \end{bmatrix} \cdot [V_{j-1} \; v_j]^\star . \end{aligned}$$

Thus, setting $T_j = \begin{bmatrix} T_{j-1} & t_+ \\ 0 & \tau_j \end{bmatrix}$ completes the proof. □

Once the block is precessed, we will have to update

$$\begin{aligned} A \leftarrow Q^\star A Q &= (I - V T^\star V^\star) A (I - V T V^\star) \\ &= (I - V T^\star V^\star)(A - Y V^\star) , \end{aligned} \tag{5.10}$$

where $T = T_b$, $Q = Q_b$ and $Y = AVT$. The auxiliary matrix Y will be used in the algorithm for fast update of the matrix A "from the right-hand side" ($A \leftarrow A - YV^\star$).

Proposition 5.5. *The construction of $Y_j = Y(:, 1:j) = AV_jT_j$ can be carried out as follows:*

$$\begin{aligned} Y_1 &= \tau_1 A v_1 \\ Y_j &= \left[\; Y_{j-1} \quad \tau_j (-Y_{j-1} \cdot V_{j-1}^\star v_j + A v_j) \; \right] . \end{aligned} \tag{5.11}$$

Proof. Starting with $Y_1 = \tau_1 A v_1$ and using Proposition 5.4, we have

$$\begin{aligned} Y_j &= A \cdot \left[\; V_{j-1} \quad v_j \; \right] \cdot \begin{bmatrix} T_{j-1} & t_+ \\ 0 & \tau_j \end{bmatrix} \\ &= \left[\; A V_{j-1} T_{j-1} \quad A V_{j-1} t_+ + \tau_j A v_j \; \right] = (\text{since } t_+ = -\tau_j T_{j-1} V_{j-1}^\star v_j) \\ &= \left[\; Y_{j-1} \quad \tau_j (-Y_{j-1} \cdot V_{j-1}^\star v_j + A v_j) \; \right] . \end{aligned} \tag{5.12}$$

□

ALGORITHM 14: Processing of a block to reduce it to m-Hessenberg form

Input: block $A^{(i)}$, block size b, index k
Output: partially updated block $A^{(i)}$, transformation matrices Y and T, auxiliary array TAU

```
1  for j = 1, 2, ..., b do
2      if j > 1 then
3          Update a_j^(i)(k+1:n):
4              a_j^(i)(k+1:n) = a_j^(i)(k+1:n) − (Y_{j−1} V*_{j−1})(k+1:n, k−j+m);
5          Update a_j^(i)(k+1:n) by applying I − V_{j−1} T*_{j−1} V*_{j−1} on it from the left;
6      end
7      Generate the elementary reflector H_j to annihilate a_j^(i)(k+j+1:n);
8      Compute Y_j(k+1:n, j);
9      Compute T_j(:, j);
10 end
11 Compute Y(1:k, 1:b);
```

In order to compute the reflector that annihilates the elements in the j-th column of the block, we must have already updated this part of the matrix A with the accumulated product of the previous $j-1$ reflectors. To accomplish that, we have to calculate Y_{j-1}, as formula (5.10) suggests. Note that computing the reflectors requires only the elements of A in the rows from $(k+1)$-st on – thus the update will also require only the elements of Y_{j-1} with the same row indices.

To obtain a more efficient implementation, one has to choose carefully the appropriate moment for application of each transformation. In order to convert as many as possible matrix-vector (BLAS 2) operations into matrix-matrix operations (BLAS 3), we have to postpone transformations on parts of matrices until the very last moment in which the updated content is actually required for some further calculation. Algorithm 14 sketches the processing of a single block.

Note that the Householder vector from line 7 is stored in $a_j^{(i)}(k+j+1:n)$, while the scalar τ is stored in an auxiliary array TAU as the element $TAU(j)$. That line reduces to a single call of the LAPACK routine `xLARFG`.

We proceed by giving more details on how to implement the remaining lines in Algorithm 14.

Updating $a_j^{(i)}(k+1:n)$ from the right. Column $a_j^{(i)}(k+1:n)$ is updated via the formula

$$a_j^{(i)}(k+1:n) = a_j^{(i)}(k+1:n) - (Y_{j-1}V_{j-1}^{\star})(k+1:n, k-j+m).$$

Figure 5.1 illustrates the situation in the case $n = 15$, $m = 2$ and the block size $b = 5$. The current block $A^{(i)}$ is shaded ($i = 2$); the parameter k is equal to $(i-1)\cdot b + m = 7$.

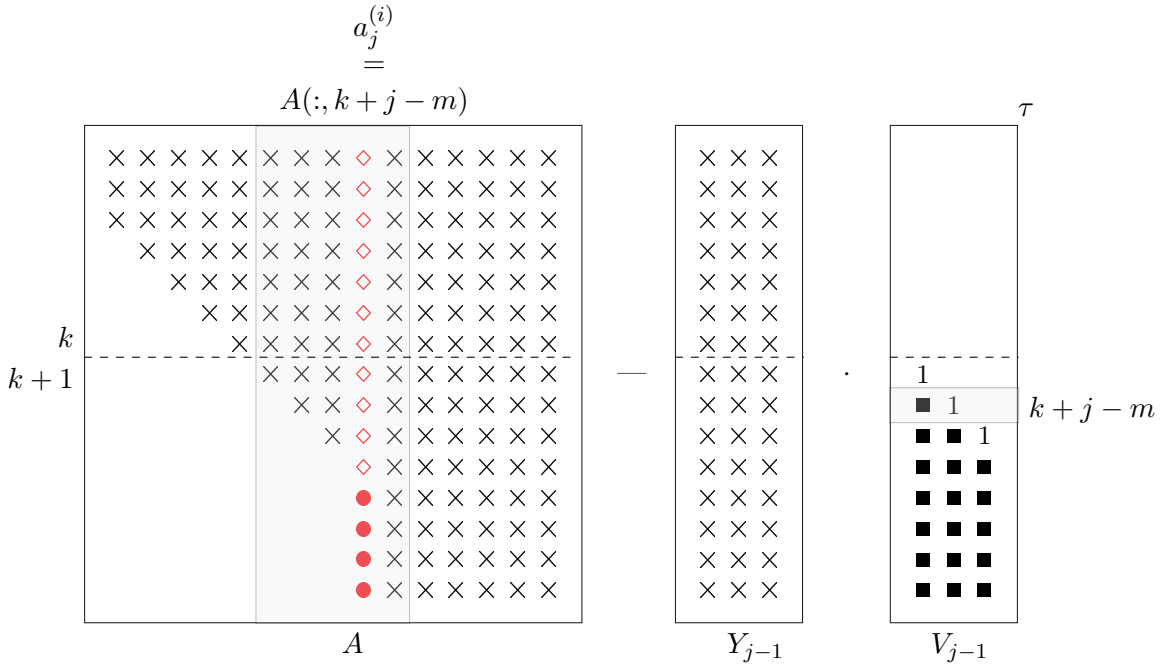


Figure 5.1: Updating column $a_j^{(i)}(k+1:n)$ from the right (line 4 in Algorithm 14)

The elements of a particular column $a_j^{(i)}$ are shown as diamonds and filled circles ($j = 4$). All elements below the m-th subdiagonal in the columns to the left of $a_j^{(i)}$ have already been set to zero by the algorithm, and our goal now is to use a Householder reflector and zero out those elements of the column $a_j^{(i)}$ that are shown as filled circles.

However, prior to that, we have to apply to this column the previously generated block reflector $I - V_{j-1}T_{j-1}V_{j-1}^{\star}$ from both the left and the right. During the block processing routine only the row indices $k+1:n$ of this column will be updated – the remaining indices are not involved in computation of the Householder reflectors for the current block. For the same reason, only the rows $k+1:n$ of Y_{j-1} and V_{j-1} will be referenced in this computation. The horizontal dashed line separates the rows $1:k$ and $k+1:n$ of all matrices in the figure. All action takes place below this line.

The non-trivial elements of the matrix V_{j-1} are shown as ■ – these are actually stored in places of the elements of $A^{(i)}$ that have already been zeroed out. All elements above the ones in the matrix V_{j-1} are equal to zero.

Note that the j-th column of block $A^{(i)}$ is the $((i-1)\cdot b + j = k + j - m)$-th column of the original matrix A. Thus, updating the column $a_j^{(i)}$ involves only the row $k+j-m$ of the matrix V_{j-1} – the shaded one in Figure 5.1. In fact, what is needed from this row are the leading $j-m-1$ elements stored in $A^{(i)}(k+j-m, 1:j-m-1)$, then a single element 1 (thus the trick with introduction of 1 in line 4.3).

ALGORITHM 15: Details of Line 4 in Algorithm 14

4.1 **if** $j > m$ **then**
4.2 $\quad temp = a^{(i)}_{j-m}(k+j-m)$;
4.3 $\quad a^{(i)}_{j-m}(k+j-m) = 1$;
4.4 $\quad$ Call `xGEMV` to compute
$\quad\quad a^{(i)}_{j}(k+1:n) = a^{(i)}_{j}(k+1:n) -$
$\quad\quad\quad Y_{j-1}(k+1:n, 1:j-m) \cdot \left(A^{(i)}(k+j-m, 1:j-m)\right)^{\star}$;
4.5 $\quad a^{(i)}_{j-m}(k+j-m) = temp$;
end

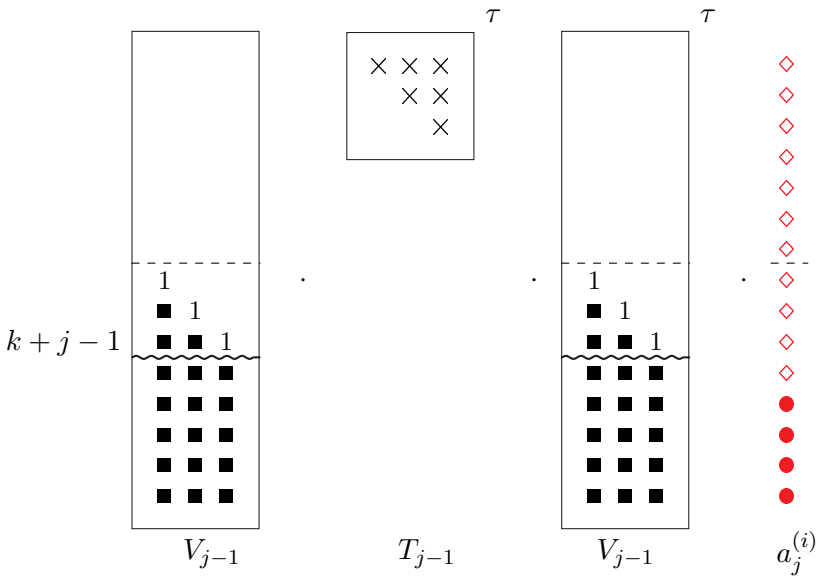


Figure 5.2: Updating column $a^{(i)}_{j}(k+1:n)$ from the left

Updating $a^{(i)}_{j}(k+1:n)$ from the left. Algorithm 16 gives the details of the implementation of line 5 in Algorithm 14.

The lines 5.2 and 5.3 compute the product $V^{\star}_{j-1}a^{(i)}_{j}$. This multiplication is carried out in two phases: first, we multiply the triangular block $V_{j-1}(k+1:k+j-1, 1:j-1)$ using the `xTRMV` routine. This routine can be deployed to "pretend" as if the lower triangular matrix has ones on its diagonal without explicitly looking up matrix elements, enabling us to avoid the temporary backup of elements of $A^{(i)}$ which occupy that place. Then the rectangular block $V_{j-1}(k+j:n, 1:j-1)$ multiplies the rest of vector $a^{(i)}_{j}$.

In line 5.4, the vector $T^{\star}_{j-1}V^{\star}_{j-1}a^{(i)}_{j}$ is formed, and lines 5.5–5.7 complete the transformation by multiplying that vector by V_{j-1} from the left.

Figure 5.2 shows this update from the left. The triangular block of V_{j-1} is between the dashed and the wavy line.

Algorithm 16: Details of Line 5 in Algorithm 14

5.1 Copy $a_j^{(i)}(k+1:k+j-1)$ to $T(1:j-1,j)$;
5.2 Call `xTRMV` to compute
$T(1:j-1,j) = \left(A^{(i)}(k+1:k+j-1,1:j-1)\right)^\star \cdot T(1:j-1,j)$;
5.3 Call `xGEMV` to compute
$T(1:j-1,j) = T(1:j-1,j) + \left(A^{(i)}(k+j:n,1:j-1)\right)^\star \cdot a_j^{(i)}(k+j:n)$;
5.4 Call `xGEMV` to compute $T(1:b,j) = T_{j-1}^\star \cdot T(1:b,j)$;
5.5 Call `xGEMV` to compute $a_j^{(i)}(k+j:n) = A^{(i)}(k+j:n,1:j-1) \cdot T(1:j-1,j)$;
5.6 Call `xTRMV` to compute
$T(1:j-1,j) = A^{(i)}(k+1:k+j-1,1:j-1) \cdot T(1:j-1,j)$;
5.7 $a_j^{(i)}(k+1:k+j-1) = a_j^{(i)}(k+1:k+j-1) - T(1:j-1,j)$;

Algorithm 17: Details of Line 8 in Algorithm 14

8.1 $temp = a_j^{(i)}(k+j)$; $a_j^{(i)}(k+j) = 1$;
8.2 Call `xGEMV` to compute
$Y_j(k+1:n,j) = A(k+1:n,k+j:n) \cdot a_j^{(i)}(k+j:n)$;
8.3 Call `xGEMV` to compute
$T_j(1:j-1,j) = (A(k+j:n,1:j-1))^\star \cdot a_j^{(i)}(k+1:n)$;
8.4 Call `xGEMV` to compute
$Y_j(k+1:n,j) = Y_j(k+1:n,j) - Y_{j-1}(k+1:n,:) \cdot T_j(1:j-1,j)$;
8.5 Scale $Y_j(k+1:n,j)$ by multiplying it with $\tau_j = TAU(j)$;

Other details of block processing. Next, we append a column to Y_{j-1} to define Y_j as in (5.11). Note that $a_j^{(i)}(k+j+1:n)$ now contains the non-trivial part of the j-th elementary Householder vector. Algorithm 17 gives the details.

Line 8.2 is the only line in the loop of the block processing algorithm that references elements of A outside of the current block $A^{(i)}$. This is also the only line which deals with a matrix of dimension $\mathcal{O}(n) \times \mathcal{O}(n)$; all others have at least one dimension of block size b or less. Line 8.3 computes the product $V_{j-1}^\star v_j$ – observe that $v_j(1:k+j-1) = 0$, so we can disregard the rows 1 to $k+j-1$ of the matrix V_{j-1}. The old value of the element $a_j^{(i)}(k+j)$, whose backup copy is in Line 8.1, can be restored after the update of the next column $a_{j+1}^{(i)}$.

All that is now left to do in order to complete the matrix T_j is to re-scale the current content of its last column by $TAU(j)$, and to multiply it by T_{j-1} from the left by using yet another call to `xTRMV`.

Finally, we show how to compute the first k rows of Y, outside of the main loop in Algorithm 14:

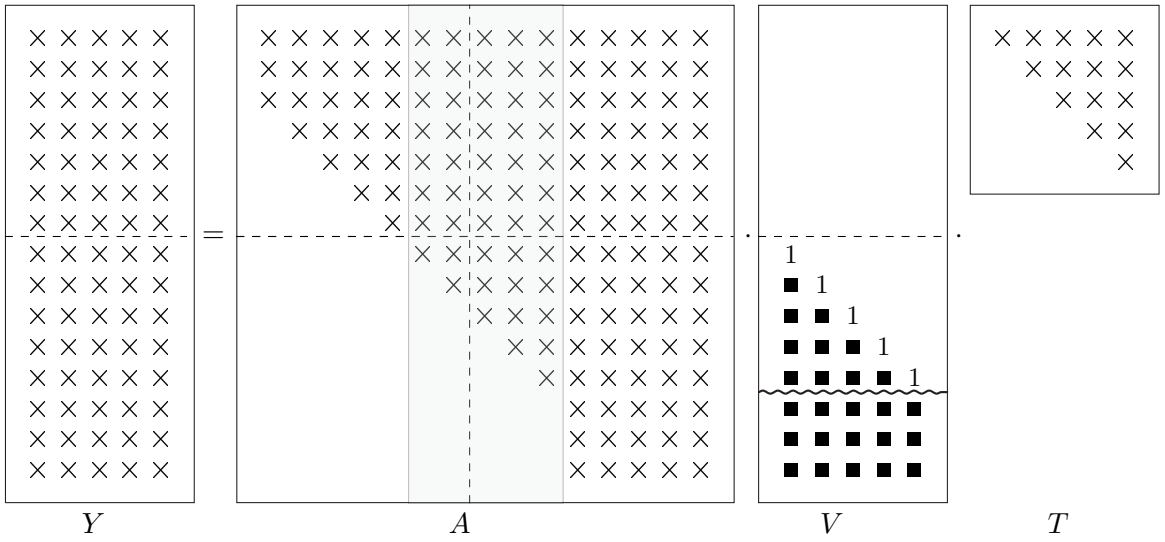


Figure 5.3: Updating first k rows of Y

ALGORITHM 18: Details of Line 11 in Algorithm 14

11.1 Copy $A(1:k, k+1:k+b)$ to $Y(1:k, 1:b)$;

11.2 Call `xTRMM` to compute

$$Y(1:k,1:b) = Y(1:k,1:b) \cdot A^{(i)}(k+1:k+b,1:b) \ ;$$

11.3 Call `xGEMM` to compute

$$Y(1:k,1:b) = Y(1:k,1:b) + A(1:k,k+b+1:n) \cdot A^{(i)}(k+b+1:n,1:b) \ ;$$

11.4 Call `xTRMM` to compute $Y(1:k,1:b) = Y(1:k,1:b) \cdot T$;

In Figure 5.3, the region below the elements denoted by $\times$ in the current block (shaded) of the matrix A has been zeroed out. It now stores the non-trivial part of the matrix V (elements denoted by ■). Multiplication by the matrix V from the right in lines 11.2 and 11.3 is once again split into triangular and rectangular parts. Line 11.3 references elements not in the current block. Note that only columns $k+1:n$ of the matrix A (the ones to the right of the vertical dashed line) are referenced while updating Y, because of the zero-pattern in V.

This completes the description of the block processing algorithm. We now turn our attention to the global algorithm on the block level.

The concept of mini-blocks. The first m columns of a block do not need to be updated from the right since the corresponding rows of matrices V_{j-1} are zero. Moreover, if m is greater than or equal to the block size, then no updates from the right are needed

at all within the current block. We are indebted to Daniel Kressner [36] for pointing this out; a similar concept is also used in [12] for the reduction of a symmetric matrix to the tridiagonal form. When m is less than the block size, this fact permits another variant of Algorithm 14 in which:

1. The block is split into several disjoint *"mini-blocks"*, each (except maybe the last one) consisting of m consecutive columns.

2. In each "mini-block", column by column is updated only from the left, and the appropriate elements are annihilated.

3. A *"mini-reflector"* $\mathcal{Q}_m = I - \mathcal{V}_m \mathcal{T}_m \mathcal{V}_m^\star$ is aggregated from m annihilations.

4. After processing of the "mini-block", the block reflector Q_{j+m} is updated:

$$\left.\begin{aligned} V_{j+m} &= \begin{bmatrix} V_j & \mathcal{V}_m \end{bmatrix}; \\ T_{j+m} &= \begin{bmatrix} T_j & T_{jm} \\ 0 & \mathcal{T}_m \end{bmatrix}, \; T_{jm} = -T_j V_j^\star \mathcal{V}_m \mathcal{T}_m; \\ Y_{j+m} &= \begin{bmatrix} Y_j & (-Y_j V_j^\star \mathcal{V}_m + A\mathcal{V}_m)\mathcal{T}_m \end{bmatrix}. \end{aligned}\right\} \tag{5.13}$$

5. The next m columns of A are updated from the right; these columns are then declared the new current "mini-block".

Note that in order to update all columns in the next "mini-block" from the right, one only needs elements of Y that lie in the current and previous "mini-blocks". Thus, an update from the right for the entire next "mini-block" and the appropriate part of the matrix Y may be computed after all columns in the current "mini-block" are annihilated.

The details are worked out in Algorithm 19 which, for $m > 1$, makes better use of BLAS 3 operations than the original Algorithm 14 and provides better performance. The minimum size of m where this variant is more effective depends on the user's machine setup – on our machine, for $m \geq 3$ the speedup was already notable. See Section 5.5 for timing results that illustrate a considerable improvement due to the modification described in Algorithm 19.

5.3.2 The outer loop

Figure 5.4 shows how to perform the update $A = A - YV^\star$, where Y and V are computed during the last block processing (the last block $A^{(i)}$ is shaded). This update only affects the columns $k+1 : n$ of the matrix A – the ones to the right of the vertical dashed line. Only the elements of A that are encircled or boxed have yet to be updated, since the block processing has already updated $A^{(i)}(k+1 : n, 1 : b)$.

The boxed elements of A will be updated by subtracting the product of Y with the boxed part of V. Since the elements of V above the ones are all zeros and V is stored in $A^{(i)}$, the appropriate part of $A^{(i)}$ has to be replaced with zeros and ones in order to use the `xGEMM` routine for the multiplication. The encircled elements of A will be updated by

ALGORITHM 19: Processing of a block to reduce it to m-Hessenberg form

Input: block $A^{(i)}$, block size b, index k
Output: partially updated block $A^{(i)}$, transformation matrices Y and T, auxiliary array TAU

for $j = 1, 2, \ldots b$ **do**
  if $j > 1$ **then**
    Update $a_j^{(i)}(k+1:n)$ only from the left by applying $I - V_{j-1}T_{j-1}^{\star}V_{j-1}^{\star}$;
  end
  Generate the elementary reflector H_j to annihilate $a_j^{(i)}(k+j+1:N)$;
  Compute $T_j(:, j)$;
  if $j \ mod \ m = 0$ *or* $j = b$ **then**
    $cMini = m$ *or* $(b \ mod \ m)$; `// current miniblock size`
    $nMini = \min\{b - j, m\}$; `// next miniblock size`
    Compute $Y_j(k+1:n, j-cMini+1:j)$;
    Call `xGEMM` to update the entire next miniblock from the right:
    $A^{(i)}(k+1:n, j+1:j+nMini) =$
    $A^{(i)}(k+1:n, j+1:j+nMini) - (Y_j V_j^{\star})(k+1:n, j+1:j+nMini)$;
  end
end
Compute $Y(1:k, 1:b)$;

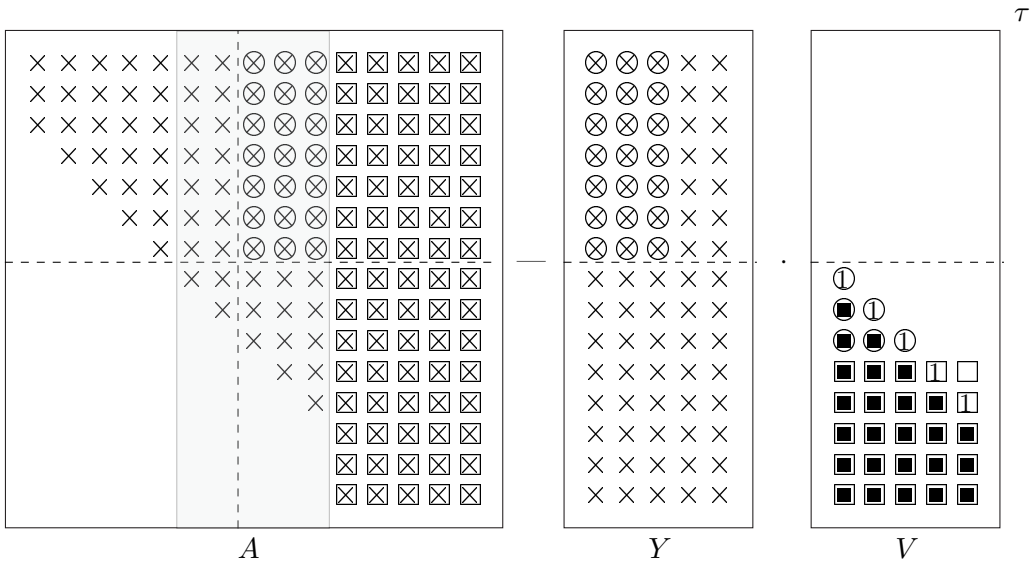


Figure 5.4: Updating matrix A from the right

Algorithm 20: Block algorithm for reduction to the m-Hessenberg form

Input: $n \times n$ matrix A, block size b, bandwidth m
Output: A converted to m-Hessenberg form

for $z = 1,\ 1+b,\ 1+2\cdot b,\ 1+3\cdot b,\ \ldots$ **do**
  $i = (z-1)/b + 1$;
  Process block $A^{(i)} = A(1:n, z:z+b-1)$ with $k = z+m-1$;
  `// Apply block reflector from the right:`
  Backup the upper triangle of $A^{(i)}(k+b-m : k+b-1, b-m+1 : b)$ into matrix *temp* and replace it with the upper triangle of identity matrix ;
  Call `xGEMM` to compute $A(1:n, z+b:n) = A(1:n, z+b:n) - Y \cdot \left(A^{(i)}(k+b-m:n, 1:b)\right)^{\star}$;
  Restore *temp* to upper triangle of $A^{(i)}(k+b-m : k+b-1, b-m+1 : b)$;
  Call `xTRMM` to compute $temp = Y(1:k, 1:b-m) \cdot \left(A^{(i)}(k+1:k+b-m, 1:b-m)\right)^{\star}$;
  $A(1:k, k+1:z+b-1) = A(1:k, k+1:z+b-1) - temp$;
  `// Apply block reflector from the left:`
  Call `xLARFB` to apply block reflector (V, T) from left onto $A(k+1:n, i+b:n)$;
end

subtracting the product of encircled elements of Y with the triangular matrix of encircled elements in V.

The Block Algorithm 20 at first just stores all of the block's reflectors, instead of annihilating columns one by one and applying the resulting reflector to the entire matrix A each time. Only after a block is processed, the whole batch of its reflectors is applied onto the matrix A, first from the left and then from the right.

This brings several important performance benefits into play. The first one is the localization of memory referencing: in non-blocked version the algorithm has to reference elements of A that are "far away" (in terms of memory addresses) from the current column when applying its reflectors, for each column being annihilated. To the contrast, the blocked version mostly references only elements local to the block being processed. Elements "far away", i.e. the ones outside of that block are referenced only after all of the block's reflectors have already been computed. If block size is chosen to fit into the computers cache memory, there will be much less cache misses during the computation of all the reflectors and less data fetching from the slower main memory. The second advantage is the more effective transformation $A \mapsto Q^{\star}AQ$. If Q were a single Householder reflector, one could only use BLAS 2 (i.e. matrix-vector) operations to apply the transformation, whereas transformation routines such as `xLARFB` use the more efficient BLAS 3 (i.e. matrix-matrix) operations (e.g. `xGEMM`, `xTRMM`) taking advantage of the underlying block reflector structure of Q.

5.4 A hybrid algorithm

The demand of the computer entertainment industry and the computed aided design for ever more realistic experience induced a rapid development of the graphics processing units (GPUs) in the last few years. The complexity and the computing power of GPUs grew tremendously and exceeded the CPU by a large margin. To appreciate the difference, let us compare the hardware we used for testing of our algorithms:

- CPU: Intel Xeon X5470 processor – 820 million transistors, 53.3 Gflops theoretical peak performance [32].
- GPU: NVIDIA GeForce GTX280 – 1.4 billion transistors, 933 Gflops theoretical peak performance [49, 79].

Although the measured performance is far from the theoretical, and depends on a number of additional factors (the memory bandwidth, for example), the observed ratio still approximately holds; see [76] for a detailed study.

However, the technology of using the GPU for the general purpose computing is still quite young at the time of this writing. The hardware architecture and design of GPUs was motivated by other applications and only recently the opportunity of using GPU as a scientific computing tool appeared. NVIDIA and AMD, market-share leaders of the video card industry, have each released their hardware and software solutions, with NVIDIA's Compute Unified Device Architecture (CUDA [48]) being the first general purpose computing engine accessible to software developers.

With the arising of such powerful tools, a notable trend in high performance computing is to make the transition from the traditional CPU-only configurations to the hybrid configurations which use mixed CPU + GPU architecture. The most powerful computer system in the world [72] in November 2010 is Tianhe-1A composed of 14336 Xeon processors and 7168 NVIDIA Fermi GPUs, and this systems' peak performance is 2.5 petaFLOPS.

The challenge of software design in such a hybrid environment is to take the best features of both, relatively complementary, worlds. Modern CPU's consist of a small (typically 2-6) number of powerful cores each capable of solid performance when running sequential code. Also, the memory hierarchy ranging from smallest, but fastest (CPU registers, different levels of cache) to largest, but slower (RAM) has to be taken into consideration. On the other hand, GPU's have a large number (a few hundreds) of small cores (*stream processors*) which can run only simultaneously, executing the same code on the different data, yearning for the design of highly parallel code. There are different types of memory on the GPU; the memory hierarchy is introduced in the latest generation of video cards (NVIDIA Fermi). The cost of communicating between the CPU and the GPU cannot be neglected as well.

In this section, we present an implementation of a hybrid algorithm for reducing the matrix to the m-Hessenberg form. For computing on the GPU we use CUBLAS [50], which is a part of NVIDIA's CUDA, providing functionality of regular BLAS libraries

using a well-known interface, plus a number of additional subroutines for moving the data from CPU to the GPU and vice versa. Using CUBLAS, the developer does not need to worry about the very details of hardware implementation in order to produce efficient code, since the CUBLAS library uses high level routines and direct communication with the hardware is abstracted away. This also makes the final code more portable to the future generations of both video cards and CUDA software development toolkit. A similar software for reduction to the (1-)Hessenberg form was already developed by Tomov et al. [71]; we have successfully obtained the performance they reported.

To begin, we first profile the developed blocked algorithm in order to detect which subtasks require the most computing resources (here $n = 5000$ and $m = 20$; all percentages are relative to the total running time):

subtask	CPU time
Processing of a block (Algorithm 19)	41.3%
• Computing $A\mathcal{V}_m$ in Line 10	(30.0%)
• Line 14: update of $Y(1:k, 1:b)$	(8.8%)
Out-of-block update (Lines 4–9 of Algorithm 20)	58.7%
• Line 5 to update A from the right	(22.5%)
• Line 9 to update A from the left	(35.3%)

As we can see, most of the CPU time is spent doing the four bulleted operations; these are actually four calls to `xGEMM` (without the "mini-blocks", computing $A\mathcal{V}_m$ is a `xGEMV`). We try to distribute these operations to both the CPU and the GPU, proportionally to their computing power:

(a) Computing Av_j within the block processing will be done one the GPU, since `xGEMV` is highly parallelizable and can be done with high flop rate on the GPU. This is also the case for the line 10 in Algorithm 19, especially when "mini-blocks" are thin enough so that cache effects of the CPU don't show up.

(b) Line 14 of Algorithm 19 will be computed on the CPU, since it can make use of the CPU-efficient BLAS 3 routines `xGEMM` and `xTRMM`.

(c) Lines 4–9 of Algorithm 20 will be split to minimize communication between the CPU and the GPU, and to put heavier load on the GPU:

1. Update of A from the left involves only $A(k+1:n, :)$; this will be done on the GPU;
2. Update of $A(k+1:n, :)$ from the right will be done on the GPU;
3. Update of $A(1:k, :)$ from the right will be done on the CPU.

At the beginning ot the algorithm, the matrix A is copied to the matrix dA on the GPU. As a block is being annihilated, each column is first updated from the left and a

reflector is computed; this takes place on the CPU. Then the computed reflector v_j is copied to the GPU where we keep them in a separate $n \times b$ matrix dV. The matrix dV has zeros above the m-th subdiagonal and ones on it, so that products with it can be computed using one `xGEMM`, without the need for a temporary backup. All calls to the GPU subroutines are done asynchronously, including the data transfers. This way, the CPU can continue computing the matrix T while v_j is being transferred to the GPU.

Once all columns in the current "mini-block" are annihilated, we already have a the matrix V_j ready on both the GPU (dV) and the CPU (below the m-th subdiagonal of A). To update the block reflector using (5.13), we first copy the last m columns (those that belong to the current "mini-block") of T to dT at the GPU. Then the GPU can update rows $k+1 : n$ of the matrix $dA \cdot dV \cdot dT$, which overwrites parts of dA not needed any more. Once computed, these elements are transferred to the matrix Y on the CPU. The matrix Y is used on the CPU to update the next "mini-block" from the right. The pseudocode for the block processing routine is shown as Algorithm 21.

Conversion of the outer loop of the algorithm is straightforward; see Algorithms 22 and 23; note that we have moved computation of $Y(1 : k, :)$ to the CPU update. As the CPU and the GPU each perform disjoint parts of the original algorithm, some data in matrices A/dA, dV, T/dT and Y may not reflect the actual situation at some point of the algorithm. For example, Figure 5.5 shows matrices A and dA after one pass of the main loop (process block + CPU update + GPU update). The green area represents valid and up-to-date data, and the red area represents data which is not. Fortunately, it is necessary to synchronize only a small part of the matrix A in order for the algorithm to continue: rows $i+b : i+b+m-1$ of the matrix dA have to be copied to the CPU (line 3 of Algorithm 22) as well as columns that belong to the next block (line 1 of Algorithm 21).

The timing breakdown now reads as follows (here $n = 6500$ and $m = 20$):

subtask	total time
Processing of a block (Algorithm 21)	20.2%
• Line 13 : compute $Y_j(k+1 : n, j)$	(17.3%)
GPU update (Algorithm 22)	27.8%
• Line 2 : update of $A(k+1, :)$ from the left	(19.1%)
• Line 1 : update of $A(k+1, :)$ from the right	(8.5%)
CPU update (Algorithm 23)	49.7%
• Line 1 : update of $Y(1 : k, 1 : b)$	(27.8%)
• Line 2 : update of $A(1 : k, z+b : n)$	(21.9%)

Most of the processing time is now spent in the CPU update task. Another major observation is that this task can be done in parallel with the GPU update of the current block and the processing of the next one. To balance the amount of work, only one core of the CPU will be dedicated to processing of a block, and all the others will be doing the CPU update. The latter task is more time demanding for the CPU and the amount of work in it grows toward the end of the reduction. This is why we split the job of the

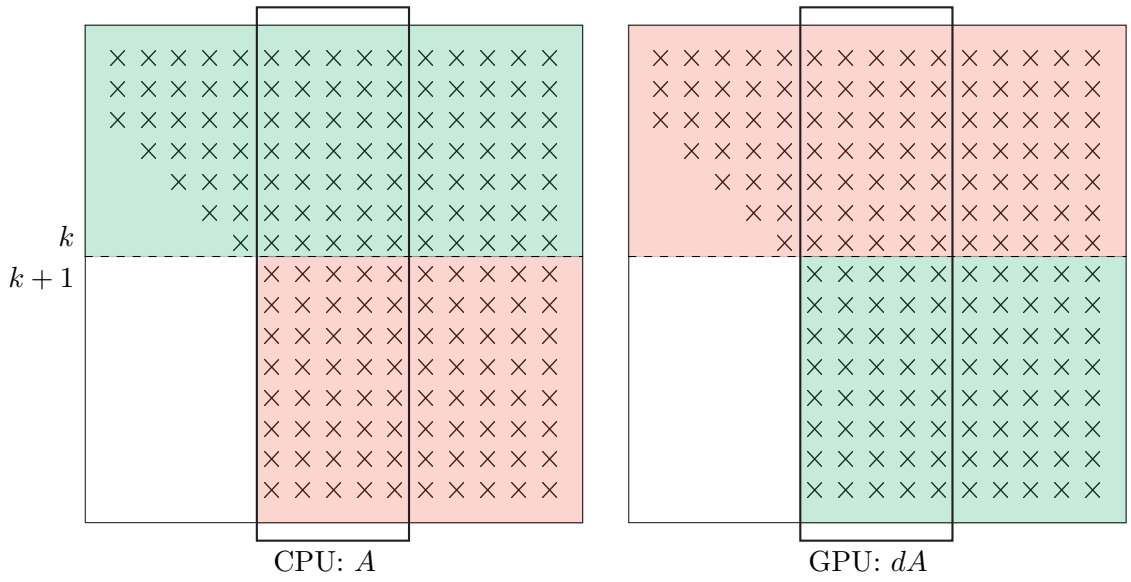


Figure 5.5: Matrices A (at the CPU) and dA (at the GPU) after one pass through the outer loop of Algorithm 24. Green color indicates elements that are up-to-date, red color indicates those that are not.

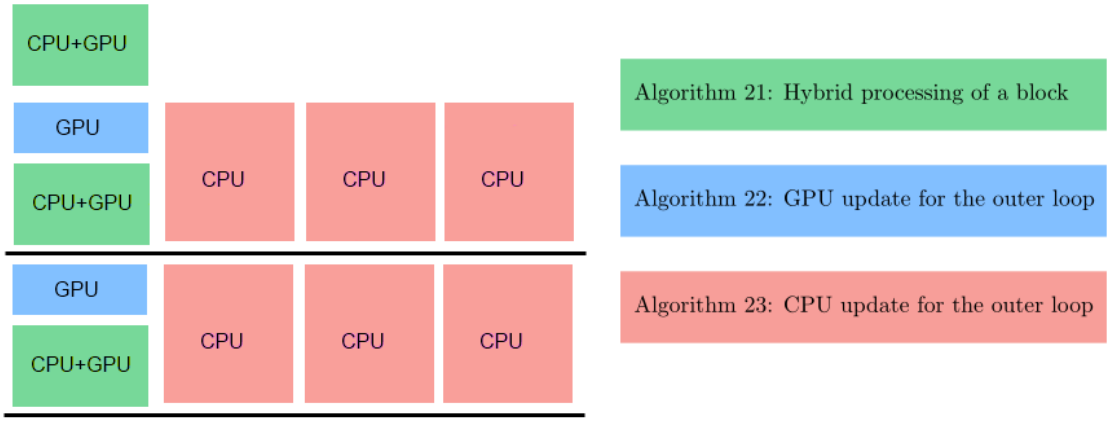


Figure 5.6: Parallelism in the hybrid algorithm for the m-Hessenberg reduction. The block processing and the GPU–update is done on a single core, while the CPU–update algorithm spreads across all other cores of the CPU. The thick black line is a synchronization point.

CPU update between the cores: each core will get several consecutive rows of Y and A to compute. This too can be done in parallel; should the number of cores be large enough, the computation can be completed just before the GPU update of the current block and the block processing of the next one have both finished. See Figure 5.6.

ALGORITHM 21: Hybrid processing of a block

Input: block $A^{(i)}$, block size b, index k
Output: partially updated block $A^{(i)}$, transformation matrices Y and T, auxiliary array TAU

```
1  Copy the current block dA(k+1:n, z:z+b-1) from the GPU to
   A^(i)(k+1:n, :) at the CPU ;
2  for j = 1, 2, ... b do
3      if j > 1 then
4          Update a^(j) only from the left on the CPU;
5      end
6      Generate the elementary reflector to annihilate a^(j) on the CPU and store it in
       a^(j)(k+j:n);
7      Copy a^(j)(k+j:n) to dV(k+j, j) on the GPU ;
8      Compute T_j(:, j) on the CPU;
9      if j mod m = 0 or j = b then
10         cMini = m or (b mod m); // current mini-block size
11         nMini = min{b-j, m}; // next mini-block size
12         Copy last cMini columns of T_j from CPU to dT_j on the GPU ;
13         Compute dY_j(k+1:n, j-cMini+1:j) on the GPU ;
14         Copy computed part of dY_j to Y_j on the CPU ;
15         Update the entire next mini-block from the right on the CPU;
16     end
17 end
```

ALGORITHM 22: GPU update for the outer loop

```
1  Using (dY, dV), apply the block reflector to update dA(k+1, n, z+b:n) from the
   right on the GPU ;
2  Using (dV, dT), apply the block reflector to update dA(k+1, n, z+b:n) from the
   left on the GPU ;
3  Copy dA(z+m:z+m+b-1, z+b:n) to A ;
```

ALGORITHM 23: CPU update for the outer loop

```
1  Compute Y(1:k, 1:b);
2  Using Y, apply the block reflector to update A(1:k, z+b:n) from the right on
   the GPU ;
```

Algorithm 24: Hybrid algorithm for the m-Hessenberg reduction of A

Input: $n \times n$ matrix A, block size b, bandwidth m
Output: A converted to m-Hessenberg form

```
1 Copy the matrix A from CPU to dA on the GPU ;
2 for z = 1, 1 + b, 1 + 2 · b, 1 + 3 · b, ... do
3 |   i = (z − 1)/b + 1; k = z + b + 1;
4 |   Process block A^(i) = A(1 : n, z : z + b − 1) using hybrid Algorithm 21 ;
5 |   Call Algorithm 23 asynchronously to compute Y(1 : k, :) and update A(1 : k, :)
  |   on the CPU from the right ;
6 |   Call Algorithm 22 to update dA(k + 1 : n, :) on the GPU from both the left
  |   and the right ;
7 end
```

5.5 Numerical experiments

We have run a series of tests in order to assess the performances of the new software. The test machines were equipped with

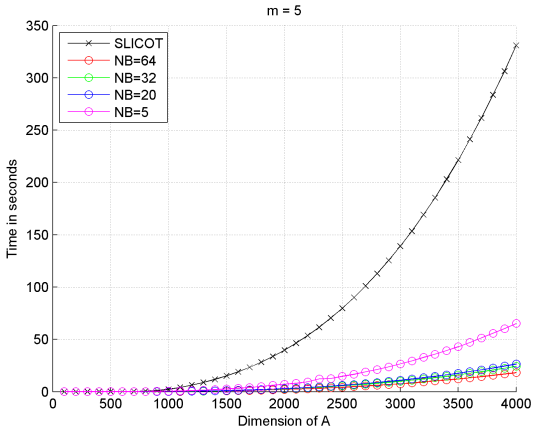
- Intel Xeon X5470 (quad-core) processor, 3.33GHz, 6+6MB of Level 2 cache memory and 8GB of RAM;
- NVIDIA GeForce GTX280 video card with 1GB of memory.

Programs were compiled on Ubuntu Linux 8.10 using:

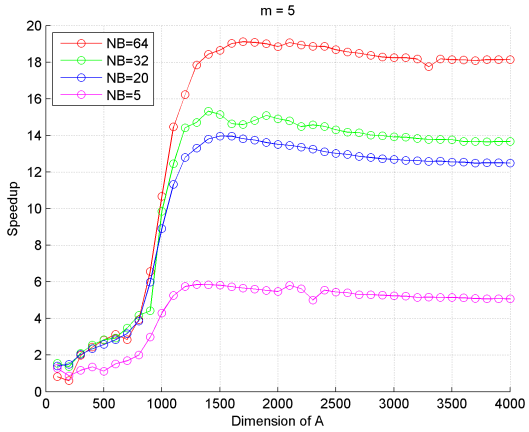
- Intel Fortran Compiler 11.0 + MKL 11.0.
- CUBLAS 3.2.
- pthreads library for multithreading.

All tests were run using the IEEE double precision arithmetic. The code was tested for correctness: if the computed reduction was $\tilde{H} \approx \tilde{Q}^\star A \tilde{Q}$, then the size of $\left\| \tilde{H} - \tilde{Q}^\star A \tilde{Q} \right\|$ was checked and the unitarity of $\tilde{Q}$ was confirmed by computing $\left\| I - \tilde{Q}^\star \tilde{Q} \right\|$. Both of these numbers are expected to be $\mathcal{O}(n\epsilon)$ for a correctly computed reduction; here $\epsilon \approx 2.22e-16$ is the *machine epsilon*.

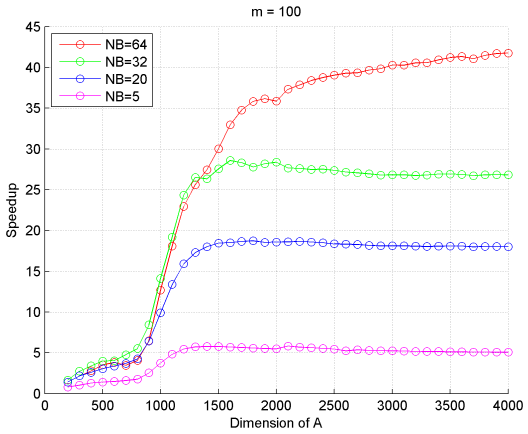
We have compared the default non-blocked implementation found in routine `TB01MD` from SLICOT with our new algorithms: blocked versions with and without the "mini-blocks" using various block sizes, and the hybrid algorithm described in the last section. Only the reduction of the matrix A to the m-Hessenberg form is tested, and the code for the QR–factorization of the matrix B is removed from the SLICOT routine, along with the transformation of the matrix A that brings the LTI system to (5.9). We note that



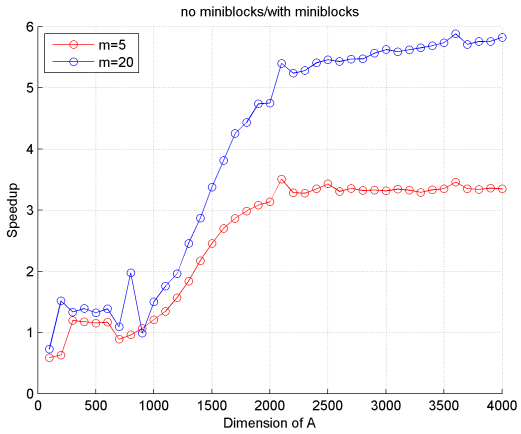
(a) Timings of TB01MD and the "mini-block" version of Algorithm 20 using various block sizes b



(b) Speedup factors relative to TB01MD; $m = 5$



(c) Speedup factors relative to TB01MD; $m = 100$



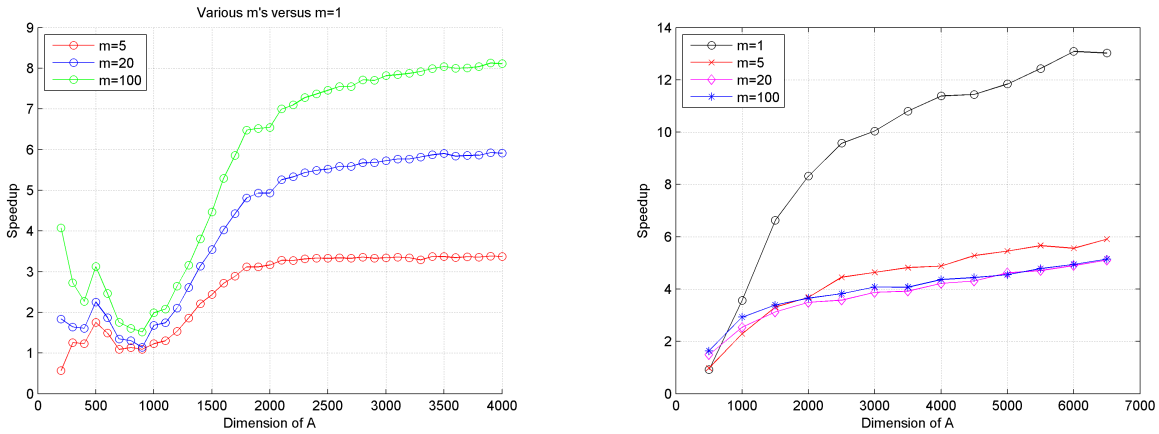
(d) The blocked algorithm: using "mini-blocks" versus no "mini-blocks"

Figure 5.7: Comparisons of the new algorithms and TB01MD from SLICOT .

the blocked implementation of these transformations would even more increase the gap between the algorithms.

Figure 5.7(a) shows the timings for the SLICOT routine and the "mini-block" version of the CPU algorithm using $m = 5$ and the block sizes of $b = 5$, 20, 32 and 64. The tests were run for a batch of matrices A with order up to 4000. We clearly see how blocking effects the performance: the larger the block size, the more efficient the usage of the fast cache memory. Ratios time(non-blocked routine)/time(blocked routine) are shown in Figure 5.7(b); we obtain speedup up to the factor of 19.

The blocking has even more effect when m is larger. Figure 5.7(c) shows that the speedup for $m = 100$ can reach a factor of up to 42.



(a) The blocked algorithm: reduction to larger band is faster

(b) Speedup factors of the hybrid algorithm relative to the algorithm using "mini-blocks"

Figure 5.8: Comparisons of the new subroutines.

The significance of using the "mini-blocks" is shown in Figure 5.7(d). Here we compare two versions of our algorithms: the one using "mini-reflectors" and "mini-blocks" and the one updating each column of the block from the right prior to its annihilation. Using the "mini-blocks", we obtain speedup up to a factor of 3.25 when $m = 5$, and up to a factor of 6 when $m = 20$. Clearly, larger m shifts this ratio even more in favor of the "mini-block" version.

A rather surprising fact is demonstrated in Figure 5.8(a), where only the "mini-block" version of our algorithm was used to generate the image. We measure how fast does it take to reduce a matrix to 5–, 20– and 100-Hessenberg form compared to the reduction to the 1-Hessenberg form. I turns out that in order to remove 3900 subdiagonals of a 4000×4000 matrix our algorithm needs only 1/8 of the time needed for the removal of 3999 subdiagonals! One could use this fact to design an algorithm for successive band removal: in order to reduce a matrix to 1-Hessenberg, it is first very quickly reduced to e.g. 100-Hessenberg, and then using a similar process this intermediate matrix is further reduced to a 1-Hessenberg form. However, attempts to design such an algorithm have failed so far: reduction from a 100-Hessenberg form to a 1-Hessenberg form simply takes too much time! In the symmetric case, there has been better success; see [12].

Finally, Figure 5.8(b) compares the hybrid algorithm to the "mini-block" version of the CPU algorithm. Speedup of up to the factor 13 is obtained for matrices of the dimension 6500 when $m = 1$. This confirms the results shown by Tomov et al. [71]. They have used 8 CPU cores and were able to completely overlap the CPU update with the block processing and the GPU update. Using only 4 cores of our test machine was not sufficient for this task, especially in the later phases of the reduction; thus a small discrepancy from the factor 15 obtained by Tomov et al.; note that they were also able to run the algorithm for matrices of dimensions up to 8000. When m is larger, the "mini-block" effect boosts the CPU-only algorithm. Nevertheless, we obtain respectful speedup factors of 5.9 when

$m = 5$, 5.11 when $m = 20$ and 5.14 when $m = 100$.

Bibliography

[1] S. Adam, P. Arbenz, and R. Geus, *Eigenvalue solvers for electromagnetic fields in cavities*, Tech. Report 275, Institute of Scientific Computing, ETH Zürich, 1997. [cited at p. 8, 9]

[2] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, third ed., 1999. [cited at p. v, 114]

[3] J. Baglama, *Augmented block Householder Arnoldi method*, Linear Algebra Appl., 429 (2008), pp. 2315–2334. [cited at p. 89, 116]

[4] ——, *ahbeigs – Augmented block Arnoldi method for computing eigenvalues of large non-hermitian matrices.* `http://www.math.uri.edu/~jbaglama/software/ahbeigs.m`, 2010. [cited at p. 89]

[5] J. Baglama, D. Calvetti, and L. Reichel, *Fast Leja points*, Electronic Transactions on Numerical Analysis, 7 (1998), pp. 124–140. [cited at p. 150, 151]

[6] J. Baglama, D. Calvetti, L. Reichel, and A. Ruttan, *Computation of a few small eigenvalues of a large matrix with application to liquid crystal modeling*, J. Comput. Phys., 146 (1998), pp. 203–226. [cited at p. 75]

[7] Z. Bai and J. W. Demmel, *On swapping diagonal blocks in real Schur form*, Linear Algebra and its Applications, 186 (1993), pp. 75–95. [cited at p. 70]

[8] C. Beattie, *Harmonic ritz and lehmann bounds*, Electronic Transactions on Numerical Analysis, 7 (1998), pp. 18–39. [cited at p. 32]

[9] C. Beattie, Z. Drmač, and S. Gugercin, *A note on shifted Hessenberg systems and frequency response computation*, tech. report, Virginia Tech and University of Zagreb, 2009. [cited at p. 121]

[10] C. Beattie, M. Embree, and D. C. Sorensen, *Convergence of polynomial restart Krylov methods for eigenvalue computations*, SIAM Review, 47 (2005), pp. 492–515. [cited at p. 54, 73, 75]

[11] M. Bellalij, Y. Saad, and H. Sadok, *On the convergence of the arnoldi process for eigenvalue problems*, SIAM Journal on Numerical Analysis, 48 (2010), pp. 393–407. [cited at p. 54]

[12] C. Bischof, B. Lang, and X. Sun, *A framework for symmetric band reduction*, ACM Transactions on Mathematical Software (TOMS), 26 (2000), pp. 581–601. [cited at p. 128, 138]

[13] N. Bosner and Z. Drmač, *Subspace gap residuals for Rayleigh–Ritz approximations*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 54–67. [cited at p. 29]

[14] S. Brin and L. Page, *The anatomy of a large-scale hypertextual web search engine*, Proceedings of the 7th international conference on World Wide Web (WWW), Brisbane, Australia, (1998), pp. 107–117. [cited at p. 12]

[15] J. J. Cullum and R. A. Willoughby, *Computing eigenvalues of very large symmetric matrices – an implementation of a Lanczos algorithms with no reorthogonalization*, J. Comput. Phys., (1981), pp. 329–358. [cited at p. 60]

[16] J. J. M. Cuppen, *A divide and conquer mathod for the symmetric eigenproblem*, Numerische Mathematik, 36 (1981), pp. 177–195. [cited at p. 35]

[17] J. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart., *Reorthogonalization and stable algorithms for updating the Gram–Schmidt QR factorization*, Mathematics of Computation, 30 (1976), pp. 772–795. [cited at p. 58]

[18] B. Datta, *Numerical methods for linear control systems*, Elsevier Academic Press, 2003. [cited at p. 80, 118]

[19] B. Datta and M. Arnold, *Single-input eigenvalue assignment algorithms: A close-look*, SIAM J. Matrix Anal. Appl., 19 (1998), pp. 447–467. [cited at p. 80]

[20] T. A. Davis and Y. Hu, *The University of Florida sparse matrix collection*, ACM Transactions on Mathematical Software (to appear). [cited at p. 14]

[21] J. W. Demmel, *A Numerical Analyst's Jordan Canonical Form*, PhD thesis, University of California, Berkeley, 1983. [cited at p. 6]

[22] J. W. Demmel, O. A. Marques, B. N. Parlett, and C. Vömel, *Performance and accuracy of LAPACK's symmetric tridiagonal eigensolvers*, SIAM J. Sci. Comput., 30 (2008), pp. 1508–1526. [cited at p. 35]

[23] J. W. Demmel and K. Veselić, *Jacobi's method is more accurate than QR*, SIAM J. Matrix Anal. Appl, 13 (1992), pp. 1204–1245. [cited at p. 35]

[24] T. Driscoll, *Schwarz–Christoffel toolbox for MATLAB*. `http://www.math.udel.edu/~driscoll/software/SC/`, 2010. [cited at p. 152]

[25] Z. Drmač and V. Hari, *Relative residual bounds for the eigenvalues of a Hermitian semidefinite matrix*, SIAM J. Matrix Anal. Appl., 18 (1997), pp. 21–29. [cited at p. 29]

[26] Z. Drmač and K. Veselić, *New fast and accurate Jacobi SVD algorithm, parts I and II*, SIAM J. Matrix Anal. Appl., 35 (2008), pp. 1322–1342, 1343–1362. [cited at p. 35]

[27] L. Elden, *Matrix methods in data mining and pattern recognition*, SIAM, 2007. [cited at p. 12]

[28] M. Embree, *The Arnoldi eigenvalue iteration with exact shifts can fail*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1–10. [cited at p. iv, 72, 91, 109]

[29] J. G. F. Francis, *The QR transformation, parts I and II*, Computer Journal, 4 (1961, 1962), pp. 265–271, 332–345. [cited at p. 35]

[30] D. Gaier, *Lectures on Complex Approximation*, Birkhäuser, 1980. [cited at p. 152]

[31] G. Golub and J. Wilkinson, *Ill–conditioned eigensystems and the computations of the Jordan canonical form*, SIAM Review, 18 (1976), pp. 578–619. [cited at p. 6]

[32] Intel, *Intel microprocessor export compliance metrics*. `http://www.intel.com/support/processors/sb/CS-023143.htm`, 2010. [cited at p. 131]

[33] C. G. J. Jacobi, *Über ein leichtes Verfahren die in der Theorie der Säkularstörungen vorkommenden Gleichungen numerisch aufzulösen*, Journal für die reine und angewandte Mathematik, (1846), pp. 51–94. [cited at p. iii, 35]

[34] Z. Jia, *The convergence of harmonic Ritz values, harmonic Ritz vectors, and refined harmonic Ritz vectors*, Mathematics of Computation, 74 (2004), pp. 1441–1456. [cited at p. 32]

[35] A. Knyazev, *Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method*, SIAM Journal on Scientific Computing, 23 (2001), pp. 517–541. [cited at p. 36]

[36] D. Kressner, *Private communication*, January 2010. [cited at p. 128]

[37] V. N. Kublanovskaya, *On some algorithms for the solution of the complete eigenvalue problem*, USSR Computational Mathematics and Mathematical Physics, 1 (1961), pp. 637–657. [cited at p. 35]

[38] M. Kvasnica, P. Grieder, and M. Baotić, *Multi-parametric toolbox (MPT)*. `http://control.ee.ethz.ch/~mpt/`, 2004. [cited at p. 105]

[39] A. Laub and A. Linnemann, *Hessenberg and Hessenberg/triangular forms in linear system theory*, International Journal of Control, 6 (1986), pp. 1523–1547. [cited at p. 118]

[40] R. Lehoucq and K. Maschhoff, *Implicitly restarted Arnoldi methods and subspace iteration*. Preprint MCS-P649-0297, Argonne National Laboratory, 1997. [cited at p. 116]

[41] R. Lehoucq, K. Maschhoff, D. Sorensen, and C. Yang, *ARPACK – Solution of large scale eigenvalue problems with implicitly restarted Arnoldi methods*. `http://www.caam.rice.edu/software/ARPACK/`, 1997. [cited at p. iv]

[42] R. Lehoucq and D. Sorensen, *Deflation techniques for an implicitly restarted Arnoldi iteration*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 789–821. [cited at p. 67]

[43] S. M. Malamud, *Inverse spectral problem for normal matrices and the Gauss–Lucas theorem*, Transactions of the American Mathematical Society, 357 (2005), pp. 4043–4064. [cited at p. 111]

[44] MatrixMarket, *A collection of test matrices*. `http://math.nist.gov/MatrixMarket/`, 2007. [cited at p. 59, 83, 86]

[45] V. Mehrmann and H. Xu, *An analysis of the pole placement problem: I. the single-input case*, Electronic Transactions on Numerical Analysis, 4 (1996), pp. 89–105. [cited at p. 81]

[46] J. Möller, *Implementations of the implicitly restarted block Arnoldi method*, tech. report, Royal Institute of Technology (KTH), Dept. of Numerical Analysis and Computer Science, 2004. TRITA-NA-0446. [cited at p. 116]

[47] R. Morgan and M. Zeng, *A harmonic restarted Arnoldi algorithm for calculating eigenvalues and determining multiplicity*, Linear Algebra and its Applications, 415 (2006), pp. 96–113. [cited at p. 75]

[48] NVIDIA, *Compute unified device architecture*. `http://www.nvidia.com/object/cuda_home_new.html`, 2006. [cited at p. 131]

[49] ——, *GeForce GTX 280 video card*. `http://www.nvidia.com/object/product_geforce_gtx_280_us.html`, 2009. [cited at p. 131]

[50] ——, *CUBLAS User's guide*. `http://developer.download.nvidia.com/compute/cuda/3_2_prod/toolkit/docs/CUBLAS_Library.pdf`, 2010. [cited at p. 131]

[51] C. Paige, *The computation of eigenvalues and eigenvectors of very large sparse matrices*, PhD thesis, Univ. of London, 1971. [cited at p. 60]

[52] B. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, 1980. [cited at p. 49, 106]

[53] B. Parlett and J. Le, *Forward instability of tridiagonal QR*, SIAM J. Matrix Anal. Appl., 14 (1993), pp. 279–316. [cited at p. 67]

[54] B. Parlett and D. Scott, *The Lanczos algorithm with selective reorthogonalization*, Math. Comp., 33 (1979), pp. 217–238. [cited at p. 61]

[55] R. Plato, *Concise Numerical Mathematics*, AMS, Graduate Studies in Mathematics, 2003. [cited at p. 150]

[56] J. F. Queir and A. L. Duarte, *Imbedding conditions for normal matrices*, Linear Algebra and its Applications, 430 (2009), pp. 1806 – 1811. [cited at p. 92]

[57] S. Roman, *Advanced Linear Algebra*, Springer, 2008. [cited at p. 104]

[58] Y. Saad, *Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems*, Mathematics of Computation, 42 (1984), pp. 567–588. [cited at p. 75]

[59] ——, *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press, 1992. [cited at p. 38, 55, 150]

[60] R. Schreiber and C. van Loan, *A storage–efficient WY representation for products of Householder transformations*, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 53–57. [cited at p. 122]

[61] H. D. Simon, *Analysis of the symmetric Lanczos algorithm with reorthogonalization methods*, Linear Algebra and Its Applications, 61 (1984), pp. 101–131. [cited at p. 61]

[62] G. L. G. Sleijpen and J. van den Eshof, *On the use of harmonic ritz pairs in approximating internal eigenpairs*, Linear Algebra and its Applications, 358 (2003), pp. 115–137. [cited at p. 32]

[63] G. L. G. Sleijpen and H. A. V. D. Vorst, *A Jacobi-Davidson iteration method for linear eigenvalue problems*, SIAM J. Matrix Anal. Appl, 17 (2000), pp. 401–425. [cited at p. 31, 35]

[64] SLICOT, *The control and systems library.* `http://www.slicot.org/`, 2009. [cited at p. v, 114, 119]

[65] D. C. Sorensen, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385. [cited at p. iv, 64, 72, 91]

[66] ——, *Deflation for implicitly restarted Arnoldi methods*, Tech. Report TR98-12, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 1998. [cited at p. 67]

[67] G. W. Stewart, *A Krylov–Schur algorithm for large eigenproblems*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 601–614. [cited at p. iv, 69, 71, 75]

[68] ——, *Matrix Algorithms: Volume II: Eigensystems*, SIAM, 2001. [cited at p. 6, 30, 54]

[69] ——, *Addendum to "A Krylov–Schur algorithm for large eigenproblems"*, SIAM J. Matrix Anal. Appl., 24 (2002), pp. 599–601. [cited at p. 75, 81]

[70] G. W. Stewart and J. guang Sun, *Matrix Perturbation Theory*, Academic Press, 1990. [cited at p. 4, 16, 25]

[71] S. Tomov, R. Nath, and J. Dongarra, *Accelerating the reduction to upper Hessenberg, tridiagonal, and bidiagonal forms through hybrid GPU-based computing*, Parallel Computing, 36 (2010), pp. 645–654. [cited at p. v, 132, 138]

[72] top500, *Top 500 supercomputer sites*. `http://www.top500.org/`, 2010. [cited at p. 131]

[73] W. F. Trench and P. A. Scheinok, *On the inversion of a Hilbert type matrix*, SIAM Review, 8 (1966), pp. 57–61. [cited at p. 101]

[74] A. van der Sluis, *Condition numbers and equilibration of matrices*, Numer. Math., 14 (1969), pp. 14–23. [cited at p. 23]

[75] C. van Loan, *Using the Hessenberg decomposition in control theory*, Mathematical Programming Study, (1982), pp. 102–111. [cited at p. 118]

[76] V. Volkov and J. W. Demmel, *Benchmarking GPUs to tune dense linear algebra*, in Proceedings of the 2008 ACM/IEEE conference on Supercomputing, IEEE Press, 2008, pp. 31:1–31:11. [cited at p. 131]

[77] D. Watkins, *A case where balancing is harmful*, Electronic Transactions on Numerical Analysis, 23 (2006), pp. 1–4. [cited at p. 86]

[78] ——, *The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods*, SIAM, 2007. [cited at p. 42, 49]

[79] Wikipedia, *GeForce 200 series*. `http://en.wikipedia.org/wiki/GeForce_200_Series`, 2010. [cited at p. 131]

[80] G. Wu, *A modified harmonic block Arnoldi algorithm with adaptive shifts for large interior eigenproblems*, J. Comput. Appl. Math., 205 (2007), pp. 343–363. [cited at p. 75]

Appendices

Appendix A

Shift selection in the restarted Arnoldi algorithm

This Appendix gives a brief overview of various methods for choosing the shifts in the restarted Arnoldi algorithm. The motivation for defining a particular set of points in the complex plane and using it as a set of shifts stems from the polynomial optimization problems and their approximate solutions.

A.1 Chebyshev polynomials

Definition A.1. *For $k = 0, 1, 2, \ldots$ and $t \in \mathbb{R}$ define a function $T_k : \mathbb{R} \to \mathbb{R}$ as*

$$T_k(t) = \begin{cases} \cos(k \arccos t), & -1 \leq t \leq 1 \\ \cosh(k \operatorname{arccosh} t), & |t| > 1 \end{cases}$$

Function T_k is a k–th Chebyshev polynomial of the first kind.

That T_k is indeed a polynomial of degree k is easily seen from the three–term recurrence:

$$\begin{aligned} T_0(t) &= 1; \\ T_1(t) &= t; \\ T_k(t) &= 2t \cdot T_{k-1}(t) - T_{k-2}(t), \text{ for } k = 2, 3, \ldots \end{aligned}$$

This recurrence is used for a fast computation of T_k at a given point t. Furthermore, the recurrence is used to expand the definition of a Chebyshev polynomial to a function of a complex variable.

The polynomial T_k has roots at

$$x_j = \frac{\cos(2j+1)\pi}{2n} \in \langle -1, 1 \rangle, \quad j = 0, 1, \ldots, k-1.$$

The maximum absolute value T_k reaches at $[-1, 1]$ is equal to 1. Departing the $[-1, 1]$ interval, the value of $T_k(x)$ increases very quickly:

$$T_k(t) = \frac{1}{2}\left((t + \sqrt{t^2 - 1})^k + (t + \sqrt{t^2 - 1})^{-k}\right), \quad \text{for } |t| > 1.$$

We can approximate this as

$$T_k(1 + 2\epsilon) \approx \frac{1}{2} e^{2k\sqrt{\epsilon}}, \qquad \text{for } k\sqrt{\epsilon} > 1.$$

The basic optimality property of the Chebyshev polynomials is the main tool that provides convergence results for the Krylov subspace methods, at least in the Hermitian case.

Theorem A.2. *Let $[\alpha, \beta] \subseteq \mathbb{R}$ denote a non–empty interval and let $\gamma > \beta$. Then*

$$\min_{\substack{\pi \in \mathbb{P}_k \\ \pi(\gamma)=1}} \max_{t \in [\alpha, \beta]} |\pi(t)| = \frac{1}{\left|T_k\left(\frac{\gamma-\mu}{\nu}\right)\right|},$$

where $\mu = \frac{\alpha+\beta}{2}$, $\nu = \frac{\beta-\alpha}{2}$. The minimum is obtained for the polynomial

$$\hat{T}_k(t) = \frac{T_k\left(\frac{t-\mu}{\nu}\right)}{T_k\left(\frac{\gamma-\mu}{\nu}\right)}.$$

See e.g. [55] for a proof. In the complex case, the asymptotic optimality for Chebyshev polynomials can be shown when interval from the Theorem is replaced by an ellipse. Let $E(c, d)$ be an ellipse with center at c and focal distance d and $\gamma \notin E(c, d)$. Then (see [59]), for large k, the polynomial

$$z \mapsto T_k\left(\frac{z - c}{d}\right)$$

is close to a solution of the optimization problem

$$\min_{\substack{\pi \in \mathbb{P}_k \\ \pi(\gamma)=1}} \max_{z \in E(c,d)} |\pi(z)|.$$

In the setting of the restarted Arnoldi algorithm, suppose that the unwanted part of the spectrum can be enclosed within an ellipse. The result above then justifies the use of the roots of the Chebyshev polynomials as the shifts in the restarted Arnoldi algorithm.

A.2 Leja points

Unlike the Chebyshev points, the Leja points aim to solve a polynomial optimization problem for a more general subset of the complex plane. We follow the definitions from [5].

Definition A.3. *Let Ω be a compact subset of $\mathbb{C}$ such that $\mathbb{C} \setminus \Omega$ is connected and let $w : \Omega \to [0, +\infty\rangle$ denote a weight function. (Weighted) Leja points are elements of the sequence $(z_n)_{n=0}^{\infty}$, $z_n \in \Omega$, defined by the following recursive property:*

$$\begin{aligned} w(z_0)|z_0| &= \max_{z\in\Omega} w(z)|z|; \\ w(z_n)\prod_{k=0}^{n-1}|z_n - z_k| &= \max_{z\in\Omega} w(z)\prod_{k=0}^{n-1}|z - z_k|. \end{aligned}$$

Note that the sequence in the definition is not necessarily unique for a given Ω; any such sequence is called a sequence of Leja points. When the weight function is $w(z) \equiv 1$, then the maximum principle for analytical functions implies that all Leja points fall onto the boundary of the set Ω. One can show that the distribution of Leja points is uniform on the boundary of Ω with respect to the density function

$$\frac{1}{2\pi}\frac{\partial G(x,y)}{\partial n}, \quad z = x + yi \in \partial\Omega.$$

Here $G(x, y)$ is the Green function for $\mathbb{C} \setminus \Omega$ with logarithmic singularity at infinity, and $\partial/\partial n$ is a normal derivative to the boundary. For example, Leja points for a disc are uniformly distributed on its circumference. Leja points for an interval $[\alpha, \beta]$ have the same limit distribution as the roots of the Chebyshev polynomials do.

Solving the optimization problem given in the Definition is not a computationally simple task. Instead of computing the maximum over the whole (boundary of) Ω, Baglama et al. [5] suggest a discretized version resulting in computation of the *fast Leja points*.

Suppose $\partial\Omega$ is a closed Jordan curve, and let $t : [0, 1] \to \partial\Omega$ be a continuous parametric representation, $t(0) = t(1)$. Define $\tilde{z}_0 = t(0)$, $\tilde{z}_1 = t(0.5)$ to be the first two fast Leja points. Let S denote a set of candidate points; at first let $S = \{t(0.25), t(0.75)\}$ – note that there is exactly one candidate point between two fast Leja points. The next fast Leja point $\tilde{z}_n$, $n = 2$, is chosen so that

$$w(\tilde{z}_n)\prod_{k=0}^{n-1}|\tilde{z}_n - \tilde{z}_k| = \max_{s\in S} w(s)\prod_{k=0}^{n-1}|s - \tilde{z}_k|. \tag{A.1}$$

This only requires a simple loop through all elements of S. The computed point $\tilde{z}_n = t(\alpha_n)$ is removed from the set of candidate points; it has two adjacent fast Leja points $\tilde{z}_p = t(\alpha_p)$ and $\tilde{z}_q = t(\alpha_q)$. Two new elements are added to the set of candidate points:

$$S \mapsto S \cup \left\{ t\left(\frac{\alpha_n + \alpha_p}{2}\right), t\left(\frac{\alpha_n + \alpha_q}{2}\right) \right\}.$$

This keeps the property of exactly one candidate point between two fast Leja points, so we can proceed once again with solving (A.1) to compute another point.

Numerical examples show that fast Leja points inherit the good properties of Leja points. This allows for their use in polynomial interpolation or as shifts in the restarted Arnoldi algorithm.

A.3 Fejer points

Another way of computing well–distributed points along the boundary of a given compact set $\Omega \subseteq \mathbb{C}$ is given by the construction of *Fejer points*.

Definition A.4. *Let Ω be a compact subset of $\mathbb{C}$ such that $\mathbb{C} \setminus \Omega$ is connected. Then there exists a conformal mapping*

$$\Psi : \{z \ : \ |z| > 1\} \to \mathbb{C} \setminus \Omega.$$

Suppose that Ψ has a continuous extension to $\{z \ : \ |z| \geq 1\}$. For an integer $n \geq 1$, the points

$$f_k = \Psi\left(e^{2\pi i k/(n+1)}\right), \quad k = 0, 1, \ldots, n,$$

are called the Fejer points of order n on Ω.

The Fejer points are uniformly distributed on $\partial\Omega$; this and other theoretical properties are studied in [30]. To compute the Fejer points, one has to construct the conformal mapping Ψ. The Schwarz-Christoffel toolbox for MATLAB [24] is a very good software for such a purpose.

Summary

This thesis is devoted to the large scale eigenvalue problem, in particular to the Arnoldi algorithm for computing a few eigenvalues of large matrices.

It is shown how the Krylov–Schur restarting method can be used with any choice of shifts. On the other hand, the connection with the pole placement problem is made, and this problem is known to generally be ill conditioned.

Next, geometry of the Ritz values for normal matrices is studied; this is an important problem in convergence theory of the Arnoldi algorithm restarted using the exact shifts. The necessary and the sufficient condition for a given set of k complex numbers to appear as a set of Ritz values from a Krylov subspace is shown to be the existence of a positive solution to a linear system with a Cauchy matrix. This fact is used for derivation of simple proofs for some known facts for the Ritz values. An example of a normal matrix for which the restarted Arnoldi algorithm fails to compute the second largest eigenvalue is constructed. Also a variant of the Cauchy interlacing lemma is shown to hold in the setting of normal matrices.

Finally, a new blocked algorithm for the reduction of a matrix to the m-Hessenberg form is presented. This algorithm is superior to the existing implementation in the SLICOT software library. A variant of the algorithm that uses hybrid CPU+GPU computing is derived, exhibiting even higher performance.

Sažetak

Tema ove disertacije je problem svojstvenih vrijednosti za matrice velike dimenzije, a posebno Arnoldijev algoritam za izračunavanje nekoliko svojstvenih parova velikih matrica.

Pokazano je kako Krilov–Schurova metoda restartanja može biti upotrijebljena s bilo kojim izborom pomaka. S druge strane, uspostavljena je veza između ovakvog restarta i *pole placement* problema; poznato je da je taj problem općenito loše uvjetovan.

Nadalje, promatrana je geometrija Ritzovih vrijednosti za normalne matrice; ona predstavlja važan problem u teoriji konvergencije Arnoldijevog algoritma koji prilikom restarta koristi egzaktne pomake. Pokazano je da je nužan i dovoljan uvjet da bi skup k kompleksnih brojeva mogao biti skup Ritzovih vrijednosti iz Krilovljevog potprostora egzistencija pozitivnog rješenja jednog linearnog sustava sa Cauchyevom matricom. Ova činjenica je iskorištena za izvod jednostavnih dokaza nekih poznatih činjenica o Ritzovim vrijednostima. Konstruiran je primjer normalne matrice za koju restartani Arnoldijev algoritam ne uspijeva izračunati drugu po veličini svojstvenu vrijednost. Također, pokazano je da varijanta Cauchyeve leme o ispreplitanju vrijedi i za normalne matrice.

Konačno, izveden je novi blok-algoritam za redukciju matrice na m-Hessenbergovu formu. Ovaj algoritam je superioran postojećoj implementaciji u biblioteci SLICOT . Napravljena je i varijanta algoritma koja koristi hibridno CPU+GPU računanje. Ta varijanta pokazuje još bolje performanse.

Curriculum Vitae

Personal Information:

- born on June 10, 1980 in Zagreb, Croatia

Education:

- January 2003 – March 2007, M.Sc. in Mathematics, Department of Mathematics, University of Zagreb; thesis title: *"Numerical Analysis of the Large Scale Eigenvalue Problem"* (in Croatian), supervisor: prof. Zlatko Drmač
- October 1998 – December 2002, B.Sc. in Mathematics, Department of Mathematics, University of Zagreb; thesis title: *"Singular Value Decomposition"* (in Croatian), supervisor: prof. Zlatko Drmač
- September 1994 – June 1998, XV. Gimnazija (Highschool for mathematics and natural sciences), Zagreb
- September 1986 – June 1994, primary school "Bartol Kašić", Zagreb

Work Experience:

- January 2003 – , teaching and research assistant, Department of Mathematics, University of Zagreb

Publications:

- Z. Drmač, Z.Bujanović: *"On the Failure of Rank Revealing QR Factorization Software – A Case Study"*, ACM TOMS, Vol. 35, Number 2, 2008.

Conferences:

- Parallel Matrix Algorithms and Applications 2010 (Basel, Switzerland): *"A Hybrid m-Hessenberg Reduction Algorithm"*
- IWASEP 7 (Dubrovnik, Croatia, 2008): *"On the Convergence of Block Kogbetliantz Methods for Calculating the SVD"* (coauthor: Z. Drmač)

- ApplMath07 (Brijuni, Croatia, 2007): *"On the Failure of Rank Revealing QR Factorization Software"* (coauthor: Z. Drmač)
- IWASEP 5 (Hagen, Germany, 2004): *"Implementation of the New Jacobi SVD Algorithm for Complex Matrices"* (poster section)

Summer Schools:

- Gene Golub SIAM Summer School 2010 (Selva di Fasano, Italy)
- SIAG/LA-SIMUMAT International Summer School on Numerical Linear Algebra (Castro Urdiales, Spain, 2008)

Awards:

- Rector's Award by the rector of University of Zagreb in 2002; thesis title: *"Numerical Simulation of the Hyperbolic Conservation Law – Shallow Water Equations"* (in Croatian), with M. Botinčan
- Scholarship of City of Zagreb, during highschool and university education
- bronze medals at International Olympiad in Informatics, 1997 and 1998
- participated at International Mathematical Olympiad, 1998