Course Material
NetworkMaths Graduate Programme

Maynooth 2010

Numerical Linear Algebra

Volker Mehrmann,
TU Berlin,

August 3, 2010

Literature

The material of this course is based on the following textbooks:

- G. Golub, C. Van Loan. *Matrix computations*. Baltimore, 1996.
- R. B. Lehoucq, D. C. Sorensen and C. Yang. *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, Philadelphia, 1989.
- Y. Saad. *Numerical methods for large eigenvalue problems*. Manchester, 1992.
- L. Trefethen, D. Bau. *Numerical linear algebra*. Philadelphia, 1997.
- D. Watkins. *Fundamentals of matrix computations*. New York, 2002.

The following books are also useful to complement the material of these notes:

- J. Demmel. *Applied numerical linear algebra*. Philadelphia, 1997.
- N.J. Higham. *Accuracy and stability of numerical algorithms*. Philadelphia, 2002.
- G.W. Stewart. *Matrix algorithms*. Philadelphia, 1998-2001, 2 Volumes.
- G.W. Stewart, J.G. Sun. *Matrix perturbation theory*. Boston, 1990.

Chapter 0

Introduction

The main topics of *Numerical Linear Algebra* are the solution of different classes of *eigenvalue problems* and *linear systems*.
For the eigenvalue problem we discuss different classes.

(a) The *standard eigenvalue problem*: For a real or complex matrix $A \in \mathbb{C}^{n,n}$, determine $x \in \mathbb{C}^n, \lambda \in \mathbb{C}$, such that

$$Ax = \lambda x.$$

The standard eigenvalue problem is a special case of the *generalized eigenvalue problem*: For real or complex matrices $A, E \in \mathbb{C}^{m,n}$, determine $x \in \mathbb{C}^n, \lambda \in \mathbb{C}$, such that

$$Ax = \lambda Ex,$$

In many applications the coefficient matrices have extra properties such as being real and symmetric or complex Hermitian.
For linear systems:

$$Ax = b, x \in \mathbb{C}^n, b \in \mathbb{C}^m$$

with $A \in \mathbb{C}^{m,n}$ we again may extra properties for the coefficient matrices.
We will concentrate in this course on the numerical solution of standard and generalized eigenvalue problems and the solution of linear systems. We will briefly review some of the standard techniques for small scale problems and put an emphasis on large scale problems.
Applications: Eigenvalue problems arise in

- the vibrational analysis of structures and vehicles (classical mechanics);
- the analysis of the spectra and energy levels of atoms and molecules (quantum mechanics);
- model reduction techniques, where a large scale model is reduced to a small scale model by leaving out weakly important parts;
- many other applications.

Linear systems arise in almost any area of science and engineering such as

(a) frequency response analysis for excited structures and vehicles;

(b) finite element methods or finite difference methods for ordinary and partial differential equations;

(c) data mining, information retrieval;

(d) and many others.

We will distinguish between small and medium class problems where the full matrices fit into main memory, these are of today sizes $n = 10^2 - 10^5$ and large sparse problems, where the coefficient matrices are stored in sparse formats, and have sizes $n = 10^6$ and larger. We will mainly discuss the case of complex matrices. Many results hold equally well in the real case, but often the presentation becomes more clumsy. We will point out when the real case is substantially different.
We will discuss the following algorithms.

	A small	A large
EVP	QR-Algorithm, QZ-Algorithm	Lanczos, Arnoldi, Jacobi-Davidson
LS		CG, GMRES

Chapter 1

Matrix theory

1.1 Basics

1.1.1 Eigenvalues and Eigenvectors

Let $A, E \in \mathbb{C}^{n,n}$, then $v \in \mathbb{C}^n \setminus \{0\}$ and $\lambda \in \mathbb{C}$ that satisfy

$$Av = \lambda Ev$$

are called *eigenvector and eigenvalue* of the pair (E, A). In the special case that E is the $n \times n$ identity matrix $I_n(I)$ we have eigenvalues and eigenvectors of the standard eigenvalue problem.
The sets

$$\begin{aligned} \sigma(A) &:= \{\lambda \in \mathbb{C} |\ \lambda \text{ eigenvalue of } A\} \\ \sigma(E, A) &:= \{\lambda \in \mathbb{C} |\ \lambda \text{ eigenvalue of } (E, A)\} \end{aligned}$$

are called *spectrum of A (the pair (E, A))*, respectively.

1.1.2 Matrix norms

Let $A \in \mathbb{C}^{m,n}$, then

$$||A||_p := \sup_{x \neq 0} \frac{||Ax||_p}{||x||_p}$$

is the *matrix p-norm*, $p \in \mathbb{N} \cup \{\infty\}$ and for invertible matrices A

$$\kappa_p(A) := ||A||_p \cdot ||A^{-1}||_p$$

is called the *p-norm condition number of A*.

Special cases:

(a) $p = 1 \rightsquigarrow$ the column-sum norm:

$$||A||_1 = \max_j \sum_{i=1}^{m} |a_{ij}|$$

(b) $p = \infty \rightsquigarrow$ the row-sum norm:

$$||A||_\infty = ||A^T||_1$$

(c) $p = 2 \rightsquigarrow$ the spectral norm

$$||A||_2 = \text{square root of the largest eigenvalue of } A^H A$$

with $A^H = \bar{A}^T$.

Convention:

$$||A|| = ||A||_2, \kappa(A) = \kappa_2(A)$$

Frobenius norm:

$$||A||_F = \sqrt{\sum_{j=1}^{n} \sum_{i=1}^{m} |a_{ij}|^2}$$

1.1.3 Isometric and unitary matrices

Definition 1 *Let $U \in \mathbb{C}^{m,n}, m \geq n$.*

(a) U is called isometric *if $U^H U = I_k$;*

(b) U is called unitary *if U is isometric and $n = k$.*

Theorem 2 *Let $U \in \mathbb{C}^{n \times k}, k \leq n$. Then the following are equivalent.*

(a) U is isometric;

(b) the columns of U are orthonormal;

(c) $\langle Ux, Uy \rangle = \langle x, y \rangle$ for all $x, y \in \mathbb{C}^k$ ($\langle \cdot, \cdot \rangle$: standard real or complex scalar product);

(d) $||Ux|| = ||x||$ for all $x \in \mathbb{C}^k$;

For $k = n$, (a)-(d) are equivalent to

(e) $UU^H = I_n$;

(f) $U^{-1} = U^H$;

(g) the rows of U are orthonormal.

In this case, furthermore,

$$||U|| = 1 = ||U^{-1}|| = \kappa(U).$$

1.1.4 Subspaces

Definition 3 *A space $\mathcal{U} \subset \mathbb{C}^n$ is called subspace, if for all $x, y \in \mathcal{U}, \alpha \in \mathbb{C}$ we have*

$$x + y \in \mathcal{U},\ \alpha x \in \mathcal{U}.$$

Theorem 4 *Let $\mathcal{U} \subset \mathbb{C}^n$ be a subspace with basis $(x_1, \ldots, x_m)$ and $X = [x_1, \ldots, x_m]$, i.e. Rank$(X) = m$.*

(a) Then $\mathcal{U} = \mathcal{R}(X) := \{Xy |\ y \in \mathbb{C}^m\}$ (Range or column space of X).

(b) Let $Y \in \mathbb{C}^{n,m}$ with Rank$(Y) = m$, then

$$\mathcal{R}(X) = \mathcal{R}(Y) \Leftrightarrow X = YB,\ B \in \mathbb{C}^{m,m}.$$

In particular then B is invertible and

$$XB^{-1} = Y.$$

(c) The Gram-Schmidt method for $(x_1, \ldots, x_m)$ delivers an orthonormal basis $(q_1, \ldots, q_m)$ of $\mathcal{U}$ with

$$Span\{q_1, \ldots, q_j\} = Span\{x_1, \ldots, x_j\}$$

for $j = 1, \ldots, m$. This condition is equivalent to:
There exists an upper triangular matrix $R \in \mathbb{C}^{m,m}$ with $X = QR$ where $Q = [q_1, \ldots, q_m]$. (QR-decomposition)

1.1.5 Invariant subspaces

Definition 5 *Let $A \in \mathbb{C}^{n,n}$ and $\mathcal{U} \subset \mathbb{C}^n$. Then $\mathcal{U}$ is called A-*invariant*, if*

$$x \in \mathcal{U} \Rightarrow Ax \in \mathcal{U} \quad \text{for all } x \in \mathbb{C}^n.$$

Theorem 6 *Let $A \in \mathbb{C}^{n,n}, X \in \mathbb{C}^{n,n}$ and $\mathcal{U} = \mathcal{R}(X)$. Then the following are equivalent:*

(a) $\mathcal{U}$ is A-invariant;

(b) There exists $B \in \mathbb{C}^{k,k}$, such that:

$$AX = XB.$$

Furthermore, in this case for $\lambda \in \mathbb{C}$ and $v \in \mathbb{C}^k$:

$$Bv = \lambda v \Rightarrow AXv = \lambda Xv,$$

i.e., every eigenvalue von B is also an eigenvalue von A.

Remark 7 *If A, X, B satisfy $AX = XB$ and if X has only one column x, then B is a scalar λ and we obtain the eigenvalue equation*

$$Ax = x\lambda,$$

i.e., X can be viewed as a generalization of the concept of eigenvector.

1.2 Matrix decompositions

1.2.1 Schur decomposition

Theorem 8 (Schur, 1909)
Let $A \in \mathbb{C}^{n,n}$. Then there exists $U \in \mathbb{C}^{n,n}$ unitary such that

$$T := U^H A U$$

is upper triangular.

Proof: By induction: $n = 1$ is trivial.
"$n-1 \Rightarrow n$": Let $v \in \mathbb{C}^n$ be an eigenvector of A to the eigenvalue $\lambda \in \mathbb{C}$. Let $q_1 := \frac{v}{||v||}$ and complete q_1 to an orthonormal basis $(q_1, \ldots, q_n)$ of $\mathbb{C}^n$. Then $Q = [q_1, \ldots, q_m]$ is unitary and

$$Q^{-1} A Q = \left[\begin{array}{c|c} \lambda & A_{12} \\ \hline 0 & A_{22} \end{array}\right]$$

By the inductive assumption there exists U_{22} unitary, such that $T_{22} := U_{22}^H A_{22} U_{22}$ is upper triangular. Setting

$$U = Q \left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & U_{22} \end{array}\right],$$

then $T = U^H A U$ is upper triangular. □

Remark 9 *In the Schur decomposition U can be chosen such that the eigenvalues of A appear in arbitrary order on the diagonal.*

Definition 10 *A matrix $T \in \mathbb{R}^{n,n}$ is called* quasi-upper triangular matrix*, if T is a block-upper triangular matrix and the diagonal blocks have size maximal 2×2.*

Theorem 11 (Murnaghan, Wintner, 1931)
Let $A \in \mathbb{R}^{n,n}$. Then there exists a real orthogonal matrix Q, (i.e., $Q^T Q = I$) such that

$$T = Q^T A Q$$

is quasi-upper triangular.

Proof: The proof is similar to that of the Theorem of Schur: If A has a real eigenvector then we can proceed the induction as in the complex case. Otherwise for a complex eigenvector $v = v_1 + i v_2$ to the complex eigenvalue $\lambda = \lambda_1 + i\lambda_2$, with $v_1, v_2 \in \mathbb{R}^n$ and $\lambda_1, \lambda_2 \in \mathbb{R}, \lambda_2 \neq 0$ we have from

$$A\,(v_1 + i v_2) = (\lambda_1 + i\lambda_2)\,(v_1 + i v_2)$$

that

$$\left.\begin{array}{rcl} A v_1 &=& \lambda_1 v_1 - \lambda_2 v_2 \\ A v_2 &=& \lambda_1 v_2 + \lambda_2 v_1 \end{array}\right\} \Rightarrow A\,[v_1\, v_2] = [v_1\, v_2] \left[\begin{array}{cc} \lambda_1 & \lambda_2 \\ -\lambda_2 & \lambda_1 \end{array}\right].$$

Hence Span $\{v_1, v_2\}$ is an A-invariant subspace. Let (q_1, q_2) be an orthonormal basis of Span $\{v_1, v_2\}$ and let $Q = [q_1, q_2, q_3, \ldots, q_n]$ be orthogonal, then

$$Q^H A Q = \left[\begin{array}{cc|c} \lambda_1 & \lambda_2 & \\ -\lambda_2 & \lambda_1 & A_{12} \\ \hline 0 & & A_{22} \end{array}\right].$$

The remaining steps of the induction are as in the complex case. □

Definition 12 *$A \in \mathbb{C}^{n,n}$ is called* normal *if $AA^H = A^H A$.*

Example 13 *Hermitian matrices, skew-Hermitian matrices and unitary matrices are normal.*

Theorem 14 *Let $A \in \mathbb{C}^{n,n}$ be normal and $U \in \mathbb{C}^{n,n}$ unitary such that*

$$U^H A U = \left[\begin{array}{cc} A_{11} & A_{12} \\ 0 & A_{22} \end{array}\right],$$

then $A_{12} = 0$.

Proof: Exercise. □

Corollary 15 *If $A \in \mathbb{C}^{n,n}$ is normal, then there exists $U \in \mathbb{C}^{n,n}$ unitary, such that $U^H A U$ is diagonal.*

Proof: Exercise. □

Theorem 16 (Generalized Schur form) *Let $A, E \in \mathbb{C}^{n,n}$ be such that the pair (E, A) is* regular*, i.e., $\det \lambda E - A \neq 0$ for all $\lambda \in \mathbb{C}$. Then there exist $U, V \in \mathbb{C}^{n,n}$ unitary such that*

$$S = U^H E V, \; T := U^H A V$$

are upper triangular.

Proof: Let (E_k) be a sequence of nonsingular matrices that converges to E. For every let

$$Q_k^H A E_k^{-1} Q_k = T_k$$

be a Schur decomposition of AE_k^{-1} and let $Z_k^H (E_k^{-1} Q_k = S_k^{-1}$ be a QR decomposition. Then both $Q_k^H A Z_k = R_k S_k$ and $Q_k^H E_k Z_k = S_k$ are upper triangular.
Using the Bolzano-Weierstraß it follows that the bounded sequence (Q_k, Z_k) has a converging subsequence with a limit (Q, Z), where Q, Z are unitary. Then $Q^H A Z = T$ and $Q^H E Z = S$ are upper triangular. □

1.2.2 The singular value decomposition (SVD)

Theorem 17 (Singular value decomposition, SVD)
Let $A \in \mathbb{C}^{m,n}$ with Rank $(A) = r$. *Then there exist unitary matrices $U \in \mathbb{C}^{m,m}$ and $V \in \mathbb{C}^{n,n}$ such that*

$$A = U\Sigma V^H,\ \Sigma = \left[\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & 0 \\ & & \sigma_r & \\ \hline & 0 & & 0 \end{array}\right] \in \mathbb{C}^{m,n}.$$

Furthermore, $\sigma_1 = ||A||_2$ and $\sigma_1, \ldots, \sigma_r$ are uniquely determined.

Proof: Exercise. □

Definition 18 *Let $A, U = [u_1, \ldots, u_m], V = [v_1, \ldots, v_n], \Sigma$ be as in the SVD and $\sigma_k := 0$ for $k = r+1, \ldots, \min\{m,n\}$. Then*

(a) $\sigma_1, \ldots, \sigma_{\min\{m,n\}}$ *are called* singular values *of A.*

(b) $u_1, \ldots, u_m$ *are called* left singular vectors *of A.*

(c) $v_1, \ldots, v_n$ *are called* right singular vectors *of A.*

Remark 19 *(a) From the SVD one obtains*

$$A^H A = V\Sigma^H U^H U\Sigma V^H = V\Sigma^H\Sigma V^H = V\Sigma^2 V^H$$

and

$$AA^H = U\Sigma V^H V\Sigma^H U^H = U\Sigma\Sigma^H U^H = U\Sigma^2 U^H,$$

i.e. $\sigma_1^2, \ldots, \sigma_r^2$ are the nonzero eigenvalues of AA^H and $A^H A$, respectively.

(b) Since $AV = U\Sigma$ one has Kernel $(A) = Span\{v_{r+1}, \ldots, v_n\}$ *and* Image $(A) = \mathcal{R}(A) = Span\{u_1, \ldots, u_r\}$.

(c) The SVD allows optimal low-rank approximation of A, since

$$\begin{aligned} A &= U\Sigma V^H \\ &= U\left(\left[\begin{array}{ccc} \sigma_1 & & \\ & 0 & \\ & & \ddots \\ & & 0 \end{array}\right] + \ldots + \left[\begin{array}{ccc} 0 & & \\ & \ddots & \\ & & \sigma_n \end{array}\right]\right) V^H \\ &= \sum_{j=1}^{r} \sigma_j u_j v_j^H. \end{aligned}$$

Here $u_j v_j^H$ is a rank one matrix of size $m \times n$. For $0 \le \nu \le r$ the matrix

$$A_\nu := \sum_{i=1}^{\nu} \sigma_i u_j v_i^H$$

is the best rank ν approximation to A in the sense that

$$||A - A_\nu|| = \inf_{\substack{B \in \mathbb{C}^{m,n} \\ \text{Rank}\,(B) \leq \nu}} ||A - B|| = \sigma_{\nu+1},$$

where $\sigma_{r+1} := 0$.

(d) If A is real, then also U and V can be chosen real.

1.3 Perturbation theory

In the analysis of numerical methods, we will have to study the eigenvalues, eigenvectors invariant subspaces under small perturbations ?

1.3.1 Canonical angles and vectors

Question: let $\mathcal{U}, \mathcal{V} \subset \mathbb{C}^n$ be subspaces of dimension k. How 'near' are $\mathcal{U}$ and $\mathcal{V}$?
Strategy: Compute successively the angles between $\mathcal{U}$ and $\mathcal{V}$ beginning with the smallest. Choose normalized vectors $x \in \mathcal{U}$ and $y \in \mathcal{V}$, such that

$$|\langle x, y \rangle| \stackrel{!}{=} \max.$$

W.l.o.g. we can choose x and y such that their scalar product is real and nonnegative. Otherwise we can take $z \in \mathbb{C}$ with $|z| = 1$, so that $\langle zx, y \rangle = z \langle x, y \rangle$ is real and nonnegative. Then $|\langle x, y \rangle| = |\langle zx, y \rangle|$.

(a) Choose $x_1 \in \mathcal{U}$ and $y_1 \in \mathcal{V}$ with $||x_1|| = ||y_1|| = 1$ such that

$$\langle x_1, y_1 \rangle = \max \{ \text{Re}\,\langle x, y \rangle \mid x \in \mathcal{U}, y \in \mathcal{V}, ||x|| = ||y|| = 1 \}$$

Then $\langle x_1, y_1 \rangle$ is real, $\vartheta_1 = \arccos \langle x_1, y_2 \rangle$ is called first *canonical angle* and x_1, y_1 are called first *canonical vectors*.

(b) Suppose that we have determined $j-1$ canonical angles and vectors, i.e.,

$$x_1, \ldots, x_{j-1} \in \mathcal{U},\ y_1, \ldots, y_{j-1} \in \mathcal{V}$$

are determined with $(x_1, \ldots, x_{j-1})$ and $(y_1, \ldots, y_{j-1})$ orthonormal.

Choose $x_j \in \mathcal{U}$ and $y_j \in \mathcal{V}$ with $x_j \perp x_1, \ldots, x_{j-1}$ and $y_j \perp, y_1, \ldots, y_{j-1}$ and $||x_j|| = ||y_i|| = 1$, so that

$$\langle x_j, y_j \rangle$$

has maximal real part. Then $\langle x_j, y_j \rangle$ is real,

$$\vartheta_j := \arccos \langle x_j, y_j \rangle$$

is the j-th canonical angle, and x_j, y_j are j-th canonical vectors. Proceeding inductively we obtain k canonical angles $0 \leq \vartheta_1 \leq \ldots \leq \vartheta_k \leq \frac{\pi}{2}$ and orthonormal bases $(x_1, \ldots, x_k)$ and $(y_1, \ldots, y_k)$ of $\mathcal{U}$, $\mathcal{V}$, respectively.

Lemma 20 *For $i, j = 1, \ldots, k$ and $i \neq j$ the canonical vectors satisfy $\langle x_i, y_j \rangle = 0$.*

Proof: Exercise. □

Corollary 21 *Let $X = [x_1, \ldots, x_k]$ and $Y = [y_1, \ldots, y_k]$. Then*

$$X^H Y = (\langle x_i, y_j \rangle) = \begin{bmatrix} \cos\vartheta_1 & & 0 \\ & \ddots & \\ 0 & & \cos\vartheta_k \end{bmatrix}$$

with $\cos\vartheta_1 \geq \ldots \geq \cos\vartheta_k \geq 0$ and this is SVD.

Practical computation of canonical angles and vectors

(a) Determine orthonormal bases of $\mathcal{U}$ and $\mathcal{V}$, i.e., isometric matrices $P, Q \in \mathbb{C}^{n,k}$ with

$$\mathcal{R}(P) = \mathcal{U},\ \mathcal{R}(Q) = \mathcal{V}.$$

(b) Compute the SVD of $P^H Q$

$$P^H Q = U \Sigma V^H$$

with the diagonal matrix

$$\Sigma = \underbrace{U^H P^H}_{X^H} \underbrace{QV}_{Y}$$

(c) Set $U = [u_1, \ldots, u_k]$ and $V = [v_1, \ldots, v_k]$. Then

(a) $\vartheta_j = \arccos\sigma_j, j = 1, \ldots, k$ are the canonical angles and
(b) $Pu_j, Qv_j, j = 1, \ldots, k$ are the canonical vectors.

1.3.2 Distance between subspaces

Definition 22 *Let $\mathcal{U}, \mathcal{V} \in \mathbb{C}^n$ be subspaces of dimension k.*

(a) For $x \in \mathcal{U}$ we call

$$d(x, \mathcal{V}) := \min_{y \in \mathcal{V}} ||x - y||$$

the distance from x to $\mathcal{V}$ *and*

(b)

$$d(\mathcal{U}, \mathcal{V}) := \max_{\substack{x \in \mathcal{U} \\ ||x|| = 1}} d(x, \mathcal{V})$$

the distance of $\mathcal{U}$ and $\mathcal{V}$.

Theorem 23 *Let $\mathcal{U}, \mathcal{V} \subset \mathbb{C}^n$ be subspaces of dimension k with canonical angles $\vartheta_1 \leq \ldots \leq \vartheta_k$, then*

$$d(\mathcal{U}, \mathcal{V}) = \sin\vartheta_k.$$

Proof: See Stewart/Sun. *Matrix perturbation theory.* Boston, 1990. □

Lemma 24 *Let*

$$\mathcal{U} = \mathcal{R}\left(\begin{bmatrix} I_m \\ 0 \end{bmatrix}\right) \quad \text{and} \quad \hat{\mathcal{U}} = \mathcal{R}\left(\begin{bmatrix} I_m \\ X \end{bmatrix}\right),$$

with $X \in \mathbb{C}^{(n-m)\times m}$ *be* m*-dimensional subspaces of* $\mathbb{C}^n$ *and let* $\theta_1, \ldots, \theta_m$ *be the canonical angles between* $\mathcal{U}$ *and* $\hat{\mathcal{U}}$*. Then*

$$\tan\theta_1, \ldots, \tan\theta_n$$

are the singular values of X*, in particular then* $||X|| = \tan\theta_m$*.*

Proof: See Stewart/Sun. *Matrix perturbation theory.* Boston, 1990. □

Chapter 2

Eigenvalue problems with full matrices

Situation: $A \in \mathbb{C}^{n,n}$, where n is small enough so that the matrix A can be fully stored and that we can manipulate the whole matrix by similarity transformations.

2.1 The power method

Idea: Take an arbitrary $q \in \mathbb{C}^n \setminus \{0\}$ and form the sequence $q, Aq, A^2q, \ldots$. What will happen?

Assumption: A is diagonalizable. Let $\lambda_1, \ldots, \lambda_n$ with $|\lambda_1| \geq \ldots \geq |\lambda_n|$ be the eigenvalues of A and let $(v_1, \ldots, v_n)$ be a basis of eigenvectors. Then there exist $c_1, \ldots, c_n$ with

$$q = c_1 v_1 + \ldots + c_n v_n.$$

Further assumption: $c_1 \neq 0$ (this happens with probability 1 if q is random). Then,

$$\begin{aligned} Aq &= c_1\lambda_1 v_1 + \ldots + c_n\lambda_n v_n, \\ A^k q &= c_1\lambda_1^k v_1 + \ldots + c_n\lambda_n^k v_n. \end{aligned}$$

For $|\lambda_1| > 1$ the powers $|\lambda_1^k|$ will grow, so we scale as

$$\frac{1}{\lambda_1^k} A^k q = c_1 v_1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k v_2 + \ldots + c_n \left(\frac{\lambda_n}{\lambda_1}\right)^k v_n.$$

Third assumption: $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n|$. Then,

$$\begin{aligned} \left\| \frac{1}{\lambda_1^k} A^k q - c_1 v_1 \right\| &\leq |c_2| \left|\frac{\lambda_2}{\lambda_1}\right|^k ||v_2|| + \ldots + |c_n| \left|\frac{\lambda_n}{\lambda_1}\right|^k ||v_n|| \\ &\leq \Big(|c_2|||v_2|| + \ldots + |c_n|||v_n||\Big) \left|\frac{\lambda_2}{\lambda_1}\right|^k \overset{k\to\infty}{\longrightarrow} 0, \end{aligned}$$

and hence $\lim_{k\to\infty} \frac{1}{\lambda_1^k} A^k q = c_1 v_1$ and the convergence is linear with convergence rate $r \leq |\frac{\lambda_2}{\lambda_1}|$.

Definition 25 *A sequence* (x_k) converges linearly *to* x*, if there exists* r *with* $0 < r < 1$ *such that*

$$\lim_{k\to\infty} \frac{||x_{k+1} - x||}{||x_k - x||} = r.$$

Then r *is called the* convergence rate *of the sequence.*
We say that the convergence $(x_k) \to x$ *is of order* $m \geq 2$ *if*

$$\lim_{k\to\infty} \frac{||x_{k+1} - x||}{||x_k - x||^m} = c \neq 0.$$

If $m = 2$ *then we speak of quadratic convergence and if* $m = 3$ *of cubic convergence.*

In practice we do not know $1/\lambda_1^k$, thus we normalize differently and divide by the largest (in modulus) component of $A^k q$.

Algorithm: (Power method)
Computes the dominant eigenvalue λ_1 and the associated eigenvector v_1.

(a) Choose $q_0 \in \mathbb{C}^n \setminus \{0\}$

(b) Iterate, for $k = 1, 2, \ldots$ to convergence

$$q_k := \frac{1}{\alpha_k} A q_{k-1},$$

where α_k is the largest (in modulus) component of Aq_{k-1}.

The power method can also be used for large scale problem where only matrix vector multiplication is available. It computes only one eigenvalue and eigenvector and is for example used in the Google page rank method. By the presented analysis we have proved the following theorem.

Theorem 26 *Suppose that* $A \in \mathbb{C}^{n,n}$ *has the eigenvalues* $\lambda_1, \ldots, \lambda_n$ *with* $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n|$*. If* $q_0 \in \mathbb{C}^n \setminus \{0\}$ *has a component in the invariant subspace associated to* λ_1*, (i.e.,* $c_1 \neq 0$*), then the sequence* (q_k) *defined in the power method converges to an eigenvector associated with* λ_1*. The convergence is linear with rate* $r \leq |\frac{\lambda_2}{\lambda_1}|$*. Furthermore, the sequence* (α_k) *converges to the eigenvalue* λ_1*.*

Remark 27 *(a) This theorem also holds for non-diagonalizable matrices.*

(b) Forming the full products Aq_k *costs* $2n^2$ *flops and the scaling* $O(n)$ *flops. Hence* m *iterations will cost* $2n^2 m$ *flops.*

(c) If (as is very common) $|\frac{\lambda_2}{\lambda_1}| \approx 1$ *then the convergence is very slow.*

2.2 Shift-and-Invert and Rayleigh-Quotient-Iteration

Observations: Let $A \in \mathbb{C}^{n,n}$ and $(\lambda, v) \in \mathbb{C} \times \mathbb{C}^n$ with $Av = \lambda v$. Then

(a) $A^{-1}v = \lambda^{-1}v$ for A invertible, and

(b) $(A - \varrho I)v = (\lambda - \varrho)v$ for all $\varrho \in \mathbb{C}$.

If $\lambda_1, \ldots, \lambda_n$ are again the eigenvalues of A with $|\lambda_1| \geq \ldots \geq |\lambda_n|$, then we can perform the following iterations.

Inverse Iteration This is the power method applied to A^{-1}. If $|\lambda_n| < |\lambda_{n-1}|$, then the inverse iteration converges to an eigenvector to λ_n with convergence rate $\leq |\frac{\lambda_n}{\lambda_{n-1}}|$ (which is small if $|\lambda_n| \ll |\lambda_{n-1}|$).

Shift and Invert Power Method This is the power method applied to $(A - \varrho I)^{-1}$. Let λ_j, λ_k be the eigenvalues that are closest to ϱ, and suppose that $|\lambda_j - \varrho| < |\lambda_k - \varrho|$. Then the power method for $(A - \varrho I)^{-1}$ converges to an eigenvector associated with λ_j with rate

$$\left| \frac{\lambda_j - \varrho}{\lambda_k - \varrho} \right|.$$

This is small if $|\lambda_j - \varrho| \ll |\lambda_k - \varrho|$ and $\lambda_j \approx \varrho$ would be optimal, but where do we get such good shifts? To answer this question we need some results on residuals and backward errors.

Definition 28 *Let $A \in \mathbb{C}^{n,n}$ and $(\mu, w) \in \mathbb{C} \times \mathbb{C}^n$. Then $Aw - \mu w$ is called the* eigenvalue residual *of (μ, w) with respect to A.*

Theorem 29 *Let $\mu \in \mathbb{C}, \varepsilon > 0$, $A \in \mathbb{C}^{n \times n}$, and $w \in \mathbb{C}^n$ with $||w|| = 1$. If $||Aw - \mu w|| = \varepsilon$, then there exists a matrix (*the backward error matrix*) $E \in \mathbb{C}^{n,n}$ with $||E|| \leq \varepsilon$ such that*

$$(A + E)w = \mu w.$$

Proof: Let $r := Aw - \mu w$ and $E = -rw^H$. Then

$$(A + E)w = Aw - r \underbrace{w^H w}_{=1} = \mu w$$

$$\text{and} \quad ||E|| = ||rw^H|| \leq ||r||||w^H|| = ||r|| = \varepsilon.$$

□

Small residual implies small backward error in eigenvalue/eigenvector pair.

The idea to determine a good eigenvalue approximation (shift) from a given eigenvector approximation is to minimize the residual $||Aw - \mu w||$. Consider the over-determined linear system

$$w\mu = Aw$$

with the $n \times 1$-Matrix w, the unknown vector μ and the right hand side Aw. We can use the normal equations to solve $||Aw - \mu w|| = \min!$, i.e., we use

$$w^H w\mu = w^H Aw \qquad \text{respect.} \qquad \mu = \frac{w^H Aw}{w^H w}.$$

Definition 30 *Let $A \in \mathbb{C}^{n,n}$ and $w \in \mathbb{C}^n \setminus \{0\}$. Then*

$$r(w) := \frac{w^H A w}{w^H w}$$

is called the Rayleigh-quotient *of w with respect to A.*

The following theorem gives an estimate for the distance of the Rayleigh-quotient from an eigenvalue.

Theorem 31 *Let $A \in \mathbb{C}^{n,n}$ and $(\lambda, v) \in \mathbb{C} \times \mathbb{C}^n$ be an eigenvalue/eigenvector pair of A with $\|v\| = 1$. Then for $w \in \mathbb{C}^n$ with $\|w\| = 1$ the following estimate holds*

$$|\lambda - r(w)| \leq 2\|A\| \cdot \|v - w\|.$$

This gives an the idea for an iteration to iterate computing an approximate eigenvector and from this a Rayleigh-quotient, i.e., the following algorithm:

Algorithm: Rayleigh-Quotient-Iteration (RQI)
This algorithm computes an eigenvalue/eigenvector pair $(\lambda, v) \in \mathbb{C} \times \mathbb{C}^n$ of the matrix $A \in \mathbb{C}^{n,n}$.

(a) Start: Choose $q_0 \in \mathbb{C}^n$ with $||q_0|| = 1$ and set $\lambda_0 := q_0^H A q_0$.

(b) Iterate for $k = 1, 2, \ldots$ to convergence

  (a) Solve the linear system $(A - \lambda_{k-1} I)x = q_{k-1}$ for x.
  (b) $q_k := \dfrac{x}{||x||}$
  (c) $\lambda_k := q_k^H A q_k$

Remark 32 *(a) It is difficult to analyze the convergence of this algorithm but one observes practically that it almost always converges. The convergence rate is typically quadratic. For Hermitian matrices $A = A^H$ there is more analysis and one can even show cubic convergence.*

(b) The Rayleigh-quotient iteration can also be applied to very large matrices provided that a linear system solver is available (we get back to this later).

(c) **Costs:** *$O(n^3)$ flops per step if the linear system is solved with full Gaussian elimination. The costs are $O(n^2)$ for Hessenberg matrices (see Chapter 2.4.2) and they can be even smaller for banded or other sparse matrices.*

(d) $A - \lambda_{k-1} I$ is 'almost singular', i.e., if λ_{k-1} is close to an eigenvalue, then linear systems with $A - \lambda_{k-1} I$ are generally ill-conditioned. But if we use a backward stable method then we get a small backward error, i.e.,

$$(A + \Delta A - \lambda_{k-1} I)\hat{x} = q_{k-1}$$

with $||\Delta A||$ small. Thus we can expect good results only if the eigenvalue/eigenvector computation is well conditioned.

Conditioning of eigenvalues: if λ is a simple eigenvalue of A, i.e., the algebraic multiplicity is 1 and if v, w are normalized right and left eigenvectors, i.e.,

$$Av = \lambda v, \quad w^H A = \lambda w^H, \quad \|v\| = 1 = \|w\|,$$

then for small perturbations ΔA we have in first approximation that $A+\Delta A$ has an eigenvalue $\lambda + \Delta\lambda$ with

$$|\Delta\lambda| \leq \frac{1}{|w^H v|}\|\Delta A\|.$$

We then have that $1/|w^H v|$ is a *condition number for simple eigenvalues.* For normal matrices we have $v = w$ and thus $|w^H v| = 1$. Normal matrices thus have well-conditioned eigenvalues.

2.3 Subspace iteration

To compute several eigenvalues and the associated invariant subspace, we can generalize the power method to the subspace iteration. Consider $A \in \mathbb{C}^{n,n}$ with eigenvalues $\lambda_1, \ldots, \lambda_n$, where $|\lambda_1| \geq \ldots \geq |\lambda_n|$.

Idea: Instead of $q_0 \in \mathbb{C}^n$, consider a set of linearly independent vectors $\{w_1, \ldots, w_m\} \subset \mathbb{C}^n$. Set

$$W_0 := [w_1, \ldots, w_m] \in \mathbb{C}^{n\times m},$$

and form the sequence $W_0, AW_0, A^2W_0, \ldots$ via

$$W_k := A^k W_0 = \left[A^k w_1, \ldots, A^k w_m\right], \quad k \geq 1.$$

In general, we expect $\mathcal{R}(W_k)$ to converge to the invariant subspace $\mathcal{U}$ associated with the m eigenvalues $\lambda_1, \ldots, \lambda_m$. This iteration is called *subspace iteration.*

Theorem 33 *Let $A \in \mathbb{C}^{n,n}$ with eigenvalues $\lambda_1, \ldots, \lambda_n$ satisfy*

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

Let $\mathcal{U}, \mathcal{V}$ be the invariant subspaces associated with $\lambda_1, \ldots, \lambda_m$, and $\lambda_{m+1}, \ldots, \lambda_n$ respectively. Furthermore, let $W \in \mathbb{C}^{n\times m}$ with $\operatorname{Rank}(W) = m$ and $\mathcal{R}(W) \cap \mathcal{V} = \{0\}$. Then for the iteration $W_0 := W$, $W_{k+1} = AW_k$ for $k \geq 0$ and for every ϱ with $\left|\frac{\lambda_{m+1}}{\lambda_m}\right| < \varrho < 1$, there exists a constant c such that

$$d(\mathcal{R}(W_k), \mathcal{U}) \leq c \cdot \varrho^k, \quad k \geq 1.$$

Proof: We prove the theorem for the case that A is diagonalizable. We perform a similarity transformation

$$A_{\text{new}} = S^{-1} A_{\text{old}} S = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix},$$

with $A_1 = \operatorname{diag}(\lambda_1, \ldots, \lambda_m)$ and $A_2 = \operatorname{diag}(\lambda_{m+1}, \ldots, \lambda_n)$. Then A_1 is nonsingular, since $|\lambda_1| \geq \ldots \geq |\lambda_m| > 0$. Set

$$\mathcal{U}_{\text{new}} = S^{-1}\mathcal{U}_{\text{old}} = \mathcal{R}\left(\begin{bmatrix} I_m \\ 0 \end{bmatrix}\right)$$

and

$$\mathcal{V}_{\text{new}} = S^{-1}\mathcal{V}_{\text{old}} = \mathcal{R}\left(\begin{bmatrix} 0 \\ I_{n-m} \end{bmatrix}\right).$$

Furthermore, let

$$W_{\text{new}} = S^{-1}W_{\text{old}} = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix}$$

for some $Z_1 \in \mathbb{C}^{m,m}$ and $Z_2 \in \mathbb{C}^{n-m,m}$. Then

(a) $d(\mathcal{R}(W_{\text{new}}),\mathcal{U}_{\text{new}}) \leq \kappa(S)d(\mathcal{R}(W_{\text{old}}),\mathcal{U}_{\text{old}})$ (Exercise)
(Here $\kappa(S) = \|S\|\|S^{-1}\|$ is the condition number of S with respect to inversion.)

(b) $\mathcal{R}(W_{\text{new}}) \cap \mathcal{V}_{\text{new}} = \{0\} \Leftrightarrow Z_1$ is nonsingular (Exercise)

In the following we drop the index 'new'. Then

$$W = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \begin{bmatrix} I_m \\ Z_2 Z_1^{-1} \end{bmatrix} Z_1 = \begin{bmatrix} I \\ X_0 \end{bmatrix} Z_1$$

with $X_0 = Z_2 Z_1^{-1}$, and hence

$$\mathcal{R}(W) = \mathcal{R}\left(\begin{bmatrix} I_m \\ X_0 \end{bmatrix}\right)$$

as well as

$$\mathcal{R}(W_k) = \mathcal{R}(A^k W) = \mathcal{R}(A^k \begin{bmatrix} I \\ X_0 \end{bmatrix}).$$

Then it follows that

$$A^k \begin{bmatrix} I \\ X_0 \end{bmatrix} = \begin{bmatrix} A_1^k & 0 \\ 0 & A_2^k \end{bmatrix} \begin{bmatrix} I \\ X_0 \end{bmatrix} = \begin{bmatrix} A_1^k \\ A_2^k X_0 \end{bmatrix} = \begin{bmatrix} I_m \\ \underbrace{A_2^k X_0 A_1^{-k}}_{=:X_k} \end{bmatrix} A_1^k,$$

and thus,

$$\mathcal{R}(W_k) = \mathcal{R}\left(\begin{bmatrix} I_m \\ X_k \end{bmatrix}\right).$$

It remains to show that

$$d(\mathcal{R}(W_k),\mathcal{U}) \to 0.$$

Let $\Theta_m^{(k)}$ be the largest canonical angle between $\mathcal{R}(W_k)$ and $\mathcal{U}$. Then

$$\begin{aligned} d(\mathcal{R}(W_k),\mathcal{U}) &= \sin\Theta_m^{(k)} \leq \tan\Theta_m^{(k)} = ||X_k|| \leq ||A_2^k||||X_0||||A_1^{-k}|| \\ &= |\lambda_{m+1}^k|||X_0|||\lambda_m^{-k}|, \end{aligned}$$

which implies that

$$d(R(W_k),\mathcal{U}) \leq \tilde{c}\left|\frac{\lambda_{m+1}}{\lambda_m}\right|^k.$$

Undoing the similarity transformation we obtain the desired result. For the diagonalizable case we do not need the bound ϱ. This will be only needed in the non-diagonalizable case. □

Remark 34 *For $W_0 = [w_1, \ldots, w_m]$ we have*

$$A^k W_0 = \left[A^k w_1, \ldots, A^k w_m\right],$$

i.e., we perform the iteration not only for W_0 but simultaneously also for all $W_0^{(j)} = [w_1, \ldots, w_j]$, since

$$A^k W_0^{(j)} = \left[A^k w_1, \ldots, A^k w_j\right].$$

Under appropriate assumptions, we then have convergence of

$$Span\left\{A^k w_1, \ldots, A^k w_j\right\}$$

to the invariant subspace associated with $\lambda_1, \ldots, \lambda_j$ for all $j = 1, \ldots, m$. For this reason one often speaks of 'simultaneous subspace iteration'.

Problems with subspace iteration in finite precision arithmetic:

Theory: $A^k W_0 = \left[A^k w_1, \ldots, A^k w_m\right]$ in general has Rank m (for generic starting values).

Practice: Unfortunately, in finite precision arithmetic rounding errors lead to linear dependence in $\mathcal{R}(W_k)$ already after few iterations.

The basic idea to cope with this problem is to orthonormalize the columns in every step.

Step 1: Factor $W_0 = [w_1, \ldots, w_m] = Q_0 R_0$ with $Q_0 \in \mathbb{C}^{n,m}$ isometric and $R_0 \in \mathbb{C}^{m,m}$ upper triangular. Then $\mathcal{R}(W_0) = \mathcal{R}(Q_0)$ and furthermore,

$$\text{Span}\left\{w_1, \ldots, w_j\right\} = \text{Span}\left\{q_1^{(0)}, \ldots, q_j^{(0)}\right\},\ j = 1, \ldots, m,$$

where $Q_0 = \left[q_1^{(0)}, \ldots, q_m^{(0)}\right]$. (This follows form the triangular form of R_0.)

Situation after step $k-1$: $\mathcal{R}(W_{k-1}) = \mathcal{R}(Q_{k-1})$ with

$$Q_{k-1} = \left[q_1^{(k-1)}, \ldots, q_m^{(k-1)}\right] \in \mathbb{C}^{n,m}$$

isometric and

$$\text{Span}\left\{A^{k-1} w_1, \ldots, A^{k-1} w_j\right\} = \text{Span}\left\{q_1^{(k-1)}, \ldots, q_j^{(k-1)}\right\}, \quad j = 1, \ldots, m.$$

Step k: Let $AQ_{k-1} = Q_k R_k$ be a QR decomposition with $Q_k \in \mathbb{C}^{n,m}$ isometric and $R_k \in \mathbb{C}^{m,m}$ upper triangular. Then

$$\mathcal{R}(W_k) = \mathcal{R}(AW_{k-1}) = \mathcal{R}(AQ_{k-1}) = \mathcal{R}(Q_k),$$

and moreover

$$\text{Span}\left\{A^k w_1, \ldots, A^k w_j\right\} = \text{Span}\left\{q_1^{(k)}, \ldots, q_j^{(k)}\right\}, \quad j = 1, \ldots, m.$$

Algorithm: Unitary Subspace Iteration

(a) Start: Choose $Q_0 \in \mathbb{C}^{n,m}$ isometric.

(b) Iterate. For $k = 1, 2, \ldots$ to convergence:

  (a) Compute $Z_k = AQ_{k-1}$

  (b) Compute QR-decomposition $Z_k = Q_k R_k$.

Remark 35 *Theoretically the convergence behavior of the unitary subspace iteration is as for the subspace iteration but the described problems in finite precision arithmetic do not arise.*

2.4 The Francis QR Algorithm

2.4.1 Simple QR Algorithm Without Shifts

Let $A \in \mathbb{C}^{n,n}$ have the eigenvalues $\lambda_1, \ldots, \lambda_n$ where $|\lambda_1| \geq \ldots \geq |\lambda_n|$.

Idea: Use the n-dimensional unitary subspace iteration, i.e., choose $m = n$ and $Q_0 = I_n$. If $Q_k = \left[q_1^{(k)}, \ldots, q_n^{(k)}\right]$, then for every $1 \leq m \leq n$

$$\text{Span}\left\{q_1^{(k)}, \ldots, q_m^{(k)}\right\}$$

converges to the invariant subspace associated with $\lambda_1, \ldots, \lambda_m$ with a rate $\left|\frac{\lambda_{m+1}}{\lambda_m}\right|$ provided that $|\lambda_{m+1}| < |\lambda_m|$ and one does not run into an exceptional situation.
To observe the convergence, we form $A_k = Q_k^{-1} A Q_k$. If $\text{Span}\{q_1^{(k)}, \ldots, q_m^{(k)}\}$ converges to an invariant subspace, then we expect that in the matrix

$$A_k = \begin{array}{c} \\ m \\ n-m \end{array} \begin{array}{c} \begin{array}{cc} m & n-m \end{array} \\ \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \end{array}$$

the block A_{21} converges to 0 for $k \to \infty$. Since this happens for all m simultaneously, it follows that A_k converges to block-upper triangular matrix.
Another question is whether we can directly move from A_{k-1} to A_k? To see this, observe that

$$\begin{aligned} A_{k-1} &= Q_{k-1}^{-1} A Q_{k-1} \\ A_k &= Q_k^{-1} A Q_k \end{aligned}$$

and hence

$$A_k = Q_k^{-1} Q_{k-1} A_{k-1} \underbrace{Q_{k-1}^{-1} Q_k}_{=:U_k} = U_k^{-1} A_{k-1} U_k.$$

Thus we can reformulate the k-th step of the unitary subspace iteration

$$AQ_{k-1} = Q_k R_k$$

as

$$A_{k-1} = Q_{k-1}^{-1} A Q_{k-1} = Q_{k-1}^{-1} Q_k R_k = U_k R_k.$$

This is a QR decomposition of A_{k-1} and we have

$$A_k = U_k^{-1} A_{k-1} U_k = U_k^{-1} U_k R_k U_k = R_k U_k.$$

Algorithm: (QR Algorithm)(Francis and Kublanovskaya 1961)
For a given matrix $A \in \mathbb{C}^{n,n}$ this algorithm constructs a sequence (A_k) of similar matrices that converges to block upper-triangular form.

(a) Start with $A_0 = A$

(b) Iterate for $k = 1, 2, \ldots$ to convergence.

  (a) Compute a QR-decomposition of A_{k-1}: $A_{k-1} = U_k R_k$

  (b) Compute A_k via $A_k = R_k U_k$.

Theorem 36 (Convergence of the QR algorithm)
Let $A \in \mathbb{C}^{n,n}$ have eigenvalues $\lambda_1, \ldots, \lambda_n$, where $|\lambda_1| \geq \ldots \geq |\lambda_m| > |\lambda_{m+1}| \geq \ldots \geq |\lambda_n|$. Let $\mathcal{V} \subset \mathbb{C}^n$ be the invariant subspace associated with $\lambda_{m+1}, \ldots, \lambda_n$, and let (A_k) be the sequence generated by the QR Algorithm. If

$$Span\{e_1, \ldots, e_m\} \cap \mathcal{V} = \{0\}$$

and

$$A_k = \begin{matrix} \\ m \\ n-m \end{matrix} \begin{matrix} \begin{matrix} m & n-m \end{matrix} \\ \begin{bmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ A_{21}^{(k)} & A_{22}^{(k)} \end{bmatrix} \end{matrix},$$

then for every ϱ with $\left|\frac{\lambda_{m+1}}{\lambda_m}\right| < \varrho < 1$ there exists a constant $\tilde{c}$ such that

$$||A_{21}^{(k)}|| \leq \tilde{c} \varrho^k.$$

Proof: (Sketch) Let $\mathcal{U}$ be the invariant subspace associated with $\lambda_1, \ldots, \lambda_m$ and

$$\mathcal{U}_k = \text{Span}\left\{q_1^{(k)}, \ldots, q_m^{(k)}\right\},$$

where

$$Q_k = \left[q_1^{(k)}, \ldots, q_n^{(k)}\right]$$

is the unitary matrix with $Q_k^{-1} A Q_k = A_k$ from the unitary subspace iteration. One first shows that

$$||A_{21}^{(k)}|| \leq 2\sqrt{2}||A|| d(\mathcal{U}, \mathcal{U}_k)$$

Then using the convergence results for the subspace iteration there exists a constant $c > 0$ with

$$d(\mathcal{U}, \mathcal{U}_k) \leq c \varrho^k.$$

Then choose $\tilde{c} := 2\sqrt{2}||A||c$. □

In the special case that A is Hermitian, the sequence A_k converges to a diagonal matrix.

Remark 37 *In the presented form the algorithm has two major disadvantages:*

(a) It is expensive, since it costs $O(n^3)$ flops per iteration step.

(b) The convergence is slow (only linear).

A way to address the two problems is the Hessenberg reduction and the use of shifts.

2.4.2 Hessenberg reduction

Definition 38 *A matrix $A = [a_{ij}]$ is called* Hessenberg matrix *or* in Hessenberg form*, if $a_{ij} = 0$ for $i > j + 1$. A Hessenberg matrix A is called* unreduced *if $a_{i+1,i} \neq 0$ for all $i = 1, \ldots, n-1$.*

Householder transformations

The QR decomposition and other unitary transformations can be realized via *Householder transformations*

$$P = I - \frac{2}{v^H v} v v^H$$

for $v \in \mathbb{C}^n \setminus \{0\}$. Householder Transformations are Hermitian and unitary. (Exercise) Multiplication with Householder transformations is geometrically a reflection of a vector $x \in \mathbb{C}^n$ at the hyperplane $\text{Span}\{v\}^\perp$. (Exercise.)

A typical task: Reflect $x \in \mathbb{C}^n \setminus \{0\}$ to a multiple of the first unit vector, i.e., determine v and from this P such that

$$Px = \pm ||x|| e_1.$$

To determine such a v we make an ansatz $v = x + \alpha e_1$ and obtain

$$v = x \pm ||x|| e_1 \qquad \text{and} \qquad Px = \mp ||x|| e_1. \qquad \text{(Exercise)}$$

For numerical reasons we take

$$v = \begin{cases} x + ||x|| e_1, & x_1 \geq 0, \\ x - ||x|| e_1, & x_1 < 0. \end{cases}$$

Advantage: The multiplication with Householder transformations is cheap. For $B \in \mathbb{C}^{n,m}$ the computation of PB only needs $\sim 4mn$ flops (instead of $\sim 2n^2 m$ flops for classical matrix/matrix multiplication).

Givens rotations

Another tool for unitary operations are *Givens rotations*

$$G_{i,j}(c,s) = \begin{bmatrix} 1 & & & & & & & & & & \\ & \ddots & & & & & & & & & \\ & & 1 & & & & & & & & \\ & & & c & & & & s & & & \\ & & & & 1 & & & & & & \\ & & & & & \ddots & & & & & \\ & & & & & & 1 & & & & \\ & & & -\bar{s} & & & & \bar{c} & & & \\ & & & & & & & & 1 & & \\ & & & & & & & & & \ddots & \\ & & & & & & & & & & 1 \end{bmatrix},$$

where $|c|^2+|s|^2 = 1$ and where the matrix differs from an identity only in positions $(i,i),(i,j),(j,i),(j,j)$. Multiplication of a matrix with a Givens rotation allows to zero an element in any position. E.g. choose

$$\hat{G}_{1,2}(c,s) = \begin{bmatrix} c & s \\ -\bar{s} & \bar{c} \end{bmatrix}$$

with $|c|^2 + |s|^2 = 1$ such $-\bar{s}a_{11} + \bar{c}a_{21} = 0$, then we have

$$\hat{G}_{1,2}(c,s) \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} * & * \\ 0 & * \end{bmatrix}.$$

(a) For a Hessenberg matrix $H \in \mathbb{C}^{n,n}$ the QR decomposition can be performed in $O(n^2)$ flops using Givens rotations.

(b) Hessenberg matrices are invariant under QR iterations.

Theorem 39 (Implicit Q Theorem) *Let $A \in \mathbb{C}^{n,n}$ and let $Q = [q_1, \ldots, q_n]$, $U = [u_1, \ldots, u_n]$ be unitary matrices such that*

$$H = Q^{-1}AQ = (h_{ij}) \qquad \text{and} \qquad G = U^{-1}AU = [g_{ij}]$$

are Hessenberg matrices. If $q_1 = u_1$ and H is unreduced, then

$$q_i = c_i u_i$$

for $c_i \in \mathbb{C}$ with $|c_i| = 1$ and $|h_{i,i-1}| = |g_{i,i-1}|$ for $i = 2, \ldots, n$, i.e., Q is determined already essentially uniquely by q_1.

Proof: Exercise. □

2.4.3 The Francis QR Algorithm with Shifts

Deflation: Let $H \in \mathbb{C}^{n,n}$ be in Hessenberg form. If H is not unreduced, i.e., if $h_{m+1,m} = 0$ for some m, then

$$H = \begin{matrix} \\ m \\ n-m \end{matrix} \begin{matrix} \begin{matrix} m & n-m \end{matrix} \\ \begin{bmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{bmatrix} \end{matrix}$$

i.e., we can split our problem into two subproblems H_{11}, H_{22}.

Algorithm (QR Algorithm with Hessenberg reduction and shifts)
Given: $A \in \mathbb{C}^{n,n}$:

(a) Compute U_0 unitary such that

$$H_0 := U_0^H A U_0$$

is in Hessenberg form. We may assume that H_0 is unreduced, otherwise we can deflate right away.

(b) Iterate for $k = 1, 2, \ldots$ until deflation happens, i.e.,

$$h_{m+1,m}^{(k)} = O(\texttt{eps})(|\texttt{h}_{\texttt{m,m}}| + |\texttt{h}_{\texttt{m+1,m+1}}|)$$

for some m and the machine precision `eps`.

(i) Choose shift $\mu_k \in \mathbb{C}$.
(ii) Compute a QR decomposition $H_{k-1} - \mu_k I = Q_k R_k$ of $H_{k-1} - \mu_k I$.
(iii) Form $H_k = R_k Q_k + \mu_k I$.

Remark 40 *(a) Steps (ii) and (iii) of this algorithm correspond to a QR iteration step for $H_0 - \mu_k I$.*

(b) The sub-diagonal entry $h_{m+1,m}^{(k)}$ in H_k converges with rate $\left|\frac{\lambda_{m+1}-\mu_k}{\lambda_m-\mu_k}\right|$ to 0.

(c) If $h_{m+1,m}^{(k)} = 0$ or $h_{m+1,m}^{(k)} = O(eps)$, then we have deflation and we can continue with smaller problems.

(d) If μ_k is an eigenvalue then deflation happens immediately after one step.

Shift strategies:

(a) **Rayleigh-quotient shift:** For the special case that A is Hermitian, the sequence A_k converges to a diagonal matrix. Then $q_n^{(k)}$ is a good approximation to an eigenvector and a good approximation to the eigenvalue is the Rayleigh-quotient

$$r(q_n^{(k)}) = (q_n^{(k)})^H A q_n^{(k)}$$

which is just the n-th diagonal entry $a_{n,n}^{(k)}$ of $Q_k^H A Q_k$.

Heuristic: We expect in general $a_{n,n}^{(k)}$ to be a good approximation to an eigenvalue and therefore may choose

$$\mu_k = a_{n,n}^{(k)}$$

With this choice $h_{n,n-1}^{(k)}$ typically converges quadratically to 0.

(b) **Wilkinson-shift:** Problems with the Rayleigh-quotient shift arise when the matrix is real and has nonreal eigenvalues, e.g., for

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

A QR iteration for $A_0 = A$ yields $Q_0 = A_0, R_0 = I$ and hence,

$$R_0 Q_0 = I A_0 = A_0,$$

i.e., the algorithm stagnates. To avoid such situations, for $A \in \mathbb{C}^{n,n}$ in the k-th step, one considers the submatrix B in

$$A_k = \begin{matrix} \\ n-2 \\ 2 \end{matrix} \begin{matrix} \begin{matrix} n-2 & 2 \end{matrix} \\ \begin{bmatrix} * & * \\ * & B \end{bmatrix} \end{matrix}$$

and chooses the μ_k as shift that is nearest to $a_{nn}^{(k)}$.

(c) Another strategy, the double-shift will be discussed below.

(d) On the average 2-3 iterations are needed until a 1×1 or 2×2 block deflates.

2.4.4 Implicit Shifts and 'Bulge-Chasing'

Let $H \in \mathbb{C}^{n,n}$ be a Hessenberg matrix and $\mu_1, \ldots, \mu_l \in \mathbb{C}$. Carry out l steps of the QR Algorithm with shifts $\mu_1, \ldots, \mu_l$.

$$\begin{aligned} H - \mu_1 I &= Q_1 R_1 \\ H_1 &= R_1 Q_1 + \mu_1 I \\ &\vdots \\ H_{l-1} - \mu_l I &= Q_l R_l \\ H_l &= R_l Q_l + \mu_l I. \end{aligned}$$

Then

$$H_l = Q_l^H Q_l R_l Q_l + \mu_l Q_l^H Q_l = Q_l^H (Q_l R_l + \mu_l I) Q_l = Q_l^H H_{l-1} Q_l$$

and thus per induction

$$H_l = Q_l^H \ldots Q_1^H H \underbrace{Q_1 \ldots Q_l}_{=:Q} = Q^H H Q.$$

This opens the question whether we can compute Q directly without carrying out l QR-iterations.

Lemma 41 $M := (H - \mu_l I) \ldots (H - \mu_1 I) = Q_1 \ldots Q_l \underbrace{R_l \ldots R_1}_{=:R} = QR.$

Proof: By induction we show that

$$(H - \mu_j I) \ldots (H - \mu_1 I) = Q_1 \ldots Q_j R_j \ldots R_1, \quad j = 1, \ldots, l.$$

$j = 1$**:** This is just the first step of the QR algorithm.

$j - 1 \to j$**:**

$$\begin{aligned} & Q_1 \dots Q_j R_j \dots R_1 \\ = \; & Q_1 \dots Q_{j-1}(H_{j-1} - \mu_j I) R_{j-1} \dots R_1 \\ = \; & Q_1 \dots Q_{j-1}\Big(Q_{j-1}^H \dots Q_1^H H Q_1 \dots Q_{j-1} - \mu_j I\Big) R_{j-1} \dots R_1 \\ = \; & (H - \mu_j I) Q_1 \dots Q_{j-1} R_{j-1} \dots R_1 \\ \overset{I.A.}{=} \; & (H - \mu_j I)(H - \mu_{j-1} I) \dots (H - \mu_1 I). \end{aligned}$$

□

This leads to the idea to compute M and then the Householder QR decomposition of M, i.e., $M = QR$, and to set

$$\tilde{H} = Q^H R Q = H_l.$$

This means that one just needs one QR decomposition instead of l QR decompositions in each QR step. On the other hand we would have to compute M, i.e., $l - 1$ matrix-matrix multiplications. But this can be avoided by computing $\tilde{H}$ directly from H using the implicit Q Theorem.

Implicit shift-strategy:

(a) Compute

$$Me_1 = (H - \mu_l I) \dots (H - \mu_1 I) e_1,$$

the first column of M. Then the first $l + 1$ entries are in general nonzero. If l is not too large, then this costs only $O(1)$ flops.

(b) Determine a Householder matrix P_0 such that $P_0(Me_1)$ is a multiple of e_1. Transform H with P_0 as

$$\begin{aligned} & & & l+1 & n-l-1 \\ P_0 &= & l+1 & \begin{bmatrix} * & 0 \\ 0 & I \end{bmatrix} \\ & & n-l-1 & \end{aligned}$$

$$\begin{aligned} & & & l+2 & n-l-2 \\ P_0 H P_0 &= & l+2 & \begin{bmatrix} * & * \\ 0 & \hat{H} \end{bmatrix} \\ & & n-l-2 & \end{aligned}$$

P_0 changes only rows and columns $1, \dots, l + 1$ of H. This gives a Hessenberg matrix with a bulge.

(c) Determine Householder matrices $P_1, \dots, P_{n-2}$ to restore the Hessenberg form. This is called *bulge chasing*, since we chase the bulge the down the diagonal. This yields

$$\tilde{H} := P_{n-2} \dots P_1 P_0 H P_0 \dots P_{n-2}$$

that is again in Hessenberg form and $P_k e_1 = e_1$ for $k = 1, \dots, n - 2$.

(d) P_0 has the same first column as Q. As in the first step for P_0 we have $P_k e_1 = e_1$, then also

$$P_0 P_1 \dots P_{n-2}$$

has the same first column as P_0 and Q, respectively. With the implicit Q Theorem then also Q and $P_0 \dots, P_{n-2}$ and therefore also $\tilde{H}$ and $Q^H H Q$ are essentially equal, thus we have computed $\tilde{H}$ and H_l directly from H.

Algorithm (Francis QR algorithm with implicit double-shift strategy): (Francis 1961)
Given $A \in \mathbb{C}^{n,n}$:

(a) Determine U_0 unitary so that $H_0 := U_0^H A U_0$ is in Hessenberg form.

(b) Iterate for $k = 1, 2, \dots$ to convergence (deflation):

(a) Compute the eigenvalues μ_1, μ_2 of the lower right 2×2 submatrix of H_{k-1}.

(b) Compute (with the implicit shift strategy) for $l = 2$ the matrix $\tilde{Q}_k$ that one obtains with 2 steps of the QR Algorithm with shifts μ_1, μ_2.

(c)

$$H_k := \tilde{Q}_k^H H_{k-1} \tilde{Q}_k$$

Remark 42 *(a) The empirical costs for the computation of all eigenvalues of A are approx. $10n^3$ flops, If also the transformation matrix Q is needed then this leads to approximately $25n^3$ flops.*

(b) The convergence analysis is difficult, no global convergence proof is known.

(c) The method works also for real problems in real arithmetic, since the double shift can be chosen as complex conjugate pairs.

2.5 The QZ algorithm

For the solution of the generalized eigenvalue problem $\lambda E x = A x$ with regular (E, A), $E, A \in \mathbb{C}^{n,n}$ we can extend the ideas of the QR algorithm. The basic idea is to apply the QR algorithm implicitly to $E^{-1}A$, without really computing the inverse and the product.
The first step is a transformation to Hessenberg-triangular form, i.e., one computes unitary matrices U_0, V_0 such that

$$U_0^H E V_0 = S_0, \; U_0^H A V_0 = T_0,$$

with S_0 upper triangular and T_0 upper Hessenberg. If S_0 was invertible, then $S_0^{-1} T_0$ would be of upper Hessenberg form.

Algorithm. Hessenberg-triangular reduction

For a given regular pair (E, A) with $E, A \in \mathbb{C}^{n,n}$ the algorithm computes unitary matrices U_0, V_0 such that

$$U_0^H E V_0 = S_0, \; U_0^H A V_0 = T_0,$$

with S_0 upper triangular and T_0 upper Hessenberg.

Use the QR decomposition to compute $\hat{U}$ such that $\hat{U}^H E$ is upper triangular and set $\hat{A} = \hat{U}^H A = [a_{ij}],\ \hat{E} = \hat{U}^H E = [e_{ij}]$.

For $j = 1, \ldots, n-2$

for $i = n, n-1, \ldots, j+2$

Determine a 2×2 Householder or Givens matrix $\hat{P}$ such that

$$\hat{P} \begin{bmatrix} a_{i-1,j} \\ a_{ij} \end{bmatrix} = \begin{bmatrix} * \\ 0 \end{bmatrix}$$

and set $A := PA = [a_{ij}],\ E = PE = [e_{ij}]$, where $P = \text{diag}\ (I_{i-2}, \hat{P}, I_{n-i})$.
Determine a 2×2 Householder or Givens matrix $\hat{Q}$ such that

$$\begin{bmatrix} e_{i,i-1} & a_{ii} \end{bmatrix} \hat{Q} = \begin{bmatrix} 0 \end{bmatrix}$$

and set $A := AQ,\ E = EQ$, where $Q = \text{diag}\ (I_{i-2}, \hat{Q}, I_{n-i})$.

end

end

This algorithm costs about $5n^3$ flops plus extra $23/6n^3$ if $\hat{U}, \hat{V}$ are desired.
If A is not unreduced then we can again deflate the problem into subproblems, i.e., as

$$\lambda E - A = \lambda \begin{bmatrix} E_{11} & E_{12} \\ 0 & E_{22} \end{bmatrix} - \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}$$

and continue with the subproblems.
Every 0 element in the diagonal of E corresponds to an infinite eigenvalue, i.e. a zero eigenvalue of $\lambda A - E$. If an element on the diagonal of E is zero then we can introduce a 0 in the position $(n, n-1)$ of A and move the 0 to the bottom of the diagonal of E (Exercise). This can be repeated until all zero elements are in the bottom part of the diagonal of E and the corresponding part of A is triangular. Then we have deflated all the infinite eigenvalues and have obtained

$$U_0^H E V_0 = S_0 = \begin{bmatrix} E_{11} & E_{12} \\ 0 & E_{22} \end{bmatrix},\ U_0^H A V_0 = T_0 = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix},$$

with E_{22} strictly upper triangular and A_{22} is nonsingular upper triangular by the regularity of (E, A).
In the top pair (E_{11}, A_{11}) then E_{11} is invertible and in principle we can imply the implicit QR algorithm to $E_{11}^{-1} A_{11}$. This leads to the QZ algorithm of Moler and Stewart from 1973 (Exercise).

Chapter 3

Eigenvalue problems with large sparse matrices

Situation: Given a matrix $A \in \mathbb{C}^{n,n}$ with n very large (e.g., $n \approx 10^6, 10^7, \ldots$) and A sparse, i.e., A has only very few nonzero elements. A sparse matrix $\tilde{A} \in \mathbb{C}^{m,n}$ is described by 6 parameters.

(a) m: no. of rows;

(b) n: no. of columns;

(c) nnz: no. of nonzero elements;

(d) a: list of nonzero elements;

(e) $irow$: list of row indices;

(f) $jcol$: list of column indices.

The programming environment MATLAB for example has the data structure `sparse`.

(a) For $x \in \mathbb{C}^n$ we can compute the product Ax, this often works with $O(n)$ flops instead of $O(n^2)$.

(b) We cannot apply standard similarity transformations, since the transformed matrix in general is not sparse any more.

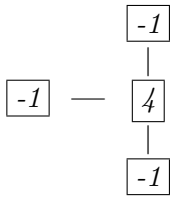
Often even A is not given but just a subroutine, that computes the product Ax for a given $x \in \mathbb{C}^n$, i.e.,

$$x \longrightarrow \boxed{\text{black box}} \rightarrow Ax.$$

This is then the only possibility to obtain information about the matrix A.

Example 43 *(a) Discretized Laplace operator on a uniform grid.*

For a given function $u(x, y)$, we get on a two-dimensional grid in each grid point (i, j) approximations $u_{ij} = u(x_i, y_j)$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Here the approximation to $v = \Delta u$ in the grid point (i, j) obtained via the 5-point difference star satisfies



$$vij = 4u_{ij} - u_{i,j+1} - u_{i,j-1} - u_{i+1,j} - u_{i-1,j},$$

together with further boundary conditions. This we can write as matrix equation.

$$\begin{aligned} v &= [v_{11}, \ldots, v_{1n}, v_{21}, \ldots, v_{2n}, \ldots, v_{m1}, \ldots, v_{mn}] = Au \\ u &= [u_{11}, \ldots, u_{1n}, u_{21}, \ldots, u_{2n}, \ldots, u_{m1}, \ldots, u_{mn}] \end{aligned}$$

Instead of explicitly storing A we can write a subroutine that computes $v = Au$ from u via

```
for i=1,...,m
  for j=1,...,n
    v[i,j]=4*u[i,j]-u[i,j+1]-u[i,j-1]-u[i+1,j]-u[i-1,j]
  end
end
```

(b) Toeplitz matrices:

$$A = \begin{bmatrix} a_0 & a_{-1} & \ldots & a_{-n} \\ a_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ a_n & \ldots & a_1 & a_0 \end{bmatrix}$$

A is uniquely determined by the row vector

$$[a_n, \ldots, a_1, a_0, a_{-1}, \ldots, a_{-n}]$$

and is suffices to store this vector and to write a subroutine to compute the product Ax.

Then the question arises, how to compute some eigenvalues and eigenvectors of A? In general we are only interested in a few eigenvalues, e.g., the ones that are largest or smallest in modulus. (We would certainly not be able to store all eigenvectors because that would need $O(n^3)$ storage.)

1. Idea: *Unitary subspace iteration*

(a) Start with $m \ll n$ orthonormal vectors $q_1^{(0)}, \ldots, q_m^{(0)}$ and set $Q_0 = \left[q_1^{(0)}, \ldots, q_m^{(0)}\right]$.

(b) Compute $AQ_{k-1} = \tilde{Q}_k$, hence m-matrix vector products. For sparse matrices this costs $O(mn)$ flops instead of $O(mn^2)$ for general matrices.

(c) Compute a QR decomposition:

$$\tilde{Q}_k = Q_k R_k,$$

where $Q_k \in \mathbb{C}^{n,m}$ is isometric and $R_k \in \mathbb{C}^{m,m}$ is upper triangular.

In general this method does not work very well.

2. Idea: *Petrov-Galerkin projection methods*: Make use of the fact that for $(\lambda, v) \in \mathbb{C} \times \mathbb{C}^n \setminus \{0\}$ we have

$$(\lambda, v) \text{ is an eigenvalue/eigenvector pair} \quad \Leftrightarrow \quad Av - \lambda v = 0, \quad \Leftrightarrow \quad Av - \lambda v \perp \mathbb{C}^n.$$

(a) Thus we may construct $\mathcal{K} \subseteq \mathbb{C}^n$, the *search space*.

(b) Choose a second subspace $\mathcal{L} \subseteq \mathbb{C}^n$, the *test space*. Then determine $(\tilde{\lambda}, \tilde{v}) \in \mathbb{C} \times \mathcal{K}$ with

$$A\tilde{v} - \tilde{\lambda}\tilde{v} \perp \mathcal{L}.$$

Often one uses $\mathcal{L} = \mathcal{K}$, this is called *Galerkin projection*. *Hope*: Some of the $(\tilde{\lambda}, \tilde{v})$ are good approximations to eigenvalue/eigenvectors pairs of A.

3.1 Krylov Spaces

Question: How do we find a subspace $\mathcal{K}$ that contains good approximation to eigenvectors of A?

As motivation we consider the special case that $A \in \mathbb{R}^{n,n}$ is symmetric with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_n$. Then

$$\lambda_1 = \max_{x \neq 0} r(x) \quad \text{and} \quad \lambda_n = \min_{x \neq 0} r(x), \qquad \text{where} \quad r(x) = \frac{x^T A x}{x^T x} \quad \text{(Exercise)}$$

Let $\mathcal{R}(Q_k)$ be given via $Q_k = [q_1, \ldots, q_k] \in \mathbb{R}^{n,k}$ and let

$$M_k := \max_{x \in \mathcal{R}(Q_k) \setminus \{0\}} r(x) = \max_{y \neq 0} \frac{y^T Q_k^T A Q_k y}{y^T Q_k^T Q_k y},$$

and analogously

$$m_k := \min_{x \in \mathcal{R}(Q_k) \setminus \{0\}} r(x).$$

Then clearly $\lambda_1 \geq M_k \geq m_k \geq \lambda_n$ and, in particular, M_k is an approximation to λ_1 and m_k to λ_n. It is then our goal to construct the vectors $q_1, q_2, \ldots, q_k, \ldots$ such that M_k and m_k quickly approximate λ_1 and λ_n, i.e., we want

$$M_k \approx \lambda_1 \quad \text{and} \quad m_k \approx \lambda_n \quad \text{for small } k.$$

Let $q_1, \ldots, q_k$ be already constructed and choose $u_k, w_k \in \text{Span}\{q_1, \ldots, q_k\}$ such that

$$M_k = r(u_k) \quad \text{and} \quad m_k = r(w_k).$$

It is clear that $r(x)$ grows fastest in the direction of the gradient

$$\nabla r(x) = \frac{2}{x^T x}\Big(Ax - r(x)x\Big) \qquad \text{(Exercise)}.$$

Then $M_{k+1} > M_k$ if $\nabla r(u_k) \neq 0$ and

$$\nabla r(u_k) \in \text{Span}\{q_1, \ldots, q_k, q_{k+1}\}. \tag{3.1}$$

Analogously $m_{k+1} < m_k$ if $\nabla r(w_k) \neq 0$ and

$$\nabla r(w_k) \in \text{Span}\{q_1, \ldots, q_k, q_{k+1}\}. \tag{3.2}$$

Since $\nabla r(x) \in \text{Span}\{x, Ax\}$, the conditions 3.1 and 3.2 are satisfied if

$$\begin{aligned}
\text{Span}\{q_1, q_2\} &= \text{Span}\{q_1, Aq_1\}, \\
\text{Span}\{q_1, q_2, q_3\} &= \text{Span}\{q_1, Aq_1, A^2q_1\}, \\
&\vdots \\
\text{Span}\{q_1, \ldots, q_{k+1}\} &= \text{Span}\Big\{q_1, Aq_1, A^2q_1, \ldots, A^kq_1\Big\}.
\end{aligned}$$

Definition 44 *Let $A \in \mathbb{C}^{n,n}$, $x \in \mathbb{C}^n$ and $l \in \mathbb{N}$.*

(a) $K_l(A,x) := \big[x, Ax, A^2x, \ldots, A^{l-1}x\big]$ *is called* Krylov matrix *for A and x.*

(b)

$$\mathcal{K}_l(A,x) := \mathcal{R}(K_l(A,x)) = \mathit{Span}\Big\{x, Ax, A^2x, \ldots, A^{l-1}x\Big\}$$

is called Krylov space *for A and x.*

We have just observed that for symmetric matrices $A \in \mathbb{R}^{n,n}$ already after few matrix vector multiplications, Krylov spaces yield good approximations to eigenvalues λ_1 and λ_n, i.e., eigenvalues at the exterior of the spectrum. We expect that a similar property holds for general matrices $A \in \mathbb{C}^{n,n}$. **Heuristic**: Krylov spaces are 'good' search spaces!

In the following we present a few properties of Krylov spaces. In particular, we construct the relationship to minimal polynomials and Hessenberg matrices.

Reminder: Let $A \in \mathbb{C}^{n,n}, x \in \mathbb{C}^n$, then there exists a unique normalized polynomial p of smallest degree such that

$$0 = p(A)\,x = A^m x + \alpha_{m-1}A^{m-1}x + \ldots + \alpha_1 Ax + \alpha_0 x.$$

Then p is called *minimal polynomial* of x with respect to A.

Lemma 45 *Let $A \in \mathbb{C}^{n,n}$, $x \in \mathbb{C}^n$ and let ν be the degree of the minimal polynomial of x with respect to A. Then*

(a) $\dim \mathcal{K}_m(A,x) = m \iff m \leq \nu$.

(b) $\mathcal{K}_\nu(A,x)$ *ist A-invariant.*

(c) $\mathcal{K}_m(A,x) = \mathcal{K}_\nu(A,x)$ *für* $m \geq \nu$

Proof: Exercise. □

Lemma 46 *Let $A \in \mathbb{C}^{n,n}$ and $g_1 \in \mathbb{C}^n$ be such that $g_1, Ag_1, \ldots, A^{m-1}g_1$ are linearly independent. Suppose that $g_2, \ldots, g_n$ are such that $G = [g_1, g_2, \ldots, g_n]$ is nonsingular and let $B = G^{-1}AG = [b_{ij}]$. Then the following are equivalent*

(a) $b_{jk} = 0$ *for* $k = 1, \ldots, m-1$ *and* $j = k+2, \ldots, n$, *i.e.,*

$$B = \begin{bmatrix} b_{11} & \ldots & \ldots & \ldots & \ldots & b_{1n} \\ b_{21} & & & & & \vdots \\ 0 & \ddots & & & & \vdots \\ \vdots & \ddots & b_{m,m-1} & b_{mm} & \ldots & b_{mn} \\ \vdots & & 0 & \vdots & \ddots & \vdots \\ \vdots & & \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & 0 & b_{n,m} & \ldots & b_{nn} \end{bmatrix}.$$

(b) $Span\{g_1, \ldots, g_l\} = \mathcal{K}_l(A, g_1)$ *for* $l = 1, \ldots, m$

If one of the conditions is satisfied and if $m = n$, then B is an unreduced Hessenberg matrix.

Proof: Exercise. □

3.2 The Arnoldi algorithm

Let $A \in \mathbb{C}^{n,n}$ be given. We already know the decompositions

QR decomposition	Hessenberg reduction
$A = QR$	$A = QHQ^H$
Householder	Householder
Gram-Schmidt	*new: Arnoldi*

In the transformation with Householder matrices we construct the structure (triangular/Hessenberg) by applying orthogonal transformations to the whole matrix, in the Gram-Schmidt/Arnoldi method we construct the structure (triangular/Hessenberg) column by column without ever transforming the matrix itself.

Ansatz: Let $Q = [q_1, \ldots, q_n]$ be unitary and H a Hessenberg matrix. We want $Q^H AQ = H$ or

$$A\,[q_1, \ldots, q_n] = [q_1, \ldots, q_n] \begin{bmatrix} h_{11} & \ldots & \ldots & h_{1n} \\ h_{21} & & & \vdots \\ & \ddots & & \vdots \\ 0 & & h_{n,n-1} & h_{nn} \end{bmatrix},$$

respectively. Comparing the k-th columns we obtain

$$Aq_k = h_{1k}q_1 + \ldots + h_{kk}q_k + h_{k+1,k}q_{k+1}.$$

Thus,

$$q_{k+1} = \frac{1}{h_{k+1,k}} \left(Aq_k - \sum_{i=1}^{k} h_{ik}q_i \right).$$

Due to the orthonormality of the q_i we get

$$q_j^H Aq_k = h_{jk}, \quad j = 1, \ldots, k$$

Thus we can determine q_{k+1} from $q_1, \ldots, q_k$ if $h_{k+1,k} \neq 0$, which holds if H is unreduced.

Algorithm (Arnoldi, 1951):
Determines for $x \in \mathbb{C}^n \setminus \{0\}$ and $A \in \mathbb{C}^{n,n}$ a unitary matrix $Q = [q_1, \ldots, q_n]$, such that $Q^{-1}AQ = H$ is in Hessenberg form.

1) Start: $q_1 = \frac{x}{||x||}$.

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

   (a) $\tilde{q}_{k+1} := Aq_k - \sum_{i=1}^{k} h_{ik}q_i, h_{ik} = q_i^H Aq_k.$

   (b) $h_{k+1,k} := ||\tilde{q}_{k+1}||.$

   (c) $q_{k+1} = \frac{1}{h_{k+1,k}} \tilde{q}_{k+1}.$

Remark 47 *(a) The algorithm stops if $h_{m+1,m} = 0$ for some m (*good breakdown*). Then*

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

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

$$Aq_m = \sum_{j=1}^{m} h_{jm}q_j.$$

Thus,

$$A\,[q_1,\ldots,q_m] = [q_1,\ldots,q_m] \underbrace{\begin{bmatrix} h_{11} & \ldots & h_{1,m-1} & h_{1m} \\ h_{21} & & & \vdots \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \ldots & h_{m,m-1} & h_{mm} \end{bmatrix}}_{=:H_m},$$

i.e., the subspace Span$\{q_1,\ldots,q_m\}$ is A-invariant.

(b) If $h_{m+1,m} \neq 0$ then

$$A\,[q_1,\ldots,q_m] = [q_1,\ldots,q_{m+1}] \begin{bmatrix} h_{11} & \ldots & h_{1m} \\ h_{21} & \ddots & \vdots \\ 0 & \ddots & \vdots \\ 0 & \ldots & h_{m+1,m} \end{bmatrix}.$$

In other words we have

$$\begin{aligned} AQ_m &= Q_m H_m + q_{m+1}\,[0,\ldots,0,h_{m+1,m}] \\ &= Q_m H_m + h_{m+1,m} q_{m+1} e_m^T \ . \end{aligned}$$

This is called the Arnoldi relation*. Due to the orthonormality of the q_i we also have $Q_m^H A Q_m = H_m$.*

Consequence: Due to the relation between Hessenberg matrices and Krylov spaces it follows that

$$\text{Span}\,\{q_1,\ldots,q_l\} = \mathcal{K}_l\,(A,x)$$

for $l = 1,\ldots,m+1$. Thus the Arnoldi algorithm computes orthonormal bases of Krylov spaces.

Application: The Arnoldi algorithm as projection method for $A \in \mathbb{C}^{n,n}$, n large.

1) For a given start vector $x \neq 0$ compute the vectors $q_1, q_2, \ldots$ with the Arnoldi method.

2) Stop after $m \ll n$ steps

   (a) either due a good breakdown in step m;

   (b) or because we cannot store more vectors or because we have convergence. Note that the computation of

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

   is more and more expensive in each further step (concerning flops and storage).

This yields an orthonormal basis of the Krylov spaces:

$$\mathcal{K}_l(A,x) = \mathrm{Span}\{q_1,\ldots,q_l\} = \mathcal{R}(Q_l), \quad l = 1,\ldots,m$$

3) If $h_{m+1,m} = 0$, then $\mathcal{R}(Q_m) = \mathcal{K}_m(A,x)$ is invariant.

4) If $h_{m+1,m} \neq 0$, then choose $\mathcal{K}_m(A,x) = \mathcal{R}(Q_m)$ as search and test space for a projection method, i.e., determine $\mu \in \mathbb{C}$ and $v \in \mathcal{R}(Q_m) \setminus \{0\}$ with

$$Av - \mu v \perp \mathcal{R}(Q_m).$$

Hope: Since $\mathcal{R}(Q_m)$ is a Krylov space, some of the (μ, v) are good approximations to eigenvalue/eigenvector pairs of A.

Definition 48 *Let $A \in \mathbb{C}^{n,n}$ and $\mathbb{C}^n \supset \mathcal{K} \neq \{0\}$ be a subspace. Then $(\mu, v) \in \mathbb{C}, \mathbb{C}^n$ is called a* Ritz pair of A with respect to $\mathcal{K}$, *μ is called* Ritz value *and v* Ritz vector *if*

$$v \in \mathcal{K} \setminus \{0\} \quad \textit{und} \quad Av - \mu v \perp \mathcal{K}.$$

The Ritz pairs are obtained from the eigenvalues of $H_m = Q_m^H A Q_m \in \mathbb{C}^{m,m}$. This follows form the following lemma.

Lemma 49 *Let $A \in \mathbb{C}^{n,n}$, let $Q_m \in \mathbb{C}^{n,m}$ be isometric, $\mu \in \mathbb{C}$, $z \in \mathbb{C}^m$ and $v = Q_m z$. Then*

$$Q_m^H A Q_m z = \mu z \quad \Longleftrightarrow \quad Av - \mu v \perp \mathcal{R}(Q_m).$$

Proof:

$$Q_m^H A Q_m z = \mu z = \mu Q_m^H Q_m z \quad \Longleftrightarrow \quad Q_m^H (Av - \mu v) = 0 \quad \Longleftrightarrow \quad Av - \mu v \perp \mathcal{R}(Q_m).$$

□

Remark 50 *We can compute the eigenvalues of H_m by the Francis QR algorithm and here we can exploit that H_m is already in Hessenberg form. To compute the eigenvectors, we carry out one step of inverse iteration with a computed eigenvalue $\tilde{\mu}$ of H_m as shift, i.e., we choose a start vector $w_0 \in \mathbb{C}^m$, $||w_0|| = 1$, solve*

$$(H_m - \tilde{\mu} I_m)\, \tilde{w}_1 = w_0$$

for $\tilde{w}_1$ and set $w_1 = \frac{\tilde{w}_1}{||\tilde{w}_1||}$. Since $\tilde{\mu}$ is already a good eigenvalue approximation, i.e., a good shift, one step is usually enough.

To check whether a Ritz pair is a good approximation, we can compute the residual. A small residual means a small backward error, i.e., if $Av - \mu v$ is small and the eigenvalue is well-conditioned, then (μ, v) is a good approximation to an eigenvalue/eigenvector pair of A.

Theorem 51 *Let $A, Q_m, H_m, h_{m+1,m}$ be the results of m steps of the Arnoldi algorithm. Furthermore, let $z = [z_1,\ldots,z_m]^T \in \mathbb{C}^m$ be an eigenvector of H_m associated with $\mu \in \mathbb{C}$. Then (μ, v), with $v = Q_m z$, is a Ritz pair of A with respect to $\mathcal{R}(Q_m)$ and*

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

Proof: Using the Arnoldi relation, it follows that

$$\begin{aligned} Av - \mu v &= AQ_m z - \mu Q_m z \\ &= \left(Q_m H_m + h_{m+1,m} q_{m+1} e_m^T\right) z - \mu Q_m z \\ &= \underbrace{Q_m \left(H_m z - \mu z\right)}_{=0} + h_{m+1,m} z_m q_{m+1} \\ \Rightarrow ||Av - \mu v|| &= |h_{m+1,m}||z_m|. \end{aligned}$$

□

Remark 52 *(a) For the computation of the residual $Av - \mu v$ we do not need to determine the Ritz vector v explicitly.*

(b) After some iterations in finite precision arithmetic the orthonormality of the q_i deteriorates. This happens in particular when $|h_{m+1,m}|$ is small. Then the Ritz values deteriorate as well. This can be fixed by re-orthonormalization *using the modified Gram-Schmidt method for $q_1, \ldots, q_m$.*

(c) We know how to detect good approximations but we are not sure that they occur.

3.3 The Symmetric Lanczos Algorithm

Special case: $A = A^H \in \mathbb{C}^{n,n}$ is Hermitian.
Let $H = Q^H A Q$ be in Hessenberg form, then $T := H$ is tridiagonal. Suppose we have computed T with the Arnoldi algorithm. Then

$$T = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & \ldots & 0 \\ \beta_1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \beta_{n-1} \\ 0 & \ldots & 0 & \beta_{n-1} & \alpha_n \end{bmatrix} \in \mathbb{R}^{n,n}$$

is even real, since the diagonal of an Hermitian matrix is real and, furthermore ,we have $\beta_m = h_{m+1,m} = ||\tilde{q}_{m+1}||$ for $m = 1, \ldots, n-1$. With $Q = [q_1, \ldots, q_m]$, comparing the columns in $AQ = QT$, we obtain that

$$\begin{aligned} Aq_1 &= \alpha_1 q_1 + \beta_1 q_2 \\ Aq_k &= \beta_{k-1} q_{k-1} + \alpha_k q_k + \beta_k q_{k+1}, \quad k = 2, \ldots, n-1 \\ Aq_n &= \beta_{n-1} q_{n-1} + \alpha_n q_n \end{aligned}$$

This is called a *3-term recursion.* Since $q_1, \ldots, q_n$ are orthonormal we have, furthermore, that

$$\alpha_k = q_k^H A q_k.$$

Algorithm (Lanczos, 1950)
Given $A^H = A \in \mathbb{C}^{n,n}$.

1) Start: Choose $x \neq 0$. Then set $q_0 := 0, q_1 := \dfrac{x}{||x||}$ and $\beta_0 := 0$.

2) For $k = 1, 2, \ldots, m$

   (a) $\alpha_k := q_k^H A q_k$,

   (b) $r_k = A q_k - \beta_{k-1} q_{k-1} - \alpha_k q_k$,

   (c) If $r_k = 0$ STOP. Otherwise set $\beta_k := ||r_k||$ and $q_{k+1} := \dfrac{1}{\beta_k} r_k$.

Remark 53 *(a) The Lanczos algorithm is essentially the Arnoldi algorithm for $A = A^H$.*

(b) As in the Arnoldi algorithm, after m steps we have (due to Lemma 46) that

$$Span\{q_1, \ldots, q_l\} = \mathcal{K}_l(A, x), \quad l = 1, \ldots, m$$

The eigenvalue/eigenvector pairs of the matrix

$$T_m = \begin{bmatrix} \alpha_1 & \beta_1 & & 0 \\ \beta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_{m-1} \\ 0 & & \beta_{m-1} & \alpha_m \end{bmatrix}$$

yield Ritz pairs for A with respect to $\mathcal{R}(Q_m)$.

(c) As in the Arnoldi method, in finite precision arithmetic the orthonormality of $q_1, \ldots, q_m$ deteriorates and re-orthonormalization is necessary. Otherwise one obtains spurious eigenvalues.

(d) Due to the 3-term recursion we do no need to store more than three vectors q_i, thus we can choose m much bigger than in the Arnoldi algorithms. However, if we do not store the q_i then we cannot re-orthonormalize. The it is necessary to detect the spurious eigenvalues and remove them. (Cullum-Willoughby method 1979.)

3.4 The Nonsymmetric Lanczos Algorithm

A disadvantage of Arnoldi algorithm is that the recursion becomes more and more expensive the more steps we perform. So we might ask whether there also exists a 3-term recursion for $A \neq A^H$?

Idea: Transform A to tridiagonal form and allow the transformation matrix to become non-unitary, i.e., we want to determine

$$X^{-1} A X = T$$

with T tridiagonal, or equivalently

$$AX = X \begin{bmatrix} \alpha_1 & \gamma_1 & & 0 \\ \beta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \gamma_{n-1} \\ 0 & & \beta_{n-1} & \alpha_n \end{bmatrix}.$$

If we write X as

$$X = [x_1, \ldots, x_k],$$

then we obtain

$$Ax_k = \gamma_{k-1}x_{k-1} + \alpha_k x_k + \beta_k x_{k+1}$$

for $k = 1, \ldots, n-1$ and with $\gamma_0 := 0, x_0 := 0$. Furthermore,

$$T^H = X^H A^H X^{-*} = Y^{-1} A^H Y$$

with $Y := X^{-*}$. Then clearly $Y^H X = I$, i.e., $y_j^H x_i = \delta_{ij}$ with

$$Y = [y_1, \ldots, y_n].$$

Such families of vectors $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ are called *bi-orthogonal*. We again compare columns in $A^H Y = Y T^H$. This means that

$$A^H y_k = \bar{\beta}_{k-1} y_{k-1} + \bar{\alpha}_k y_k + \bar{\gamma}_k y_{k+1}$$

for $k = 1, \ldots, n-1$ and $\beta_0 := 0, y_0 := 0$. Since $Y^H X = I$, it follows that

$$\alpha_k = y_k^H A x_k.$$

Moreover,

$$\beta_k x_{k+1} = Ax_k - \gamma_{k-1}x_{k-1} - \alpha_k x_k =: r_k,$$

and

$$\bar{\gamma}_k y_{k+1} = A^H y_k - \bar{\beta}_{k-1} y_{k-1} - \bar{\alpha}_k y_k =: s_k.$$

Furthermore, we have

$$1 = y_{k+1}^H x_{k+1} = \frac{1}{\beta_k \gamma_k} s_k^H r_k$$

for all $k \geq 1$. But there is still freedom in the computation of β_k, γ_k. We could in principle use one of the variants

$$\beta_k = ||r_k||, \gamma_k = ||s_k||, \beta_k = \gamma_k, \ldots$$

Algorithm (Nonsymmetric Lanczos algorithm, with choice $\beta_k = ||r_k||$)
Given: $A \in \mathbb{C}^{n,n}$, $x_1, y_1 \in \mathbb{C}^n$ with $y_1^H x_1 = 1$.

1) Start: $\beta_0 := 0, \gamma_0 := 0, x_0 := 0, y_0 := 0$

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

$$\begin{aligned} \alpha_k &:= y_k^H A x_k, \\ r_k &:= Ax_k - \gamma_{k-1}x_{k-1} - \alpha_k x_k, \\ s_k &:= A^H y_k - \bar{\beta}_{k-1} y_{k-1} - \bar{\alpha}_k y_k, \end{aligned}$$

If $s_k^H r_k = 0$, then STOP. Otherwise

$$\begin{aligned} \beta_k &= ||r_k||, \\ \gamma_k &= \frac{1}{\beta_k} s_k^H r_k, \\ x_{k+1} &= \frac{1}{\beta_k} r_k, \\ y_{k+1} &= \frac{1}{\gamma_k} s_k. \end{aligned}$$

Remark 54 *1) After m steps, if we do not have a breakdown (since we cannot divide by 0), we have*

$$A\underbrace{[x_1,\ldots,x_m]}_{:=X_m} = [x_1,\ldots,x_m,x_{m+1}]\begin{bmatrix} \alpha_1 & \gamma_1 & & 0 \\ \beta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \gamma_{m-1} \\ & & \beta_{m-1} & \alpha_m \\ 0 & & & \beta_m \end{bmatrix}.$$

If we denote the submatrix consisting of the first m rows with T_m, then we have

$$AX_m = X_m T_m + \beta_m x_{m+1} e_m^T$$

and analogously

$$A^H Y_m = Y_m T_m^H + \gamma_m^H y_{m+1} e_m^T.$$

2) Due to the relationship with Krylov spaces and Hessenberg matrices, we have

$$Span\{x_1,\ldots,x_l\} = \mathcal{K}_l(A,x_1), \quad l=1,\ldots,m,$$

and

$$Span\{y_1,\ldots,y_l\} = \mathcal{K}_l\left(A^H,y_1\right), \quad l=1,\ldots,m.$$

3) For $A = A^H$ and $x_1 = y_1$, i.e., $||x_1|| = 1$, the algorithm is identical to the symmetric Lanczos algorithm.

4) The nonsymmetric Lanczos algorithm breaks down with a serious breakdown *if $s_k^H r_k = 0$ or is very small in modulus.*

- *(a) If $r_k = 0$, then Span$\{x_1,\ldots,x_k\}$ is an A-invariant subspace (good breakdown).*
- *(b) If $s_k = 0$, then Span$\{y_1,\ldots,y_k\}$ is an A^H-invariant subspace.*
- *(c) If $s_k^H r_k = 0$, but $s_k, r_k \neq 0$, then the algorithm breaks down without delivering information about invariant subspaces. There are tricks to fix this problem in some but not all cases (look ahead Lanczos, see below), but in general this algorithm has to be used with great care.*

5) The nonsymmetric Lanczos algorithm is not numerically stable, whenever $s_k^H r_k \approx 0$.

6) **The Lanczos algorithm as projection method.** *After $m \ll n$ steps we have*

$$AX_m = X_m T_m + \beta_m\left(x_{m+1}e_m^T\right).$$

Since $Y_m^H X_m = I_m$, we have

$$T_m = Y_m^H A X_m.$$

Some of the eigenvalues of this matrix are typically good approximations to an eigenvalue of A. Let $\mu \in \mathbb{C}$ and $z \in \mathbb{C}^m$, $v = X_m z$. If (μ, z) is an eigenvalue/eigenvector pair of T_m, then

$$Y_m^H A X_m z = T_m z = \mu z = \mu Y_m^H X_m z$$

and this is equivalent to

$$Y_m^H \left(AX_m z - \mu X_m z \right) = 0.$$

We thus have $Y_m^H \left(Av - \mu v \right) = 0$ *or*

$$Av - \mu v \perp \mathcal{R}\left(Y_m \right).$$

Such a pair (μ, v) *is often called* Petrov pair. *Here we use* $\mathcal{R}\left(X_m \right) = \mathcal{K}_m \left(A, x_1 \right)$ *as search space and* $\mathcal{R}\left(Y_m \right) = \mathcal{K}\left(A^H, y_1 \right)$ *as test space.*

7) **The Lanczos algorithm in practice**

(a) In the look-ahead-variant one weakens the bi-orthonormality to avoid the serious breakdowns (QMR algorithm).

(b) The bi-orthonormality deteriorates due to round-off errors in finite precision arithmetic.

3.5 Convergence of Krylov Space Methods

For $A \in \mathbb{C}^{n,n}$ and $x \in \mathbb{C}^n$ Krylov space methods construct $X_m = [x_1, \ldots, x_m] \in \mathbb{C}^{n,m}$ such that

$$\operatorname{Span}\left\{ x_1, \ldots, x_l \right\} = \mathcal{K}_l \left(A, x_1 \right), \quad l = 1, \ldots m$$

We stop the method after $m \ll n$ steps and choose $\mathcal{K} = \mathcal{K}_m \left(A, x_1 \right)$ as search space together with a test space $\mathcal{L} \subset \mathbb{C}^n$ and then we compute pairs

$$\mu \in \mathbb{C}, \quad v \in \mathcal{K} \setminus \{0\},$$

such that

$$Av - \mu v \perp \mathcal{L}.$$

In the Arnoldi and the symmetric Lanczos algorithm we choose $\mathcal{L} = \mathcal{K}$, in the nonsymmetric Lanczos method $\mathcal{L} = \mathcal{K}_m \left(A^H, y_1 \right)$.

Then the obvious question is whether in the computed pairs (μ, v) there are good approximations to eigenvalue/eigenvector pairs?

Lemma 55 *Denoting by* Π_{m-1} *the set of polynomials of degree less than or equal to* $m-1$, *then*

$$\mathcal{K}_m \left(A, x \right) = \left\{ p(A)x \,|\, p \in \Pi_{m-1} \right\}.$$

Proof: Let $w \in \mathcal{K}_m \left(A, x \right)$. Then there exist $\alpha_0, \ldots, \alpha_{m-1} \in \mathbb{C}$ with

$$w = \alpha_0 x + \alpha_1 Ax + \ldots + \alpha_{m-1} A^{m-1} x = p\left(A \right) x,$$

where $p\left(t \right) = \alpha_0 + \alpha_1 t + \ldots + \alpha_{m-1} t^{m-1}$. □

The convergence is typically very good for eigenvalue in the outer part of the spectrum and very slow for the eigenvalues in the interior. Quantitative results using Lemma 55 are based on optimal polynomial approximations, but a complete convergence analysis in all cases is an open problem.

3.6 The Implicitely Restarted Arnoldi algorithm

The Arnoldi algorithm becomes more and more expensive the more iterations one performs, and the alternative nonsymmetric Lanczos is unstable. Can we resolve the problem with Arnoldi algorithm?

Ideas:

1) **Use restarts**: After m steps of the Arnoldi algorithm with startvector x choose $p \in \Pi_{m-1}$ with

$$|p(\lambda_i)| = \begin{cases} \text{large} & \text{for desired} \lambda_i, \\ \text{small} & \text{for undesired} \lambda_i. \end{cases}$$

Then choose $p(A)x$ as new start vector and restart the Arnoldi algorithm. Since the start vector has been enlarged in the components of the direction of the desired eigenvectors, we except that the Krylov spaces contain, in particular, good approximations to the desired eigenvectors.

The disadvantage is that we start with a single new vector, we loose a lot of already obtained information. It would be ideal if we choose the new start vector $p(A)x$ optimally in the sense that we keep as much information as possible, i.e., maximally many approximation to desired eigenvalues. This is again an open problem.

2) To preserve more information, we proceed as follows. If k eigenvalues are desired then we run the Arnoldi algorithm for $m = k + l$ steps. Then keep k vectors and throw away l vectors. This leads to the

IRA (*implicitly restarted Arnoldi method*), Sorensen, 1992.

The strategy of the implicitly restarted Arnoldi algorithm is as follows:

1) After m steps of the Arnoldi algorithm (without good breakdown) we have the Arnoldi relation

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

$Q_m = [q_1, \ldots, q_m]$ is isometric and $H_m \in \mathbb{C}^{m,m}$ is in Hessenberg form.

2) Choose l Shifts $\nu_1, \ldots, \nu_l \in \mathbb{C}$. (Details will be described later!)

3) Carry out l steps of the QR algorithm with shifts $\nu_1, \ldots, \nu_l$.

$$\begin{aligned} H^{(1)} &= H_m \\ \text{For } j &= 1, \ldots, l, \\ &\quad H^{(j)} - \nu_j I = U_j R_j \quad \text{(QR decomposition)} \\ &\quad H^{(j+1)} = R_j U_j + \nu_j I \\ \text{end} & \\ \hat{H}_m &= H^{(l+1)} \\ U &= U_1 \ldots U_l \end{aligned}$$

Then $\hat{H}_m = U^H H_m U$ and furthermore every U_i has the form $U_i = G_{12}^{(i)} \ldots G_{m-1,m}^{(i)}$ with Givens rotation matrices $G_{12}^{(i)}, \ldots, G_{m-1,m}^{(i)}$. Every U_i is a Hessenberg matrix, thus U is a product of l Hessenberg matrices and hence a band matrix with lower bandwidth l.

4) We have $AQ_m = Q_m H_m + f_{m+1} e_m^T$ with $f_{m+1} = h_{m+1,m} q_{m+1}$. Hence,

$$A \underbrace{Q_m U}_{=:\hat{Q}_m} = Q_m U \underbrace{U^H H_m U}_{\hat{H}_m} + f_{m+1} \underbrace{e_m^T U}_{=:u_m^T},$$

where u_m^T is the last column of U. Thus, we have

$$A\hat{Q}_m = \hat{Q}_m \hat{H}_m + f_{m+1} u_m^T$$

with

$$u_m^T = [\underbrace{0, \ldots, 0}_{k-1}, \alpha, \underbrace{*, \ldots, *}_{l}].$$

for some $\alpha \in \mathbb{C}$. We then partition $\hat{Q}_m$ as

$$\hat{Q}_m = \left[\begin{array}{cc} \hat{Q}_k & \tilde{Q}_l \end{array} \right] = [\hat{q}_1, \ldots, \hat{q}_m]$$

and set $\hat{Q}_j = [\hat{q}_1, \ldots, \hat{q}_j]$ for $j = 1, \ldots, m$. In the same way we partition $\hat{H}_m$. Then

$$A \left[\hat{Q}_k \, \tilde{Q}_l \right] = \left[\hat{Q}_k \, \tilde{Q}_l \right] \left[\begin{array}{cc} \hat{H}_k & * \\ \beta e_1 e_k^T & \tilde{H}_l \end{array} \right] + f_{m+1} \left[0, \ldots, 0, \alpha, *, \ldots, * \right].$$

We then delete the last l columns and obtain

$$\begin{aligned} A\hat{Q}_k &= \hat{Q}_k \hat{H}_k + \beta \tilde{Q}_l e_1 e_k^T + f_{m+1} \left[0, \ldots, 0, \alpha \right] \\ &= \hat{Q}_k \hat{H}_k + \underbrace{\left(\beta \tilde{Q}_l e_1 + \alpha f_{m+1} \right)}_{=:\hat{f}_{k+1}} e_k^T \\ &= \hat{Q}_k \hat{H}_k + \hat{f}_{k+1} e_k^T. \end{aligned}$$

This is again an Arnoldi relation, since one easily see that $\hat{Q}_k^H \hat{f}_{k+1}$. In particular, this Arnoldi relation is the same as that after k steps of the restarted Arnoldi algorithm with the restart vector $\hat{q}_1$. This is the reason for the terminology *'implicit restart'*.

5) Then we perform l further Arrnoldi steps and begin again with 1) until sufficiently many eigenvalues are found.

This concept works very nicely but some questions remain.

1) How to choose the shifts $\nu_1, \ldots, \nu_l$?

2) What is the relation between q_1 and $\hat{q}_1$?

3) What is the relation between the corresponding Krylov spaces?

Lemma 56 *Let $p(t) = (t - \nu_1) \ldots (t - \nu_l)$. Then, with the notation and assumptions above,*

$$p(A) Q_m = Q_m U R + F_m,$$

where $F_m = \left[\begin{array}{cc} 0 & \tilde{F}_m \end{array} \right]$ and $R = R_k \ldots R_1$.

Proof: We have already shown that

$$p(H_m) = (H_m - \nu_l I)\dots(H_m - \nu_1 I) = UR.$$

Then we show via induction on l that

$$p(A)\,Q_m = Q_m p(H_m) + F_m.$$

'$l = 1$': We have the Arnoldi relation $AQ_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^T$. Then

$$(A - \nu_1 I)\,Q_m = Q_m\,(H_m - \nu_1 I) + \underbrace{h_{m+1,m} q_{m+1} e_m^T}_{=:F_1}.$$

'$l - 1 \Rightarrow l$': We have

$$\begin{aligned}
& p(A)\,Q_m \\
= \;& (A - \nu_1 I)\,(A - \nu_2 I)\dots(A - \nu_l I)\,Q_m \\
\overset{\text{I.V.}}{=} \;& (A - \nu_1 I)\left(Q_m\,(H_m - \nu_2 I)\dots(H_m - \nu_l I) + F_{m-1}\right) \\
= \;& \left[Q_m\,(H_m - \nu_1 I) + h_{m+1,m} q_{m+1} e_m^T\right](H_m - \nu_2 I)\dots(H_m - \nu_l I) + (A - \nu_1 I)\,F_{m-1} \\
= \;& Q_m p(H_m) + \underbrace{h_{m+1,m} q_{m+1}\, e_m^T \underbrace{(H_m - \nu_2 I)\dots(H_m - \nu_l I)}_{\text{has band width } l-1}}_{= \left[0,\dots,0,\underbrace{*,\dots,*}_{l}\right]} + \underbrace{\left[\begin{array}{cccc} m-l+1 & & l-1 & \mathcal{R} \end{array}\;\; 0\;\; (A - \nu_1 I)\,\tilde{F}_{m-1}\right]}_{=:F_m},
\end{aligned}$$

where $F_m = \left[\begin{array}{cc} 0 & \tilde{F}_m \end{array}\right]$ has the desired form. □

Theorem 57 *With the notation and assumptions above*

1) $\hat{q}_1 = \alpha p(A)\,q_1$ *for a* $\alpha \in \mathbb{C}$.

2) $\mathcal{R}(\hat{Q}_j) = p(A)\,\mathcal{R}(Q_j) = \mathcal{R}(p(A)\,Q_j)$, *for* $j = 1,\dots,m$.

Proof:

1) One has

$$\begin{aligned}
p(A)\,Q_m &= \underbrace{Q_m U}_{=\hat{Q}_m} R + F_m = \hat{Q}_m R + F_m \\
&= \hat{Q}_m \begin{bmatrix} r_{11} & \dots & r_{1m} \\ & \ddots & \vdots \\ 0 & & r_{mm} \end{bmatrix} + \left[\begin{array}{cc} 0 & \tilde{F}_m \end{array}\right].
\end{aligned}$$

Comparing the first columns yields

$$p(A)\,q_1 = \hat{q}_1 r_{11}.$$

Therfore, choose $\alpha = 1/r_{11}$. Here $r_{11} \neq 0$, otherwise we would have $p(A)\, q_1 = 0$, i.e., the minimal polynomial of q_1 with respect to A would have degree $\leq l$. But then the Krylov space $\mathcal{K}_l(A, q_1)$ is invariant, i.e., the Arnoldi would have had a good breakdown after l steps which is a contradiction to the fact that we have done $m > l$ steps without breakdown.

2) We could proceed as in 1) but we use the following approach: Since we know already that

$$\mathcal{R}(Q_j) = \mathcal{K}_j(A, q_1), \quad j = 1, \ldots, m$$

and

$$\mathcal{R}(\hat{Q}_j) = \mathcal{K}_j(A, \hat{q}_1), \quad j = 1, \ldots, m$$

it follows that

$$\begin{aligned} \mathcal{R}(\hat{Q}_j) &= \operatorname{Span}\left\{p(A)\, q_1, Ap(A)\, q_1, \ldots, A^{j-1}p(A)\, q_1\right\} \\ &= \operatorname{Span}\left\{p(A)\, q_1, p(A)\, Aq_1, \ldots, p(A)\, A^{j-1}q_1\right\} \\ &= p(A) \operatorname{Span}\left\{q_1, Aq_1, \ldots, A^{j-1}q_1\right\} \\ &= p(A)\, \mathcal{R}(Q_j) \end{aligned}$$

□

As a consequence we obtain that $\mathcal{R}(\hat{Q}_j) = (A - \nu_1 I) \ldots (A - \nu_l I)\, \mathcal{R}(Q_j), \quad j = 1, \ldots, m.$ This correspond to l steps of subspace iteration with shifts $\nu_1, \ldots, \nu_l$. Then we can expect that $\mathcal{K}_j(A, q_1)$ contains better approximations to eigenvectors than $\mathcal{K}_j(A, q_1)$.
Choice of shifts: Suppose that A is diagonalizable and $(v_1, \ldots, v_n)$ is a basis of eigenvectors to eigenvalues $\lambda_1, \ldots, \lambda_n$. Then

$$q_1 = c_1 v_1 + \ldots + c_n v_n \quad \text{with } c_i \in \mathbb{C}.$$

Since $\hat{q}_1 = \alpha p(A) q_1$ it follows that

$$\hat{q}_1 = \alpha c_1 p(\lambda_1)\, v_1 + \ldots + \alpha c_n p(\lambda_n)\, v_n.$$

But then $|p(\lambda_i)|$ is large (small) if λ_i if far from ν_j (respectively near to a) ν_j. The component to eigenvalues from $\nu_1, \ldots, \nu_l$ will be enlarged $\hat{q}_1$. Choose $\nu_1, \ldots, \nu_l$ far away of the desired eigenvalues.

Example 58 *Search k eigenvalues $\mu \in \mathbb{C}$. Compute the $m = k + l$ eigenvalues of H_m and choose the l eigenvalues of H_m furthest away from μ as shifts.*

Remark 59 *1)* Locking and purging. *If (λ, v) is a converged Ritz pair, then we would like to avoid that further Ritz pairs converge to this one. This can be done in different ways.*

(a) If λ is desired then v is locked by working in the orthogonal complement.

(b) If λ is undesired then v is removed (purged).

Details can be found in Lehoucq/Sorensen 1996 'Deflation techniques for an IRA iteration'.

2) The function `eigs` *in* MATLAB *is based on the ARPACK package that contains the best implementation of the implicitly restarted Arnoldi method, see also the ARPACK user guide.*

3.7 The Jacobi-Davidson Algorithm

If only a specific pair (λ, u) of a large sparse matrix $A \in \mathbb{C}^{n,n}$ is desired (e.g., the eigenvalue largest in modulus or the one nearest to $\tilde{\mu} \in \mathbb{C}$, then alternative methods can be considered.

1. Idea. Apply the Arnoldi algorithm with shift-and-invert, i.e., apply the Arnoldi algorithm to $(A - \nu I)^{-1}$. This has the disadvantage that per step we have to solve a linear system with $A - \nu I$. In practice, this has to be done very accurately, otherwise the convergence deteriorates. If a sparse LR decomposition of $(A - \nu I)^{-1}$ can be determined, then this is acceptable otherwise this is problematic.

2. Idea. We use again a projection method. We choose an isometric matrix $Q_m \in \mathbb{C}^{n,m}$ with $m \ll n$ and use $\mathcal{R}(Q_m)$ as search and test space. Then we have again

$$Q_m^H A \underbrace{Q_m z}_{=:v} = \mu z \quad \Leftrightarrow \quad Av - \mu v \perp \mathcal{R}(Q_m),$$

i.e., we obtain Ritz pairs (μ, v) from $Q_m^H A Q_m$.
The new idea is that $\mathcal{R}(Q_m)$ does not have to be a Krylov space.

This leads to the following task: For $q_1 \in \mathbb{C}^n$, $||q_1|| = 1$ construct an isometric matrix

$$Q_m = [q_1, \ldots, q_m],$$

such that (λ, u) is quickly approximated by Ritz pairs of A with respect too $\mathcal{R}(Q_m)$. When $q_1, \ldots, q_k$ are constructed, then we compute Ritz pairs of A with respect to $\mathcal{R}(Q_k)$. Let (μ_k, v_k) be the Ritz pair with $||v_k|| = 1$ that approximates (λ, u) best.
Ansatz:

$$\begin{aligned} u &= v_k + x, \qquad \text{with } x \perp v_k, \\ \lambda &= \mu_k + \eta. \end{aligned}$$

Here the unknown u is scaled so that $(u - v_k) \perp v_k$. This can be achieved by determining an approximation to x and computing q_{k+1} from x by orthonormalizing against $q_1, \ldots, q_k$.
Computation of x**.** Consider

$$P = I - v_k v_k^H.$$

This is the orthogonal projection to $\text{Span}(v_k)^\perp$. Furthermore, let

$$r_k := Av_k - \mu_k v_k$$

be the residual of (μ_k, v_k) with respect to A. Then

$$Pv_k = 0, \quad Px = x, \quad Pr_k = r_k,$$

since (μ_k, v_k) is a Ritz pair, and hence $r_k \perp \mathcal{R}(Q_k) \ni v_k$, i.e., $r_k \perp v_k$. Moreover,

$$\begin{aligned} & Au &&= \lambda u \\ \Longleftrightarrow \quad & A(v_k + x) &&= \lambda(v_k + x) \\ \Longleftrightarrow \quad & (A - \lambda I)x &&= -(A - \lambda I)v_k \\ & &&= -(A - \mu_k I)v_k + \eta v_k = -r_k + \eta v_k \\ \Longleftrightarrow \quad & P(A - \lambda I)x &&= -Pr_k = -r_k \\ \Longleftrightarrow \quad & P(A - \lambda I)Px &&= -r_k \quad \text{and } x \perp v_k. \end{aligned}$$

Unfortunately we cannot solve the last equation, since we do not know λ. Therefore, we replace λ by the Ritz value μ_k,

$$P\,(A - \mu_k I)\,Px = -r_k, \quad x \perp v_k. \tag{3.3}$$

This equation is called the *Jacobi correction equation.*

Remark 60 *$P\,(A - \mu_k I)\,P$ is the orthogonal projection of $A - \mu_k I$ on $Span\{v_k\}^\perp$.*

Algorithm: (Jacobi-Davidson algorithm, Sleijpen/van der Vorst, 1996)
For a given matrix $A \in \mathbb{C}^{n,n}$, this algorithm computes $Q_m \in \mathbb{C}^{n,m}$ isometric, such that $M_m = Q_m^H A Q_m$ contains good approximations to a given pair (λ, u).

1) Start: Choose $q_1 \in \mathbb{C}^n, ||q_1|| = 1$.

2) Iterate for $k = 1, 2, \ldots$ to convergence

   (a) $Q_k = [q_1, \ldots, q_k]\,,\ w_k = Aq_k,\ W_k = [w_1, \ldots, w_k] = AQ_k$ and $M_k = Q_k^H W_k$.

   (b) Compute the Ritz pairs (i.e., the eigenvalue/eigenvector pairs of M_k) and choose the Ritz pair (μ_k, v_k), that approximates (λ, u). Then $v_k = Q_k z$ for $z \in \mathbb{C}^k$.

   (c) Compute $r_k = W_k z - \mu_k Q_k z$. (This is the residual, since $r_k = W_k z - \mu_k Q_k z = AQ_k z - \mu_k Q_k z = Av_k - \mu_k v_k$.)

   (d) If $||r_k||$ is sufficiently small, then we stop the iteration since we have a converged eigenvalue/eigenvector pair. Otherwise solve the Jacobi correction equation (3.3) for x.

   (e) Compute q_{k+1} from x by orthonormalization against $q_1, \ldots, q_k$.

Interpretation: Since $Px = x$ it follows from the Jacobi correction equation (3.3) that

$$(A - \mu_k I)\,x = -r_k + \alpha v_k$$

with $\alpha = v_k^H\,(A - \mu_k I)\,x$. Hence,

$$\begin{aligned} x &= -\,(A - \mu_k I)^{-1}\,r_k + \alpha\,(A - \mu_k I)^{-1}\,v_k \\ &= -v_k + \alpha\,(A - \mu_k I)^{-1}\,v_k \end{aligned}$$

Since v_k is already in the search space, we have extended $\mathcal{R}\,(Q_k)$ by the scaled vector $(A - \mu_k I)^{-1}\,v_k$. Since

$$v_k^H\,(Av_k - \mu_k v_k) = 0, \quad \mu_k = \frac{v_k^H A v_k}{v_k^H v_k},$$

this corresponds to the vector that one obtains by applying one step of the Rayleigh quotient iteration to v_k, which converges cubically for $A = A^H$ and at least quadratically otherwise.

As a consequence in general, the Jacobi-Davidson method converges quadratically to an eigenvalue/eigenvector pair (λ, u) if the Jacobi correction equation is solved exactly in every step. The Jacobi-Davidson method can be interpreted as a Newton method.

Remark 61 *1) In general, it is too expensive to solve the correction equation exactly, instead one uses approximations such as*

(a) $P\,(A-\mu_k I)\,P \approx I$, *hence* $x = -r_k$. *This is formally equivalent to the Arnoldi algorithm (Exercise).*

(b) $P\,(A-\mu_k I)\,P \approx (D-\mu_k I)$ *where* D *is the diagonal part of* A.
(This is the Davidson method in quantum chemistry.) It works very well for diagonally dominant matrices.

(c) *Solve the correction equation with iterative methods (see next chapter).*

2) *The algorithm is called 'Jacobi-Davidson' since as in the Davidson algorithm it uses a stepwise extension of the search space. For the approximation* $P\,(A-\mu_k I)\,P \approx (D-\mu_k I)$ *we get back the Davidson algorithm.*

Otherwise the following ansatz is due to Jacobi.

$$A \begin{bmatrix} 1 \\ z \end{bmatrix} = \begin{bmatrix} \alpha & c^T \\ b & F \end{bmatrix} \begin{bmatrix} 1 \\ z \end{bmatrix} = \lambda \begin{bmatrix} 1 \\ z \end{bmatrix}.$$

This is equivalent to

$$\begin{aligned} \lambda &= \alpha + c^T z, \\ (F-\lambda I)\,z &= -b. \end{aligned}$$

Jacobi suggested to solve $(F-\lambda I)z = -b$ *iteratively and to compute an improved eigenvalue from* $\lambda = \alpha + c^T z$ *approximation. The basic idea is the search for corrections in the orthogonal complement, i.e.,* $[0, z^T]^T$ *is orthogonal to the ansatz vector* $[1,0]^T$. *In the Jacobi-Davidson-algorithm we search orthogonal to the last approximation.*

3) *We can compare the Jacobi-Davidson method and the Arnoldi algorithm with shift-and-Invert for the computation of a desired eigenvalue/eigenvector pair. Jacobi-Davidson consists in the solution of the Jacobi correction equation* $P\,(A-\mu_k I)\,Px = -r_k$ *and the computation of the new search direction. Then we must determine the new eigenvalue of* M_k *by applying the operator* A *in the step* $W_k = AQ_k$. *We can solve the Jacobi correction equation approximatively, since it is only used for the search direction. The spectral information of the operator* A *will be injected in the step* $W_k = AQ_k$ *where* M_k *and its eigenvalues are computed. When we solve Jacobi correction equation approximatively the convergence may be slowed down.*

(a) *In the Arnoldi algorithm in both steps, the computation of the search direction and the application of the operator are combined in the equation*

$$\tilde{q}_{k+1} = (A-\nu I)^{-1}\,q_k - \sum_{i=1}^{k} h_{ik} q_i.$$

The solution of the linear system has to be done very accurately. If this is not done accurately enough, then the information about the operator is not injected well enough. Furthermore, restarts, locking and purging is easier in the Jacobi-Davidson method, since we do not need to preserve an Arnoldi relation.

(b) *On the other hand the Jacobi-Davidson algorithm approximates only one pair at a time while the Arnoldi method determines several eigenvalues at the same time.*

3.8 Large Scale Generalized Eigenvalue Problems

For large scale generalized eigenvalue problems $\lambda Ex = Ax$ with regular pairs (E, A) we can apply all the methods from the previous sections whenever a solution of linear systems $(\lambda_0 E - A)x = b$ are feasible for chosen shifts λ_0.
Set $\mu = (\lambda + \lambda_0)^{-1}$ and replace $\lambda Ex = Ax$ by $\mu x = (\lambda_0 E - A)^{-1} Ex =: \hat{A}x$ and apply the chosen algorithm to the matrix $\hat{A}$. Whenever a matrix-vector multiplication with $\hat{A}$ is needed we have to carry out a matrix-vector multiplication with E and solve a linear system with $(\lambda_0 E - A)$.
If an eigenvalue/eigenvector pair (μ, x) has been computed then it corresponds to an eigenvalue $\lambda = \mu^{-1} - \lambda_0$ with the same eigenvector, using the spectral transformation $\lambda \to \mu = (\lambda + \lambda_0)^{-1}$. Using this relation we can make any eigenvalue an exterior eigenvalue by choosing an appropriate λ_0.
This spectral transformation can also be used in the case $E = I$ to achieve fast convergence near any chosen shift.

Chapter 4

Iterative methods for Large Sparse Linear Systems

Situation: $A \in \mathbb{C}^{n,n}$ sparse, n large, $b \in \mathbb{C}^n$.

Goal: Determine $x \in \mathbb{C}^n$ with $Ax = b$.

4.1 Splitting-Methods

The basic idea of splitting methods is to split the matrix A in two summands $A = M - N$ and to transfer the linear system to a fix-point equation

$$Mx = Nx + b.$$

This immediately leads to an iterative method via the recursion

$$Mx^{(k+1)} = Nx^{(k)} + b.$$

Remark 62 *(a) If A, M are nonsingular and $\varrho\left(M^{-1}N\right) < 1$, where*

$$\varrho\left(B\right) := \max_{\lambda \in \sigma(B)} |\lambda|,$$

then the iteration converges for every start vector x_0 to $A^{-1}b$ (Exercise).

(b) The convergence is linear with convergence rate $\varrho\left(M^{-1}N\right)$. This is typically too slow compared with other methods so that splitting methods are today used mostly only in the context of preconditioning.

Example 63 *We split A as* $A = \underbrace{\begin{bmatrix} 0 & \dots & 0 \\ * & \ddots & \vdots \\ * & * & 0 \end{bmatrix}}_{=:L} + \underbrace{\begin{bmatrix} * & & 0 \\ & \ddots & \\ 0 & & * \end{bmatrix}}_{=:D} + \underbrace{\begin{bmatrix} 0 & * & * \\ \vdots & \ddots & * \\ 0 & \dots & 0 \end{bmatrix}}_{=:R}.$

(a) Set $M = D$ and $N = -L - R$. This is called Jacobi-method.

(b) Set $M = L + D$ and $N = -R$. This is called Gauß-Seidel method.

4.2 The Conjugate Gradient Method (CG)

Special case: $A \in \mathbb{R}^{n,n}$ symmetric and positive definite, $b \in \mathbb{R}^n$.

Basic idea: Consider

$$\varphi : \mathbb{R}^n \to \mathbb{R},\ x \mapsto \varphi(x) = \frac{1}{2}x^T Ax - x^T b.$$

Then the gradient is $\nabla\varphi(x) = Ax - b$ the Hessian is $Hess\,\varphi(x) = A$, i.e., our linear system corresponds to a minimization problem, since $\hat{x} = A^{-1}b$ is the unique global minimum of φ.

$$\varphi(\hat{x}) = -\frac{1}{2}b^T A^{-1} b.$$

Thus we may hope that a stepwise iterative minimization method for φ will converge to a solution of the linear system.

4.2.1 Steepest Descent

Idea: φ decreases most in the direction of the negative gradient

$$-\nabla\varphi(x) = b - Ax.$$

Definition 64 *Let $A \in \mathbb{C}^{n,n}$ and $x, b \in \mathbb{C}^n$. Then*

$$r = b - Ax$$

is called residual *of x with respect to A and b.*

For $r \neq 0$ we have that $\varphi(x + \alpha r) < \varphi(x)$ for some $\alpha > 0$. Thus we can decrease φ by choosing the parameter α to minimize the residual.

Lemma 65 *The minimum in $\alpha \mapsto \varphi(x + \alpha r)$ is given for*

$$\alpha = \frac{r^T r}{r^T A r}.$$

Proof: Exercise. □

Algorithm ('Steepest Descent')
Determines for $A \in \mathbb{R}^{n,n}$ symmetric positive definite and $b \in \mathbb{R}^n$ the solution $x = A^{-1}b$ of $Ax = b$.

1) Start: Choose $x_0 \in \mathbb{R}^n$.

2) Iterate for $k = 1, 2, \ldots$ to convergence

   (a) $r_{k-1} = b - Ax_{k-1}$

   (b) If $r_{k-1} = 0$ then stop and use $x_{k-1} = A^{-1}b$. Otherwise set $\alpha_k = \dfrac{r_{k-1}^T r_{k-1}}{r_{k-1}^T A r_{k-1}}$.

(c) $x_k = x_{k-1} + \alpha_k r_{k-1}$.

Remark 66 *One can show that*

$$\varphi(x_{k+1}) + \frac{1}{2} b^T A^{-1} b \leq \left(1 - \frac{1}{\kappa_2(A)}\right) \left(\varphi(x_k) + \frac{1}{2} b^T A^{-1} b\right).$$

We thus have global convergence for all start vectors. But the method has many disadvantages.

(a) The convergence is very slow if $\kappa_2(A)$ is large.

(b) Furthermore, even if φ becomes small very quickly, then this is not automatically true for the residual.

The main reasons for these disadvantages are the following.

1) We minimize only in one search direction r_k, but we have many more directions than one (namely $r_0, \ldots, r_k$).

2) The search directions are not different enough.

4.2.2 A-Conjugate Search Directions

To improve the convergence behavior of the descent method we add a little modification. The basic idea is to choose in every step instead of the negative gradient a direction $p \in \mathbb{R}^n$ with $p \not\perp r$. Then we also find in this direction a decrease of φ. Thus in every step instead of r_k we choose a search direction p_k with $p_k^T r_k \neq 0$.
We require the following conditions for p_{k+1} and x_{k+1}.

R1) $p_1, \ldots, p_{k+1}$ are linearly independent.

R2) $\varphi(x_{k+1}) = \min\limits_{x \in \mathcal{R}_{k+1}} \varphi(x)$, where $\mathcal{R}_{k+1} := x_0 + \operatorname{Span}\{p_1, \ldots, p_{k+1}\}$.

R3) x_{k+1} is easily computed from x_k.

The first two conditions R1) and R2) together guarantee convergence in at most n steps in exact arithmetic, since we minimize φ over the whole space $\mathbb{R}^n$.

To compute p_{k+1} and x_{k+1}, suppose that the search directions $p_1, \ldots, p_k \in \mathbb{R}^n$ and x_k with $\varphi(x_k) = \min\limits_{x \in \mathcal{R}_k} \varphi(x)$ are already computed and then determine p_{k+1} and x_{k+1} with $\varphi(x_{k+1}) = \min\limits_{x \in \mathcal{R}_{k+1}} \varphi(x)$, such that all three conditions R1)-R3) are satisfied.

In order to achieve this we set $x_k = x_0 + P_k y_k$ with $P_k = [p_1, \ldots, p_k]$ and $y_k \in \mathbb{R}^k$ and make the ansatz

$$x_{k+1} = x_0 + P_k y + \alpha p_{k+1}$$

for $y \in \mathbb{R}^k, \alpha \in \mathbb{R}$. Then we determine the parameters y and α. We have

$$\begin{aligned}
\varphi(x_{k+1}) &= \frac{1}{2}(x_0 + P_k y + \alpha p_{k+1})^T A(x_0 + P_k y + \alpha p_{k+1}) - (x_0 + P_k y + \alpha p_{k+1})^T b \\
&= \varphi(x_0 + P_k y) + \alpha p_{k+1}^T A(x_0 + P_k y) - \alpha p_{k+1}^T b + \frac{1}{2}\alpha^2 p_{k+1}^T A p_{k+1} \\
&= \underbrace{\varphi(x_0 + P_k y)}_{\text{nur } y} + \alpha p_{k+1}^T A P_k y + \underbrace{\frac{1}{2}\alpha^2 p_{k+1}^T A p_{k+1} - \alpha p_{k+1}^T r_0}_{\text{nur } \alpha}.
\end{aligned}$$

If the mixed term was not there then we could minimize separately over the two variables. Thus we choose p_{k+1} so that

$$p_{k+1}^T A P_k = 0$$

and obtain

$$\min_{x \in \mathcal{R}_{k+1}} \varphi(x) = \underbrace{\min_{y \in \mathbb{R}^k} \varphi(x_0 + P_k y)}_{\text{Sol. } y = y_k} + \underbrace{\min_{\alpha \in \mathbb{R}} \left(\frac{1}{2}\alpha^2 p_{k+1}^T A p_{k+1} - \alpha p_{k+1}^T r_0 \right)}_{\text{Sol. } \alpha_{k+1} = \frac{p_{k+1}^T r_0}{p_{k+1}^T A p_{k+1}}}.$$

The first minimization problem is solved by $y = y_k$, since $x_k = x_0 + P_k y_k$ satisfies

$$\varphi(x_k) = \min_{x \in \mathcal{R}_k} \varphi(x).$$

The second minimization is just a scalar minimization and solved by $\alpha_{k+1} = \frac{p_{k+1}^T r_0}{p_{k+1}^T A p_{k+1}}$. Thus we have satisfied conditions R2) and R3).

As a consequence, we choose *A-conjugate search directions* p_k, i.e., we choose

$$p_{k+1} \in \text{Span}\{Ap_1, \ldots, Ap_k\}^\perp, k = 1, 2, \ldots.$$

Then

$$p_i^T A p_j = 0, \quad i \neq j, \quad i, j = 1, \ldots, k$$

i.e., $p_1, \ldots, p_k$ are orthogonal with respect to the scalar product

$$\langle x, y \rangle_A := y^T A x.$$

Then the question arises whether we can always find A-conjugate search directions.

Lemma 67 *If $r_k = b - Ax_k \neq 0$, then there exists $p_{k+1} \in Span\{Ap_1, \ldots, Ap_k\}^\perp$ with $p_{k+1}^T r_k \neq 0$.*

Proof: For $k = 0$ this is clear (choose e.g. $p_1 = r_0$). For $k \geq 1$ then with $r_k \neq 0$ it follows that

$$A^{-1}b \notin \mathcal{R}_k = x_0 + \text{Span}\{p_1, \ldots, p_k\},$$

since $A^{-1}b$ is the unique minimum, which however is not reached yet, since $r_k \neq 0$. Therefore,

$$b \notin Ax_0 + \text{Span}\{Ap_1, \ldots, Ap_k\}$$

or

$$r_0 = b - Ax_0 \notin \text{Span}\{Ap_1, \ldots, Ap_k\}.$$

Thus there exists $p_{k+1} \in \text{Span}\{Ap_1, \ldots, Ap_k\}^\perp$ with $p_{k+1}^T r_0 \neq 0$. Since $x_k \in x_0 + \text{Span}\{p_1, \ldots, p_k\}$, we have

$$r_k = b - Ax_k \in r_0 + \text{Span}\{Ap_1, \ldots, Ap_k\}$$

and thus also

$$p_{k+1}^T r_k = p_{k+1}^T r_0 \neq 0.$$

□

Remark 68 *From the proof of the Lemma 67 we have the following observation. Since* $p^T r_k = p^T r_0$ *for* $p \in Span\{Ap_1, \ldots, Ap_k\}^T$*, we have, in particular, that* $p_{k+1}^T r_k = p_{k+1}^T r_0$*, and thus*

$$\alpha_{k+1} = \frac{p_{k+1}^T r_0}{p_{k+1}^T A p_{k+1}} = \frac{p_{k+1}^T r_k}{p_{k+1}^T A p_{k+1}}.$$

We then can finally show that also the first requirement R1) is satisfied.

Lemma 69 *The search directions* $p_1, \ldots, p_k$ *are linearly independent.*

Proof: The matrix $P_k^T A P_k = \text{diag}\left(p_1^T A p_1, \ldots, p_k^T A p_k\right)$ is invertible, since A is positive definite. Thus P_k has full rank, i.e., the columns $p_1, \ldots, p_k$ are linearly independent. □

Algorithm (A-conjugate search directions)
The algorithm computes for $A \in \mathbb{R}^{n,n}$ symmetric positive definite and $b \in \mathbb{R}^n$ the solution $x = A^{-1}b$ of $Ax = b$.

1) Start: Choose $x_0 \in \mathbb{R}^n$

2) Iterate for $k = 1, 2, \ldots$ until convergence

   (a) $r_k = b - Ax_k$,

   (b) If $r_k = 0$ then stop and use $x_k = A^{-1}b$ as solution. Otherwise choose $p_{k+1} \in \text{Span}\{Ap_1, \ldots, Ap_k\}^\perp$ with $p_{k+1}^T r_k \neq 0$ and compute

$$\alpha_{k+1} = \frac{p_{k+1}^T r_k}{p_{k+1}^T A p_{k+1}}.$$

   (c) $x_{k+1} = x_k + \alpha_{k+1} p_{k+1}$.

Note that we still have freedom in the choice of p_{k+1}.

4.2.3 The Conjugate Gradient Algorithm, CG

We have seen that the choice of A-conjugate search directions has many advantages (an easy computation of x_{k+1} from x_k and a guaranteed convergence in at most n steps in exact arithmetic). On the other hand we would like to keep the advantage of steepest descent that the function φ decreases maximally in the direction of the negative gradient, i.e., this is heuristically a good search direction. The idea is then to use the freedom in p_{k+1} to choose that p_{k+1} which is nearest to r_k, the direction of the negative gradient, i.e., to choose p_{k+1} so that

$$||p_{k+1} - r_k|| = \min_{p \in \text{Span}\{Ap_1,\ldots,Ap_k\}^\perp} ||p - r_k||. \tag{4.1}$$

At first sight this looks strange, since we wanted to choose directions that allow an easy solution of the optimization problem and here we introduce another optimization problem. We will see now that this optimization problem is easy to solve since it will turn out that p_{k+1} is just a linear combination of p_k and r_k.

In the following, under the same assumptions as before, we choose the A-conjugate search directions to minimize (4.1) for $k = 0, \ldots, m$. Let $P_k = [p_1, \ldots, p_k]$ and show then that $p_{k+1} \in \text{Span}\{p_k, r_k\}$.

Lemma 70 *Let $k \in \{1, \ldots, m\}$ and $z_k \in \mathbb{R}^k$, such that*

$$||r_k - AP_k z_k|| = \min_{z \in \mathbb{R}^k} ||r_k - APz||.$$

Then $p_{k+1} = r_k - AP_k z_k$.

Proof: Let $\hat{p} := AP_k z_k$, then by assumption $\hat{p}$ is the orthogonal projection of r_k to $\mathcal{R}(AP_k)^\perp$. Hence

$$||\hat{p} - r_k|| = \min_{p \in \mathcal{R}(AP_k)^\perp} ||p - r_k||$$

and therefore $\hat{p} = p_{k+1}$. □

Theorem 71 *If $r_k \neq 0$ for $k = 0, \ldots, m$, then for $k = 0, \ldots, m$ the following statements hold.*

1) $r_{k+1} = r_k - \alpha_{k+1} A p_{k+1}$,

2) $\text{Span}\{p_1, \ldots, p_{k+1}\} = \text{Span}\{r_0, \ldots, r_k\} = \mathcal{K}_{k+1}(A, r_0)$,

3) $r_{k+1} \perp r_j$ for $j = 0, \ldots, k$,

4) $p_{k+1} \in \text{Span}\{p_k, r_k\}$ for $k \geq 1$.

Proof:

1) Since $x_{k+1} = x_k + \alpha_{k+1} p_{k+1}$, it follows that

$$r_{k+1} = b - Ax_{k+1} = \underbrace{b - Ax_k}_{= r_k} - \alpha_{k+1} A p_{k+1}.$$

2) Applying 1) inductively we obtain

$$\text{Span}\{Ap_1, \ldots, Ap_k\} \subseteq \text{Span}\{r_0, \ldots, r_k\}, \quad k = 1, \ldots, m.$$

We have already shown that for all $k = 0, \ldots, m$ we have

$$p_{k+1} = r_k - AP_k z_k \in \text{Span}\{r_0, \ldots, r_k\}.$$

Thus, we have

$$\text{Span}\{p_1, \ldots, p_{k+1}\} \subseteq \text{Span}\{r_0, \ldots, r_k\}$$

for $k = 0, \ldots, m$. Moreover, with 1) it follows that

$$r_{k+1} \in \text{Span}\{r_k, Ap_{k+1}\} \subseteq \text{Span}\{r_k, Ar_0, \ldots, Ar_k\}$$

for $k = 0, \ldots, m$. Therefore,

$$\begin{aligned} r_1 &\in \text{Span}\{r_0, Ar_0\}, \\ r_2 &\in \text{Span}\{r_0, Ar_0, Ar_1\} \subseteq \text{Span}\{r_0, Ar_0, A^2 r_0\}, \\ &\vdots \quad . \end{aligned}$$

By induction we then have finally

$$\text{Span}\{p_1, \ldots, p_{k+1}\} \subseteq \text{Span}\{r_0, \ldots, r_k\} \subseteq \mathcal{K}_{k+1}(A, r_0).$$

Equality follows by a dimension argument.

3) We show that $P_k^T r_k = 0$ i.e., $p_1, \ldots, p_k \perp r_k$ for all $k = 1, \ldots, m$. By 2) we then also have $r_0, \ldots, r_{k-1} \perp r_k$ desired. We have $x_{k+1} = x_0 + P_k y_k$, where y_k minimizes the function

$$\begin{aligned} \varphi(x_0 + P_k y) &= \frac{1}{2}(x_0 + P_k y)^T A(x_0 + P_k y) - (x_0 + P_k y)^T b \\ &= \varphi(x_0) + y^T P_k^T (Ax_0 - b) + \frac{1}{2} y^T P_k^T A P_k y. \end{aligned}$$

The gradient of $y \mapsto \varphi(x_0 + P_k y)$ therefore vanishes for $y = y_k$, i.e.,

$$P_k^T A P_k y_k + P_k^T (Ax_0 - b) = 0.$$

This is equivalent to $0 = P_k^T(b - Ax_0 - AP_k y_k) = P_k^T(b - Ax_k) = P_k^T r_k$.

4) If $k = 1$, then by 2), it follows that $p_2 \in \text{Span}\{r_0, r_1\}$. Since $p_1 = r_0$ we then have $p_2 \in \text{Span}\{p_1, r_1\}$. For $k > 1$ we partition z_k from Lemma 70 as

$$z_k = \begin{bmatrix} w \\ \mu \end{bmatrix}, \quad w \in \mathbb{R}^{k-1}, \ \mu \in \mathbb{R}.$$

with $r_k = r_{k-1} - \alpha_k A p_k$. By 1) we then obtain from Lemma 70 that

$$\begin{aligned} p_{k+1} &= r_k - AP_k z_k \\ &= r_k - AP_{k-1} w - \mu A p_k \\ &= r_k - AP_{k-1} w + \frac{\mu}{\alpha_k}(r_k - r_{k-1}) \\ &= \left(1 + \frac{\mu}{\alpha_k}\right) r_k + s_k, \end{aligned}$$

where

$$\begin{aligned} s_k &= -\frac{\mu}{\alpha_k} r_{k-1} - AP_{k-1}w \\ &\in \text{Span}\{r_{k-1}, AP_{k-1}w\} \\ &\subseteq \text{Span}\{r_{k-1}, Ap_1, \ldots, Ap_{k-1}\} \\ &\subseteq \text{Span}\{r_0, \ldots, r_{k-1}\}. \end{aligned}$$

(Observe that α_k is nonzero by construction!) By 3) then r_k and s_k are orthogonal. Then we can solve the optimization problem in Lemma 4.4 by determining w and μ such that

$$\|p_{k+1}\|^2 = \left(1 + \frac{\mu}{\alpha_k}\right)^2 \|r_k\|^2 + \|s_k\|^2$$

is minimal. Then, in particular, s_k so that $\|s_k\|$ is minimal (for fixed μ and variable w). But $\|r_{k-1} - AP_{k-1}z\|$ will be minimized (see Lemma 70) via $z = z_{k-1}$ and one obtains $p_k = r_{k-1} - AP_{k-1}z_{k-1}$. Hence, s_k is a multiple of p_k, and therefore

$$p_{k+1} \in \text{Span}\{r_k, s_k\} \subseteq \text{Span}\{r_k, p_k\}.$$

□

Corollary 72 *After scaling p_{k+1} we have*

$$p_{k+1} = r_k + \beta_k p_k.$$

Since $p_k^T A p_{k+1} = 0$, we furthermore have

$$\beta_k = -\frac{p_k^T A r_k}{p_k^T A p_k}.$$

Hence p_{k+1} can be constructed directly from p_k and r_k without solving a minimization problem.

Algorithm: (CG, conjugate-gradient method - Hesteness/Stiefels, 1952)
For $A \in \mathbb{R}^{n,n}$ symmetric, positive definite and $b \in \mathbb{R}^n$ the algorithm computes the solution $x = A^{-1}b$ of $Ax = b$.

1) Start: $x_0 \in \mathbb{R}^n$, $r_0 = b - Ax_0$, $p_1 = r_0$.

2) Iterate for $k = 1, 2, \ldots$ until convergence

   (a) $\alpha_k = \dfrac{p_k^T r_{k-1}}{p_k^T A p_k}$,
   (b) $x_k = x_{k-1} + \alpha_k p_k$,
   (c) $r_k = b - Ax_k$,
   (d) $\beta_{k+1} = -\dfrac{p_k^T A r_k}{p_k^T A p_k}$,
   (e) $p_{k+1} = r_k + \beta_{k+1} p_k$.

Remark 73 *There are many theoretical results behind this simple algorithm, for example the convergence after at most n steps in exact arithmetic, since the CG-algorithm is a special case of the algorithm of A-conjugate search directions. The iterate x_k satisfies*

$$\varphi(x_k) = \min_{x \in \mathcal{R}_k} \varphi(x),$$

where $\varphi(x) = \frac{1}{2}x^T Ax - x^T b$ and $\mathcal{R}_k = x_0 + Span\{p_1, \ldots, p_k\}$. But $Span\{p_1, \ldots, p_k\} = \mathcal{K}_k(A, r_0)$ by Theorem 71, i.e., we minimize φ over the affine Krylov space $x_0 + \mathcal{K}_k(A, r_0)$. Hence x_k satisfies

$$\varphi(x_k) = \min_{x \in x_0 + \mathcal{K}_k(A, r_0)} \varphi(x).$$

For this reason one calls the CG-algorithm a **Krylov space method**.

4.2.4 Convergence properties of the CG method

Using the relationship of the *CG* algorithm with Krylov spaces allows a detailed convergence analysis. For this we introduce a special norm.

Definition 74 *Let $A \in \mathbb{R}^{n,n}$ be symmetric and positive definite. Then the norm defined by*

$$||x||_A := \sqrt{x^T Ax}$$

on $\mathbb{R}^n$ is called the A-Norm or energy norm.

We would like to estimate the error

$$e_k := A^{-1}b - x_k = A^{-1}(b - Ax_k) = A^{-1}r_k$$

where (x_k) is the sequence generated by the *CG* algorithm.

Theorem 75 (Optimality of CG in the A-Norm) *Let $A \in \mathbb{R}^{n,n}$ be symmetric and positive definite and let (x_k) be the CG sequence generated from a starting vector x_0. If $r_{k-1} \neq 0$, then*

$$||e_k||_A = ||A^{-1}b - x_k||_A < ||A^{-1}b - x||_A$$

for all $x \in x_0 + \mathcal{K}_k(A, r_0)$ with $x_k \neq x$.

Proof: We know that $x_k \in x_0 + \mathcal{K}_k(A, r_0)$. Let $x \in x_0 + \mathcal{K}_k(A, r_0)$ be arbitrary and $\Delta x = x_k - x$, i.e., $\Delta x \in \mathcal{K}_k(A, r_0)$, and moreover

$$\hat{e} := A^{-1}b - x = A^{-1}b - (x_k - \Delta x) = e_k + \Delta x$$

Then

$$\begin{aligned} ||\hat{e}||_A^2 &= \hat{e}^T A \hat{e} = (e_k + \Delta x)^T A (e_k + \Delta x) \\ &= e_k^T A e_k + 2e_k^T A \Delta x + \Delta x^T A \Delta x \end{aligned}$$

and

$$2e_k^T A \Delta x = 2r_k^T A^{-1} A \Delta x = 2r_k^T \Delta x = 0$$

since $\Delta x \in \mathcal{K}_k(A, r_0) = \text{Span}\{r_0, \ldots, r_{k-1}\}$ and $r_k \perp r_j$ for $j = 0, \ldots, k-1$ due to Theorem 71. Thus we obtain

$$||\hat{e}||_A^2 = ||e_k||_A^2 + ||\Delta x||_A^2 > ||e_k||_A^2, \quad \text{if } \Delta x \neq 0.$$

□

Corollary 76 *Let* $\widetilde{\Pi}_k := \{p \in \Pi_k | p(0) = 1\}$. *With the notation and assumptions of Theorem 75 (in particular* $r_{k-1} \neq 0$*), there exists a unique polynomial* $p_k \in \widetilde{\Pi}_k$ *with*

$$||p_k(A) e_0||_A = \min_{p \in \widetilde{\Pi}_k} ||p(A) e_0||_A.$$

Furthermore, $e_k = p_k(A) e_0$ *and*

$$\frac{||e_k||_A}{||e_0||_A} = \min_{p \in \widetilde{\Pi}_k} \frac{||p(A) e_0||_A}{||e_0||_A} \leq \inf_{p \in \widetilde{\Pi}_k} \max_{\lambda \in \sigma(A)} |p(\lambda)| \tag{4.2}$$

Proof: There exists $x_k \in x_0 + \mathcal{K}_k(A, r_0)$, i.e.,

$$x_k = x_0 + \hat{p}_{k-1}(A) r_0$$

for some $\hat{p}_{k-1} \in \Pi_{k-1}$. Furthermore,

$$r_k = b - Ax_k = \underbrace{b - Ax_0}_{=:r_0} - A\hat{p}_{k-1}(A) r_0.$$

Thus, we have

$$e_k = A^{-1} r_k = \underbrace{A^{-1} r_0}_{=e_0} - \hat{p}_{k-1}(A) r_0 = e_0 - \hat{p}_{k-1}(A) A e_0 = \underbrace{(I - \hat{p}_{k-1}(A) A)}_{=p_k(A) \in \tilde{\Pi}_k} e_0.$$

Then the uniqueness of p_k, and the first equality in (4.2) follows from Theorem 75. To prove the inequality in (4.2), let $(v_1, \ldots, v_n)$ be an orthonormal basis of eigenvectors of A to the eigenvalues $\lambda_1, \ldots, \lambda_n$. Furthermore let $p \in \tilde{\Pi}_k$ and

$$e_0 = c_1 v_1 + \ldots + c_n v_n \qquad \text{with } c_1, \ldots, c_n \in \mathbb{R}.$$

Then,

$$p(A) e_0 = c_1 p(\lambda_1) v_1 + \ldots + c_n p(\lambda_n) v_n.$$

By the orthogonality of the v_i we obtain

$$||e_0||_A^2 = e_0^T A e_0 = \sum_{i=1}^{n} c_i^2 \lambda_i$$

and

$$||p(A) e_0||_A^2 = \sum_{i=1}^{n} c_i^2 p(\lambda_i)^2 \lambda_i \leq \max_{\lambda \in \sigma(A)} p(\lambda)^2 \sum_{i=1}^{n} c_i^2 \lambda_i.$$

But this implies that

$$\frac{||p(A) e_0||_A^2}{||e_0||_A^2} \leq \max_{\lambda \in \sigma(A)} |p(\lambda)|^2.$$

□

Remark 77 *1) From Corollary 76 we conclude that the CG algorithm converges fast if A has an appropriate spectrum, i.e., one for which there exist polynomials p with $p(0) = 1$ and small degree such that $|p(\lambda)|$ is small for all $\lambda \in \sigma(A)$. This is e.g. the case if*

(a) the eigenvalues occur in clusters,

(b) all eigenvalues are far from the origin, (then $\kappa_2(A) = \frac{\lambda_{max}}{\lambda_{min}}$ is not too large).

2) With the help of Chebysheff polynomials one can prove that

$$\frac{||e_k||_A}{||e_0||_A} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k,$$

where $\kappa := \kappa_2(A)$ and

$$\frac{||e_k||_2}{||e_0||_2} \leq 2\sqrt{\kappa}\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k.$$

3) We can improve the convergence of the CG algorithm via preconditioning*:*

(a) For general linear systems $Ax = b$ consider

$$M^{-1}Ax = M^{-1}b$$

where $M^{-1}A$ has an appropriate spectrum and $Mz = c$ is easy to solve.

(b) For $Ax = b$ with A symmetric and positive definite consider

$$\left(C^{-1}AC^{-T}\right)\left(C^T x\right) = C^{-1}b$$

where $C^{-1}AC^{-T}$ has an appropriate spectrum and $C^T z = d$ is easy to solve. $C^{-1}AC^{-T}$ is again symmetric and positive definite.

4.2.5 The CG and the Lanczos Algorithm

In this section we use the same notation as in previous sections. Consider the matrices

$$R_k = [r_0, \ldots, r_{k-1}], \quad P_k = [p_1, \ldots, p_k], \quad B_k = \begin{bmatrix} 1 & -\beta_2 & & 0 \\ & 1 & \ddots & \\ & & \ddots & -\beta_n \\ 0 & & & 1 \end{bmatrix}.$$

Using the equations $p_1 = r_0$ and $p_i = r_{i-1} + \beta_i p_{i-1}$ for $i = 2, \ldots, n$ (see Section 4.2.3) we obtain

$$R_k = P_k B_k.$$

Then the matrix $R_k^T A R_k$ is tridiagonal, since

$$R_k^T A R_k = B_k^T P_k^T A P_k B_k = B_k^T \begin{bmatrix} p_1^T A p_1 & & 0 \\ & \ddots & \\ 0 & & p_k^T A p_k \end{bmatrix} B_k.$$

Furthermore, we know from Theorem 75, that the $r_0, \ldots, r_{k-1}$ are orthogonal and span a Krylov space, i.e., $\frac{r_0}{\|r_0\|}, \ldots, \frac{r_{k-1}}{\|r_{k-1}\|}$ is an orthonormal basis of $\mathcal{K}_k(A, r_0)$.
This leads to an interesting conclusion. If $q_1 := \frac{r_0}{\|r_0\|}$ and if $q_1, \ldots, q_k$ are the vectors generated by the Lanczos algorithm, then by the implicit Q Theorem

$$q_j = \pm \frac{r_{j-1}}{\|r_{j-1}\|}, \quad j = 1, \ldots, k.$$

Thus, the tridiagonal matrix generated by the Lanczos algorithm is (except for signs) the matrix $R_k^T A R_k$, i.e.,

$$\boxed{\text{'}CG \approx \text{Lanczos'}}$$

Application: In the course of the CG algorithm we can generate the tridiagonal matrix $R_k^T A R_k$ and obtain information about extremal eigenvalues of A and the condition number $\kappa_2(A) = \frac{\lambda_{max}}{\lambda_{min}}$.

4.2.6 The GMRES algorithm

Situation: $A \in \mathbb{C}^{n,n}$ general and invertible, n large, A sparse, $b \in \mathbb{C}^n$.

Goal: Determine $x \in \mathbb{C}^n$ with $Ax = b$.

In Section 4.2.4 we have noticed that certain affine Krylov spaces are good search spaces. This suggests to use again a Krylov space method. In the CG algorithm we have used that the solution $\hat{x} = A^{-1}b$ is the unique minimum of $\varphi = \frac{1}{2}x^T Ax - x^T b$. This, however, holds in general only if $A \in \mathbb{R}^{n,n}$ is symmetric positive definite.

Idea: For a given starting vector $x_0 \in \mathbb{C}^n$ and $r_0 := b - Ax_0$, determine x_k via

$$\|b - Ax_k\|_2 = \min_{x \in x_0 + \mathcal{K}_k(A, r_0)} \|b - Ax\|_2. \tag{4.3}$$

A Hermitian	$\rightsquigarrow$	MINRES (*min*inmal *res*iduals), Paige/Saunders 1975
A general	$\rightsquigarrow$	GMRES (*g*eneralized *m*inimal *res*iduals), Saad/Schultz 1986

This means that we have to solve in each step the least-squares problem

$$\|b - Ax_k\|_2 = \min_{x \in x_0 + \mathcal{K}_k(A, r_0)} \|b - Ax\|_2.$$

In Section 4.2.5 we have seen that the CG algorithm corresponds to the Lanczos algorithm. We expect that in the general case

$$\boxed{\text{'GMRES} \approx \text{Arnoldi'}}$$

After k steps of the Arnoldi algorithm (without breakdown) we have the Arnoldi relation

$$AQ_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T = Q_{k+1} H_{k+1,k},$$

with $Q_k = [q_1, \dots q_k]$, $Q_{k+1} = [Q_k, q_{k+1}]$ isometric and

$$H_{k+1,k} = \begin{bmatrix} h_{11} & \dots & \dots & h_{1k} \\ h_{21} & \ddots & & \vdots \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & h_{k,k-1} & h_{kk} \\ 0 & \dots & 0 & h_{k+1,k} \end{bmatrix} \in \mathbb{C}^{k+1,k}.$$

If $q_1 = \frac{r_0}{\|r_0\|}$, then $\mathrm{Span}\{q_1, \dots, q_k\} = \mathcal{K}_k(A, r_0)$. Let $x \in x_0 + \mathcal{K}_k(A, r_0)$, i.e., $x = x_0 + Q_k y$ for an $y \in \mathbb{C}^k$. Then

$$\begin{aligned} \|b - Ax\| &= \|b - A(x_0 + Q_k y)\| \\ &= \|r_0 - AQ_k y\| \\ &= \|r_0 - Q_{k+1} H_{k+1,k} y\| \\ &= \|Q_{k+1}^H r_0 - H_{k+1,k} y\| \qquad \text{since } Q_{k+1} \text{ is isometric,} \\ &= \left\| \|r_0\| \cdot e_1 - H_{k+1,k} y \right\| \qquad \text{since } q_2, \dots, q_{k+1} \perp q_1 = \tfrac{r_0}{\|r_0\|}. \end{aligned} \tag{4.4}$$

Reminder. For the solution of least-squares problems $\|c - My\| \stackrel{!}{=} \min$, with $M \in \mathbb{C}^{k,n}$, $k \le n$ we may

1) compute QR decomposition of M,

$$M = QR, \quad Q \in \mathbb{C}^{n,n} \text{ unitary}, \quad R = \begin{bmatrix} R_1 & 0 \end{bmatrix}.$$

2) Since Q is unitary, we have

$$\|c - My\|^2 = \|Q^H c - Ry\|^2 = \left\| \begin{bmatrix} c_1 - R_1 y \\ c_2 \end{bmatrix} \right\|^2, \qquad \text{where} \quad Q^H c = \begin{bmatrix} c_1 & c_2 \end{bmatrix}.$$

If R_1 is invertible, then this is minimal if $R_1 y = c$. Thus we solve $R_1 y = c$.

In the least-squares problem (4.3), the matrix $H_{k+1,k}$ is in Hessenberg form and we need to solve this problem for every k. Suppose that we have solved the problem for $k-1$, i.e., we already have a QR decomposition for $H_{k,k-1}$,

$$H_{k,k-1} = \widetilde{Q}_k \widetilde{R}_k, \quad \widetilde{Q}_k \text{ unitary}, \quad \widetilde{R}_{k-1} = \begin{bmatrix} R_{k-1} \\ 0 \end{bmatrix}, \quad R_{k-1} \text{ upper triangular.}$$

Then

$$\begin{aligned} \begin{bmatrix} \widetilde{Q}_k^H & 0 \\ 0 & 1 \end{bmatrix} \cdot H_{k+1,k} &= \begin{bmatrix} \widetilde{Q}_k^H & 0 \\ 0 & 1 \end{bmatrix} \left[\begin{array}{c|c} H_{k,k-1} & h_{kk} \\ \hline 0 & h_{k+1,k} \end{array}\right] = \left[\begin{array}{c|c} \widetilde{R}_{k-1} & \widetilde{Q}_k^H h_{kk} \\ \hline 0 & h_{k+1,k} \end{array}\right] \\ &= \begin{bmatrix} R_{k-1} & 0 \\ 0 & h_{k+1,k} \end{bmatrix}. \end{aligned}$$

Thus, the element $h_{k+1,k}$ can be eliminated by a single Givens rotation and we obtain the QR decomposition of $H_{k+1,k}$ from $H_{k,k-1}$ by this Givens rotation (this costs only $\mathcal{O}(n)$ flops).

Algorithm (GMRES)
For $A \in \mathbb{C}^{n,n}$ invertible, $b \in \mathbb{C}^n$, and a starting vector $x_0 \in \mathbb{C}^n$, the algorithm computes the solution $\hat{x} = A^{-1}b$ of $Ax = b$.

1) Start: $r_0 = b - Ax_0$, $h_{10} = \|r_0\|$.

2) Iterate for $k = 1, 2, \ldots$ to convergence

a) $q_k = \dfrac{r_k}{h_{k,k-1}}$,

b) $r_k = Aq_k - \sum_{j=1}^{k} h_{jk} q_j$ with $h_{jk} = Q_j^H r_k$,

c) $h_{k+1,k} = \|r_k\|$,

d) Determine y_k such that $\left\| \|r_0\| \cdot e_1 - H_{k+1,k} y_k \right\|$ is minimal,

e) $x_k = x_0 + Q_k y_k$.

Remark 78 *As the CG algorithm, also GMRES can be analyzed via polynomial approximation in* $\widetilde{\Pi}_k = \{p \in \Pi_k |\ p(0) = 1\}$.

$$x = x_0 + \hat{p}(A)r_0 \qquad \textit{for } \hat{p} \in \Pi_{k-1},$$

since $x \in x_0 + \mathcal{K}_k(A, r_0)$. *Therefore,*

$$r_k := b - Ax_k = b - Ax_0 - A\hat{p}(A)r_0 = \Big(I - A\hat{p}(A)\Big)r_0 = p(A)r_0 \qquad \textit{for } p \in \widetilde{\Pi}_k.$$

Then we can reformulate GMRES as

$$\textit{Determine } p \in \widetilde{\Pi}_k, \textit{ such that } \|p(A)r_0\| \textit{ is minimal.}$$

If $p_k \in \widetilde{\Pi}_k$ *is such that* $r_k = p_k(A)r_0$, *then*

$$\|r_k\| = \|p_k(A)r_0\| \leq \|p(A)r_0\|$$

for all $p \in \widetilde{\Pi}_k$.

Theorem 79 *Let* $A \in \mathbb{C}^{n,n}$ *be diagonalizable and* $V^{-1}AV = \Lambda$ *diagonal. Then,*

$$\frac{\|r_k\|}{\|r_0\|} \leq \kappa(V) \inf_{p \in \widetilde{\Pi}_k} \max_{\lambda \in \sigma(A)} \Big|p(\lambda)\Big|.$$

Proof: For every $p \in \widetilde{\Pi}_k$ we have

$$\begin{aligned} \|p(A)\| &= \|p(V\Lambda V^{-1}\| = \|Vp(\Lambda)V^{-1}\| \\ &\leq \|V\| \cdot \|p(\Lambda)\| \cdot \|V^{-1}\| = \kappa(V) \cdot \|p(\Lambda)\|, \end{aligned}$$

and, furthermore,

$$\|p(\Lambda)\| = \max_{\lambda \in \sigma(A)} |p(\lambda)|,$$

since Λ is diagonal. Thus, we obtain

$$\begin{aligned} \|r_k\| &= \|p_k(A)r_0\| \leq \inf_{p \in \widetilde{\Pi}_k} \|p(A)r_0\| \leq \inf_{p \in \widetilde{\Pi}_k} \|p(A)\| \cdot \|r_0\| \\ &\leq \|r_0\| \cdot \kappa(V) \inf_{p \in \widetilde{\Pi}_k} \max_{\lambda \in \sigma(A)} |p(\lambda)|. \end{aligned}$$

□

Consequence The GMRES algorithm (in most cases) converges fast if

1) the spectrum A is appropriate and

2) $\kappa(V)$ is small, i.e., if A is not too far from a normal matrix (since for a normal matrix V can be chosen unitary, i.e., with condition number 1).

Remark 80 *Convergence acceleration can again be achieved via preconditioning, i.e., instead of* $Ax = b$ *we solve* $M^{-1}Ax = M^{-1}b$, *where* $My = c$ *is easy to solve and chosen such that the spectrum is appropriate.*

Remark 81 *Other methods for the solution of $Ax = b$, A invertible with $A \neq A^H$:*

1) CGN. Instead of $Ax = b$ consider the normal equations, i.e, $A^H Ax = A^H b$ with positive definite $A^H A$ and apply to this the CG algorithm without forming the product.

Disadvantage. *The condition number is squared $\kappa(A^H A) = \kappa(A)^2$.*

Advantage. *The eigenvalues of $A^H A$ are the squares of the singular values of A. Therefore CGN is a good approach for matrices A with 'bad spectrum' but 'good singular values'.*

2) BiCG (Bi-conjugate gradients).

CG: *The computed $x_k \in x_0 + \mathcal{K}_k(A, r_0)$ delivers $r_k \perp r_0, \ldots, r_{k_1}$, hence $r_k \perp \mathcal{K}_k(A, r_0)$.*

BiCG: *choose s_0 with $s_0^H r_0 = 1$ and determine $x_k \in x_0 + \mathcal{K}_k(A, r_0)$ with $r_k \perp \mathcal{K}_k(A, s_0)$*

BICG corresponds to the nonsymmetric Lanczos algorithm.

	$Ax = \lambda x$	$Ax = b$
$A = A^H$	*Lanczos*	*CG*
$A \neq A^H$	*Arnoldi*	*GMRES*
	Lanczos	*BiCG*

3) Survey of Krylov space methods

Common Krylov space: $\mathcal{K} = \mathcal{K}_k(A, r_0)$

Important quantity: $r_k = b - Ax_k$ (residual)

a) Ritz-Galerkin-Ansatz: choose $x_k \in x_0 + \mathcal{K}$ such that $r_k \perp \mathcal{K}$
$\rightsquigarrow$ CG, FOM, GENCG

b) Minimal-residual-Ansatz: choose $x_k \in x_0 + \mathcal{K}$ such that $\|r_k\|$ is minimal
$\rightsquigarrow$ MINRES, GMRES, ORTHODIR

c) Petrov-Galerkin-Ansatz: choose $x_k \in x_0 + \mathcal{K}$ such that $r_k \perp \mathcal{L}$, $\mathcal{L} \subseteq \mathbb{C}^n$, $\dim \mathcal{L} = k$
$\rightsquigarrow$ BiCG, QMR

d) Minimal error-Ansatz: choose $x_k \in x_0 + \mathcal{K}$ such that $\|x_k - A^{-1}b\|$ is minimal.
$\rightsquigarrow$ SYMMLQ, GMERR

There are further hybrid methods, like (CGS,Bi-CGSTAB,...)

But none of the methods is really efficient without preconditioning. To obtain a good preconditioner depends very much on the problem and usually it has to be chosen based on knowledge about the background of the problem.