

ITERATIVE VERFAHREN FÜR LINEARE GLEICHUNGSSYSTEME UND EIGENWERTPROBLEME

VORLESUNGSSKRIPT, SOMMERSEMESTER 2016

Christian Clason

Stand vom 20. Juli 2016

Fakultät für Mathematik
Universität Duisburg-Essen

INHALTSVERZEICHNIS

I ALGEBRAISCHE GRUNDLAGEN

- 1 VEKTOREN UND MATRIZEN 3
 - 1.1 Elementare Definitionen 3
 - 1.2 Matrizen und Unterräume 4
 - 1.3 Vektor- und Matrixnormen 5
- 2 ORTHOGONALITÄT UND PROJEKTIONEN 7
 - 2.1 Skalarprodukt und Orthogonalität 7
 - 2.2 Projektionen 9
 - 2.3 Orthonormalisierungsverfahren 13
- 3 EIGENWERTE UND EIGENVEKTOREN 16
 - 3.1 Definitionen und elementare Eigenschaften 16
 - 3.2 Ähnlichkeitstransformationen 19

II LINEARE GLEICHUNGSSYSTEME

- 4 STATIONÄRE VERFAHREN 24
 - 4.1 Konvergenz stationärer Iterationsverfahren 25
 - 4.2 Klassische Zerlegungs-Strategien 27
- 5 PROJEKTIONSVERFAHREN 30
 - 5.1 Allgemeine Projektionsverfahren 30
 - 5.2 Eindimensionale Projektionsverfahren 34
- 6 KRYLOVRAUM-VERFAHREN 40
 - 6.1 Der Arnoldi-Prozess 41
 - 6.2 Der Lanczos-Prozess 43
 - 6.3 Arnoldi-Verfahren für Gleichungssysteme 43

7	DAS CG-VERFAHREN	45
	7.1 Algorithmus	45
	7.2 Konvergenz	48
	7.3 Präkonditioniertes CG-Verfahren	50
8	DAS GMRES-VERFAHREN	53
	8.1 Algorithmus	53
	8.2 Konvergenz	56
	8.3 Präkonditioniertes GMRES-Verfahren	57
9	BIORTHOGONALE KRYLOVRAUM-VERFAHREN	59
	9.1 Der Bi-Lanczos-Prozess	59
	9.2 Das Bi-CG-Verfahren	62
	9.3 Das CGS-Verfahren	63
	9.4 Das Bi-CGStab-Verfahren	65

III EIGENWERTPROBLEME

10	VEKTOR- UND QR-ITERATION	68
	10.1 Vektoriteration	68
	10.2 Inverse Vektoriteration	71
	10.3 Das QR-Verfahren	72
	10.4 Praktische Durchführung des QR-Verfahrens	77
11	PROJEKTIONSVERFAHREN	83
	11.1 Orthogonale Projektionsverfahren	85
	11.2 Schiefe Projektionsverfahren	87
12	KRYLOVRAUM-VERFAHREN	89
	12.1 Arnoldi- und Lanczos-Verfahren	89
	12.2 Konvergenz	91
	12.3 Neustarts und Deflation	93

IV AUSGLEICHSPROBLEME

13	KRYLOVRAUM-VERFAHREN	99
	13.1 Das CGLS-Verfahren	99
	13.2 Das CGME-Verfahren	101
14	DAS LSQR-VERFAHREN	103
	14.1 Der Paige–Saunders–Lanczos-Prozess	103
	14.2 LSQR-Verfahren	104

ÜBERBLICK

Der Inhalt dieser Vorlesung sind iterative Verfahren zur Lösung linearer Gleichungssysteme und Eigenwertprobleme; der Schwerpunkt liegt dabei auf modernen Projektionsverfahren und insbesondere auf Krylovraum-Methoden. Im Gegensatz zu den sogenannten *direkten* Verfahren wie zum Beispiel der LR-Zerlegung liefern iterative Verfahren zwar immer nur Näherungslösungen, benötigen jedoch in jedem Schritt nur die Multiplikation eines Vektors mit einer Matrix. Dies ist insbesondere für Probleme mit *dünn besetzten* Matrizen wichtig, die nur wenige von Null verschiedene Einträge in jeder Zeile haben: In diesem Fall wächst der Aufwand für eine Matrix-Vektor-Multiplikation nur linear mit der Anzahl der Zeilen, während der der LR-Zerlegung kubisch wächst. Außerdem sind die Faktoren der LR-Zerlegung in der Regel voll besetzt. Da etwa bei der numerischen Lösung von partiellen Differentialgleichungen routinemäßig (dünn besetzte) Matrizen mit mehreren Hunderttausend bis hin zu Milliarden Zeilen auftreten, kommen hier selbst auf modernsten Großrechnern nur noch iterative Verfahren in Betracht.

Dieses Skriptum basiert vor allem auf den folgenden Werken:

- [1] G. H. Golub und C. F. Van Loan (2013). *Matrix Computations*. 4. Aufl. Johns Hopkins University Press, Baltimore, MD
- [2] Y. Saad (2003). *Iterative Methods For Sparse Linear Systems*. 2. Aufl. SIAM, Philadelphia, PA. URL: http://www-users.cs.umn.edu/~saad/IterMethBook_2ndEd.pdf
- [3] H. A. van der Vorst (2009). *Iterative Krylov Methods for Large Linear Systems*. Cambridge University Press, Cambridge
- [4] D. S. Watkins (2007). *The Matrix Eigenvalue Problem*. GR and Krylov subspace methods. SIAM, Philadelphia, PA. DOI: [10.1137/1.9780898717808](https://doi.org/10.1137/1.9780898717808)
- [5] D. S. Watkins (2008). *The QR algorithm revisited*. SIAM Review 50.1, S. 133–145. DOI: [10.1137/060659454](https://doi.org/10.1137/060659454).
- [6] Y. Saad (2011). *Numerical Methods for Large Eigenvalue Problems*. 2. Aufl. SIAM, Philadelphia, PA. URL: http://www-users.cs.umn.edu/~saad/eig_book_2ndEd.pdf
- [7] Å. Björck (2015). *Numerical Methods in Matrix Computations*. Bd. 59. Texts in Applied Mathematics. Springer, Cham. DOI: [10.1007/978-3-319-05089-8](https://doi.org/10.1007/978-3-319-05089-8)

Teil I

ALGEBRAISCHE GRUNDLAGEN

VEKTOREN UND MATRIZEN

In diesem Kapitel stellen wir einige grundlegende Definitionen und Eigenschaften zusammen. Zum größten Teil sollten diese aus früheren Vorlesungen bereits bekannt sein, weshalb auf Beweise verzichtet wird. Im Allgemeinen betrachten wir dabei komplexe Vektorräume; auf Eigenschaften, die nur für reelle Matrizen gelten, wird explizit hingewiesen.



1.1 ELEMENTARE DEFINITIONEN

Wir beginnen mit der Festlegung von Notation. Sei $n \in \mathbb{N}$ beliebig. Ein *Vektor* $x \in \mathbb{C}^n$ ist stets ein *Spaltenvektor*; er hat die Einträge $x_i \in \mathbb{C}$, $1 \leq i \leq n$. Der zugehörige *Zeilenvektor* wird mit $x^T = (x_1, \dots, x_n)$ bezeichnet. Der *adjungierte* Vektor zu x ist $x^* := \bar{x}^T = (\bar{x}_1, \dots, \bar{x}_n)$. Hier ist \bar{z} die komplex konjugierte Zahl zu $z \in \mathbb{C}$.

Eine *Matrix* $A \in \mathbb{C}^{n \times n}$ hat die Einträge $a_{ij} \in \mathbb{C}$, $1 \leq i, j \leq n$, wobei der erste Index (hier i) den *Zeilenindex* und der zweite Index (hier j) den *Spaltenindex* bezeichnet. Die Matrix $A \in \mathbb{C}^{2 \times 2}$ hat also die Einträge

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}.$$

Sind n Vektoren $a_1, \dots, a_n \in \mathbb{C}^n$ gegeben, können wir daraus eine Matrix $A \in \mathbb{C}^{n \times n}$ bilden, deren j -te Spalte der Vektor a_j ist. Dies drücken wir aus durch

$$A = [a_1, a_2, \dots, a_n].$$

Die *transponierte Matrix* zu $A \in \mathbb{C}^{n \times n}$ bezeichnen wir mit A^T ; sie hat die Einträge a_{ji} . Die *adjungierte Matrix* $A^* := \bar{A}^T$ hat dagegen die Einträge \bar{a}_{ji} . Ist $A \in \mathbb{R}^{n \times n}$, so ist natürlich $A^* = A^T$.

Damit können wir nun bestimmte Klassen von Matrizen unterscheiden. Die Matrix $A \in \mathbb{C}^{n \times n}$ heißt

- *symmetrisch*, falls $A^T = A$ gilt,

- *hermitesch* oder *selbstadjungiert*, falls $A^* = A$ gilt,
- *schiefsymmetrisch* (bzw. *schiefadjungiert*), falls $A^T = -A$ (bzw. $A^* = -A$) gilt,
- *normal*, falls $A^*A = AA^*$ gilt,
- *unitär*, falls $A^*A = I$ gilt, wobei I die Einheitsmatrix (oder *Identität*) bezeichnet. (Ist $A \in \mathbb{R}^{n \times n}$, so bedeutet dies, dass $A^T A = I$ ist, und man nennt A *orthogonal*.)

1.2 MATRIZEN UND UNTERRÄUME

Ein Unterraum von \mathbb{C}^n ist eine nichtleere Teilmenge $U \subset \mathbb{C}^n$, die abgeschlossen bezüglich Addition und Multiplikation mit (komplexen) Skalaren ist. Eine Menge $\{u_1, \dots, u_k\} \subset \mathbb{C}^n$, $k \in \mathbb{N}$, von Vektoren erzeugt einen Unterraum $U \subset \mathbb{C}^n$ – genannt *Spann* der u_i – indem wir die Menge aller Linearkombinationen der u_i betrachten:

$$U = \text{span}\{u_1, \dots, u_k\} := \left\{ \sum_{i=1}^k \alpha_i u_i : \alpha_i \in \mathbb{C}, 1 \leq i \leq k \right\}.$$

Sind die u_i linear unabhängig, so bilden die Vektoren u_1, \dots, u_k eine *Basis* von U , und U hat die *Dimension* k (geschrieben: $\dim(U) = k$).

Eine Matrix $A \in \mathbb{C}^{n \times n}$ definiert nun zwei wichtige Unterräume von \mathbb{C}^n :

1. Das *Bild* von A ist der Spann aller Spalten von A , kurz

$$\text{Im}(A) := \{Ax : x \in \mathbb{C}^n\}.$$

Der (*Spalten-*)*Rang* von A ist definiert als die maximale Anzahl der linear unabhängigen Spalten von A :

$$\text{rank}(A) := \dim(\text{Im}(A)).$$

2. Der *Kern* von A ist die Menge aller Vektoren, die durch A auf $0 \in \mathbb{C}^n$ abgebildet werden:

$$\text{Ker}(A) = \{x \in \mathbb{C}^n : Ax = 0\}.$$

Falls $\text{Ker}(A) \neq \{0\}$ ist, heißt A *singulär*, ansonsten *regulär*. In letzterem Fall hat das lineare Gleichungssystem $Ax = b$ für alle $b \in \mathbb{C}^n$ eine eindeutige Lösung $x^* \in \mathbb{C}^n$.

Ein Fundamentalsatz der Linearen Algebra ist nun, dass jeder Vektor $x \in \mathbb{C}^n$ *eindeutig* als Summe eines Vektors aus dem Bild von A und eines Vektors aus dem Kern von A^* (und umgekehrt) dargestellt werden kann. Der Raum \mathbb{C}^n ist also die *direkte Summe* aus Bild und Kern:

$$\mathbb{C}^n = \text{Im}(A) \oplus \text{Ker}(A^*) = \text{Im}(A^*) \oplus \text{Ker}(A).$$

Oft werden wir auch das *Bild* eines Unterraums $U \subset \mathbb{C}^n$ *unter einer Matrix* $A \in \mathbb{C}^{n \times n}$ betrachten:

$$AU := \{Ax : x \in U\}$$

Gilt $AU \subset U$, so nennen wir U *invariant* unter A .

1.3 VEKTOR- UND MATRIXNORMEN

Da ein iteratives Verfahren immer nur Näherungen an einen gesuchten Vektor liefern kann, ist es wichtig, Fehler messen zu können. Dafür werden Vektornormen eingeführt:

Die Abbildung $\|\cdot\| : \mathbb{C}^n \rightarrow \mathbb{R}$ heißt *Norm*, falls

(N1) $\|x\| \geq 0$ für alle $x \in \mathbb{C}^n$, und $\|x\| = 0$ genau dann, wenn $x = 0$;

(N2) Für alle $\alpha \in \mathbb{C}$ und $x \in \mathbb{C}^n$ gilt $\|\alpha x\| = |\alpha| \|x\|$;

(N3) Für alle $x, y \in \mathbb{C}^n$ gilt die *Dreiecksungleichung*

$$\|x + y\| \leq \|x\| + \|y\|.$$

Die wichtigsten Beispiele sind die *p-Normen* auf \mathbb{C}^n :

$$\|x\|_p := \left(\sum_{i=1}^n |x_i|^p \right)^{\frac{1}{p}} \quad \text{für } 1 \leq p < \infty,$$

$$\|x\|_\infty := \max_{i=1, \dots, n} |x_i|.$$

Wie wir sehen werden, hat die *Euklidische Norm* $\|\cdot\|_2$ unter allen p -Normen eine ausgezeichnete Stellung. Wir vereinbaren daher, dass – soweit nicht anders angegeben – eine nicht weiter spezifizierte Norm $\|x\|$ stets die Euklidische Norm $\|x\|_2$ bezeichnet.

Auf endlichdimensionalen Vektorräumen (also insbesondere auf \mathbb{C}^n) sind alle Normen äquivalent: Sind $\|\cdot\|$ und $\|\cdot\|'$ zwei Normen, so existieren zwei Konstanten $c_1, c_2 > 0$, so dass für alle $x \in \mathbb{C}^n$ gilt:

$$c_1 \|x\| \leq \|x\|' \leq c_2 \|x\|.$$

Jede Matrix $A \in \mathbb{C}^{n \times n}$ lässt sich als Vektor im Raum $\mathbb{C}^{(n^2)}$ auffassen, womit sich jede dieser Normen auch zur Untersuchung einer Matrix verwenden ließe. Jedoch sind Matrizen auch lineare Operatoren, die auf Vektoren wirken. Um dies zu berücksichtigen, verlangen wir von einer Matrixnorm über die Normaxiome hinaus zusätzliche Eigenschaften: Eine Matrixnorm $\|\cdot\|_M : \mathbb{C}^{n \times n} \rightarrow \mathbb{R}$ heißt

1. *submultiplikativ*, falls $\|AB\|_M \leq \|A\|_M \|B\|_M$ für alle $A, B \in \mathbb{C}^{n \times n}$ gilt.
2. *verträglich* mit einer Vektornorm $\|\cdot\|_V$, wenn für alle $A \in \mathbb{C}^{n \times n}$ und $x \in \mathbb{C}^n$ gilt:

$$\|Ax\|_V \leq \|A\|_M \|x\|_V.$$

Für eine Vektornorm $\|\cdot\|_V$ auf \mathbb{C}^n heißt die durch

$$(1.1) \quad \|A\|_M := \sup_{x \neq 0} \frac{\|Ax\|_V}{\|x\|_V}$$

definierte Norm die (durch $\|\cdot\|_V$) *induzierte* Norm. Jede induzierte Norm ist submultiplikativ und mit der sie induzierenden Vektornorm verträglich. Auch hier stellen die p -Normen die wichtigsten Beispiele, und wir bezeichnen die durch die p -(Vektor-)Norm induzierte Matrixnorm wieder mit $\|\cdot\|_p$, wo dies nicht zu Verwechslung führen kann. Einige induzierte p -Normen haben eine explizite Darstellung:

- $p = 1$ (Spaltensummennorm): $\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^n |a_{ij}|$,
- $p = \infty$ (Zeilensummennorm): $\|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}|$.

Wieder hat die Norm für $p = 2$ (die sogenannte *Spektraln*orm) eine besondere Stellung; für diese existiert aber keine explizite Darstellung über die Einträge (sie kann aber mit Hilfe von Eigenwerten charakterisiert werden).

Nicht alle Matrixnormen sind induzierte Normen. Auch die *Frobenius-Norm* ist submultiplikativ und verträglich mit der Euklidischen Norm¹:

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

Der Wert der Frobeniusnorm besteht darin, dass sie eine scharfe obere Schranke für die – schwer zu berechnende – Spektraln

orm darstellt. (Wie Vektornormen sind auch alle Matrixnormen äquivalent.)

¹Sie entspricht der Euklidischen Norm von $A \in \mathbb{C}^{n \times n}$, aufgefasst als Vektor in $\mathbb{C}^{(n^2)}$.

ORTHOGONALITÄT UND PROJEKTIONEN

Moderne Iterationsverfahren nutzen in fundamentaler Weise die Geometrie des Euklidischen Vektorraums \mathbb{C}^n aus. Diese Geometrie wird im Wesentlichen durch den Begriff des Skalarprodukts und der dadurch definierten Orthogonalität vermittelt.

2

2.1 SKALARPRODUKT UND ORTHOGONALITÄT

Eine Abbildung $\langle \cdot, \cdot \rangle : \mathbb{C}^n \times \mathbb{C}^n \rightarrow \mathbb{C}$ heißt *Skalarprodukt*, falls

(s1) Für alle $x, y \in \mathbb{C}^n$ gilt $\langle x, y \rangle = \overline{\langle y, x \rangle}$;

(s2) Für alle $\alpha, \beta \in \mathbb{C}$ und $x, y, z \in \mathbb{C}^n$ gilt

$$\langle \alpha x + \beta y, z \rangle = \alpha \langle x, z \rangle + \beta \langle y, z \rangle,$$

$$\langle z, \alpha x + \beta y \rangle = \bar{\alpha} \langle z, x \rangle + \bar{\beta} \langle z, y \rangle;$$

(s3) Für alle $x \in \mathbb{C}^n$ gilt $\langle x, x \rangle \geq 0$, wobei $\langle x, x \rangle = 0$ genau dann gilt, wenn $x = 0$ ist.¹

Jedes Skalarprodukt induziert eine Norm durch

$$\|x\|^2 := \langle x, x \rangle,$$

und für eine so induzierte Norm gilt die *Cauchy-Schwarzsche Ungleichung*:

$$|\langle x, y \rangle| \leq \|x\| \|y\|.$$

Das kanonische Skalarprodukt im \mathbb{C}^n ist

$$\langle x, y \rangle_2 := y^* x = \sum_{i=1}^n x_i \bar{y}_i,$$

¹Wegen (s1) ist $\langle x, x \rangle = \overline{\langle x, x \rangle}$ und damit $\langle x, x \rangle \in \mathbb{R}$.

welches die Euklidische Norm $\|x\|_2$ induziert.² In den nachfolgenden Kapiteln wird das kanonische Skalarprodukt daher in der Regel einfach mit $\langle \cdot, \cdot \rangle$ bezeichnet.

Von besonderem Interesse für uns ist das Zusammenspiel von Matrizen und Skalarprodukten. Zunächst folgt aus der Definition des kanonischen Skalarprodukts und der adjungierten Matrix sofort, dass für alle $A \in \mathbb{C}^{n \times n}$ und $x, y \in \mathbb{C}^n$ gilt:

$$\langle Ax, y \rangle = \langle x, A^*y \rangle.$$

Eine *reelle* Matrix $A \in \mathbb{R}^{n \times n}$ heißt *positiv definit*, wenn

$$\langle Ax, x \rangle > 0 \quad \text{für alle } x \neq 0,$$

$x \in \mathbb{R}^n$, gilt. Ist $A \in \mathbb{R}^{n \times n}$ symmetrisch und positiv definit, so wird durch

$$\langle x, y \rangle_A := \langle x, Ay \rangle$$

ein Skalarprodukt definiert. Dieses induziert die sogenannte *Energienorm* $\|x\|_A^2 := \langle x, Ax \rangle$. Beide werden für die Krylovraum-Verfahren von fundamentaler Wichtigkeit sein. (Diese Begriffe können auf komplexe Matrizen übertragen werden; dies wird im nächsten Kapitel behandelt.)

Zwei Vektoren $x, y \in \mathbb{C}^n$ heißen *orthogonal* (bezüglich des Skalarprodukts $\langle \cdot, \cdot \rangle$), falls $\langle x, y \rangle = 0$ gilt. Eine Menge $\{x_1, \dots, x_m\} \subset \mathbb{C}^n$ heißt *orthogonal*, falls ihre Elemente paarweise orthogonal sind, d. h. es gilt

$$\langle x_i, x_j \rangle = 0 \quad \text{für alle } i \neq j.$$

Ist zusätzlich $\langle x_i, x_i \rangle = 1$ für alle $1 \leq i \leq m$, so nennt man die Menge *orthonormal*. Zwei Mengen $\{x_1, \dots, x_m\}$ und $\{y_1, \dots, y_m\}$ heißen *biorthogonal*, wenn für alle $1 \leq i, j \leq m$ gilt:

$$\langle x_i, y_j \rangle = \delta_{ij} := \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases}$$

Ein Vektor $x \in \mathbb{C}^n$ ist *orthogonal* zu einem Unterraum $U \subset \mathbb{C}^n$ (geschrieben $x \perp U$), falls x orthogonal zu allen $u \in U$ ist. Aus der Sesquilinearität des Skalarprodukts (Eigenschaft (s2)) folgt, dass eine äquivalente Bedingung dafür die Orthogonalität zu einer Basis u_1, \dots, u_m von U ist:

$$x \perp U \quad \Leftrightarrow \quad \langle x, u_i \rangle = 0 \quad \text{für alle } 1 \leq i \leq m.$$

²Dies begründet die bereits angesprochene Sonderstellung unter den p-Normen.

Die Menge aller zu U orthogonalen Vektoren nennt man das *orthogonale Komplement* von U :

$$U^\perp := \{x \in \mathbb{C}^n : x \perp U\}.$$

Insbesondere ist $(\mathbb{C}^n)^\perp = \{0\}$. Weiterhin ist $(U^\perp)^\perp = U$ für jeden Unterraum U . Außerdem gilt für jeden Unterraum $U \subset \mathbb{C}^n$, dass

$$(2.1) \quad \mathbb{C}^n = U \oplus U^\perp$$

ist. Jeder Vektor $x \in \mathbb{C}^n$ kann also auf eindeutige Weise dargestellt werden als

$$x = u + u_\perp, \quad u \in U, u_\perp \in U^\perp.$$

Durch die Zuordnung $x \mapsto u$ wird eine lineare Abbildung $P : \mathbb{C}^n \rightarrow U$ definiert, die $P(P(x)) = P(x)$ für alle $x \in \mathbb{C}^n$ erfüllt. Solche Abbildungen nennt man Projektionen, und sie stellen die Grundlage vieler Iterationsverfahren dar.

2.2 PROJEKTIONEN

Wir betrachten zunächst möglichst allgemeine Projektionen, bevor wir auf den Spezialfall der orthogonalen Zerlegung zurückkommen. Im folgenden sei $\langle \cdot, \cdot \rangle$ ein beliebiges Skalarprodukt und $\|\cdot\|$ die dadurch induzierte Norm.

Definition 2.1. Eine lineare Abbildung $P : \mathbb{C}^n \rightarrow \mathbb{C}^n$ heißt *Projektion*, falls gilt

$$P^2 := P \circ P = P.$$

Die folgenden Eigenschaften sind eine direkte Folgerung aus der Definition. Sei P eine Projektion. Dann gilt:

- $I - P$ ist eine Projektion;
- $\text{Ker}(P) = \text{Im}(I - P)$;
- $\text{Ker}(P) \cap \text{Im}(P) = \{0\}$.

Aus den letzten beiden Eigenschaften und der Gleichung $x = x - Px + Px = (I - P)x + Px$ folgt sofort

$$\mathbb{C}^n = \text{Ker}(P) \oplus \text{Im}(P).$$

Sind umgekehrt zwei Unterräume M und S mit $\mathbb{C}^n = M \oplus S$ gegeben, können wir mit ihrer Hilfe eine Projektion P definieren: Haben wir die (eindeutige) Darstellung $x = v + w$ mit $v \in M$ und $w \in S$, so setzen wir $Px := v$. Wir nennen dann P die *Projektion auf M entlang*

S. Offensichtlich ist $\text{Im}(P) = M$ und $\text{Ker}(P) = S$. Wir halten also fest: Eine Projektion wird durch ihr Bild und ihren Kern eindeutig festgelegt.

Oft ist es bequemer, statt S das orthogonale Komplement $L := S^\perp$ zu spezifizieren. Wir sprechen dann von der *Projektion auf M senkrecht zu L* . Wieder gilt: Durch $M (= \text{Im}(P))$ und $L (= \text{Ker}(P)^\perp = \text{Im}(I-P)^\perp)$ ³ wird eine Projektion P eindeutig festgelegt, wenn $\mathbb{C}^n = M \oplus L^\perp$ ist. Eine hinreichende Bedingung für die Zerlegbarkeit von \mathbb{C}^n in eine direkte Summe von M und L^\perp kann wie folgt gegeben werden:

Satz 2.2. Seien M und L Unterräume von \mathbb{C}^n mit $\dim(M) = \dim(L) = m < n$ und $M \cap L^\perp = \{0\}$. Dann wird durch die Zuordnung $P : x \mapsto v$ mit

1. $v \in M$,
2. $(x - v) \perp L$,

eine Projektion $P : \mathbb{C}^n \rightarrow M$ eindeutig festgelegt.

Die durch die Zerlegung in (2.1) vermittelte Projektion erkennen wir also als den Spezialfall $M = L = U$ wieder, der sich durch einige nützliche Eigenschaften auszeichnet.

Definition 2.3. Eine Projektion heißt *orthogonal*, wenn $\text{Ker}(P) = \text{Im}(P)^\perp$ ist. Eine nicht-orthogonale Projektion nennt man *schiefe Projektion*.

Um orthogonale Projektionen zu charakterisieren, ohne Kern und Bild bestimmen zu müssen, betrachten wir zuerst die zu einer (schiefen) Projektion P adjungierte Abbildung P^* , definiert durch

$$(2.2) \quad \langle P^*x, y \rangle = \langle x, Py \rangle \quad \text{für alle } x, y \in \mathbb{C}^n.$$

Diese ist selber eine Projektion, denn für alle $x, y \in \mathbb{C}^n$ gilt

$$\langle (P^*)^2x, y \rangle = \langle P^*x, Py \rangle = \langle x, P^2y \rangle = \langle x, Py \rangle = \langle P^*x, y \rangle.$$

Also ist $(P^*)^2x - P^*x \in (\mathbb{C}^n)^\perp = \{0\}$ und damit $(P^*)^2 = P^*$. Weiterhin folgt aus (2.2), dass $y \in \text{Ker}(P)$ impliziert, dass $y \perp \text{Im}(P^*)$ ist, und umgekehrt. Genauso ist $x \in \text{Ker}(P^*)$ äquivalent zu $x \perp \text{Im}(P)$. Es gilt also:

$$(2.3) \quad \begin{cases} \text{Ker}(P^*) = \text{Im}(P)^\perp, \\ \text{Ker}(P) = \text{Im}(P^*)^\perp. \end{cases}$$

Da Bild und Kern die Projektion eindeutig charakterisieren, folgern wir, dass P^* die Projektion auf $\text{Ker}(P)^\perp$ senkrecht zu $\text{Im}(P)$ ist. Außerdem bedeutet (2.3) auch, dass eine Projektion P orthogonal ist, wenn $P^* = P$, also P selbstadjungiert ist. Tatsächlich gilt auch die Umkehrung:

³Daraus folgt sofort die nützliche Äquivalenz $Px = 0 \Leftrightarrow x \perp L$

Lemma 2.4. *Eine Projektion P ist orthogonal genau dann, wenn sie selbstadjungiert ist.*

Beweis. Dass selbstadjungierte Projektionen orthogonal sind, folgt direkt aus der Definition und (2.3). Sei nun P eine orthogonale Projektion. Dann ist

$$\begin{aligned}\operatorname{Ker}(P) &= \operatorname{Im}(P)^\perp = \operatorname{Ker}(P^*), \\ \operatorname{Im}(P) &= \operatorname{Ker}(P)^\perp = \operatorname{Im}(P^*),\end{aligned}$$

wieder nach Definition und (2.3). Da eine Projektion durch Bild und Kern eindeutig festgelegt wird, muss $P^* = P$ gelten. \square

Wir sammeln nun einige nützliche Eigenschaften orthogonaler Projektionen. Ist P eine orthogonale Projektion, so gilt $\operatorname{Im}(P) = \operatorname{Ker}(P)^\perp = \operatorname{Im}(I - P)^\perp$. Daraus folgt sofort:

$$\langle Px, (I - P)x \rangle = 0 \quad \text{für alle } x \in \mathbb{C}^n.$$

Also ist für $x \in \mathbb{C}^n$

$$\|x\|^2 = \|Px + (I - P)x\|^2 = \|Px\|^2 + \|(I - P)x\|^2,$$

da der gemischte Term verschwindet. Weil der letzte Summand nichtnegativ ist, muss

$$\|Px\|^2 \leq \|x\|^2$$

gelten. Da $\|Px\| = \|x\|$ für $x \in \operatorname{Im} P$ gilt, folgt aus der Definition (1.1), dass (falls $P \neq 0$)

$$\|P\| = 1$$

ist.⁴ Die für uns wichtigste Eigenschaft ist jedoch die Optimalität der orthogonalen Projektion bezüglich bestimmter Minimierungsprobleme:

Satz 2.5. *Sei P eine orthogonale Projektion auf einen Unterraum $U \subset \mathbb{C}^n$ und $z \in \mathbb{C}^n$ gegeben. Dann ist*

$$\|Pz - z\| = \min_{x \in U} \|x - z\|.$$

Beweis. Sei $x \in U$ beliebig. Dann ist auch $Pz - x \in U$ und wegen $U = \operatorname{Im}(P) = \operatorname{Im}(I - P)^\perp$ ist $\langle z - Pz, Pz - x \rangle = 0$. Daraus folgt aber:

$$\|x - z\|^2 = \|z - Pz + Pz - x\|^2 = \|x - Pz\|^2 + \|z - Pz\|^2 \geq \|z - Pz\|^2.$$

Es ist also $\|z - x\|$ mindestens gleich $\|z - Pz\|$, und dieses Minimum wird auch angenommen für $x = Pz \in U$. \square

⁴Dagegen gilt für *schiefe* Projektionen stets $\|P\| > 1$!

Wir geben noch für spätere Verwendung eine alternative Formulierung dieser Tatsache an:

Folgerung 2.6. Sei U ein Unterraum von \mathbb{C}^n und $z \in \mathbb{C}^n$. Dann ist

$$\|x^* - z\| = \min_{x \in U} \|x - z\|$$

genau dann, wenn

$$x^* \in U \quad \text{und} \quad x^* - z \perp U$$

gilt.

Um Projektionen einfach berechnen zu können, suchen wir nun nach Matrixdarstellungen von schiefen und orthogonalen Projektionen. Dazu benötigen wir eine Basis der betroffenen Unterräume. Seien M und L wieder Unterräume von \mathbb{C}^n der Dimension $m < n$, und P die Projektion auf M senkrecht zu L (bezüglich des kanonischen Skalarprodukts). Wir betrachten nun eine Basis v_1, \dots, v_m von M und eine Basis w_1, \dots, w_m von L , sowie die Matrizen

$$V := [v_1, \dots, v_m], \quad W := [w_1, \dots, w_m] \in \mathbb{C}^{n \times m}.$$

Sei nun $Px \in M$ beliebig. Dann existiert ein $\xi \in \mathbb{C}^m$ mit

$$Px = \sum_{i=1}^m \xi_i v_i = V\xi$$

und die Bedingung $x - Px \perp L$ ist äquivalent zu

$$w_j^*(x - V\xi) = \langle x - V\xi, w_j \rangle = 0 \quad \text{für alle } 1 \leq j \leq m.$$

Schreiben wir diese m Bedingungen als einen Vektor, so erhalten wir die Gleichung

$$W^*(x - V\xi) = 0.$$

Wir betrachten nun zwei Fälle:

1. V und W sind biorthogonal (d. h. $\langle v_i, w_j \rangle = \delta_{ij}$). Dann ist $W^*V = I$ und wir können nach ξ auflösen: $\xi = W^*x$. Also ist

$$(2.4) \quad Px = V\xi = VW^*x.$$

2. V und W sind nicht biorthogonal. Dann ist $W^*V\xi = W^*x$. Da nach Voraussetzung $M \cap L^\perp = \{0\}$ (also kein nicht-trivialer Vektor in M orthogonal zu L) ist, ist $W^*V \in \mathbb{C}^{m \times m}$ nicht singular (sonst müsste $w_i^*v = 0$ für ein $v = V\xi \in M$, $\xi \neq 0$, und alle $1 \leq i \leq m$ sein). Wir erhalten dann

$$Px = V(W^*V)^{-1}W^*x.$$

Ist P eine orthogonale Projektion auf M , und v_1, \dots, v_m eine orthonormale Basis, dann vereinfacht sich (2.4) zu

$$(2.5) \quad P\chi = VV^*\chi.$$

Es ist hilfreich, diese Matrixmultiplikation komponentenweise zu betrachten: Der i -te Eintrag des Vektors $V^*\chi$ ist $v_i^*\chi$. Die Matrixmultiplikation mit V bildet dann eine Linearkombination der Spalten von V :

$$(2.6) \quad VV^*\chi = \sum_{i=1}^m \langle \chi, v_i \rangle v_i.$$

Dies entspricht der gewohnten Berechnung der Projektion mit Hilfe einer Orthonormalbasis. Effiziente und stabile Algorithmen zur Konstruktion solch einer Basis stellen also einen Kern aller projektionsbasierten Verfahren dar.

2.3 ORTHONORMALISIERUNGSVERFAHREN

Jeder Unterraum besitzt eine orthonormale Basis, die sich aus einer gegebenen Basis x_1, \dots, x_m mit Hilfe des *Gram-Schmidt-Verfahrens* konstruieren lässt. Dabei wird schrittweise die neue Basis q_1, \dots, q_m so konstruiert, dass die einzelnen Vektoren q_i normiert (d. h. $\|q_i\| = 1$) und orthogonal zu den bereits konstruierten Vektoren q_j , $j < i$, sowie im Spann der x_k , $k \leq i$ sind. Wir subtrahieren von x_i also alle Beiträge, die bereits in Richtung der q_j liegen. Bezeichnet P_i die orthogonale Projektion auf $\{q_i\}$, so setzen wir

$$\tilde{q}_i := x_i - P_1x_i - P_2x_i - \dots - P_{i-1}x_i$$

und normieren anschließend

$$q_i := \|\tilde{q}_i\|^{-1} \tilde{q}_i.$$

Mit Hilfe der Darstellung (2.6) der Projektion auf $\{v_i\}$ erhalten wir den Algorithmus 2.1, von dem man sich leicht vergewissert, dass er genau dann vorzeitig abbricht, wenn die x_i linear abhängig sind, und ansonsten orthonormale Vektoren produziert.

Betrachten wir Zeile 6 und 11 in Algorithmus 2.1 und lösen wir nach x_j auf, so erhalten wir

$$x_j = \sum_{i=1}^j r_{ij} q_i.$$

Bezeichnen wir mit R die obere Dreiecksmatrix mit den Einträgen r_{ij} , $i \leq j$ und schreiben $Q = [q_1, \dots, q_m]$, so ist x_j die j te Spalte von QR und daher

$$X := [x_1, \dots, x_m] = QR.$$

Algorithmus 2.1 Gram–Schmidt-Verfahren**Input:** $x_1, \dots, x_m \in \mathbb{C}^n \setminus \{0\}$

```

1:  $r_{11} = \|x_1\|$ 
2:  $q_1 = x_1/r_{11}$ 
3: for  $j = 2, \dots, m$  do
4:   for  $i = 1, \dots, j - 1$  do
5:      $r_{ij} = \langle x_j, q_i \rangle$ 
6:    $\hat{q} = x_j - \sum_{i=1}^{j-1} r_{ij} q_i$ 
7:    $r_{jj} = \|\hat{q}\|$ 
8:   if  $r_{jj} = 0$  then
9:     stop
10:  else
11:     $q_j = \hat{q}/r_{jj}$ 

```

Output: q_1, \dots, q_m

Weiter ist Q unitär (da die Spalten von Q nach Konstruktion orthonormal sind). Das Gram–Schmidt-Verfahren berechnet also die QR-Zerlegung der Matrix X .

Aufgrund von Rundungsfehlern kann die Orthogonalität der Vektoren aber verloren gehen, wenn viele Projektionen subtrahiert werden müssen. Ein numerisch stabileres Verfahren erhält man, indem man statt der Subtraktion jeweils auf das orthogonale Komplement von $\{q_j\}$ projiziert:

$$\tilde{q}_i := (I - P_{i-1}) \dots (I - P_2)(I - P_1)x_i.$$

Dieses Vorgehen führt auf das *modifizierte Gram–Schmidt-Verfahren* (Algorithmus 2.2).

Algorithmus 2.2 Modifiziertes Gram–Schmidt-Verfahren**Input:** $x_1, \dots, x_m \in \mathbb{C}^n \setminus \{0\}$

```

1:  $r_{11} = \|x_1\|$ 
2:  $q_1 = x_1/r_{11}$ 
3: for  $j = 2, \dots, m$  do
4:    $\hat{q} = x_j$ 
5:   for  $i = 1, \dots, j - 1$  do
6:      $r_{ij} = \langle \hat{q}, q_i \rangle$ 
7:      $\hat{q} = \hat{q} - r_{ij} q_i$ 
8:    $r_{jj} = \|\hat{q}\|$ 
9:   if  $r_{jj} = 0$  then
10:    stop
11:  else
12:     $q_j = \hat{q}/r_{jj}$ 

```

Output: q_1, \dots, q_m

Neben Projektionen bieten sich auch Givens-Rotationen und Householder-Reflexionen als Grundlage für Orthonormalisierungsverfahren beziehungsweise die Berechnung der QR-Zerlegung einer Matrix an. Wir werden später auch spezialisierte Verfahren kennenlernen, die an die Struktur der Basisvektoren bestimmter Unterräume angepasst sind.

EIGENWERTE UND EIGENVEKTOREN

3

Eigenwerte (und Eigenvektoren) sind nützliche Begriffe, um die Abbildungseigenschaften von Matrizen zu charakterisieren. Sie stellen daher sehr wichtige Hilfsmittel zur Analyse iterativer Verfahren dar. Wir sammeln deshalb einige fundamentale Konzepte. Kernpunkt dieses Kapitels sind die Jordan-Normalform und insbesondere die Schurfaktorisierung als zentrales Werkzeug in allen weiteren Kapiteln.

3.1 DEFINITIONEN UND ELEMENTARE EIGENSCHAFTEN

Sei $A \in \mathbb{C}^{n \times n}$. Ein $\lambda \in \mathbb{C}$ heißt *Eigenwert* von A , falls es ein $v \in \mathbb{C}^n \setminus \{0\}$ gibt, so dass

$$Av = \lambda v$$

gilt. Solch ein v heißt dann *Eigenvektor* zum Eigenwert λ ; das Tupel (λ, v) wird auch *Eigenpaar* genannt. Wenn v ein Eigenvektor ist, dann ist auch jedes skalare Vielfache von v ein Eigenvektor zum selben Eigenwert. Wir können daher stets einen Eigenvektor mit Norm 1 finden.

Damit λ ein Eigenwert ist, muss also die Matrix $A - \lambda I$ singulär sein, und umgekehrt. Dies ist aber äquivalent mit der Bedingung $\det(A - \lambda I) = 0$. Aus dem Fundamentalsatz der Algebra folgt nun, dass jede Matrix in $\mathbb{C}^{n \times n}$ genau n Eigenwerte besitzt (die nicht verschieden sein müssen, und auch bei reellen Matrizen komplex sein können). Umgekehrt bedeutet dies, dass A singulär ist genau dann, wenn 0 ein Eigenwert von A ist.

Die Menge aller Eigenwerte von A nennt man das *Spektrum* von A ,

$$\sigma(A) := \{\lambda \in \mathbb{C} : \det(A - \lambda I) = 0\}.$$

Der *Spektralradius* von A ist definiert als der betragsgrößte Eigenwert,

$$\rho(A) := \max_{\lambda \in \sigma(A)} |\lambda|.$$

Aus der Charakterisierung von Eigenwerten über die Determinante folgt direkt: Ist λ Eigenwert von A , so ist

- λ Eigenwert von A^T und
- $\bar{\lambda}$ Eigenwert von A^* .

Einen Eigenvektor zum Eigenwert $\bar{\lambda}$ von A^* nennt man auch *Links-Eigenvektor* zum Eigenwert λ von A . Will man den Unterschied betonen, spricht man vom *Rechts-Eigenvektor* v zum Eigenwert λ von A , wenn $Av = \lambda v$ gilt.

Eine weitere nützliche Charakterisierung ist folgende: Ist v Eigenvektor zum Eigenwert λ von A , so gilt (für das kanonische Skalarprodukt)

$$(3.1) \quad \lambda = \frac{\langle Av, v \rangle}{\langle v, v \rangle}.$$

Dies bedeutet, dass selbstadjungierte Matrizen reelle Eigenwerte haben: Ist nämlich v ein normierter Eigenvektor der selbstadjungierten Matrix $A \in \mathbb{C}^{n \times n}$ zum Eigenwert λ , so gilt

$$\lambda = \langle Av, v \rangle = \langle v, Av \rangle = \overline{\langle Av, v \rangle} = \bar{\lambda},$$

weshalb der Imaginärteil von λ gleich 0 sein muss. Für Eigenwerte selbstadjungierter Matrizen gilt auch das folgende *min-max-Prinzip*:

Satz 3.1 (Courant–Fischer). Sei $A \in \mathbb{C}^{n \times n}$ selbstadjungiert und seien

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$$

die Eigenwerte von A . Dann gilt für alle $1 \leq k \leq n$:

$$\lambda_k = \min_{S, \dim(S)=n-k+1} \max_{x \in S \setminus \{0\}} \frac{\langle x, Ax \rangle}{\langle x, x \rangle} = \max_{S, \dim(S)=k} \min_{x \in S \setminus \{0\}} \frac{\langle x, Ax \rangle}{\langle x, x \rangle}.$$

Folgerung 3.2. Seien λ_i , $1 \leq i \leq n$, die wie in Satz 3.1 geordneten Eigenwerte von $A \in \mathbb{C}^{n \times n}$. Dann gilt

$$\lambda_1 = \max_{x \in \mathbb{C}^n \setminus \{0\}} \frac{\langle x, Ax \rangle}{\langle x, x \rangle},$$

$$\lambda_n = \min_{x \in \mathbb{C}^n \setminus \{0\}} \frac{\langle x, Ax \rangle}{\langle x, x \rangle}.$$

Damit erhalten wir eine explizite Darstellung der Spektralnorm einer Matrix $A \in \mathbb{C}^{n \times n}$:

$$\|A\|_2^2 = \max_{x \in \mathbb{C}^n \setminus \{0\}} \frac{\|Ax\|_2^2}{\|x\|_2^2} = \max_{x \in \mathbb{C}^n \setminus \{0\}} \frac{\langle Ax, Ax \rangle}{\langle x, x \rangle} = \max_{x \in \mathbb{C}^n \setminus \{0\}} \frac{\langle x, A^*Ax \rangle}{\langle x, x \rangle} = \sigma_1,$$

wobei σ_1 der größte Eigenwert der (selbstadjungierten) Matrix A^*A ist. Da wegen $\langle x, A^*Ax \rangle = \langle Ax, Ax \rangle = \|Ax\|_2^2 \geq 0$ und Satz 3.1 alle Eigenwerte von A^*A nichtnegativ sind, erhalten wir

$$\|A\|_2 = \sqrt{\rho(A^*A)}.$$

Umgekehrt können wir den Spektralradius einer Matrix durch eine beliebige Matrixnorm abschätzen.

Lemma 3.3. Sei $A \in \mathbb{C}^{n \times n}$ und $\|\cdot\|_M$ eine Matrixnorm, die verträglich ist mit der Vektornorm $\|\cdot\|_V$. Dann gilt

$$\rho(A) \leq (\|A^k\|_M)^{\frac{1}{k}} \quad \text{für alle } k \in \mathbb{N}.$$

Beweis. Sei $\lambda \in \mathbb{C}$ ein Eigenwert von A mit zugehörigem Eigenvektor v . Dann ist

$$|\lambda|^k \|v\|_V = \|\lambda^k v\|_V = \|A^k v\|_V \leq \|A^k\|_M \|v\|_V.$$

Division auf beiden Seiten mit $\|v\|_V \neq 0$ liefert die gewünschte Abschätzung. □

Für positiv definite Matrizen erhalten wir aus der Charakterisierung (3.1), dass alle Eigenwerte $\lambda > 0$ erfüllen: Sowohl Zähler als auch Nenner müssen positiv sein. Wir bekommen sogar eine quantitative Abschätzung:

Satz 3.4. Sei $A \in \mathbb{R}^{n \times n}$ positiv definit. Dann ist A nicht singulär, und es existiert ein $\alpha > 0$ mit

$$\langle Ax, x \rangle \geq \alpha \|x\|_2^2 \quad \text{für alle } x \in \mathbb{R}^n.$$

Beweis. Wäre A singulär, so müsste ein $x \in \mathbb{R}^n \setminus \{0\}$ mit $Ax = 0$ existieren. Dann wäre aber $\langle Ax, x \rangle = 0$, was im Widerspruch zur Definitheit von A steht. Um die Abschätzung zu zeigen, schreiben wir

$$A = \frac{1}{2}(A + A^T) + \frac{1}{2}(A - A^T) =: S_1 + S_2.$$

Man prüft leicht nach, dass S_1 symmetrisch und positiv definit ist, und dass $\langle x, S_2 x \rangle = 0$ für alle $x \in \mathbb{R}^n$ gilt. Folgerung 3.2 ergibt dann für beliebiges $x \in \mathbb{R}^n \setminus \{0\}$:

$$\frac{\langle Ax, x \rangle}{\langle x, x \rangle} \geq \min_{x \in \mathbb{R}^n \setminus \{0\}} \frac{\langle Ax, x \rangle}{\langle x, x \rangle} = \min_{x \in \mathbb{R}^n \setminus \{0\}} \frac{\langle x, S_1 x \rangle}{\langle x, x \rangle} = \sigma_n,$$

wobei σ_n der kleinste Eigenwert von S_1 ist. Da A positiv definit ist, können wir $\alpha = \sigma_n > 0$ wählen. □

Da eine selbstadjungierte Matrix $A \in \mathbb{C}^{n \times n}$ reelle Eigenwerte hat, kann man obige Überlegungen auch auf komplexe selbstadjungierte und positiv definite Matrizen (d. h. $\langle Ax, x \rangle > 0$ gilt für alle $x \in \mathbb{C}^{n \times n} \setminus \{0\}$) anwenden. (Im Gegensatz zum reellen Fall ist positive Definitheit aber nur für selbstadjungierte Matrizen definiert.)

3.2 ÄHNLICHKEITSTRANSFORMATIONEN

In diesem Abschnitt betrachten wir die Geometrie der Eigenvektoren genauer. Insbesondere fragen wir uns, ob es eine Basis des \mathbb{C}^n gibt, in der die Eigenwerte einer Matrix direkt ablesbar sind. Matrizen, die dieselbe lineare Abbildung, aber bezüglich unterschiedlicher Basen, vermitteln, nennt man *ähnlich*.

Definition 3.5. Zwei Matrizen $A, B \in \mathbb{C}^{n \times n}$ heißen *ähnlich*, falls eine reguläre Matrix $X \in \mathbb{C}^{n \times n}$ existiert mit

$$A = XBX^{-1}.$$

Ist X unitär (orthogonal), so nennt man A *unitär (orthogonal) ähnlich* zu B .

Ist λ ein Eigenwert von B mit Eigenvektor v , so ist λ auch Eigenwert von A mit Eigenvektor $w := Xv$, denn

$$\lambda w = X(\lambda v) = XBv = AXv.$$

Der einfachste Fall ist sicherlich, wenn die Matrix B *diagonal* ist (d. h. $b_{ij} = 0$ für $i \neq j$ gilt), da dann die Eigenwerte genau die Diagonaleinträge sind. Solche Matrizen nennen wir *diagonalisierbar*. Die Spalten von X bilden in diesem Fall eine Basis von \mathbb{C}^n aus Eigenvektoren von A :

Satz 3.6. Eine Matrix $A \in \mathbb{C}^{n \times n}$ ist dann und nur dann diagonalisierbar, wenn A genau n linear unabhängige Eigenvektoren hat.

Beweis. Sei $X \in \mathbb{C}^{n \times n}$ regulär und $D \in \mathbb{C}^{n \times n}$ eine Diagonalmatrix mit $A = XDX^{-1}$ beziehungsweise $AX = XD$. Dies ist äquivalent zu der Bedingung, dass alle Spalten x_i , $1 \leq i \leq n$, von X linear unabhängig sind und dass $Ax_i = d_{ii}x_i$ gilt. Das bedeutet aber, dass d_{ii} Eigenwert von A mit Eigenvektor x_i ist für alle $1 \leq i \leq n$. \square

Man prüft leicht nach, dass Eigenvektoren zu unterschiedlichen Eigenwerten linear unabhängig sind.¹ Allerdings gibt es Matrizen mit mehrfachen Eigenwerten (d. h. $\lambda_i = \lambda_j$ für $i \neq j$), deren zugehörige Eigenvektoren zusammenfallen. Was können wir in diesem Fall erwarten?

Verlangen wir nur Ähnlichkeit zu einer *Block-Diagonalmatrix*, so liefert uns die *Jordan-Normalform* das Gewünschte:

¹Ist A normal, sind die Eigenvektoren sogar orthogonal, wie wir später sehen werden.

Satz 3.7. Jede Matrix $A \in \mathbb{C}^{n \times n}$ ist ähnlich zu einer Block-Diagonalmatrix $J \in \mathbb{C}^{n \times n}$,

$$J = \begin{pmatrix} J_1 & & 0 \\ & J_2 & \\ & & \ddots \\ 0 & & & J_p \end{pmatrix}, \quad J_i = \begin{pmatrix} J_{i1} & & 0 \\ & J_{i2} & \\ & & \ddots \\ 0 & & & J_{im_i} \end{pmatrix}, \quad J_{ik} = \begin{pmatrix} \lambda_i & 1 & & 0 \\ & \ddots & \ddots & \\ & & \lambda_i & 1 \\ 0 & & & \lambda_i \end{pmatrix},$$

wobei jeder Block J_i zu einem verschiedenen Eigenwert λ_i von A und jeder Unterblock J_{ik} zu einem verschieden Eigenvektor v_{ik} zum Eigenwert λ_i korrespondiert.

Die Jordan-Normalform ist ein sehr wichtiges theoretisches Hilfsmittel, aber nicht für numerische Berechnungen geeignet, da sie sehr schlecht konditioniert ist. Für unsere Zwecke genügt es jedoch, eine obere Dreiecksmatrix R zu finden (d. h. $t_{ij} = 0$ für alle $i > j$), da auch dort die Eigenwerte auf der Diagonalen stehen. (Dies folgt direkt aus dem Laplaceschen Entwicklungssatz für $\det(A - \lambda I)$.) Dass dies immer möglich ist, garantiert die *Schur-Normalform*.

Satz 3.8. Sei $A \in \mathbb{C}^{n \times n}$. Dann existiert eine unitäre Matrix $Q \in \mathbb{C}^{n \times n}$ und eine obere Dreiecksmatrix $R \in \mathbb{C}^{n \times n}$ mit $A = Q^* R Q$.

Beweis. Wir verwenden Induktion nach n . Für $n = 1$ ist die Aussage trivial. Sei nun $n > 1$ und $A \in \mathbb{C}^{n \times n}$ beliebig. Dann hat A einen Eigenwert λ mit zugehörigem Eigenvektor v , den wir normiert wählen können ($\|v\| = 1$). Die Menge $\{v\}$ kann dann zu einer orthonormalen Basis v, u_2, \dots, u_n ergänzt werden, die wir als Matrix $V = [v, U]$ mit $U = [u_2, \dots, u_n]$ schreiben. Dann ist $AV = [\lambda v, AU]$ und deshalb

$$(3.2) \quad V^* A V = \begin{bmatrix} v^* \\ U^* \end{bmatrix} [\lambda v, AU] = \begin{pmatrix} \lambda & v^* A U \\ 0 & U^* A U \end{pmatrix}.$$

Wir wenden nun die Induktionsvoraussetzung auf $B := U^* A U \in \mathbb{C}^{(n-1) \times (n-1)}$ an, und erhalten eine unitäre Matrix $Q_1 \in \mathbb{C}^{(n-1) \times (n-1)}$ und eine obere Dreiecksmatrix $R_1 \in \mathbb{C}^{(n-1) \times (n-1)}$ mit $R_1 = Q_1^* B Q_1$. Multiplizieren wir nun (3.2) mit der unitären Matrix

$$\hat{Q} := \begin{pmatrix} 1 & 0 \\ 0 & Q_1 \end{pmatrix}$$

von rechts und mit \hat{Q}^* von links, so ergibt das

$$\hat{Q}^* (V^* A V) \hat{Q} = \begin{pmatrix} \lambda & v^* A U \\ 0 & Q_1^* B Q_1 \end{pmatrix} = \begin{pmatrix} \lambda & v^* A U \\ 0 & R_1 \end{pmatrix},$$

mit einer oberen Dreiecksmatrix R_1 . Da die Spalten von V nach Konstruktion orthonormal sind, ist $Q := V \hat{Q} \in \mathbb{C}^{n \times n}$ als Produkt unitärer Matrizen unitär, woraus die Aussage folgt. \square

Die Spalten von Q werden als *Schurvektoren* bezeichnet; ist A normal, so sind die Schurvektoren auch Eigenvektoren. Für eine reelle Matrix A ist auch eine rein reelle Zerlegung möglich. Da die Eigenwerte komplex sein können, erhalten wir in diesem Fall nur noch eine obere Block-Dreiecksmatrix.

Folgerung 3.9 (Reelle Schur-Normalform). *Sei $A \in \mathbb{R}^{n \times n}$. Dann existiert eine orthogonale Matrix $Q \in \mathbb{R}^{n \times n}$, so dass*

$$(3.3) \quad Q^T A Q = \begin{pmatrix} R_1 & & * \\ & \ddots & \\ 0 & & R_m \end{pmatrix} =: R$$

gilt, wobei alle R_i reell und entweder 1×1 - oder 2×2 -Matrizen sind. (In letzterem Fall hat R_i ein Paar komplex konjugierter Eigenwerte.) Weiterhin gilt:

$$\sigma(A) = \bigcup_{i=1}^m \sigma(R_i).$$

Die Schurform erlaubt uns, einige interessante Aussagen über normale Matrizen zu beweisen.

Lemma 3.10. *Sei $A \in \mathbb{C}^{n \times n}$ normal und habe obere Dreiecksform. Dann ist A eine Diagonalmatrix.*

Beweis. Für eine normale Matrix A gilt $A^* A = A A^*$. Berechnen wir das Matrixprodukt und betrachten jeweils das erste Diagonalelement, so erhalten wir

$$|a_{11}|^2 = \sum_{j=1}^n |a_{1j}|^2.$$

Dies ist aber nur möglich, wenn $a_{1j} = 0$ ist für $1 < j \leq n$. Damit bleibt auch im zweiten Diagonalelement von $A^* A$ nur der quadratische Term übrig, und wir folgern ebenso, dass $a_{2j} = 0$ für $2 < j \leq n$ ist. Wenn wir diese Prozedur bis zum $(n - 1)$ -ten Diagonalelement fortsetzen, erhalten wir wie gewünscht, dass $a_{ij} = 0$ ist für alle $i < j$. \square

Damit können wir nun folgendes Resultat beweisen:

Satz 3.11. *Eine Matrix $A \in \mathbb{C}^{n \times n}$ ist normal genau dann, wenn sie unitär ähnlich zu einer Diagonalmatrix ist.*

Beweis. Wenn A unitär ähnlich zu einer Diagonalmatrix ist, dann existiert eine unitäre Matrix Q und eine Diagonalmatrix D mit $A = Q D Q^*$. Damit gilt

$$\begin{aligned} A^* A &= (Q D Q^*)^* (Q D Q^*) = Q D^* (Q^* Q) D Q^* = Q D^* D Q^* \\ &= Q D D^* Q^* = Q D (Q^* Q) D^* Q^* = (Q D Q^*) (Q D Q^*)^* \\ &= A A^*, \end{aligned}$$

da Diagonalmatrizen miteinander kommutieren.

Sei nun umgekehrt A normal. Die Schurform von A ist dann $A = Q^*RQ$ mit einer unitären Matrix Q und einer oberen Dreiecksmatrix R . Daraus folgt

$$Q^*R^*RQ = (Q^*R^*Q)(Q^*RQ) = A^*A = AA^* = (Q^*RQ)(Q^*R^*Q) = Q^*RR^*Q.$$

Multiplizieren wir mit Q von links und Q^* von rechts, erhalten wir $R^*R = RR^*$. Also ist R eine normale, obere Dreiecksmatrix und darum nach Lemma 3.10 diagonal. Die Schurform liefert also die gesuchte Ähnlichkeitstransformation. \square

Eine normale Matrix ist also diagonalisierbar, und insbesondere bilden ihre normierten Eigenvektoren eine *orthonormale* Basis von \mathbb{C}^n (nämlich die Spalten von Q). Normale Matrizen sind gewissermaßen der Idealfall für viele Algorithmen. Das gilt besonders für die selbstadjungierten Matrizen, die offensichtlich auch normal sind. Wir haben bereits gesehen, dass solche Matrizen reelle Eigenwerte besitzen. Umgekehrt muss eine normale Matrix mit reellen Eigenwerten selbstadjungiert sein: Ist A normal, so gibt es eine unitäre Matrix Q und eine Diagonalmatrix D mit $A = QDQ^*$. Da A und D die gleichen (reellen) Eigenwerte haben, muss $D = D^*$ sein, also $A = QDQ^* = QD^*Q^* = (QDQ^*)^* = A^*$.

Teil II

LINEARE GLEICHUNGSSYSTEME

STATIONÄRE VERFAHREN

4

In diesem und den folgenden Kapiteln suchen wir zu gegebener regulärer Matrix $A \in \mathbb{C}^{n \times n}$ und gegebenem $b \in \mathbb{C}^n$ die Lösung $x^* \in \mathbb{C}^n$ des linearen Gleichungssystems $Ax = b$. Wir betrachten hier *iterative Verfahren*, die ausgehend von einem *Startwert* $x^0 \in \mathbb{C}^n$ immer bessere Näherungen x^1, x^2, \dots an x^* konstruieren. Die Verfahren, die wir in diesem Kapitel behandeln, verwenden für die Berechnung der neuen Näherung x^{k+1} nur die Informationen des letzten Schrittes x^k . Ein *stationäres Iterationsverfahren* hat die Form

$$(4.1) \quad x^{k+1} = Gx^k + c$$

für feste $G \in \mathbb{C}^{n \times n}$ und $c \in \mathbb{C}^n$ sowie ein frei gewähltes x^0 . Natürlich können G und c nicht beliebig sein. Damit zum Beispiel der Grenzwert $\hat{x} = G\hat{x} + c$ von (4.1) die Gleichung $A\hat{x} = b$ löst, muss

$$(I - G)^{-1}c = \hat{x} = A^{-1}b$$

gelten. (Die Invertierbarkeit von $I - G$ werden wir später untersuchen.)

Ein klassisches Beispiel ist die *Richardson-Iteration*¹, die auf folgender Idee beruht: Wenn $Ax^* = b$ gilt, so ist

$$x^* = x^* - Ax^* + b.$$

Dies führt auf die *Fixpunktiteration*

$$(4.2) \quad x^{k+1} = (I - A)x^k + b.$$

Die Richardson-Iteration ist also ein stationäres Iterationsverfahren mit $G = I - A$ und $c = b$.

¹Diese Methode geht auf [Lewis Fry Richardson](#) zurück. Er propagierte auch 1922 die heutige Methode der Wettervorhersage auf Basis von numerischer Simulation. (Ein eigener erster Versuch von 1910 – durchgeführt von Hand! – war grundsätzlich korrekt, lieferte aber wegen gestörter Eingabedaten ein falsches Ergebnis.)

4.1 KONVERGENZ STATIONÄRER ITERATIONSVERFAHREN

Wir sagen, das Iterationsverfahren (4.1) *konvergiert* (in der Norm $\|\cdot\|$), wenn gilt

$$\lim_{k \rightarrow \infty} \|x^k - x^*\| = 0 \quad \text{für jeden Startwert } x^0.$$

Betrachten wir nun den Fehler $e^k := x^k - x^*$ im Schritt k und verwenden die Fixpunktgleichung $x^* = Gx^* + c$, so erhalten wir

$$\begin{aligned} x^k - x^* &= Gx^{k-1} + c - x^* = Gx^{k-1} - Gx^* \\ &= G(x^{k-1} - x^*) = G^2(x^{k-2} - x^*) = \dots \\ &= G^k(x^0 - x^*). \end{aligned}$$

Damit haben wir

$$\|e^k\| = \|G^k e^0\| \leq \|G^k\| \|e^0\| \leq \|G\|^k \|e^0\|,$$

wobei $\|\cdot\|$ eine beliebige verträgliche Matrixnorm sein kann.

Ist also $\|G\| < 1$, so konvergiert das Iterationsverfahren. Da die Anwendung der Submultiplikativität nur eine sehr grobe Abschätzung liefert, suchen wir eine schärfere notwendige Bedingung für die Konvergenz. Dafür benötigen wir folgendes Resultat über Matrixpotenzen.

Lemma 4.1. *Für $A \in \mathbb{C}^{n \times n}$ gilt $\|A^k\| \rightarrow 0$ dann und nur dann, wenn $\rho(A) < 1$ ist.*

Beweis. Es gelte $\|A^k\| \rightarrow 0$. Betrachte den betragsgrößten Eigenwert λ_1 von A mit zugehörigem Eigenvektor v mit $\|v\| = 1$. Dann gilt $A^k v = \lambda_1^k v$, und damit

$$|\lambda_1|^k = \|\lambda_1^k v\| = \|A^k v\| \leq \|A^k\| \|v\| \rightarrow 0.$$

Dies kann aber nur dann sein, wenn $\rho(A) = |\lambda_1| < 1$ ist.

Sei umgekehrt $\rho(A) < 1$, und betrachte die Jordan-Normalform von $A = XJX^{-1}$. Dann ist

$$A^k = XJ^k X^{-1}.$$

Es genügt also zu zeigen, dass $\|J^k\|$ gegen Null konvergiert. Dafür verwenden wir die Struktur der Jordan-Matrix: Das Produkt von Block-Diagonalmatrizen ist wieder eine Block-Diagonalmatrix, das heißt in unserem Fall

$$J^k = \begin{pmatrix} J_1^k & & \\ & \ddots & \\ & & J_r^k \end{pmatrix}.$$

Also können wir die Jordan-Blöcke separat untersuchen. Jeder Block hat die Form

$$J_i = \lambda_i I + E_i \in \mathbb{C}^{l_i \times l_i}, \quad (E_i)_{jk} = \begin{cases} 1, & k = j + 1, \\ 0, & \text{sonst.} \end{cases}$$

Die Matrizen E_i sind *nilpotent*, da $E_i^{l_i} = 0$ gilt. Für $k \geq l_i$ ist also nach der binomischen Formel (die Identität kommutiert mit jeder Matrix)

$$J_i^k = (\lambda_i I + E_i)^k = \sum_{j=0}^{l_i-1} \binom{k}{j} \lambda_i^{k-j} E_i^j.$$

Die Dreiecksungleichung liefert daher

$$\|J_i^k\| \leq \sum_{j=0}^{l_i-1} \binom{k}{j} |\lambda_i|^{k-j} \|E_i^j\|.$$

Da $|\lambda_i| < 1$ gilt, konvergiert jeder Term in der endlichen Summe gegen 0 (für festes j ist $\binom{k}{j} = \mathcal{O}(k^j)$), und damit auch $\|J_i^k\| \rightarrow 0$. Daraus folgt die Aussage. \square

Wir können also die Konvergenz stationärer Iterationsverfahren wie folgt charakterisieren:

Satz 4.2. Die Iteration (4.1) für $G \in \mathbb{C}^{n \times n}$ und $c \in \mathbb{C}^n$ konvergiert für alle Startwerte $x^0 \in \mathbb{C}^n$ gegen $(I - G)^{-1}c$, wenn $\rho(G) < 1$ gilt. Ist $\rho(G) \geq 1$, so existiert zumindest ein Startwert, für den die Iteration nicht konvergiert.

Da jede Norm eine obere Schranke für den Spektralradius darstellt, konvergiert die Richardson-Iteration (4.2) also gegen die Lösung x^* von $Ax = b$, wenn $\|I - A\| < 1$ ist. Dies bedeutet, dass A bereits „nahe“ an der Identität sein muss, was offensichtlich eine sehr schwere Einschränkung darstellt. Deshalb sucht man nach besseren Iterationsmatrizen, die auch in anderen Fällen zur Konvergenz führen. Einige klassische Ansätze werden im nächsten Abschnitt beschrieben.

Aus der Darstellung des Iterationsfehlers $e^k = G^k e^0$ lässt sich auch eine Abschätzung der Konvergenzgeschwindigkeit herleiten: Im (geometrischen) Mittel wird der Fehler um den Faktor $\|G^k\|^{1/k}$ reduziert. Ist man an der asymptotischen Rate interessiert (also für $k \rightarrow \infty$), kann man folgendes Lemma verwenden, dessen Beweis ähnlich dem von Lemma 4.1 auf der Jordan-Normalform basiert.

Lemma 4.3. Sei $\|\cdot\|$ *submultiplikativ und verträglich mit einer Vektornorm*. Dann gilt für $A \in \mathbb{C}^{n \times n}$:

$$\lim_{k \rightarrow \infty} \|A^k\|^{1/k} = \rho(A).$$

Daraus kann man abschätzen, wie viele Schritte nötig sind, um den Fehlern unter eine vorgegebene Schranke $\tilde{\varepsilon} := \varepsilon \|e^0\|$ zu bekommen: Umformen von $\|G^k\|^{1/k} \leq \varepsilon$ und Grenzübergang auf einer Seite ergibt

$$k \geq \frac{\ln \varepsilon}{\ln \rho(G)}.$$

4.2 KLASSISCHE ZERLEGUNGS-STRATEGIEN

Ist $C \in \mathbb{C}^{n \times n}$ eine reguläre Matrix, so gilt $Ax^* = b$ genau dann, wenn $CAx^* = Cb$ ist. Die zugehörige Fixpunktiteration lautet

$$x^{k+1} = (I - CA)x^k + Cb.$$

Wir suchen also Matrizen, für die $\|I - CA\| < 1$ ist, das heißt $CA \approx I$ gilt.

Die klassischen Methoden gehen dafür alle von der Zerlegung $A = L + D + R$ mit einer unteren Dreiecksmatrix L , einer Diagonalmatrix D und einer oberen Dreiecksmatrix R aus (wobei in L und R die Diagonaleinträge 0 sind).

JACOBI-ITERATION: $C = D^{-1}$

Diese Wahl² führt zu der Iterationsvorschrift

$$x^{k+1} = (I - D^{-1}A)x^k + D^{-1}b,$$

die durchführbar ist, solange $a_{ii} \neq 0$ ist für alle $1 \leq i \leq n$. Einsetzen von $A = L + D + R$ liefert

$$Dx^{k+1} = b - (L + R)x^k.$$

Durch zeilenweise Betrachtung erhält man daraus den folgenden Algorithmus.

Algorithmus 4.1 Jacobi-Iteration

Input: $a_{ij}, b_i, x_i^0, \varepsilon > 0, k = 0$

1: **while** $\|b - Ax^k\| > \varepsilon \|b - Ax^0\|$ **do**

2: **for** $i = 1$ to n **do**

3: $x_i^{k+1} = \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^k - \sum_{j=i+1}^n a_{ij}x_j^k \right) / a_{ii}$

4: $k \leftarrow k + 1$

Output: x_i^k

Die Konvergenz kann nur für bestimmte Klassen von Matrizen gezeigt werden. Eine Matrix $A \in \mathbb{C}^{n \times n}$ heißt *streng diagonal dominant*, falls

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}| \quad \text{für alle } 1 \leq i \leq n.$$

²Dies ist eine stark vereinfachte Fassung des [Jacobi-Verfahrens](#) für die Eigenwertberechnung, welches [Carl Gustav Jacobi](#) 1846 für die Berechnung von Störungen von Planetenbahnen entwickelte.

Satz 4.4. Sei $A \in \mathbb{C}^{n \times n}$ streng diagonal dominant. Dann konvergiert die Jacobi-Iteration.

Beweis. Es genügt zu zeigen, dass für eine beliebige induzierte Matrixnorm $\|I - CA\| < 1$ gilt. Wir wählen die Zeilensummennorm und erhalten

$$\|I - D^{-1}A\|_{\infty} = \|D^{-1}(L + R)\|_{\infty} = \max_{1 \leq i \leq n} \sum_{j \neq i} \frac{|a_{ij}|}{|a_{ii}|} < 1,$$

da A streng diagonal dominant ist. Aus Satz 4.2 folgt dann die Konvergenz. □

Dieses Verfahren wird trotz seiner im Allgemeinen vergleichsweise schlechten Konvergenzeigenschaften in der Praxis noch öfters verwendet (nicht zuletzt wegen seiner einfachen Implementierung). Ein weiterer Vorteil ist, dass die Berechnung der x_i in jedem Schritt unabhängig voneinander erfolgt; das Verfahren ist also sehr gut parallelisierbar.

GAUSS-SEIDEL-ITERATION: $C = (D + L)^{-1}$

Der Iterationsschritt³ dazu lautet also:

$$x^{k+1} = (I - (D + L)^{-1}A)x^k + (D + L)^{-1}b.$$

Dabei berechnet man $(D + L)^{-1}$ nicht explizit; da $(D + L)$ eine untere Dreiecksmatrix ist, lässt sich das zugehörige Gleichungssystem leicht durch Vorwärtssubstitution lösen:

$$(D + L)x^{k+1} = b - Rx^k.$$

Führt man Matrix-Multiplikation und Substitution zeilenweise aus, erhält man das folgende Verfahren.

Algorithmus 4.2 Gauß–Seidel-Iteration

Input: $a_{ij}, b_i, x_i^0, \varepsilon > 0$

1: **while** $\|b - Ax^k\| > \varepsilon \|b - Ax^0\|$ **do**

2: **for** $i = 1$ to n **do**

3: $x_i^{k+1} = \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum_{j=i+1}^n a_{ij}x_j^k \right) / a_{ii}$

4: $k \leftarrow k + 1$

Output: x_i^k

Dies entspricht der Jacobi-Iteration, wenn man bereits berechnete x_i in der Gleichung für x_j einsetzt. Damit ist das Verfahren nicht mehr ohne weiteres parallelisierbar. Außerdem hängt die Konvergenzgeschwindigkeit von der Reihenfolge der Berechnung der x_i ab, ist aber für wichtige Klassen von Matrizen höher als bei der Jacobi-Iteration. Für die Konvergenz halten wir fest:

³publiziert 1874 von [Philipp Ludwig von Seidel](#)

Satz 4.5. Sei $A \in \mathbb{C}^{n \times n}$ streng diagonal dominant oder symmetrisch positiv definit. Dann konvergiert die Gauß-Seidel-Iteration.

SOR-VERFAHREN (“successive over-relaxation”)

Das SOR-Verfahren⁴ basiert auf einer anderen Schreibweise der Gauß–Seidel-Iteration

$$x_i^{k+1} = x_i^k + \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum_{j=i}^n a_{ij}x_j^k \right) / a_{ii}.$$

Wir addieren also in jedem Schritt eine “Korrektur” zu x_i^k . Konvergiert das Verfahren nun zu langsam, können wir womöglich durch eine stärkere Korrektur die Konvergenz beschleunigen. (Konvergiert das Verfahren nicht, könnte eine kleinere Korrektur die Konvergenz wiederherstellen.) Wir multiplizieren den Korrekturterm deshalb mit einem *Relaxationsparameter* ω , und erhalten

Algorithmus 4.3 SOR-Iteration

Input: $a_{ij}, b_i, x_i^0, \varepsilon > 0, k = 0, \omega$

1: **while** $\|b - Ax^k\| > \varepsilon \|b - Ax^0\|$ **do**

2: **for** $i = 1$ to n **do**

3: $x_i^{k+1} = x_i^k + \omega \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum_{j=i}^n a_{ij}x_j^k \right) / a_{ii}$

4: $k \leftarrow k + 1$

Output: x_i^k

Durch Wahl des richtigen Parameters lässt sich die Konvergenz des Verfahrens entscheidend beschleunigen. Leider ist der optimale Parameter in der Regel nicht bekannt; man kann aber zumindest beweisen, welche Parameter überhaupt zu einem konvergenten Verfahren führen:

Satz 4.6. Sei $A \in \mathbb{C}^{n \times n}$. Dann konvergiert das SOR-Verfahren genau dann, wenn gilt

- $\omega \in (0, 2)$ für symmetrisch positiv definite A ,
- $\omega \in (0, 1]$ für streng diagonal dominante A .

⁴entwickelt von [David Young](#) für seine 1950 verfasste Doktorarbeit über die numerische Lösung von partiellen Differentialgleichungen

PROJEKTIONSVERFAHREN

5

Die meisten modernen iterativen Verfahren beruhen auf Projektionen auf geeignete Unterräume. Es ist daher hilfreich, zuerst allgemeine Projektionsverfahren für die Lösung linearer Gleichungssysteme zu betrachten und die einfachsten eindimensionalen Beispiele kennenzulernen, bevor man zu verbreiteten Verfahren wie CG und GMRES übergeht. Die in diesem Kapitel vorgestellten Methoden stellen außerdem Prototypen für die später behandelten Algorithmen dar.

5.1 ALLGEMEINE PROJEKTIONSVERFAHREN

Die Grundidee von Projektionsverfahren ist, die Lösung $x^* \in \mathbb{C}^n$ des linearen Gleichungssystems $Ax = b$ für $A \in \mathbb{C}^{n \times n}$ und $b \in \mathbb{C}^n$ durch ein \tilde{x} in einem gegebenen Unterraum $\mathcal{K} \subset \mathbb{C}^n$ anzunähern. Hat \mathcal{K} die Dimension $m \leq n$, so hat \tilde{x} genau m Freiheitsgrade, die durch m linear unabhängige Bedingungen festgelegt werden können. Dafür fordert man, dass das *Residuum* $b - A\tilde{x}$ orthogonal zu einem weiteren Unterraum $\mathcal{L} \subset \mathbb{C}^n$ der Dimension m ist. Man sucht also ein \tilde{x} , das die *Galerkin-Bedingungen*¹

$$\begin{cases} \tilde{x} \in \mathcal{K} \\ b - A\tilde{x} \perp \mathcal{L} \end{cases}$$

erfüllt. Üblicherweise werden in einem Projektionsverfahren eine Folge solcher Näherungen \tilde{x} (jeweils mit neuen Unterräumen \mathcal{K} und \mathcal{L}) berechnet, wobei man in jedem Schritt die bereits berechnete Näherung ausnutzt. Man sucht für einen gegebenen Startwert x^0 also eine neue Näherung

$$\tilde{x} \in x^0 + \mathcal{K} := \{x^0 + \delta x : \delta x \in \mathcal{K}\}.$$

Die Orthogonalitätsbedingung ist in diesem Fall $b - A(x^0 + \delta x) \perp \mathcal{L}$. Führen wir das Residuum des Startwerts $r^0 := b - Ax^0$ ein, so kann man ein die Galerkin-Bedingungen erfüllendes \tilde{x} konstruieren durch

¹Nach [Boris Grigorjewitsch Galjorkin](#), einem russischen Bauingenieur. Sein Verfahren der Finiten Elemente, publiziert 1915, beruht auf einem ähnlichen Ansatz.

1. Berechne $\delta x \in \mathcal{K}$, so dass $\langle r^0 - A\delta x, w \rangle = 0$ für alle $w \in \mathcal{L}$ ist,
2. Setze $\tilde{x} = x^0 + \delta x$.

Ist v_1, \dots, v_m eine Basis von \mathcal{K} und w_1, \dots, w_m eine Basis von \mathcal{L} , so können wir den Projektionsschritt in Matrixform schreiben, indem wir wieder die Matrizen $V = [v_1, \dots, v_m]$ und $W = [w_1, \dots, w_m]$ einführen und den Koeffizientenvektor $\xi \in \mathbb{C}^m$ mit $V\xi = \delta x$ betrachten. Die Galerkin-Bedingungen sind dann äquivalent zu

$$\begin{cases} \tilde{x} = x^0 + V\xi, \\ W^*AV\xi = W^*r^0. \end{cases}$$

Ist $W^*AV \in \mathbb{C}^{m \times m}$ invertierbar, so ist die neue Näherung explizit gegeben durch

$$\tilde{x} = x^0 + V(W^*AV)^{-1}W^*r^0.$$

Ein abstraktes Projektionsverfahren hat also die folgende Form:

Algorithmus 5.1 Allgemeines Projektionsverfahren

Input: $A \in \mathbb{C}^{n \times n}$, $b \in \mathbb{C}^n$, $x^0 \in \mathbb{C}^n$, $k = 0$

- 1: $r^0 = b - Ax^0$
- 2: **while** $\|r^k\|_2 > 0$ **do**
- 3: Wähle Unterräume \mathcal{K}, \mathcal{L}
- 4: Wähle Basen V, W von \mathcal{K} und \mathcal{L}
- 5: $\xi = (W^*AV)^{-1}W^*r^k$
- 6: $x^{k+1} = x^k + V\xi$
- 7: $r^{k+1} = b - Ax^{k+1}$
- 8: $k \leftarrow k + 1$

Output: x^k

Die tatsächlich verwendeten Verfahren stellen in der Regel die Matrix W^*AV nicht explizit auf, sondern nutzen die Struktur von \mathcal{K} und \mathcal{L} und ihrer Basen aus, um den Schritt 5 effizient zu berechnen.

Man unterscheidet jetzt zwei wichtige Klassen von Verfahren:

1. *Ritz-Galerkin-Verfahren* sind orthogonale Projektionsverfahren mit $\mathcal{L} = \mathcal{K}$.
2. *Petrov-Galerkin-Verfahren* sind schiefe Projektionsverfahren mit $\mathcal{L} \neq \mathcal{K}$. Als besonders attraktive Wahl wird sich in diesem Fall $\mathcal{L} = A\mathcal{K}$ erweisen.

RITZ-GALERKIN-VERFAHREN: $\mathcal{L} = \mathcal{K}$

Als erstes müssen wir überlegen, wann wir Invertierbarkeit der Matrix W^*AV garantieren können. Dafür ist es nicht ausreichend, dass A invertierbar ist, wie man am Beispiel

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad V = W = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

erkennt. Es darf kein Vektor in $A\mathcal{K}$ orthogonal zu $\mathcal{L} = \mathcal{K}$ sein. Dies ist der Fall, wenn A selbstadjungiert und positiv definit ist: Seien V und W Basen von \mathcal{K} (wobei der Fall $W = V$ natürlich eingeschlossen ist). Dann existiert eine invertierbare Matrix $X \in \mathbb{C}^{m \times m}$ mit $W = VX$, und es gilt

$$W^*AV = X^*V^*AV.$$

Da A positiv definit ist, gilt für alle $\xi \in \mathbb{C}^m \setminus \{0\}$, dass

$$\langle V^*AV\xi, \xi \rangle = \langle AV\xi, V\xi \rangle =: \langle A\eta, \eta \rangle > 0,$$

ist, da V vollen Spaltenrang hat und daher injektiv ist. Also ist auch V^*AV positiv definit und insbesondere invertierbar, und damit hat X^*V^*AV vollen Rang.

Aufgrund der Eigenschaften von A definiert $\langle \cdot, \cdot \rangle_A := \langle \cdot, A \cdot \rangle$ eine Norm $\|\cdot\|_A$, und wir erhalten eine A -orthogonale Projektion. Analog zu Folgerung 2.6 können wir folgende Optimalitätseigenschaft dieses Projektionsverfahrens beweisen.

Satz 5.1. Sei \mathcal{K} ein Unterraum von \mathbb{C}^n , $x^0, b \in \mathbb{C}^n$ beliebig und $A \in \mathbb{C}^{n \times n}$ selbstadjungiert und positiv definit. Setze $x^* := A^{-1}b$. Dann ist

$$\|\tilde{x} - x^*\|_A = \min_{x \in x^0 + \mathcal{K}} \|x - x^*\|_A$$

genau dann, wenn

$$\begin{cases} \tilde{x} \in x^0 + \mathcal{K} \\ b - A\tilde{x} \perp \mathcal{K} \end{cases}$$

gilt.

Beweis. Wir gehen wie im Beweis von Satz 2.5 vor. Für alle $v \in \mathcal{K}$ ist

$$\langle \tilde{x} - x^*, v \rangle_A = \langle \tilde{x} - x^*, Av \rangle = \langle A(\tilde{x} - x^*), v \rangle = \langle A\tilde{x} - b, v \rangle = 0.$$

Für beliebiges $x \in x^0 + \mathcal{K}$ ist dann $x - \tilde{x} \in \mathcal{K}$ und es gilt

$$\begin{aligned} \|x - x^*\|_A^2 &= \|x - \tilde{x} + \tilde{x} - x^*\|_A^2 = \|x - \tilde{x}\|_A^2 + 2\langle x - \tilde{x}, \tilde{x} - x^* \rangle_A + \|\tilde{x} - x^*\|_A^2 \\ &\geq \|\tilde{x} - x^*\|_A^2, \end{aligned}$$

weshalb das Minimum für $x = \tilde{x} \in x^0 + \mathcal{K}$ angenommen wird. □

Für die Konvergenzanalyse von Projektionsverfahren nutzt man aus, dass auch der Fehler $\tilde{e} := x^* - \tilde{x}$ durch eine Projektion charakterisiert werden kann. Definieren wir den Startfehler $e^0 := x^* - x^0$, erhalten wir die Relation

$$A\tilde{e} = Ax^* - A\tilde{x} = Ax^* - A(x^0 + \delta x) = A(e^0 - \delta x).$$

Da $Ax^* = b$ gilt, ist die Galerkin-Bedingung $b - A\tilde{x} \perp \mathcal{K}$ äquivalent zu

$$0 = \langle b - A\tilde{x}, v \rangle = \langle A(e^0 - \delta x), v \rangle = \langle e^0 - \delta x, v \rangle_A$$

für alle $v \in \mathcal{K}$. Also ist $\delta x \in \mathcal{K}$ die A -orthogonale Projektion von e^0 auf \mathcal{K} , die wir mit $P_A : \mathbb{C}^n \rightarrow \mathcal{K}$ bezeichnen. Aus

$$e^0 = x^* - x^0 - \delta x + \delta x = \tilde{e} + \delta x$$

folgt dann, dass \tilde{e} im A -orthogonalen Komplement von \mathcal{K} liegt. Es gilt also

$$\begin{cases} \delta x = P_A e^0, \\ \tilde{e} = (I - P_A) e^0. \end{cases}$$

Solche Verfahren nennt man daher auch *Fehlerprojektionsverfahren*. Da auch $I - P_A$ eine A -orthogonale Projektion ist und damit Norm gleich 1 hat, gilt $\|\tilde{e}\|_A \leq \|e^0\|_A$. Schärfere Abschätzungen und insbesondere Konvergenzresultate erhält man erst, wenn man die spezielle Wahl von \mathcal{K} berücksichtigt.

PETROV-GALERKIN-VERFAHREN: $\mathcal{L} = A\mathcal{K}$

Seien wieder die Spalten von $V = [v_1, \dots, v_m]$ eine Basis von \mathcal{K} und die Spalten von $W = [w_1, \dots, w_m]$ eine Basis von $A\mathcal{K}$. In diesem Fall können wir die Invertierbarkeit von W^*AV in Algorithmus 5.1 für jede nichtsinguläre Matrix A garantieren. Jede Basis von $A\mathcal{K}$ hat die Form AVX für eine invertierbare Matrix $X \in \mathbb{C}^{m \times m}$. Damit ist

$$W^*AV = X^*V^*A^*AV = X^*(AV)^*AV$$

invertierbar, da $(AV)^*AV$ nicht singulär ist (sonst wäre $\|AV\xi\|_2 = 0$ für ein $\xi \neq 0$ – ein Widerspruch zur Invertierbarkeit von A und linearen Unabhängigkeit der Spalten von V).

Da $\mathcal{L} \neq \mathcal{K}$ gilt, können wir in diesem Fall die Eigenschaften orthogonaler Projektionen nicht verwenden. Die spezielle Wahl von \mathcal{L} gestattet dennoch, ähnliche Optimalitätsresultate wie im orthogonalen Fall zu beweisen.

Satz 5.2. Sei \mathcal{K} ein Unterraum von \mathbb{C}^n , $x^0, b \in \mathbb{C}^n$ beliebig und $A \in \mathbb{C}^{n \times n}$. Dann ist

$$\|b - A\tilde{x}\|_2 = \min_{x \in x^0 + \mathcal{K}} \|b - Ax\|_2$$

genau dann, wenn

$$\begin{cases} \tilde{x} \in x^0 + \mathcal{K} \\ \mathbf{b} - A\tilde{x} \perp A\mathcal{K} \end{cases}$$

gilt.

Beweis. Setze $\tilde{y} := A\tilde{x}$. Dann ist $\tilde{y} \in Ax^0 + A\mathcal{K}$ und $\mathbf{b} - \tilde{y} \perp A\mathcal{K}$. Also ist $\tilde{y} - Ax^0$ die orthogonale Projektion von $\mathbf{b} - Ax^0$ auf $A\mathcal{K}$, und wie im Beweis von Satz 5.1 zeigt man nun, dass \tilde{y} das Minimierungsproblem $\min_{y \in Ax^0 + A\mathcal{K}} \|\mathbf{b} - y\|_2$ löst. \square

Dieses Projektionsverfahren minimiert also das Residuum über alle Vektoren in \mathcal{K} . Setzen wir $r^0 := \mathbf{b} - Ax^0$ und $\tilde{r} := \mathbf{b} - A\tilde{x}$, so erhalten wir analog zum Ritz-Galerkin-Verfahren

$$\tilde{r} = \mathbf{b} - A(x^0 + \delta x) = r^0 - A\delta x.$$

Aus der Bedingung $\tilde{r} \perp A\mathcal{K}$ folgt, dass $A\delta x \in A\mathcal{K}$ die orthogonale Projektion von r^0 auf $A\mathcal{K}$ ist, und \tilde{r} im orthogonalen Komplement von $A\mathcal{K}$ liegt. Bezeichnet $P : \mathbb{C}^n \rightarrow A\mathcal{K}$ die zugehörige Abbildung, dann gilt

$$\begin{cases} A\delta x = Pr^0, \\ \tilde{r} = (I - P)r^0, \end{cases}$$

und damit $\|\tilde{r}\|_2 \leq \|r^0\|_2$. Solche Verfahren nennt man auch *Residuumsprojektionsverfahren*, und die Fehleranalyse basiert dementsprechend auf der Betrachtung des Residuums.

5.2 EINDIMENSIONALE PROJEKTIONSVERFAHREN

Wir betrachten nun die einfachsten Beispiele für die oben untersuchten Klassen, in denen die Unterräume jeweils eindimensional sind, d. h.

$$\mathcal{K} = \text{span}\{v\}, \quad \mathcal{L} = \text{span}\{w\}$$

für $v, w \in \mathbb{C}^n \setminus \{0\}$. In diesem Fall vereinfacht sich Schritt 5 im Projektionsverfahren 5.1 zu

$$\xi = (w^*Av)^{-1}w^*r^k = \frac{\langle r^k, w \rangle}{\langle Av, w \rangle} =: \alpha_k.$$

(Zum Beispiel lässt sich das Gauss-Seidel-Verfahren als Projektionsverfahren interpretieren, wobei der Schritt 3 in der innersten Schleife einem Projektionsschritt für $\mathcal{K} = \mathcal{L} = \text{span}\{e_i\}$ entspricht.)

GRADIENTENVERFAHREN: $\mathcal{L} = \mathcal{K}$

Wir nehmen an, dass die Matrix $A \in \mathbb{R}^{n \times n}$ symmetrisch und positiv definit ist. Aus Satz 5.1 wissen wir, dass ein Schritt im Projektionsverfahren den Fehler $\|x - x^*\|_A$ entlang des Vektors v minimiert. Der größte Abstieg in der Norm geschieht dabei entlang des negativen Gradientens der Funktion $f(x) = \|x - x^*\|_A^2 = \langle x - x^*, A(x - x^*) \rangle$, also

$$-\nabla f(x) = -2(Ax - Ax^*) = 2(b - Ax) = 2r.$$

Im Schritt k wählen wir daher $\mathcal{K} = \mathcal{L} = \text{span}\{r^k\}$ und erhalten

- 1: $r^k = b - Ax^k$
- 2: $\alpha_k = \langle r^k, r^k \rangle / \langle r^k, Ar^k \rangle$
- 3: $x^{k+1} = x^k + \alpha_k r^k$

Da A symmetrisch und positiv definit ist, bricht das Verfahren nur ab, wenn $r^k = 0$ ist. In diesem Fall ist aber x^k bereits die gesuchte Lösung (ein sogenannter „lucky breakdown“). Eine weitere nützliche Beobachtung ist, dass das Residuum r^{k+1} nach Konstruktion orthogonal zu $\mathcal{K}_k = \text{span}\{r^k\}$ ist.

Man kann eine Matrix-Vektor-Multiplikation sparen, indem man die Berechnungen etwas rearrangiert. Multipliziert man Schritt 3 mit $-A$ und addiert b , erhält man

$$(5.1) \quad r^{k+1} = b - Ax^{k+1} = b - Ax^k - \alpha_k Ar^k = r^k - \alpha_k Ar^k.$$

Führt man den Hilfsvektor $p^k := Ar^k$ ein, erhält man das *Gradientenverfahren*.²

Algorithmus 5.2 Gradientenverfahren

Input: $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$, $x^0 \in \mathbb{R}^n$, $\varepsilon > 0$, $k = 0$

- 1: $r^0 = b - Ax^0$, $p^0 = Ar^0$
- 2: **while** $\|r^k\| > \varepsilon$ **do**
- 3: $\alpha_k = \langle r^k, r^k \rangle / \langle r^k, p^k \rangle$
- 4: $x^{k+1} = x^k + \alpha_k r^k$
- 5: $r^{k+1} = r^k - \alpha_k p^k$
- 6: $p^{k+1} = Ar^{k+1}$
- 7: $k \leftarrow k + 1$

Output: x^k

Für den Beweis der Konvergenz benötigen wir das folgende Resultat, das eine Art Cauchy-Schwarz-Ungleichung für Eigenwerte darstellt.

²von Augustin-Louis Cauchy 1847 publiziert

Lemma 5.3 (Kantorovich³-Ungleichung). Sei $A \in \mathbb{R}^{n \times n}$ symmetrisch und positiv definit mit Eigenwerten

$$\lambda_1 \geq \dots \geq \lambda_n > 0,$$

und sei $\kappa = \lambda_1/\lambda_n$ die Konditionszahl von A (bezüglich der Euklidischen Norm). Dann gilt für alle $x \in \mathbb{R}^n \setminus \{0\}$

$$\frac{\langle x, Ax \rangle \langle x, A^{-1}x \rangle}{\langle x, x \rangle^2} \leq \frac{1}{4} \left(\kappa^{\frac{1}{2}} + \kappa^{-\frac{1}{2}} \right)^2.$$

Beweis. Sei $\mu = (\lambda_n \lambda_1)^{\frac{1}{2}}$ das geometrische Mittel der beiden Eigenwerte und $B = \mu^{-1}A + \mu A^{-1}$. Dann ist B symmetrisch und positiv definit, und besitzt die Eigenwerte $\mu^{-1}\lambda_i + \mu\lambda_i^{-1}$ für $1 \leq i \leq n$. Da die Funktion $z \mapsto z + z^{-1}$ auf $(0, 1)$ und $(1, \infty)$ monoton und $\mu^{-1}\lambda_1 \leq \mu^{-1}\lambda_1 = \kappa^{1/2}$ ist, erhalten wir

$$\mu^{-1}\lambda_i + \mu\lambda_i^{-1} \leq \kappa^{\frac{1}{2}} + \kappa^{-\frac{1}{2}}, \quad 1 \leq i \leq n.$$

Mit dieser Abschätzung der Eigenwerte von B und dem Satz von [Courant–Fischer](#) erhalten wir daher

$$\mu^{-1}\langle x, Ax \rangle + \mu\langle x, A^{-1}x \rangle = \langle x, Bx \rangle \leq (\kappa^{\frac{1}{2}} + \kappa^{-\frac{1}{2}})\langle x, x \rangle.$$

Wir wenden jetzt auf die linke Seite die *Youngsche Ungleichung*

$$(ab)^{\frac{1}{2}} \leq \frac{1}{2}(\mu^{-1}a + \mu b)$$

für $a = \langle x, Ax \rangle$ und $b = \langle x, A^{-1}x \rangle$ an, quadrieren beide Seiten, und erhalten die gewünschte Ungleichung. \square

Damit können wir nun eine Fehlerabschätzung für das Gradientenverfahren zeigen (natürlich in der Energienorm, da es sich um ein orthogonales Projektionsverfahren handelt).

Satz 5.4. Sei $A \in \mathbb{R}^{n \times n}$ symmetrisch und positiv definit und sei $x^* = A^{-1}b$. Dann gilt für die Näherungen x^k im Gradientenverfahren folgende Fehlerabschätzung:

$$\|x^* - x^k\|_A \leq \left(\frac{\kappa - 1}{\kappa + 1} \right)^k \|x^* - x^0\|_A.$$

³[Leonid Witaljewitsch Kantorowitsch](#), zwischen 1934 und 1960 Professor für Mathematik an der Universität Leningrad und später Forschungsdirektor am Moskauer Institut für Nationale Wirtschaftsplanung. Für seine Anwendung der Linearen Programmierung in der Wirtschaftsplanung teilte er sich 1975 den Wirtschaftsnobelpreis. Außerdem leistete er viele Beiträge in Funktionalanalysis und Approximationstheorie. Der hier geführte Beweis geht auf [Dietrich Braess](#) zurück.

Beweis. Ist $r^k = 0$, so ist $Ax^k = b$ und damit $x^k - x^* = 0$, und die Abschätzung ist trivialerweise erfüllt. Sei nun $r^k \neq 0$. Wir betrachten den Fehler

$$e^{k+1} = x^* - x^{k+1} = x^* - x^k - \alpha_k r^k = e^k - \alpha_k r^k,$$

der die Gleichung

$$Ae^{k+1} = Ax^* - Ax^{k+1} = r^{k+1}$$

erfüllt. Verwenden wir die Rekursion für r^{k+1} und die Orthogonalität der Residuen r^k und r^{k+1} , erhalten wir

$$\begin{aligned} \|e^{k+1}\|_A^2 &= \langle e^{k+1}, Ae^{k+1} \rangle \\ &= \langle e^{k+1}, r^{k+1} \rangle \\ &= \langle e^k - \alpha_k r^k, r^{k+1} \rangle \\ &= \langle e^k, r^k - \alpha_k Ar^k \rangle \\ &= \langle A^{-1}r^k, r^k \rangle - \alpha_k \langle A^{-1}r^k, Ar^k \rangle \\ &= \langle A^{-1}r^k, r^k \rangle \left(1 - \alpha_k \frac{\langle r^k, r^k \rangle}{\langle A^{-1}r^k, r^k \rangle} \right). \end{aligned}$$

Einsetzen von $r^k = Ae^k$ und der Definition von α_k ergibt

$$\|e^{k+1}\|_A^2 = \|e^k\|_A^2 \left(1 - \frac{\langle r^k, r^k \rangle}{\langle r^k, Ar^k \rangle} \frac{\langle r^k, r^k \rangle}{\langle r^k, A^{-1}r^k \rangle} \right).$$

Auf den Bruch wenden wir nun die Kantorovich-Ungleichung an und bringen die Klammer auf einen gemeinsamen Nenner. Durch Induktion erhalten wir daraus die Aussage. \square

Da $\kappa \geq 1$ gilt, folgt daraus die Konvergenz des Gradientenverfahrens für symmetrisch positiv definite Matrizen. Im Gegensatz zu den stationären Verfahren hängt hier die Konvergenzgeschwindigkeit aber nicht von der Größe, sondern vom *Verhältnis* der Eigenwerte ab: Je näher die Eigenwerte beieinander liegen, desto schneller konvergiert das Gradientenverfahren.

MR-ITERATION: $\mathcal{L} = A\mathcal{K}$

Sei nun $A \in \mathbb{R}^{n \times n}$ regulär. Wir wählen wieder $\mathcal{K} = \text{span}\{r^k\}$ mit $r^k = b - Ax^k$, d. h. $\mathcal{L} = \text{span}\{Ar^k\}$. Das Projektionsverfahren 5.1 hat dann die Form

- 1: $r^k = b - Ax^k$
- 2: $\alpha_k = \langle r^k, Ar^k \rangle / \langle Ar^k, Ar^k \rangle$
- 3: $x^{k+1} = x^k + \alpha_k r^k$

Wie wir bereits gesehen haben, minimiert jeder Schritt die Norm des Residuums $\|b - Ax\|_2$ in Richtung r^k . Auch hier bricht das Verfahren nur ab, wenn $r^k = 0$ ist, da A regulär ist.

Mit Hilfe der Rekursion (5.1) erhalten wir die *MR-Iteration* (nach „minimal residual“).

Algorithmus 5.3 MR-Iteration

Input: $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$, $x^0 \in \mathbb{R}^n$, $\varepsilon > 0$, $k = 0$

1: $r^0 = b - Ax^0$, $p^0 = Ar^0$

2: **while** $\|r^k\|_2 > \varepsilon$ **do**

3: $\alpha_k = \langle r^k, p^k \rangle / \langle p^k, p^k \rangle$

4: $x^{k+1} = x^k + \alpha_k r^k$

5: $r^{k+1} = r^k - \alpha_k p^k$

6: $p^{k+1} = Ar^{k+1}$

7: $k \leftarrow k + 1$

Output: x^k

Für positiv definite Matrizen können wir die Konvergenz ähnlich wie für das Gradientenverfahren beweisen. Im Gegensatz zu diesem wird hier jedoch keine Symmetrie benötigt, und die Konvergenz wird im Sinne des Residuums betrachtet.

Satz 5.5. Sei $A \in \mathbb{R}^{n \times n}$ positiv definit, und sei μ der kleinste Eigenwert von $\frac{1}{2}(A^T + A)$ sowie $\sigma = \|A\|$ für eine Matrixnorm die verträglich ist mit der Euklidischen Vektornorm. Dann gilt für die Residuen $r^k = b - Ax^k$ in der MR-Iteration folgende Fehlerabschätzung:

$$\|r^k\|_2 \leq \left(1 - \frac{\mu^2}{\sigma^2}\right)^{k/2} \|r^0\|_2.$$

Da jede Norm eine obere Schranke für den Spektralradius (d. h. den betragsgrößten Eigenwert) ist, ist der Faktor vor der Norm echt kleiner als 1, und es folgt auch hier die Konvergenz des Verfahrens aus dieser Abschätzung.

Beweis. Angenommen, $r^k \neq 0$. Nach Konstruktion ist das Residuum

$$r^{k+1} = b - Ax^{k+1} = r^k - \alpha_k Ar^k.$$

orthogonal zu Ar^k . Wir erhalten also analog zum Beweis von Satz 5.4, dass

$$\begin{aligned} \|r^{k+1}\|_2^2 &= \langle r^{k+1}, r^k - \alpha_k Ar^k \rangle \\ &= \langle r^{k+1}, r^k \rangle \\ &= \langle r^k, r^k \rangle - \alpha_k \langle r^k, Ar^k \rangle \\ &= \|r^k\|_2^2 \left(1 - \frac{\langle r^k, Ar^k \rangle}{\langle r^k, r^k \rangle} \frac{\langle r^k, Ar^k \rangle}{\langle Ar^k, Ar^k \rangle}\right) \\ &= \|r^k\|_2^2 \left(1 - \frac{\langle r^k, Ar^k \rangle^2}{\langle r^k, r^k \rangle^2} \frac{\|r^k\|_2^2}{\|Ar^k\|_2^2}\right). \end{aligned}$$

Da A positiv definit ist, erhalten wir wie in Satz 3.4, dass

$$\frac{\langle r^k, Ar^k \rangle}{\langle r^k, r^k \rangle} \geq \mu$$

gilt. Zusammen mit der Verträglichkeit $\|Ar^k\|_2 \leq \|A\|_2 \|r^k\|_2$ folgt durch Induktion die gewünschte Abschätzung. \square

KRYLOVRAUM-VERFAHREN

6

Das Gradientenverfahren und die MR-Iteration sind die einfachsten Beispiele für Krylovraum-Verfahren. Im Allgemeinen sind \mathcal{K} und \mathcal{L} dabei nicht eindimensional, sondern werden als Folge verschachtelter Unterräume gewählt.

Für eine gegebene Matrix $A \in \mathbb{C}^{n \times n}$ und einen Vektor $v \in \mathbb{C}^n \setminus \{0\}$ ist die Folge der zugehörigen *Krylovräume*¹ definiert als

$$\mathcal{K}_m(A, v) = \text{span}\{v, Av, A^2v, \dots, A^{m-1}v\} \quad \text{für } m = 1, \dots$$

Offensichtlich ist $\mathcal{K}_m(A, v) \subseteq \mathcal{K}_{m+1}(A, v)$ für alle m . Gilt $\mathcal{K}_m(A, v) = \mathcal{K}_{m+1}(A, v)$, so ist $\mathcal{K}_m(A, v)$ invariant unter A , d. h. es gilt

$$A\mathcal{K}_m(A, v) \subseteq \mathcal{K}_m(A, v).$$

In diesem Fall ist $\mathcal{K}_m(A, v)$ ein *Eigenraum* von A der Dimension m . Umgekehrt ist für jeden Eigenvektor v von A die Dimension von $\mathcal{K}_m(A, v)$ gleich 1. Invariante Unterräume sind von besonderer Bedeutung für Projektionsverfahren, wie der folgende Satz zeigt.

Satz 6.1. *Seien $\mathcal{K}, \mathcal{L} \subset \mathbb{C}^n$ und $x^0 \in \mathbb{C}^n$ beliebig. Ist \mathcal{K} ein invarianter Unterraum unter A und gilt $b - Ax^0 \in \mathcal{K}$, so konstruiert das Projektionsverfahren auf $x^0 + \mathcal{K}$ senkrecht zu \mathcal{L} ein $\tilde{x} \in x^0 + \mathcal{K}$ mit $A\tilde{x} = b$.*

Krylovräume stehen in enger Verbindung zur Approximation mit Polynomen, da jeder Vektor $x \in \mathcal{K}_m(A, v)$ geschrieben werden kann als

$$x = \sum_{i=0}^{m-1} \alpha_i A^i v = p_{m-1}(A)v,$$

wobei p_{m-1} ein Polynom vom Höchstgrad $m - 1$ ist. Existiert also solch ein Polynom $q \neq 0$ mit $q(A)v = 0$, so können die Vektoren $v, Av, \dots, A^{m-1}v$ nicht linear unabhängig sein.

Lemma 6.2. *Sei k der kleinste Grad aller Polynome p mit $p(A)v = 0$. Dann hat $\mathcal{K}_m(A, v)$ die Dimension k für alle $m \geq k$; insbesondere gilt in dem Fall $\mathcal{K}_m(A, v) = \mathcal{K}_k(A, v)$.*

¹Aleksei Nikolajewitsch Krylow arbeitete ab 1888 als Schiffsbauingenieur am Schiffbauinstitut der Marineakademie in Leningrad. Die nach ihm benannten Unterräume verwendete er in einer 1931 publizierten Arbeit über die numerische Eigenwertberechnung.

6.1 DER ARNOLDI-PROZESS

Um ein effizientes Projektionsverfahren für Krylovräume zu ermöglichen, benötigen wir eine orthonormale Basis von $\mathcal{K}_m(A, v)$. Im Prinzip könnten wir das Gram–Schmidt-Verfahren auf die Vektoren $v, Av, \dots, A^{m-1}v$ anwenden. Allerdings sind diese Vektoren sehr schlecht geeignet, da sie alle für wachsendes m gegen den Eigenvektor zum betragsgrößten Eigenwert konvergieren (siehe Vektoriteration). Im *Arnoldi-Prozess* wird daher in jedem Schritt erneut orthonormalisiert, bevor aus dem Ergebnis durch Multiplikation mit A der nächste Vektor erzeugt wird.

Algorithmus 6.1 Arnoldi-Prozess

Input: $v \in \mathbb{C}^n \setminus \{0\}$

- 1: $q_1 = v / \|v\|_2$
- 2: **for** $j = 1, \dots, m$ **do**
- 3: **for** $i = 1, \dots, j$ **do**
- 4: $h_{ij} = \langle Aq_j, q_i \rangle$
- 5: $w_j = Aq_j - \sum_{i=1}^j h_{ij} q_i$
- 6: $h_{j+1,j} = \|w_j\|_2$
- 7: **if** $h_{j+1,j} = 0$ **then**
- 8: **stop**
- 9: **else**
- 10: $q_{j+1} = w_j / h_{j+1,j}$

Output: $q_1, \dots, q_m, h_{ij}, 1 \leq i \leq j+1 \leq m$

Ob dieser Algorithmus vorzeitig abbricht, hängt von A und v ab. In den meisten Fällen ist ein vorzeitiger Abbruch sogar von Vorteil, da in diesem Fall bereits die exakte Lösung des linearen Gleichungssystems oder Eigenwertproblems erreicht wird („lucky breakdown“).

Satz 6.3. *Wenn der Arnoldi-Prozess im Schritt j abbricht, so ist $\mathcal{K}_j(A, v)$ invariant unter A . Ansonsten bilden die Vektoren q_1, \dots, q_m eine orthonormale Basis von $\mathcal{K}_m(A, v)$.*

Beweis. Wir verwenden Induktion nach j . Angenommen, der Prozess ist nicht vor Schritt j abgebrochen und die Aussage gilt für alle $i \leq j$. Aus Schritt 5 und 10 erhalten wir dann mit $q_i \in \mathcal{K}_i(A, v)$, dass

$$w_j = Aq_j - \sum_{i=1}^j h_{ij} q_i = Ap_{j-1}(A)v - \sum_{i=1}^j h_{ij} p_{i-1}(A)v =: p_j(A)v \in \mathcal{K}_{j+1}(A, v)$$

für ein Polynom p_j vom Höchstgrad j ist.

Der Arnoldi-Prozess bricht nur ab, wenn $w_j = 0$ ist, was bedeutet, dass $Aq_j \in \mathcal{K}_j(A, v)$ ist. In diesem Fall ist $\mathcal{K}_j(A, v)$ invariant unter A , da $Aq_i \in \mathcal{K}_j(A, v)$ für alle $i \leq j$. Ist $w_j \neq 0$,

so folgt (ebenso mit Induktion), dass $q_{j+1} = p_j(A)v \in \mathcal{K}_{j+1}(A, v)$ für ein Polynom vom Grad (nicht nur Höchstgrad!) j ist, d. h. $q_{j+1} \in \mathcal{K}_{j+1}(A, v)$ und $q_{j+1} \notin \mathcal{K}_i(A, v)$ für $i \leq j$. Also spannen q_1, \dots, q_{j+1} den Krylovraum $\mathcal{K}_{j+1}(A, v)$ auf und sind nach Konstruktion orthonormal. \square

Aus Schritt 5 und 10 können wir analog zum Gram–Schmidt-Verfahren eine Matrix-Zerlegung aus dem Arnoldi-Prozess erhalten. Für alle $1 \leq j < m$ ist

$$(6.1) \quad Aq_j = \sum_{i=1}^{j+1} h_{ij} q_i.$$

Definieren wir die Matrix $Q_m = [q_1, \dots, q_m] \in \mathbb{C}^{n \times m}$ sowie die Matrix $H_m \in \mathbb{C}^{m \times m}$ mit den Einträgen h_{ij} für $i \leq j + 1$ und 0 sonst, so ist (wenn der Prozess nicht abbricht) Q_m unitär und H_m eine echte obere Hessenbergmatrix (d. h. $h_{i+1,i} \neq 0$ für alle $1 \leq i \leq m$). Weiterhin gilt

$$(6.2) \quad AQ_m = Q_m H_m + w_m e_m^*.$$

Da Q_m unitär ist und w_m orthogonal zu allen q_i mit $i \leq m$ ist, erhalten wir

$$(6.3) \quad Q_m^* A Q_m = H_m.$$

Der Arnoldi-Prozess berechnet also die ersten m Spalten einer unitären Ähnlichkeitstransformation auf obere Hessenbergform. Diese Argumentation kann auch umgekehrt werden: Haben wir eine Transformation der Form (6.3), so können wir durch spaltenweise Betrachtung mit Hilfe von (6.1) sehen, dass bei gegebenen q_1, \dots, q_j und h_{ij} , $1 \leq i \leq j$, sowohl q_{j+1} als auch $h_{j+1,j}$ (bis auf ihr Vorzeichen) eindeutig durch die Forderung $\|q_{j+1}\|_2 = 1$ festgelegt sind. Diese Erkenntnis ist von großer Bedeutung für die effiziente Durchführung von QR-Verfahren für Eigenwertprobleme, und wird oft als „*implicit Q theorem*“ bezeichnet:

Satz 6.4. *Sei $m \leq n$, und seien $Q, Q' \in \mathbb{C}^{n \times m}$ unitär. Sind $H = Q^* A Q$ und $H' = (Q')^* A Q'$ echte obere Hessenbergmatrizen, und ist $q_1 = q'_1$, so gilt $q_i = c_i q'_i$ mit $|c_i| = 1$ und $|h_{i,i-1}| = |h'_{i,i-1}|$ für alle $2 \leq i \leq m$.*

Dieser Satz sagt im wesentlichen aus, dass die erste Spalte von Q_m zusammen mit der Hessenbergform von H_m sowohl Q_m als auch H_m (bis auf Ähnlichkeitstransformation mit einer Diagonalmatrix mit Einträgen ± 1) eindeutig festlegt.

Wir können den Arnoldi-Prozess auch als eine Projektion von A auf den Krylovraum $\mathcal{K}_m(A, v)$ auffassen. Wir betrachten die lineare Abbildung

$$A_m : \mathcal{K}_m(A, v) \rightarrow \mathcal{K}_m(A, v), \quad v \mapsto P_{\mathcal{K}_m}(Av),$$

wobei $P_{\mathcal{K}_m}$ die orthogonale Projektion auf $\mathcal{K}_m(A, v)$ bezeichnet. Da die Spalten von Q_m eine orthonormale Basis von $\mathcal{K}_m(A, v)$ bilden, hat die Projektion auf \mathcal{K}_m nach (2.5) die

Matrixdarstellung $P_{\mathcal{K}_m} x = Q_m Q_m^* x$ (bezüglich der Standardbasis). Da x und $A_m x$ in \mathcal{K}_m liegen, existieren $\xi, \eta \in \mathbb{C}^m$ mit $x = Q_m \xi$ und $A_m x = Q_m \eta$. Einsetzen in die Definition und Verwenden von $Q_m^* Q_m = I$ ergibt dann

$$\begin{aligned} \eta &= Q_m^* (A_m x) = Q_m^* (P_{\mathcal{K}_m} A Q_m \xi) = (Q_m^* A Q_m) \xi \\ &= H_m \xi, \end{aligned}$$

d. h. H_m ist die Darstellung von A_m bezüglich der Basis q_1, \dots, q_m .

Wie beim Gram–Schmidt-Verfahren existieren numerisch stabilere Varianten basierend auf sukzessiver Projektion (*modifizierter Arnoldi-Prozess*, vergleiche Algorithmus 2.2) oder Householder-Reflexionen.

6.2 DER LANCZOS-PROZESS

Ist die Matrix $A \in \mathbb{C}^{n \times n}$ selbstadjungiert, so gilt dies auch für $H_m = Q_m^* A Q_m$. Da $h_{ij} = 0$ ist für $i \leq j + 1$, muss auch $h_{ji} = 0$ sein. Das Arnoldi-Verfahren konstruiert in diesem Fall also eine (wegen Schritt 6 reelle) *Tridiagonalmatrix*

$$Q_m^* A Q_m = T_m = \begin{pmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \ddots & \ddots & \\ & & \ddots & \alpha_{m-1} & \beta_m \\ & & & \beta_m & \alpha_m \end{pmatrix}.$$

Wir müssen in jedem Schritt also nur gegen die jeweils letzten beiden Vektoren orthogonalisieren. Nutzen wir dazu die Symmetrie von $h_{j,j-1} = h_{j-1,j} = \beta_j$, so kann der Arnoldi-Prozess stark vereinfacht werden, und wir erhalten den *Lanczos-Prozess*². Die in Algorithmus 6.2 angegebene Form basiert auf dem modifizierten Gram–Schmidt-Verfahren.

In Krylovraum-Verfahren wird das lineare Gleichungssystem oder das Eigenwertproblem statt für A für die (üblicherweise sehr viel kleinere) Matrix H_m beziehungsweise T_m gelöst, wobei die Hessenberg- beziehungsweise Tridiagonalstruktur ausgenutzt werden kann. Als Prototyp kann das folgende Projektionsverfahren dienen.

6.3 ARNOLDI-VERFAHREN FÜR GLEICHUNGSSYSTEME

Für $A \in \mathbb{C}^{n \times n}$, $b \in \mathbb{C}^n$ und $x^0 \in \mathbb{C}^n$ betrachten wir ein orthogonales Projektionsverfahren mit $\mathcal{K} = \mathcal{L} = \mathcal{K}_m(A, r^0)$, wobei $r^0 = b - Ax^0$ ist. Wir wenden nun den Arnoldi-Prozess

²Nach [Cornelius Lanczos](#), einem ungarischen Mathematiker und theoretischem Physiker. Er arbeitete Ende der 1920er Jahre mit Albert Einstein zusammen, und publizierte 1940 eine Variante der schnellen Fouriertransformation (FFT) in Matrixform.

Algorithmus 6.2 Lanczos-Prozess**Input:** $v \in \mathbb{C}^n \setminus \{0\}$,1: $q_1 = v/\|v\|_2, q_0 = 0, \beta_1 = 0$ 2: **for** $j = 1, \dots, m$ **do**3: $w_j = Aq_j - \beta_j q_{j-1}$ 4: $\alpha_j = \langle w_j, q_j \rangle$ 5: $w_j \leftarrow w_j - \alpha_j q_j$ 6: $\beta_j = \|w_j\|_2$ 7: **if** $\beta_j = 0$ **then**

8: stop

9: **else**10: $q_{j+1} = w_j/\beta_j$ **Output:** $q_1, \dots, q_m, \alpha_1, \dots, \alpha_m, \beta_2, \dots, \beta_m$

6.1 auf den Vektor $q_1 = r^0/\beta$ mit $\beta = \|r^0\|_2$ an und erhalten die Matrix $Q_m \in \mathbb{C}^{n \times m}$ der Basisvektoren von $\mathcal{K}_m(A, r^0)$ mit

$$Q_m^* A Q_m = H_m$$

sowie

$$Q_m^* r^0 = Q_m^* (\beta q_1) = \beta Q_m^* q_1 = \beta e_1.$$

(Bricht der Arnoldi-Prozess im Schritt j vorzeitig ab, führen wir diese und folgende Schritte mit $\tilde{m} := j$ durch.) Die Matrixdarstellung der Galerkinbedingungen $x^k \in x^0 + \mathcal{K}_m(A, r^0)$ und $b - Ax^k \perp \mathcal{K}_m(A, r^0)$ hat dann die Form

$$\begin{aligned} x^k &= x^0 + Q_m (Q_m^* A Q_m)^{-1} Q_m^* r^0 \\ &= x^0 + Q_m H_m^{-1} (\beta e_1). \end{aligned}$$

Da die Matrix H_m obere Hessenbergform hat, kann die Lösung von $H_m \xi^k = \beta e_1$ sehr leicht berechnet werden, etwa durch Transformation auf obere Dreiecksform mit Givens-Rotationen und anschließende Rückwärtssubstitution.

Ist das Residuum r^k klein genug, so akzeptiert man die Lösung. Ansonsten hat man zwei Möglichkeiten:

1. Wiederhole das Verfahren mit neuem Startwert x^k und neuem Krylovraum $\mathcal{K}_{\tilde{m}}(A, r^k)$ (wobei $\tilde{m} = m$ sein kann, aber nicht muss).
2. Vergrößere den Krylovraum $\mathcal{K}_{m+1}(A, r^0)$ und berechne $x^{k+1} \in x^0 + \mathcal{K}_{m+1}(A, r^0)$.

Da die Krylovräume verschachtelt sind, kann man im zweiten Fall die Informationen aus dem vorigen Schritt wiederverwenden. Für symmetrisch und positiv definite Matrizen führt dies auf das CG-Verfahren.

DAS CG-VERFAHREN

7

Das *CG-Verfahren* oder *Verfahren der konjugierten Gradienten* (Englisch „conjugate gradient“) ist ein orthogonales Projektionsverfahren für selbstadjungierte positiv definite Matrizen, das im Schritt k den Krylovraum $\mathcal{K}_k := \mathcal{K}_k(A, r^0)$ für $r^0 = b - Ax^0$ verwendet. Setzt man analog zum Arnoldi-Verfahren in Abschnitt 6.3 den Lanczos-Prozess ein, führt das auf den Schritt

$$x^k = x^0 + Q_k T_k^{-1} Q_k^* r^0$$

mit einer unitären Matrix Q_k , deren Spalten eine Orthonormalbasis von \mathcal{K}_k bilden, sowie einer selbstadjungierten positiv definiten Tridiagonalmatrix T_k (da T_k durch eine Ähnlichkeitstransformation von A entstanden ist und daher die gleichen Eigenwerte hat). Wir können also eine Cholesky-Zerlegung $T_k = L_k D_k L_k^*$ für die Lösung des Gleichungssystems $T_k^{-1}(Q_k^* r^0)$ verwenden. Setzen wir $P_k := Q_k(L_k^{-1})^*$ und $z_k = P_k^* r^0$; dann ist $x^k = x^0 + P_k D_k^{-1} z_k$. Weiterhin spannen die Spalten von P_k und Q_k den gleichen Raum \mathcal{K}_k auf (da $L_k \in \mathbb{C}^{k \times k}$ vollen Rang hat) und es gilt

$$P_k^* A P_k = L_k^{-1} Q_k^* A Q_k (L_k^*)^{-1} = L_k^{-1} T_k (L_k^*)^{-1} = D_k,$$

d. h. die Spalten von P_k sind A -orthogonal (oder A -konjugiert, daher der Name des Verfahrens). Dass diese Methode zu den **berühmtesten Algorithmen** gehört, liegt daran, dass diese Zerlegung nicht in jedem Schritt neu berechnet werden muss, sondern jeweils nur die letzte Spalte von P_k und der letzte Eintrag von z_k .

7.1 ALGORITHMUS

Die einfachste Herleitung berechnet die Spalten von P_k direkt aus der Bedingung, dass sie eine A -orthogonale Basis von \mathcal{K}_k bilden und verwendet die allgemeine Form des Projektionsverfahrens 5.1. Wir gehen dabei induktiv vor. Angenommen, wir haben für alle $j \leq k$ eine A -orthogonale Basis p_0, \dots, p_{j-1} von \mathcal{K}_j bestimmt sowie mit Hilfe dieser Basis den Projektionsschritt $x^{k-1} \in x^0 + \mathcal{K}_{k-1}$. Setzen wir $P_k = [p_0, \dots, p_{k-1}] \in \mathbb{C}^{n \times k}$, so ist die

Matrix $P_k^*AP_k$ in Schritt 5 von Algorithmus 5.1 eine Diagonalmatrix mit Einträgen $\langle p_j, Ap_j \rangle$, und daher gilt

$$\begin{aligned} x^k &= x^0 + P_k(P_k^*AP_k)^{-1}P_k^*r^0 = x^0 + \sum_{j=0}^{k-1} \frac{\langle p_j, r^0 \rangle}{\langle p_j, Ap_j \rangle} p_j \\ &= x^0 + \sum_{j=0}^{k-2} \frac{\langle p_j, r^0 \rangle}{\langle p_j, Ap_j \rangle} p_j + \frac{\langle p_{k-1}, r^0 \rangle}{\langle p_{k-1}, Ap_{k-1} \rangle} p_{k-1}. \end{aligned}$$

Nach Induktionsvoraussetzung stellen die ersten beiden Terme den Projektionsschritt für x^{k-1} dar. Setzen wir

$$\alpha_k = \frac{\langle p_{k-1}, r^0 \rangle}{\langle p_{k-1}, Ap_{k-1} \rangle},$$

erhalten wir also

$$(7.1) \quad x^k = x^{k-1} + \alpha_k p_{k-1}.$$

Wir müssen nun eine A -orthogonale Basis von \mathcal{K}_{k+1} berechnen. Dafür nutzen wir aus, dass das neue Residuum r^k nach Konstruktion orthogonal zu \mathcal{K}_k ist. Analog zum Gradientenverfahren können wir aus (7.1) die Rekursion

$$(7.2) \quad r^k = r^{k-1} - \alpha_k Ap_{k-1}$$

herleiten, woraus außerdem $r^k \in \text{span}\{r^{k-1}, Ap_{k-1}\} \subset \mathcal{K}_{k+1}$ folgt. Gilt $r^k \neq 0$ (sonst wären wir fertig), dann ist also $r^k \in \mathcal{K}_{k+1} \setminus \mathcal{K}_k$ und wegen $\dim \mathcal{K}_{k+1} = \dim \mathcal{K}_k + 1$ ist daher $\{p_0, \dots, p_{k-1}, r^k\}$ eine Basis von \mathcal{K}_{k+1} . Wir erhalten dann eine A -orthogonale Basis, indem wir r^k gegen alle p_j , $j < k$, mit Hilfe des Gram-Schmidt-Verfahrens A -orthogonalisieren. Dies führt auf

$$p_k = r^k - \sum_{j=0}^{k-1} \frac{\langle r^k, p_j \rangle_A}{\langle p_j, p_j \rangle_A} p_j.$$

Wie im Lanczos-Prozess muss auch hier nicht die komplette Summe berechnet werden. Für $j \leq k-2$ folgt aus (7.2), dass

$$\langle r^k, p_j \rangle_A = \langle r^k, Ap_j \rangle = \alpha_{j+1}^{-1} (\langle r^k, r^j \rangle - \langle r^k, r^{j+1} \rangle) = 0,$$

da r^k orthogonal zu $r^j \in \mathcal{K}_{j+1} \subset \mathcal{K}_k$ und $r^{j+1} \in \mathcal{K}_{j+2} \subset \mathcal{K}_k$ ist. (Dass $\alpha_{j+1} \neq 0$ ist, sieht man anhand der alternativen Darstellung (7.5).) Mit der Definition

$$(7.3) \quad \beta_k = -\frac{\langle r^k, Ap_{k-1} \rangle}{\langle p_{k-1}, Ap_{k-1} \rangle},$$

erhalten wir die Rekursion

$$(7.4) \quad p_k = r^k + \beta_k p_{k-1}.$$

Starten wir mit $p^0 = r^0$, so definiert dies einen Algorithmus, der nur im Fall $r^k = 0$ abbricht.

Üblicherweise werden für die Koeffizienten α_k und β_k etwas stabilere (in exakter Arithmetik äquivalente) Rekursionen verwendet. Aus der Orthogonalität von r^k und r^{k-1} und (7.2) folgt

$$0 = \langle r^{k-1}, r^k \rangle = \langle r^{k-1}, r^{k-1} \rangle - \alpha_k \langle r^{k-1}, Ap_{k-1} \rangle.$$

Da die Vektoren p_j A-orthogonal sind für $j \leq k$, ergibt (7.4), dass

$$\langle r^{k-1}, Ap_{k-1} \rangle = \langle p_{k-1}, Ap_{k-1} \rangle + \beta_{k-1} \langle p_{k-2}, Ap_{k-1} \rangle = \langle p_{k-1}, Ap_{k-1} \rangle$$

ist. Zusammen erhalten wir

$$(7.5) \quad \alpha_k = \frac{\langle r^{k-1}, r^{k-1} \rangle}{\langle p_{k-1}, Ap_{k-1} \rangle}.$$

Einsetzen von $Ap_{k-1} = \alpha_k^{-1}(r^{k-1} - r^k)$ (aus (7.2)) und (7.5) in (7.3) ergibt außerdem

$$(7.6) \quad \beta_k = \frac{\langle r^k, r^k \rangle}{\langle r^{k-1}, r^{k-1} \rangle}.$$

Die Gleichungen (7.5), (7.1), (7.2), (7.6) und (7.4) ergeben zusammen den berühmten CG-Algorithmus.¹

Algorithmus 7.1 CG-Verfahren

Input: $A, b, x^0, \varepsilon > 0$

1: $p^0 = r^0 = b - Ax^0, k = 0$

2: **while** $\|r^k\|_2 > \varepsilon$ **do**

3: $k \leftarrow k + 1$

4: $\alpha_k = \langle r^{k-1}, r^{k-1} \rangle / \langle p_{k-1}, Ap_{k-1} \rangle$

5: $x^k = x^{k-1} + \alpha_k p_{k-1}$

6: $r^k = r^{k-1} - \alpha_k Ap_{k-1}$

7: $\beta_k = \langle r^k, r^k \rangle / \langle r^{k-1}, r^{k-1} \rangle$

8: $p_k = r^k + \beta_k p_{k-1}$

Output: x^k

¹Durch Magnus Hestenes und Eduard Stiefel unabhängig voneinander entdeckt und gemeinsam publiziert in [Hestenes und Stiefel 1952].

7.2 KONVERGENZ

Da das CG-Verfahren ein Ritz–Galerkin-Verfahren ist, folgt aus Satz 5.1, dass x^k den Fehler $e = x^* - x$ in der A -Norm über alle $x \in x^0 + \mathcal{K}_k$ minimiert. Außerdem ist \mathcal{K}_k der Spann von r^0, \dots, r^{k-1} . Gilt also $r^k \neq 0$ für alle $0 \leq k < n$, so ist $\dim(\mathcal{K}_n) = n$, und daher ist

$$\|x^* - x^n\|_A = \min_{x \in \mathbb{C}^n} \|x^* - x\|_A = 0.$$

Das CG-Verfahren konvergiert also (in exakter Arithmetik) in höchstens n Iterationen. (Die Konvergenz ist monoton, d. h. es gilt $\|e^{k+1}\|_A \leq \|e^k\|_A$, da $\mathcal{K}_k \subset \mathcal{K}_{k+1}$ ist.) Allerdings ist dies nicht sehr hilfreich, da n in der Praxis sehr groß sein kann.² Um bessere Fehlerabschätzungen zu bekommen, müssen wir die Struktur des Krylovraums $\mathcal{K}_k = \text{span}\{r^0, Ar^0, \dots, A^{k-1}r^0\}$ verwenden.

Für jeden Vektor $x \in x^0 + \mathcal{K}_k$ kann $x - x^0$ als eine Linearkombination der $A^j r^0$, $0 \leq j \leq k-1$, geschrieben werden. Weiterhin ist $r^0 = b - Ax^0 = A(x^* - x^0) = Ae^0$. Es gilt also

$$\begin{aligned} x^* - x &= x^* - \left(x^0 + \sum_{j=0}^{k-1} \gamma_j A^j r^0\right) = e^0 - \sum_{j=0}^{k-1} \gamma_j A^j (Ae^0) \\ &= \left(1 - \sum_{j=0}^{k-1} \gamma_j A^{j+1}\right) e^0 \\ &=: p_k(A) e^0 \end{aligned}$$

für ein Polynom p_k vom Höchstgrad k mit $p_k(0) = 1$. Da $x^k \in x^0 + \mathcal{K}_k$ im CG-Verfahren den Fehler $\|e^k\|_A$ minimiert, muss $e^k = x^* - x^k$ die Bedingung

$$(7.7) \quad \|e^k\|_A = \min_{p \in \mathcal{P}_k, p(0)=1} \|p(A)e^0\|_A$$

erfüllen, wobei \mathcal{P}_k die Menge aller Polynome mit Höchstgrad k bezeichnet.

Mit Hilfe der Eigenwerte von A bekommen wir daraus eine erste Fehlerabschätzung.

Lemma 7.1. *Sei $A \in \mathbb{C}^{n \times n}$ selbstadjungiert und positiv definit. Dann gilt für die Näherungen x^k im CG-Verfahren die Abschätzung*

$$\|e^k\|_A \leq \left(\min_{p \in \mathcal{P}_k, p(0)=1} \max_{\lambda \in \sigma(A)} |p(\lambda)| \right) \|e^0\|_A.$$

Beweis. Da A selbstadjungiert ist, existiert eine orthonormale Basis aus Eigenvektoren v_1, \dots, v_n von \mathbb{C}^n . Für $e^0 = \sum_{j=1}^n \gamma_j v_j$ ist also

$$\|e^0\|_A^2 = \langle e^0, Ae^0 \rangle = \left\langle \sum_{j=1}^n \gamma_j v_j, \sum_{j=1}^n \gamma_j \lambda_j v_j \right\rangle = \sum_{j=1}^n |\gamma_j|^2 \lambda_j.$$

²In der Tat wurde das CG-Verfahren anfangs als direktes Verfahren betrachtet, weshalb man rasch das Interesse an ihm verlor. Erst Jahrzehnte später kam es als iteratives Verfahren wieder in Gebrauch.

Genauso gilt für $p(A)e^0 = \sum_{j=1}^n \gamma_j p(\lambda_j) v_j$

$$\|p(A)e^0\|_A^2 = \sum_{j=1}^n |\gamma_j|^2 \lambda_j |p(\lambda_j)|^2 \leq \max_{\lambda \in \sigma(A)} |p(\lambda)|^2 \sum_{j=1}^n |\gamma_j|^2 \lambda_j = \max_{\lambda \in \sigma(A)} |p(\lambda)|^2 \|e^0\|_A^2.$$

Durch Einsetzen in (7.7) erhalten wir die gewünschte Abschätzung. \square

Folgerung 7.2. *Hat A nur k verschiedene Eigenwerte, so konvergiert das CG-Verfahren in höchstens k Iterationen.*

Beweis. Seien $\lambda_1, \dots, \lambda_k$ die verschiedenen Eigenwerte von A . Dann ist

$$q(z) := \left(1 - \frac{z}{\lambda_1}\right) \dots \left(1 - \frac{z}{\lambda_k}\right) \in \mathcal{P}_k$$

und erfüllt $q(0) = 1$ sowie $q(\lambda) = 0$ für alle $\lambda \in \sigma(A)$. Damit gilt

$$\|e^k\|_A \leq \max_j |q(\lambda_j)| \|e^0\|_A = 0.$$

und daher $x^k = x^*$. \square

Quantitative Abschätzungen erhalten wir für reelle Matrizen durch das Einsetzen geeigneter Polynome. Wir betrachten die *Tschebyschow-Polynome*³ T_k , die die Rekursion

$$\begin{aligned} T_0(t) &= 1, & T_1(t) &= t, \\ T_{k+1}(t) &= 2tT_k(t) - T_{k-1}(t) \quad \text{für } k \geq 1 \end{aligned}$$

erfüllen.⁴ Durch Induktion zeigt man leicht, dass

1. $T_k \in \mathcal{P}_k$,
2. $T_k(t) = \cos(k \cos^{-1}(t))$ und deshalb $|T_k(t)| \leq 1$ für $|t| \leq 1$ sowie
3. $T_k\left(\frac{1}{2}(z + z^{-1})\right) = \frac{1}{2}(z^k + z^{-k})$

für alle $k \in \mathbb{N}$ gilt.

Es bezeichne $\kappa = \lambda_1/\lambda_n$ wieder die Konditionszahl der Matrix A . Dann haben wir folgende Abschätzung für die Konvergenzgeschwindigkeit des CG-Verfahrens:

³Nach [Pafnuti Lwowitsch Tschebyschow](#), einem der bedeutendsten russischen Mathematiker des 19. Jahrhunderts und Vater der Approximationstheorie. Zahlreiche Transliterationen seines Namens sind in Verwendung, darunter „Tschebyscheff“ (im deutschsprachigen Raum noch häufig anzutreffen), „Tschebyschew“, „Tschebyshev“, und – insbesondere im Englischen – „Chebyshev“.

⁴Tatsächlich wird für diese Polynome sogar das Minimum angenommen, siehe z. B. [\[Gautschi 2012, Theorem 2.2.1\]](#).

Satz 7.3. Sei $A \in \mathbb{R}^{n \times n}$ symmetrisch und positiv definit. Dann gilt für die Näherungen im CG-Verfahren die Abschätzung

$$\|x^* - x^k\|_A \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \|x^* - x^0\|_A.$$

Beweis. Nach Lemma 7.1 gilt $\|x^* - x^k\|_A \leq \max_{\lambda \in \sigma(A)} |p(\lambda)| \|x^* - x^0\|_A$ für jedes Polynom $p \in \mathcal{P}_k$ mit $p(0) = 1$. Wir wählen

$$p(t) = \frac{T_k \left(\mu - \frac{2t}{\lambda_1 - \lambda_n} \right)}{T_k(\mu)}$$

mit

$$\mu = \frac{\lambda_1 + \lambda_n}{\lambda_1 - \lambda_n} = \frac{\kappa + 1}{\kappa - 1} = \frac{1}{2} \left(\frac{\sqrt{\kappa} + 1}{\sqrt{\kappa} - 1} + \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right).$$

Aus den Eigenschaften der Tschebyschow-Polynome folgt dann, dass $p(0) = 1$, $p \in \mathcal{P}_k$ und $|p(t)| \leq |T_k(\mu)^{-1}|$ gilt. Außerdem ist

$$T_k(\mu) = \frac{1}{2} \left(\left(\frac{\sqrt{\kappa} + 1}{\sqrt{\kappa} - 1} \right)^k + \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \right) \geq \frac{1}{2} \left(\frac{\sqrt{\kappa} + 1}{\sqrt{\kappa} - 1} \right)^k.$$

Einsetzen liefert dann die gewünschte Abschätzung. □

Um für den Fehler $\|e^k\|_A \leq \varepsilon \|e^0\|_A$ zu garantieren, sind daher in der Regel

$$k \geq \frac{\sqrt{\kappa}}{2} \ln \left(\frac{2}{\varepsilon} \right) = \mathcal{O}(\sqrt{\kappa})$$

Iterationen notwendig (hier wurde die Ungleichung $\ln \left(\frac{a+1}{a-1} \right) > \frac{2}{a}$ für $a > 1$ verwendet).

7.3 PRÄKONDITIONIERTES CG-VERFAHREN

Die Konvergenzgeschwindigkeit des CG-Verfahrens hängt von der Konditionszahl der Matrix A ab. Ähnlich wie bei den stationären Verfahren wird daher in der Praxis oft ein äquivalentes Gleichungssystem $M^{-1}Ax = M^{-1}b$ für eine reguläre Matrix M betrachtet. Dabei soll M

1. möglichst „ähnlich“ zu A sein, in dem Sinne, dass $\kappa(M^{-1}A) \approx 1$ gilt, und
2. $M^{-1}v$ möglichst leicht zu berechnen sein.

Man nennt diesen Ansatz *Präkonditionierung*; die Matrix M heißt dabei *Präkonditionierer*.

Im CG-Verfahren sollte M selbstadjungiert und positiv definit sein. Allerdings ist auch in diesem Falle $M^{-1}A$ nicht notwendigerweise selbstadjungiert. Falls die Cholesky-Zerlegung $M = LL^*$ zur Verfügung steht, kann man das CG-Verfahren auf die Gleichung

$$L^{-1}A(L^{-1})^*u = L^{-1}b, \quad x = (L^{-1})^*u,$$

anwenden. Dabei stellt sich heraus, dass der Algorithmus nur M^{-1} benötigt. Eine alternative Herleitung desselben Algorithmus verwendet, dass $M^{-1}A$ selbstadjungiert ist im M -Skalarprodukt, denn es gilt für alle $x, y \in \mathbb{C}^n$

$$\langle M^{-1}Ax, y \rangle_M = \langle Ax, y \rangle = \langle x, Ay \rangle = \langle x, M^{-1}Ay \rangle_M.$$

Wir modifizieren Algorithmus 7.1 nun so, dass die $(M^{-1}A)$ -Orthogonalisierung bezüglich des M -Skalarprodukts erfolgt, indem wir $\langle \cdot, \cdot \rangle$ durch $\langle \cdot, \cdot \rangle_M$ und A durch $M^{-1}A$ ersetzen. Die Konvergenztheorie aus Kapitel 7.2 bleibt dabei anwendbar, denn das abgewandelte CG-Verfahren für das System $M^{-1}Ax = M^{-1}b$ minimiert ebenso

$$\langle x^* - x, M^{-1}Ax^* - M^{-1}Ax \rangle_M = \langle x^* - x, Ax^* - Ax \rangle = \|x^* - x\|_A^2,$$

nun allerdings über den Krylovraum $\mathcal{K}_k(M^{-1}A, M^{-1}r^0)$.

Nach Einführen von $z^k := M^{-1}b - M^{-1}Ax^k = M^{-1}r^k$ für das Residuum des vorkonditionierten Systems erhalten wir:

- 1: $\alpha_k = \langle z^{k-1}, z^{k-1} \rangle_M / \langle p_{k-1}, M^{-1}Ap_{k-1} \rangle_M$
- 2: $x^k = x^{k-1} + \alpha_k p_{k-1}$
- 3: $r^k = r^{k-1} - \alpha_k Ap_{k-1}$
- 4: $z^k = M^{-1}r^k$
- 5: $\beta_k = \langle z^k, z^k \rangle_M / \langle z^{k-1}, z^{k-1} \rangle_M$
- 6: $p_k = z^k + \beta_k p_{k-1}$

Wegen $\langle z^k, z^k \rangle_M = \langle z^k, r^k \rangle$ und $\langle p_{k-1}, M^{-1}Ap_{k-1} \rangle_M = \langle p_{k-1}, Ap_{k-1} \rangle$ benötigt der *prä-konditionierte CG-Algorithmus* (PCG) nur eine zusätzliche Matrixmultiplikation.

Algorithmus 7.2 PCG-Verfahren

Input: $A, b, x^0, M^{-1}, \varepsilon > 0$

- 1: $p^0 = r^0 = b - Ax^0, z^0 = M^{-1}r^0, k = 0$
- 2: **while** $\|r^k\|_2 > \varepsilon$ **do**
- 3: $k \leftarrow k + 1$
- 4: $\alpha_k = \langle z^{k-1}, r^{k-1} \rangle / \langle p_{k-1}, Ap_{k-1} \rangle$
- 5: $x^k = x^{k-1} + \alpha_k p_{k-1}$
- 6: $r^k = r^{k-1} - \alpha_k Ap_{k-1}$
- 7: $z^k = M^{-1}r^k$
- 8: $\beta_k = \langle z^k, r^k \rangle / \langle z^{k-1}, r^{k-1} \rangle$
- 9: $p_k = z^k + \beta_k p_{k-1}$

Output: x^k

Dabei wird in der Regel der Schritt 7 durch Lösung des linearen Gleichungssystems $Mz^k = r^k$ realisiert (und im Algorithmus sollte M^{-1} als Prozedur zu dessen Lösung verstanden werden).

Die Wahl eines Präkonditionierers ist in der Praxis kritisch für eine akzeptable Konvergenzgeschwindigkeit und oft stark von dem zu lösenden Gleichungssystem abhängig. Einige typische Beispiele sind:

1. *Zerlegungsstrategien* verwenden Präkonditionierer ähnlich denen aus Kapitel 4.2. Für eine symmetrisch positiv definite Matrix $A = D + L + L^*$ kommt im CG-Verfahren in Frage:
 - a) Jacobi-Iteration, $M = D$; dies entspricht einer Skalierung der Diagonaleinträge von A auf 1, und ist trotz in der Regel geringer Wirkung eigentlich immer zu empfehlen (solange kein besserer Präkonditionierer zur Verfügung steht).
 - b) Symmetrische Gauß–Seidel-Iteration, $M = (D + L)D^{-1}(D + L^*)$, wobei das Gleichungssystem $Mz^k = r^k$ einfach durch Vor- und Rückwärtssubstitution gelöst werden kann.
2. Die *unvollständige Cholesky-Zerlegung* verwendet für $M = \tilde{L}^* \tilde{L}$ eine Cholesky-Zerlegung $A = L^* L$, wobei \tilde{L} nur diejenigen Einträge l_{ij} von L enthält, für die $a_{ij} \neq 0$ ist. Eine Variante lässt zusätzlich jene Einträge weg, für die $|l_{ij}| < \varepsilon$ für eine vorgegebene Toleranz ε ist. Die Existenz einer solchen Zerlegung ist aber nicht garantiert.
3. *Approximative Inverse* („*sparse approximate inverse*“, *SPAI*) sind Matrizen M^{-1} mit einer vorgegebenen Struktur, die $\|I - M^{-1}A\|_F$ minimieren.
4. Für bestimmte Matrixklassen, die aus der Diskretisierung elliptischer partieller Differentialgleichungen entstehen, können *Mehrgitterverfahren* sehr gute Ergebnisse liefern.

DAS GMRES-VERFAHREN



Das *GMRES-Verfahren* (englisch „generalized minimal residual“) ist ein Krylovraum-Petrov-Galerkin-Verfahren mit $\mathcal{K} = \mathcal{K}_m(A, r^0)$ und $\mathcal{L} = A\mathcal{K}$. Als solches kann es für beliebige invertierbare Matrizen angewendet werden; der Preis dafür ist ein deutlich höherer Rechenaufwand und eine weniger befriedigende Konvergenztheorie.

8.1 ALGORITHMUS

Nach Satz 5.2 konstruiert ein Petrov-Galerkin-Verfahren mit $\mathcal{K} = \mathcal{K}_m(A, r^0)$ und $\mathcal{L} = A\mathcal{K}$ eine Näherung x^m , die das Residuum $\|b - Ax\|_2$ über alle $x \in x^0 + \mathcal{K}_m(A, r^0)$ minimiert. Die Grundidee des GMRES-Verfahrens ist, dieses Minimierungsproblem effizient mit Hilfe der im Arnoldi-Prozess konstruierten Orthonormalbasis zu lösen.

Wie im Arnoldi-Verfahren starten wir den Prozess mit $q_1 = r^0/\beta$, $\beta = \|r^0\|$. Bricht der Prozess nicht ab – diesen Fall betrachten wir separat, – erhalten wir nach m Schritten die orthonormalen Vektoren q_1, \dots, q_{m+1} und h_{ij} , $1 \leq i \leq j+1 \leq m+1$. Definieren wir die Matrizen $Q_{m+1} = [q_1, \dots, q_{m+1}] \in \mathbb{C}^{n \times (m+1)}$ und $\bar{H}_m = (h_{ij})_{i,j} \in \mathbb{C}^{(m+1) \times m}$, so hat die Matrix-Zerlegung (6.2) die Form

$$AQ_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^* = Q_{m+1} \bar{H}_m.$$

Da wir jedes $x \in x^0 + \mathcal{K}_m(A, r^0)$ schreiben können als $x = x^0 + Q_m \xi$ für ein $\xi \in \mathbb{C}^m$, gilt

$$\begin{aligned} b - Ax &= b - A(x^0 + Q_m \xi) \\ &= r^0 - AQ_m \xi \\ &= \beta q_1 - Q_{m+1} \bar{H}_m \xi \\ &= Q_{m+1} (\beta e_1 - \bar{H}_m \xi). \end{aligned}$$

Nun ist Q_{m+1} unitär, so dass

$$\|b - Ax\|_2 = \|\beta e_1 - \bar{H}_m \xi\|_2$$

gilt. Die Näherung x^m kann also durch Lösung eines (typischerweise kleinen) $(m+1) \times m$ -Ausgleichsproblem für eine obere Hessenbergmatrix berechnet werden (etwa mit Hilfe der QR-Zerlegung). Das abstrakte GMRES-Verfahren besteht also aus folgenden Schritten:

Input: $A \in \mathbb{C}^{n \times n}$, $b \in \mathbb{C}^n$, $x^0 \in \mathbb{C}^n$, ε

- 1: $r^0 = b - Ax^0$, $\beta = \|r^0\|_2$, $m = 0$
- 2: **while** $\|b - Ax^{m-1}\|_2 > \varepsilon \|b - Ax^0\|$ **do**
- 3: Berechne Q_m, \bar{H}_m mit **Arnoldi-Prozess**
- 4: Löse $\min_{y \in \mathbb{C}^m} \|\beta e_1 - \bar{H}_m y\|_2$ für y^m
- 5: Setze $x^m = x^0 + Q_m y^m$, $m \leftarrow m + 1$

Output: x^{m-1}

Für eine invertierbare Matrix $A \in \mathbb{C}^{n \times n}$ kann dieses Verfahren nur versagen, wenn der Arnoldi-Prozess vorzeitig im Schritt $j < m$ abbricht. (Die Matrix \bar{H}_m hat vollen Spaltenrang, da $H_m = Q_m^* A Q_m$ vollen Rang hat.) In diesem Fall ist $\mathcal{K}_j(A, r^0)$ invariant unter A , und es ist $0 = w_j = h_{j+1,j} q_{j+1}$. Damit folgt aus der Zerlegung (6.2), dass $AQ_j = Q_j H_j$ gilt, und insbesondere, dass H_j invertierbar ist. Es existiert also ein $\xi^j = H_j^{-1}(\beta e_1)$ mit

$$0 = Q_j(\beta e_1 - H_j \xi^j) = r^0 - AQ_j \xi^j = b - Ax^j,$$

d. h. $x^j = x^0 + Q_j \xi^j$ ist die exakte Lösung des Gleichungssystems. Daraus folgt, dass das GMRES-Verfahren für invertierbare Matrizen spätestens für $m = n$ (in exakter Arithmetik) die Lösung findet. Aus den Überlegungen in Kapitel 5 schließen wir außerdem, dass die Norm des Residuums $\|r^m\|_2$ monoton (aber nicht notwendigerweise strikt monoton) fallend ist.

In der Praxis wird dabei in jedem Schritt die bereits im vorigen Schritt konstruierte Orthonormalbasis $\{q_1, \dots, q_m\}$ verwendet, so dass jeweils nur eine neue Spalte für Q_{m+1} und \bar{H}_{m+1} berechnet werden muss. Um y^m im Schritt 4 zu berechnen, muss dann eine Folge von *Givens-Rotationen* $G_{j+1,j}$ (aufgefasst als lineare Operatoren) mit

$$G_m[\bar{H}_m, e_1] = \begin{pmatrix} R_m & d_m \\ 0 & \rho_m \end{pmatrix}, \quad G_m = G_{m+1,m} \circ G_{m,m-1} \circ \dots \circ G_{2,1}$$

konstruiert, so dass $R_m \in \mathbb{C}^{m \times m}$ eine obere Dreiecksmatrix ist. Da \bar{H}_m eine obere Hessenberg-Matrix ist, wird $G_{j+1,j}$ dabei so gewählt, dass das entsprechende Subdiagonalelement $h_{j+1,j}$ annulliert wird. (In der Praxis wird dabei $G_{j+1,j}$ als 2×2 -Matrix nur auf den Vektor $(h_{j,j}, h_{j+1,j})^T$ angewendet, d. h. $G_{j+1,j}(h_{j,j}, h_{j+1,j})^T = (\gamma, 0)^T$ für ein $\gamma \in \mathbb{C}$, da die restlichen Komponenten unverändert bleiben.) Die Lösung und das Residuum erhalten wir dann durch

$$y^m = \beta R_m^{-1} d_m, \quad \|r^m\|_2 = \beta |\rho_m|.$$

Insbesondere erhalten wir das relative Residuum $\|r^m\|_2 / \|r^0\|_2 = |\rho_m|$, ohne y^m und x^m berechnen zu müssen.

Die Praktikabilität des Verfahrens beruht darauf, dass auch die QR-Zerlegung von \bar{H}_m spaltenweise aufgebaut werden kann. Angenommen, wir haben bereits eine QR-Zerlegung von \bar{H}_{m-1} konstruiert mit

$$G_{m-1} \circ \bar{H}_{m-1} = \begin{pmatrix} R_{m-1} \\ 0 \end{pmatrix} \in \mathbb{C}^{m \times (m-1)}, \quad G_{m-1}(e_1) = \begin{pmatrix} d_{m-1} \\ \rho_{m-1} \end{pmatrix} \in \mathbb{C}^m.$$

Definieren wir $h_m := (h_{1,m}, \dots, h_{m,m})^T \in \mathbb{C}^m$ und $G_{m-1}(h_m) := (c_{m-1}, c_{m,m})^T \in \mathbb{C}^m$, so ist

$$\begin{aligned} G_m \circ \bar{H}_m &= G_{m+1,m} \circ \begin{pmatrix} G_{m-1} \circ \bar{H}_{m-1} & G_{m-1}(h_m) \\ 0 & h_{m+1,m} \end{pmatrix} \\ &= G_{m+1,m} \circ \begin{pmatrix} R_{m-1} & c_{m-1} \\ 0 & c_{m,m} \\ 0 & h_{m+1,m} \end{pmatrix}, \end{aligned}$$

da G_{m-1} nur auf die ersten m Zeilen wirkt. Wir bestimmen nun $G_{m+1,m}$ so, dass

$$G_{m+1,m} \circ \begin{pmatrix} R_{m-1} & c_{m-1} \\ 0 & c_{m,m} \\ 0 & h_{m+1,m} \end{pmatrix} = \begin{pmatrix} R_{m-1} & c_{m-1} \\ 0 & \gamma_m \\ 0 & 0 \end{pmatrix}$$

ist (d. h. in der Praxis so, dass die Matrix $G_{m+1,m}$ den Vektor $(c_{m,m}, h_{m+1,m})^T$ auf ein $(\gamma_m, 0)^T$ abbildet), und setzen

$$R_m = \begin{pmatrix} R_{m-1} & c_{m-1} \\ 0 & \gamma_m \end{pmatrix} \in \mathbb{C}^{m \times m}.$$

Der Vektor $c_m := (c_{m-1}, \gamma_m)^T$ enthält also genau die von Null verschiedenen Einträge, die R_{m-1} zu R_m erweitern.

Für die rechte Seite erhalten wir dann

$$\begin{aligned} G_m(e_1) &= G_{m+1,m} \circ \begin{pmatrix} G_{m-1}(e_1) \\ 0 \end{pmatrix} = G_{m+1,m} \circ \begin{pmatrix} d_{m-1} \\ \rho_{m-1} \\ 0 \end{pmatrix} = \begin{pmatrix} d_{m-1} \\ \delta_m \\ \rho_m \end{pmatrix} \\ &=: \begin{pmatrix} d_m \\ \rho_m \end{pmatrix} \in \mathbb{C}^{m+1}. \end{aligned}$$

Weil $G_{m+1,m}$ die ersten $m-1$ Komponenten von d_m unverändert lässt, müssen wir in der Praxis nur die Rotation $G_{m+1,m}$ auf den Vektor $(\rho_{m-1}, 0)^T$ anwenden, und erhalten den neuen Eintrag δ_m , der d_{m-1} zu d_m erweitert, sowie das neue Residuum ρ_m .

Wir können daher solange neue Basisvektoren generieren, bis das relative Residuum $|\rho_m|$ in dem von ihnen aufgespannten Krylovraum klein genug ist, und erst dann y^m und x^m berechnen. Trotzdem ist das Verfahren aufwendig: Die Anzahl der zu speichernden Vektoren

sowie der arithmetischen Operationen wächst dabei quadratisch in m . In der Praxis wird daher häufig nach einer festen Anzahl von Iterationen (üblicherweise zwischen 10 und 30) die aktuelle Näherung x^m berechnet und das Verfahren mit $x^0 := x^m$ neu gestartet. Die wesentlichen Schritte im *GMRES-Verfahren mit Neustarts*¹ sind im folgenden Algorithmus aufgeführt.

Algorithmus 8.1 GMRES(m)

Input: $A \in \mathbb{C}^{n \times n}$, $b \in \mathbb{C}^n$, $x^0 \in \mathbb{C}^n$, $m, \varepsilon > 0$

- 1: $r^0 = b - Ax^0$, $\beta = \|r^0\|_2$, $k = 0$
- 2: **while** $\beta > \varepsilon$ **do**
- 3: $\rho_0 = 1$, $q_1 = r^k/\beta$, $G_{1,0} = I$, $j = 0$
- 4: **while** $|\rho_j| > \varepsilon$ und $j < m$ **do**
- 5: $j \leftarrow j + 1$
- 6: Berechne $q_{j+1}, h_{1,j}, \dots, h_{j+1,j}$ durch Schritt in **Arnoldi-Prozess**
- 7: Wende Givens-Rotationen $G_{i,i-1}$ für $i \leq j$ auf $h_j := (h_{1,j}, \dots, h_{j,j})^T$ an;
 \triangleright Ergebnis: $(c_{j-1}, c_{j,j})^T$
- 8: Berechne Givens-Rotation $G_{j+1,j}$ so, dass $G_{j+1,j}(c_{j,j}, h_{j+1,j})^T = (\gamma_j, 0)$;
 \triangleright Ergebnis: $G_{j+1,j}$, $c_j := (c_{j-1}, \gamma_j)^T$
- 9: Wende Givens-Rotation $G_{j+1,j}$ auf $(\rho_{j-1}, 0)^T$ an; \triangleright Ergebnis: δ_j, ρ_j
- 10: Bilde $R_k = [c_1, \dots, c_j]$, $Q_k = [q_1, \dots, q_j]$, $d_k = (\delta_1, \dots, \delta_j)^T$
- 11: $x^{k+1} = x^k + \beta Q_k R_k^{-1} d_k$
- 12: $r^{k+1} = b - Ax^{k+1}$, $\beta = \|r^{k+1}\|_2$, $k \leftarrow k + 1$

Output: x^k

8.2 KONVERGENZ

Das *GMRES-Verfahren mit Neustarts* bricht zwar nur ab, wenn die exakte Lösung gefunden ist, aber Konvergenzraten können im Allgemeinen nicht gezeigt werden. Für positiv definite (aber nicht unbedingt symmetrische) Matrizen folgt die Konvergenz jedoch aus Satz 5.5.

Folgerung 8.1. Sei $A \in \mathbb{R}^{n \times n}$ positiv definit. Dann konvergiert GMRES(m) (Algorithmus 8.1) für jedes $m \geq 1$.

Beweis. Für jedes $j \geq 1$ im äußeren Schritt k ist das Residuum r^k in $\mathcal{K}_j(A, r^k)$ enthalten. Da in jedem inneren Schritt j das Residuum $\|r^{k+1}\|_2$ über $x^k + \mathcal{K}_j(A, r^k)$ minimiert wird, ist die Reduktion mindestens so groß wie in der MR-Iteration. Aus der Konvergenz der MR-Iteration folgt daher die Konvergenz des GMRES-Verfahrens mit Neustarts. □

¹publiziert durch Yousef Saad und Martin H. Schultz in [Saad und Schultz 1986]

Selbst ohne Neustarts kann in der Regel nicht viel ausgesagt werden. Da das GMRES-Verfahren das Residuum $\|b - Ax\|_2$ über alle $x \in x^0 + \mathcal{K}_m(A, r^0)$ minimiert und

$$\begin{aligned} b - Ax &= b - A \left(x^0 + \sum_{j=0}^{m-1} \gamma_j A^j r^0 \right) = r^0 - \sum_{j=0}^{m-1} \gamma_j A^{j+1} r^0 \\ &= p(A)r^0 \end{aligned}$$

für ein Polynom $p \in \mathcal{P}_m$ mit $p(0) = 1$ gilt, erhalten wir analog zum CG-Verfahren die Abschätzung

$$\|r^m\|_2 = \min_{p \in \mathcal{P}_m, p(0)=1} \|p(A)r^0\|_2.$$

Ist A diagonalisierbar mit $A = XDX^{-1}$, so beweist man wie in Lemma 7.1, dass

$$\|r^m\|_2 \leq \kappa(X) \left(\min_{p \in \mathcal{P}_m, p(0)=1} \max_{\lambda \in \sigma(A)} |p(\lambda)| \right) \|r^0\|_2$$

gilt, wobei $\kappa(X)$ die Konditionszahl von X ist. Diese Abschätzung ist aber im Allgemeinen nicht sehr nützlich, da $\kappa(X)$ sehr groß sein kann und das Approximationsproblem für komplexe Polynome keine einfache Lösung hat. Ist A eine normale Matrix (für die X unitär und damit $\kappa(X) = 1$ ist) und sind die Eigenwerte von A in einer Menge enthalten, auf der ein geeignetes Polynom günstig abgeschätzt werden kann, so lässt sich die Konvergenz zeigen. Dies (und insbesondere die Erweiterung auf nichtnormale Matrizen) ist ein aktuelles Forschungsthema.

8.3 PRÄKONDITIONIERTES GMRES-VERFAHREN

Auch das GMRES-Verfahren – insbesondere mit Neustarts – wird in der Praxis präkonditioniert. Da die Matrix hier nicht symmetrisch sein muss, haben wir zwei Möglichkeiten, einen Präkonditionierer M^{-1} anzuwenden.

1. Als *Links-Präkonditionierer*: Hier wendet man das GMRES-Verfahren auf

$$M^{-1}Ax = M^{-1}b$$

an, indem mit $M^{-1}r^0 = M^{-1}(b - Ax^0)$ gestartet und im Arnoldi-Prozess die Matrix $M^{-1}A$ angewendet wird. Die Näherung x^m im GMRES-Verfahren minimiert dann

$$\|M^{-1}b - M^{-1}Ax\|_2 \quad \text{über alle } x \in x^0 + \mathcal{K}_m(M^{-1}A, M^{-1}r^0).$$

Der Nachteil ist, dass dann im Verfahren ρ_m nur das präkonditionierte Residuum angezeigt.

2. Als *Rechts-Präkonditionierer*: Hier löst man

$$AM^{-1}u = b$$

und setzt $x = M^{-1}u$. Hier verwendet das GMRES-Verfahren die ursprünglichen Residuen

$$r^m = b - Ax^m = b - AM^{-1}u^m$$

sowie die Matrix AM^{-1} . Dabei wird auch bei der Berechnung der Näherungslösung der Hilfsvektor u nicht benötigt. Ist y^m die Lösung des projizierten Ausgleichsproblem, so ist

$$x^m = M^{-1}u^m = M^{-1}(u^0 + Q_m y^m) = x^0 + M^{-1}Q_m y^m.$$

Die Näherung x^m minimiert daher

$$\|b - Ax\|_2 \quad \text{über alle} \quad x \in x^0 + M^{-1}\mathcal{K}_m(AM^{-1}, r^0).$$

Tatsächlich kann man per Induktion zeigen, dass $\mathcal{K}_m(M^{-1}A, M^{-1}r^0) = M^{-1}\mathcal{K}_m(AM^{-1}, r^0)$ ist. Die beiden Varianten unterscheiden sich also nur im zu minimierenden Residuum. Die zweite Variante hat aber den Vorteil, dass in jedem Schritt ein unterschiedlicher Präkonditionierer M_m^{-1} gewählt werden kann; für die Berechnung von x^m genügt es sogar, anstelle der M_m^{-1} nur die präkonditionierten Residuen $M_m^{-1}r_m$ (die man für den Arnoldi-Prozess sowieso braucht) zu speichern. Der entsprechende Algorithmus wird als *flexible GMRES* (*FGMRES*) bezeichnet; siehe [Saad 2003, Kapitel 9.4.1].

Als Präkonditionierer kommen wieder die Ansätze aus Kapitel 7.3 in Frage.

BIORTHOGONALE KRYLOVRAUM-VERFAHREN

Der Nachteil am GMRES-Verfahren ist die Notwendigkeit, in jedem Schritt gegen alle Basisvektoren zu orthogonalisieren. Wir suchen daher nach Verfahren, die auch für unsymmetrische Matrizen mit kurzen Rekursionen auskommen. Im CG-Verfahren ermöglicht dies die Symmetrie der Matrix A und der Ritz–Galerkin-Ansatz; für unsymmetrische Matrizen verwenden wir stattdessen einen geeigneten Petrov–Galerkin-Ansatz. Alle Verfahren in diesem Kapitel basieren auf der Wahl

$$\mathcal{K} = \mathcal{K}_m(A, v), \quad \mathcal{L} = \mathcal{K}_m(A^*, w).$$

9.1 DER BI-LANZOS-PROZESS

Für eine selbstadjungierte Matrix $A \in \mathbb{C}^{n \times n}$ konstruiert der Lanczos-Prozess 6.2 eine unitäre Matrix $[q_1, \dots, q_m] =: Q_m \in \mathbb{C}^{n \times m}$, so dass $Q_m^* A Q_m$ tridiagonal ist. Statt der Tridiagonalform (wie im Arnoldi-Prozess) geben wir nun die Ähnlichkeitstransformation auf: Wir suchen $V_m, W_m \in \mathbb{C}^{n \times m}$, so dass

$$W_m^* A V_m = T_m = \begin{pmatrix} \alpha_1 & \beta_2 & & & \\ \gamma_2 & \alpha_2 & \beta_3 & & \\ & \gamma_3 & \ddots & \ddots & \\ & & \ddots & \alpha_{m-1} & \beta_m \\ & & & \gamma_m & \alpha_m \end{pmatrix}$$

gilt. Zusätzlich fordern wir $W_m^* V_m = I_m$, d. h. dass die Spalten v_i von V_m und w_j von W_m biorthogonal sind. Analog zum Gram–Schmidt-Verfahrens können wir aus diesen Bedingungen durch Induktion einen Algorithmus konstruieren.

Dazu betrachten wir $A V_n = V_n T_n$ und $A^* W_n = W_n T_n^*$ (für $m = n$ erhält man diese Gleichungen aus $W_m^* A V_m = T_m$ durch Multiplikation mit V_m bzw. W_m^*) spaltenweise, um

9

(beginnend mit $k = 1$) Rekursionen für v_{k+1} , w_{k+1} , α_k , β_{k+1} und γ_{k+1} herzuleiten. Für $1 \leq k \leq n$ ist (mit $v_0 = w_0 = 0$)

$$\begin{aligned} Av_k &= \beta_k v_{k-1} + \alpha_k v_k + \gamma_{k+1} v_{k+1}, \\ A^* w_k &= \overline{\gamma_k} w_{k-1} + \overline{\alpha_k} w_k + \overline{\beta_{k+1}} w_{k+1}. \end{aligned}$$

Auflösen nach $\tilde{v}_{k+1} := \gamma_{k+1} v_{k+1}$ und $\tilde{w}_{k+1} := \overline{\beta_{k+1}} w_{k+1}$ ergibt

$$(9.1) \quad \tilde{v}_{k+1} = (A - \alpha_k I)v_k - \beta_k v_{k-1},$$

$$(9.2) \quad \tilde{w}_{k+1} = (A^* - \overline{\alpha_k} I)w_k - \overline{\gamma_k} w_{k-1}.$$

Als Induktionsvoraussetzung nehmen wir an, dass die v_i und w_j für $i, j \leq k$ biorthogonal sind, und zeigen nun, dass wir α_k so wählen können, dass \tilde{v}_{k+1} orthogonal zu allen w_j und \tilde{w}_{k+1} orthogonal zu allen v_j ist für $1 \leq i \leq j$. Für $j = k$ ist dann

$$\begin{aligned} \langle \tilde{v}_{k+1}, w_k \rangle &= \langle Av_k, w_k \rangle - \alpha_k \langle v_k, w_k \rangle - \beta_k \langle v_{k-1}, w_k \rangle \\ &= \langle Av_k, w_k \rangle - \alpha_k \end{aligned}$$

nach Voraussetzung. Setzen wir also

$$\alpha_k = \langle Av_k, w_k \rangle,$$

so ist wie gewünscht $\langle \tilde{v}_{k+1}, w_k \rangle = 0$. Für $j = k - 1$ gilt unabhängig von der Wahl von α_k , dass

$$\begin{aligned} \langle \tilde{v}_{k+1}, w_{k-1} \rangle &= \langle Av_k, w_{k-1} \rangle - \alpha_k \langle v_k, w_{k-1} \rangle - \beta_k \langle v_{k-1}, w_{k-1} \rangle \\ &= \langle v_k, A^* w_{k-1} \rangle - \beta_k \\ &= \langle v_k, \overline{\gamma_{k-1}} w_{k-2} + \overline{\alpha_{k-1}} w_{k-1} + \overline{\beta_k} w_k \rangle - \beta_k \\ &= \beta_k - \beta_k = 0, \end{aligned}$$

wieder aufgrund der Biorthogonalität der v_i und w_j für $i, j \leq k$. Die selbe Rechnung liefert $\langle \tilde{v}_{k+1}, w_j \rangle = 0$ für $j < k - 1$. Mit dieser Wahl von α_k gilt ausserdem

$$\begin{aligned} \langle v_k, \tilde{w}_{k+1} \rangle &= \langle v_k, A^* w_k \rangle - \langle v_k, \overline{\alpha_k} w_k \rangle - \langle v_k, \overline{\gamma_k} w_{k-1} \rangle \\ &= \langle Av_k, w_k \rangle - \alpha_k = 0 \end{aligned}$$

sowie $\langle v_j, \tilde{w}_{k+1} \rangle = 0$ für alle $j \leq k$.

Es bleibt, die Bedingung $\langle v_{k+1}, w_{k+1} \rangle = 1$ durch geeignete Wahl von β_{k+1} und γ_{k+1} zu erfüllen. Nach Definition ist

$$\langle \tilde{v}_{k+1}, \tilde{w}_{k+1} \rangle = \beta_{k+1} \gamma_{k+1} \langle v_{k+1}, w_{k+1} \rangle.$$

Ist $\langle \tilde{v}_{k+1}, \tilde{w}_{k+1} \rangle \neq 0$ und wählen wir β_{k+1} und γ_{k+1} so, dass

$$\beta_{k+1} \gamma_{k+1} = \langle \tilde{v}_{k+1}, \tilde{w}_{k+1} \rangle$$

gilt, so sind v_i und w_j biorthogonal für $1 \leq i, j \leq k+1$. (Ist $\langle \tilde{v}_{k+1}, \tilde{w}_{k+1} \rangle = 0$, müssen wir den Prozess abbrechen.) Diese Wahl ist nicht eindeutig; wir können zum Beispiel zusätzlich fordern, dass $|\beta_{k+1}| = |\gamma_{k+1}|$ ist.

Starten wir mit v_1 und w_1 mit $\langle w_1, v_1 \rangle = 1$, erhalten wir daraus einen durchführbaren Algorithmus, den *Bi-Lanczos-Prozess*.

Algorithmus 9.1 Bi-Lanczos-Prozess

Input: $A \in \mathbb{C}^{n \times n}$, $v_1, w_1 \in \mathbb{C}^n \setminus \{0\}$ mit $\langle w_1, v_1 \rangle = 1$,

- 1: $w_0 = v_0 = 0, \beta_1 = \gamma_1 = 0$
- 2: **for** $j = 1, \dots, m$ **do**
- 3: $\alpha_j = \langle Av_j, w_j \rangle$
- 4: $v_{j+1} = Av_j - \alpha_j v_j - \beta_j v_{j-1}$
- 5: $w_{j+1} = A^* w_j - \overline{\alpha_j} w_j - \overline{\gamma_j} w_{j-1}$
- 6: $\gamma_{j+1} = \sqrt{|\langle v_{j+1}, w_{j+1} \rangle|}$
- 7: **if** $\gamma_{j+1} = 0$ **then**
- 8: **stop**
- 9: **else**
- 10: $\beta_{j+1} = \langle v_{j+1}, w_{j+1} \rangle / \gamma_{j+1}$
- 11: $v_{j+1} \leftarrow v_{j+1} / \gamma_{j+1}$
- 12: $w_{j+1} \leftarrow w_{j+1} / \beta_{j+1}$

Output: $v_j, w_j, \alpha_j, 1 \leq j \leq m, \beta_{j+1}, \gamma_{j+1}, 1 \leq j \leq m-1$

Mit $V_m = [v_1, \dots, v_m]$ und $W_m = [w_1, \dots, w_m]$ können wir (9.1) und (9.2) in Matrixform schreiben als

$$\begin{aligned} AV_m &= V_m T_m + \gamma_{m+1} v_{m+1} e_m^*, \\ A^* W_m &= W_m T_m^* + \overline{\beta_{m+1}} w_{m+1} e_m^*. \end{aligned}$$

Aus (9.1) und (9.2) folgt auch per Induktion, dass

$$\begin{aligned} \text{span}\{v_1, \dots, v_m\} &= \mathcal{K}_m(A, v_1), \\ \text{span}\{w_1, \dots, w_m\} &= \mathcal{K}_m(A^*, w_1) \end{aligned}$$

gilt, falls der Bi-Lanczos-Prozess nicht abbricht. Dies passiert, wenn $\langle v_{j+1}, w_{j+1} \rangle$ verschwindet (oder, in endlicher Arithmetik, zu klein wird). Ist $v_{j+1} = 0$ oder $w_{j+1} = 0$, so ist $\mathcal{K}_j(A, v_1)$ invariant unter A bzw. $\mathcal{K}_j(A^*, w_1)$ invariant unter A^* („lucky breakdown“). Andernfalls handelt es sich um einen „serious breakdown“. Dieser kann durch so-genannte „look-ahead“-Strategien behandelt werden, in denen mehrere Vektoren in Folge blockweise biorthogonalisiert werden.

9.2 DAS BI-CG-VERFAHREN

Analog zum Arnoldi-Verfahren in Abschnitt 6.3 können wir den Bi-Lanczos-Prozess verwenden, um ein Projektionsverfahren für das lineare Gleichungssystem $Ax = b$ für unsymmetrische Matrizen $A \in \mathbb{C}^{n \times n}$ zu konstruieren. Sei wieder $x^0 \in \mathbb{C}^n$ gegeben mit $r^0 = b - Ax^0$ und setze

$$v_1 = w_1 = r^0 / \beta, \quad \beta = \|r^0\|_2.$$

Wir suchen nun eine Näherung x^k mit

$$\begin{aligned} x^k &\in x^0 + \mathcal{K}_k(A, v_1), \\ r^k &\perp \mathcal{K}_k(A^*, w_1). \end{aligned}$$

Mit Hilfe der biorthogonalen Basisvektoren $V_k = [v_1, \dots, v_k]$, $W_k = [w_1, \dots, w_k]$ und der Tridiagonalmatrix T_k aus dem Bi-Lanczos-Prozess erhalten wir

$$\begin{aligned} x^k &= x^0 + V_k(W_k^*AV_k)^{-1}W_k^*r^0 \\ &= x^0 + V_kT_k^{-1}(\beta e_1). \end{aligned}$$

Man kann die Tridiagonalform von T_k ausnutzen, um (falls sie existiert) eine LR-Zerlegung $T_k = L_kR_k$ aufzubauen und in jedem Schritt zu aktualisieren. Dazu definieren wir die Matrizen $P_k := V_kR_k^{-1}$ und $\tilde{P}_k := W_k(L_k^*)^{-1}$ sowie den Vektor $z_k := L_k^{-1}(\beta e_1)$. Dann ist

$$x^k = x^0 + P_k z_k.$$

Die Spalten von P_k und \tilde{P}_k sind dabei bi- A -orthogonal, da

$$\tilde{P}_k^*AP_k = L_k^{-1}W_k^*AV_kR_k^{-1} = L_k^{-1}T_kR_k^{-1} = I,$$

gilt. Analog zum CG-Verfahren betrachtet man daher eine Folge von bi- A -orthogonalen Suchrichtungen, so dass p_0, \dots, p_{k-1} eine Basis von $\mathcal{K}_k(A, r^0)$ und $\tilde{p}_0, \dots, \tilde{p}_{k-1}$ eine Basis von $\mathcal{K}_k(A^*, r^0)$ bilden. Dies führt auf folgenden Algorithmus:

Algorithmus 9.2 Bi-CG-Verfahren

Input: $A, b, x^0, \varepsilon > 0$

- 1: $p^0 = r^0 = \tilde{p}^0 = \tilde{r}^0 = b - Ax^0, k = 0$
- 2: **while** $\|r^k\|_2 > \varepsilon$ **do**
- 3: $k \leftarrow k + 1$
- 4: $\alpha_k = \langle r^{k-1}, \tilde{r}^{k-1} \rangle / \langle Ap_{k-1}, \tilde{p}_{k-1} \rangle$
- 5: $x^k = x^{k-1} + \alpha_k p_{k-1}$
- 6: $r^k = r^{k-1} - \alpha_k Ap_{k-1}$
- 7: $\tilde{r}^k = \tilde{r}^{k-1} - \overline{\alpha_k} A^* \tilde{p}_{k-1}$
- 8: $\beta_k = \langle r^k, \tilde{r}^k \rangle / \langle r^{k-1}, \tilde{r}^{k-1} \rangle$
- 9: $p_k = r^k + \beta_k p_{k-1}$
- 10: $\tilde{p}_k = \tilde{r}^k + \overline{\beta_k} \tilde{p}_{k-1}$

Output: x^k

Die Vektoren p_{k-1}, \tilde{p}_{k-1} entsprechen hier skalierten Spalten von P_k, \tilde{P}_k . Den Vektor \tilde{r}^k nennt man auch *Schattenresiduum*, da $\tilde{x}^k = \tilde{x}^{k-1} + \alpha_k \tilde{p}_{k-1}$ eine Näherung für die Lösung des Gleichungssystems $A^*x = b$ darstellt und $\tilde{r}^k = b - A^*\tilde{x}^k$ gilt.

Durch Induktion beweist man nun ähnlich wie für das CG-Verfahren, dass

$$\langle r^i, \tilde{r}^j \rangle = 0 \quad \text{und} \quad \langle Ap_i, \tilde{p}_j \rangle = 0$$

für alle $i \neq j$ gilt. Nach Konstruktion ist dann

$$\begin{aligned} \text{span}\{p_0, \dots, p_{k-1}\} &= \text{span}\{r^0, \dots, r^{k-1}\} = \mathcal{K}_k(A, r^0), \\ \text{span}\{\tilde{p}_0, \dots, \tilde{p}_{k-1}\} &= \text{span}\{\tilde{r}^0, \dots, \tilde{r}^{k-1}\} = \mathcal{K}_k(A^*, r^0). \end{aligned}$$

Wie im Bi-Lanczos-Prozess kann es passieren, dass die Residuen r^k und \tilde{r}^k orthogonal sind, ohne dass eines von beiden verschwindet. Da nicht pivotisiert wird, ist außerdem nicht garantiert, dass eine LR-Zerlegung existiert (ein *Abbruch zweiter Art*); in diesem Fall ist $\langle Ap^k, \tilde{p}^k \rangle = 0$. Beide Fälle können durch „look-ahead“-ähnliche Blockstrategien abgefangen werden. Außerdem weisen die Näherungen im Bi-CG-Verfahren keine Minimierungseigenschaften auf, so dass die Konvergenz sehr irregulär (und insbesondere nicht-monoton) sein kann.

9.3 DAS CGS-VERFAHREN

Jeder Schritt im Bi-CG-Verfahren benötigt jeweils eine Multiplikation mit A und A^* . Letzteres kann in der Praxis problematisch sein, da A oft nicht als explizite Matrix, sondern als Prozedur für die Berechnung von Ax gegeben ist, eine entsprechende Prozedur für A^*x aber nicht zur Verfügung steht.¹

Tatsächlich werden die Vektoren \tilde{r}^k und \tilde{p}^k für die Berechnung der neuen Näherung x^{k+1} nicht direkt benötigt, sondern tauchen nur in Skalarprodukten auf. Genauer betrachtet, erkennt man in Algorithmus 9.2, dass r^k und \tilde{r}^k sowie p^k und \tilde{p}^k die selbe Rekursion erfüllen: Für geeignete Polynome $\varphi_k(t), \psi_k(t) \in \mathcal{P}_k$ mit $\varphi_k(0) = \psi_k(0) = 1$ gilt wegen $r^0 = p^0$:

$$\begin{aligned} r^k &= \varphi_k(A)r^0, & \tilde{r}^k &= \overline{\varphi_k}(A^*)\tilde{r}^0, \\ p^k &= \psi_k(A)r^0, & \tilde{p}^k &= \overline{\psi_k}(A^*)\tilde{r}^0. \end{aligned}$$

Da $\overline{q}(A^*) = q(A)^*$ für jedes Polynom q gilt, erhalten wir

$$\langle r^k, \tilde{r}^k \rangle = \langle \varphi_k^2(A)r^0, \tilde{r}^0 \rangle, \quad \langle Ap^k, \tilde{p}^k \rangle = \langle A\psi_k^2(A)r^0, \tilde{p}^0 \rangle.$$

¹Ein häufiges Beispiel ist das Newton-Verfahren für eine komplizierte Funktion $F: \mathbb{R}^n \rightarrow \mathbb{R}^m$, für die die Jacobi-Matrix $\nabla F(x)$ nicht explizit bekannt ist, aber die Richtungsableitung $\nabla F(x)v$ in Richtung v durch den Differenzenquotienten $(F(x + \varepsilon v) - F(x))/\varepsilon$ angenähert werden kann.

Wenn wir also direkt Rekursionen für $\varphi_k^2(A)r^0$ und $\psi_k^2(A)r^0$ herleiten können, sind keine Multiplikationen mit A^* erforderlich. Diese können wir durch einfache algebraische Manipulationen aus Algorithmus 9.2 gewinnen. Aus Schritt 6 und 9 erhalten wir (nach Einsetzen und Division durch r^0 auf beiden Seiten)

$$(9.3) \quad \varphi_k = \varphi_{k-1} - \alpha_k A \psi_{k-1},$$

$$(9.4) \quad \psi_k = \varphi_k + \beta_k \psi_{k-1},$$

wobei wir hier und im folgenden der Übersichtlichkeit halber die Argumente der Polynome weglassen. Quadrieren beider Gleichungen ergibt, da Matrixpolynome mit gleichem Argument kommutieren,

$$\begin{aligned} \varphi_k^2 &= \varphi_{k-1}^2 - 2\alpha_k A \psi_{k-1} \varphi_{k-1} + \alpha_k^2 A^2 \psi_{k-1}^2, \\ \psi_k^2 &= \varphi_k^2 + 2\beta_k \psi_{k-1} \varphi_k + \beta_k^2 \psi_{k-1}^2. \end{aligned}$$

Der Trick besteht nun darin, dass wir auch für einen der gemischten Terme eine Rekursion einführen. Multiplizieren wir (9.3) mit ψ_{k-1} , erhalten wir

$$\psi_{k-1} \varphi_k = \psi_{k-1} \varphi_{k-1} - \alpha_k A \psi_{k-1}^2.$$

Damit können wir den verbleibenden gemischten Term berechnen: Multiplikation von (9.4) mit φ_k und Indexverschiebung ergibt

$$\psi_{k-1} \varphi_{k-1} = \varphi_{k-1}^2 + \beta_{k-1} \psi_{k-2} \varphi_{k-1}.$$

Wir definieren nun

$$\begin{aligned} r^k &:= \varphi_k^2(A)r^0, & q^k &:= \psi_{k-1}(A)\varphi_k(A)r^0, \\ p^k &:= \psi_k^2(A)r^0, & u^k &:= \psi_k(A)\varphi_k(A)r^0 \end{aligned}$$

und erhalten die Rekursion

$$\begin{aligned} r^k &= r^{k-1} - \alpha_k A (2u^{k-1} - \alpha_k A p^{k-1}), \\ q^k &= u^{k-1} - \alpha_k A p^{k-1}, \\ p^k &= r^k + \beta_k (2q^k + \beta_k p^{k-1}), \\ u^k &= r^k + \beta_k q^k. \end{aligned}$$

Durch eine andere Anordnung der Schritte kann man die doppelte Berechnung von Termen vermeiden, und erhält dadurch den *CGS-Algorithmus* (Englisch: „conjugate gradient squared“).

Algorithmus 9.3 CGS-Verfahren

Input: $A, b, x^0, \varepsilon > 0$

- 1: $\tilde{r} = u^0 = p^0 = r^0 = b - Ax^0, k = 0$
- 2: **while** $\|r^k\|_2 > \varepsilon$ **do**
- 3: $k \leftarrow k + 1$
- 4: $\alpha_k = \langle r^{k-1}, \tilde{r} \rangle / \langle Ap^{k-1}, \tilde{r} \rangle$
- 5: $q^k = u^{k-1} - \alpha_k Ap^{k-1}$
- 6: $x^k = x^{k-1} + \alpha_k(u^{k-1} + q^k)$
- 7: $r^k = r^{k-1} - \alpha_k A(u^{k-1} + q^k)$
- 8: $\beta_k = \langle r^k, \tilde{r} \rangle / \langle r^{k-1}, \tilde{r} \rangle$
- 9: $u^k = r^k + \beta_k q^k$
- 10: $p^k = u^k + \beta_k(q^k + \beta_k p^{k-1})$

Output: x^k

Beachten Sie, dass nun r^k und p^k a priori nicht mehr gleich dem (bi- A -orthogonalisierten) Residuum sind. Aus Schritt 6 und 7 folgt aber, dass $r^k - r^{k-1} = A(x^{k-1} - x^k)$ gilt. Aufgrund der Wahl $r^0 = b - Ax^0$ ist also nach Induktion $r^k = b - Ax^k$ für alle k , und mit $r^k \rightarrow 0$ konvergieren die x^k gegen die gesuchte Lösung von $Ax = b$. Dies ist insbesondere wichtig, da die Konvergenz wie im Bi-CG-Verfahren nicht-monoton ist.

In jedem Schritt müssen im CGS-Algorithmus nur zwei Vektoren mit A multipliziert werden. In der Regel kann man eine doppelt so schnelle Konvergenz wie im Bi-CG-Verfahren erwarten (auf Kosten der gleichzeitigen Iteration für Näherungslösungen von $A^* \tilde{x} = b$). Allerdings ist es möglich, dass durch das Quadrieren der Polynome Rundungsfehler verstärkt werden, was zu erheblichen Problemen durch Auslöschung führen kann. Dies wird im *Bi-CGStab-Verfahren* (Englisch: „biconjugate gradient stabilized“) vermieden.

9.4 DAS BI-CGSTAB-VERFAHREN

Ausgangspunkt ist einmal mehr die Galerkin-Bedingung $r^k \perp \mathcal{K}_k(A^*, r^0)$, wobei $r^k = b - Ax^k = \varphi_k(A)r^0$ für ein Polynom φ_k vom Höchstgrad k mit $\varphi_k(0) = 1$ ist. Sei nun $\chi_j, j = 0, \dots$, eine Folge von Polynomen vom Grad j (und damit linear unabhängig) mit $\chi_j(0) = 1$. Dann kann die Galerkin-Bedingung formuliert werden als

$$0 = \langle \varphi_k(A)r^0, \bar{\chi}_j(A^*)r^0 \rangle = \langle \chi_j(A)\varphi_k(A)r^0, r^0 \rangle \quad \text{für alle } 0 \leq j < k.$$

Das Bi-CG-Verfahren beruht auf der Wahl $\chi_j = \varphi_j$, und dies wurde in der Herleitung des CGS-Verfahrens ausgenutzt. Wir können die χ_j aber auch anders wählen, um eine stabilere Iteration zu erreichen. Der Grundgedanke im Bi-CGStab-Verfahren ist der Ansatz

$$\chi_j(t) = (1 - \omega_j t) \cdots (1 - \omega_1 t) = (1 - \omega_j t) \chi_{j-1}(t),$$

wobei die Konstanten ω_k jeweils so gewählt werden, dass die Norm von

$$r^k := \chi_k(A)\varphi_k(A)r^0$$

minimal ist.

Ausgehend von der Rekursion (9.3) und (9.4), leitet man nun wie im CGS-Verfahren Rekursionen für die Berechnung von r^k und

$$p^k := \chi_k(A)\psi_k(A)r^0$$

her, und erhält den Bi-CGStab-Algorithmus:²

Algorithmus 9.4 Bi-CGStab-Verfahren

Input: $A, b, \chi^0, \varepsilon > 0$

- 1: $\tilde{r} = p^0 = r^0 = b - A\chi^0, k = 0$
- 2: **while** $\|r^k\|_2 > \varepsilon$ **do**
- 3: $k \leftarrow k + 1$
- 4: $\alpha_k = \langle r^{k-1}, \tilde{r} \rangle / \langle Ap^{k-1}, \tilde{r} \rangle$
- 5: $s^k = r^{k-1} - \alpha_k Ap^{k-1}$
- 6: $\omega_k = \langle s^k, As^k \rangle / \langle As^k, As^k \rangle$
- 7: $\chi^k = \chi^{k-1} + \alpha_k p^{k-1} + \omega_k s^k$
- 8: $r^k = s^k - \omega_k As^k$
- 9: $\beta_k = (\alpha_k \langle r^k, \tilde{r} \rangle) / (\omega_k \langle r^{k-1}, \tilde{r} \rangle)$
- 10: $p^k = r^k + \beta_k (p^{k-1} - \omega_k Ap^{k-1})$

Output: χ^k

Wie bei allen Verfahren in diesem Kapitel muss eine praktische Implementierung auch einen „serious breakdown“ berücksichtigen, und zum Beispiel mit einem anderen \tilde{r} neu starten oder „look-ahead“-Strategien verwenden.

²Publiziert durch [Henk van der Vorst](#) in [[Vorst 1992](#)]. Diese Arbeit war zwischen 1990 und 2000 die meistzitierte Veröffentlichung in der angewandten Mathematik. Nebenbei fertigt van der Vorst [Linolschnitt-Porträts seiner Kollegen](#) an.

Teil III

EIGENWERTPROBLEME

VEKTOR- UND QR-ITERATION

Wir betrachten nun das *Eigenwertproblem*: Zu gegebener Matrix $A \in \mathbb{C}^{n \times n}$ suchen wir einen Eigenwert $\lambda \in \mathbb{C}$ und einen zugehörigen Eigenvektor $v \in \mathbb{C}^n \setminus \{0\}$ mit

$$Av = \lambda v.$$

Da die Eigenwertbestimmung über das charakteristische Polynom auf die Nullstellensuche bei Polynomen führt (und umgekehrt jede Nullstelle eines Polynoms Eigenwert einer geeigneten Matrix ist),¹ kann es für $n > 4$ kein direktes Verfahren für die Eigenwertberechnung geben. Numerisch geht man daher anders vor: Man bestimmt zuerst iterativ eine Näherung des gesuchten *Eigenvektors*, und berechnet daraus eine Näherung des zugehörigen Eigenwerts. Das bekannteste Verfahren, um *alle* Eigenvektoren einer Matrix zu berechnen, ist das *QR-Verfahren*. Dies wird auch als Baustein der im nächsten Kapitel vorgestellten Projektionsverfahren auftauchen.

Wir beginnen die Herleitung mit einem einfachen Prototypen, das auch die Grundlage aller weiteren Verfahren dieses Kapitels darstellt.

10.1 VEKTORITERATION

Die *Vektoriteration* (oder *Potenzmethode*) ist ein einfaches Verfahren zur Bestimmung des betragsgrößten Eigenwerts λ_1 und des dazugehörigen Eigenvektors v_1 . Ausgangspunkt ist die folgende Beobachtung: Unter gewissen Voraussetzungen konvergiert für gegebenes $x \in \mathbb{C}^n \setminus \{0\}$ und eine beliebige Norm

$$(10.1) \quad \frac{A^k x}{\|A^k x\|} \rightarrow cv_1, \text{ für } k \rightarrow \infty, c \in \mathbb{C} \setminus \{0\}.$$

¹Tatsächlich ist die Berechnung der Eigenwerte der sogenannten *Begleitmatrix* eins der effizientesten Verfahren zur Bestimmung von Nullstellen eines Polynoms; siehe [Boyd 2013].

Um dies zu zeigen, nehmen wir an, dass $\lambda_1 \in \mathbb{R}$ ein einfacher Eigenwert ist:

$$(10.2) \quad |\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|.$$

Außerdem nehmen wir der Einfachheit an, dass A diagonalisierbar ist. Dann hat \mathbb{C}^n eine Basis $\{v_1, \dots, v_n\}$ aus Eigenvektoren von A und wir können daher jedes $x \in \mathbb{C}^n$ schreiben als

$$x = \xi_1 v_1 + \xi_2 v_2 + \dots + \xi_n v_n.$$

Damit ist

$$\begin{aligned} A^k x &= \xi_1 A^k v_1 + \xi_2 A^k v_2 + \dots + \xi_n A^k v_n \\ &= \xi_1 \lambda_1^k v_1 + \xi_2 \lambda_2^k v_2 + \dots + \xi_n \lambda_n^k v_n \\ &= \lambda_1^k \left[\xi_1 v_1 + \xi_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k v_2 + \dots + \xi_n \left(\frac{\lambda_n}{\lambda_1} \right)^k v_n \right]. \end{aligned}$$

Wegen (10.2) gilt

$$1 > \frac{|\lambda_2|}{|\lambda_1|} \geq \dots \geq \frac{|\lambda_n|}{|\lambda_1|},$$

die entsprechenden Terme konvergieren also für $k \rightarrow \infty$ gegen Null. Falls nun $\xi_1 \neq 0$ ist, so gilt für $\lambda_1 > 0$

$$(10.3) \quad \frac{A^k x}{\|A^k x\|} = \frac{\lambda_1^k \left[\xi_1 v_1 + \mathcal{O} \left(\left| \frac{\lambda_2}{\lambda_1} \right|^k \right) \right]}{|\lambda_1^k| \left\| \xi_1 v_1 + \mathcal{O} \left(\left| \frac{\lambda_2}{\lambda_1} \right|^k \right) \right\|} \xrightarrow{k \rightarrow \infty} \tilde{v}_1 := \frac{v_1}{\|v_1\|}.$$

(Ist $\lambda_1 < 0$, so existieren zwei Teilfolgen, von denen eine gegen \tilde{v}_1 , die andere gegen $-\tilde{v}_1$ konvergiert.)

Die Vektoriteration liefert also für passendes² $x^0 \in \mathbb{C}^n$ eine Approximation $x^k = \frac{A^k x^0}{\|A^k x^0\|}$ von v_1 . Um daraus eine Näherung des zugehörigen Eigenwerts λ_1 zu bestimmen, kann man folgenden Ansatz nutzen: zu gegebenem x^k suchen wir diejenige Zahl λ , die das Residuum der Eigenwertgleichung

$$\|x^k \lambda - A x^k\|_2^2$$

minimiert. Dies ist ein lineares Ausgleichsproblem für λ mit der "Matrix" $x^k \in \mathbb{C}^{n \times 1}$ und der rechten Seite $A x^k \in \mathbb{C}^n$. Die Normalgleichungen dazu sind

$$\langle x^k, x^k \rangle \lambda = \langle A x^k, x^k \rangle.$$

²Ein zufällig gewählter Vektor ungleich 0 wird in der Regel eine Komponente in v_1 -Richtung haben. Ist dies nicht der Fall, erzeugen Rundungsfehler während der Iteration einen solchen Anteil.

Da x^k nach Konstruktion die Norm $\|x^k\| = 1$ hat, ist $x^k \neq 0$. Wir können daher nach λ auflösen und erhalten den *Rayleigh-Quotienten*³

$$(10.4) \quad \lambda^k = \frac{\langle Ax^k, x^k \rangle}{\langle x^k, x^k \rangle}.$$

Wählt man die euklidische Norm in (10.1), ergibt das die *Vektor- oder von-Mises-Iteration*⁴:

Algorithmus 10.1 Vektoriteration

Input: $A, x^0 \neq 0, \varepsilon, k = 0, \lambda^0 = 0$

1: **while** $\|\lambda^k x^k - Ax^k\|_2 > \varepsilon$ **do**

2: $k \leftarrow k + 1$

3: $w \leftarrow Ax^{k-1}$

4: $x^k \leftarrow w / \|w\|_2$

5: $\lambda^k \leftarrow \langle Ax^k, x^k \rangle$

Output: x^k, λ^k

Für die Konvergenz des Verfahrens folgt also aus den obigen Überlegungen und der Stetigkeit des Skalarprodukts.⁵

Satz 10.1. *Ist $A \in \mathbb{R}^{n \times n}$ diagonalisierbar, gilt $|\lambda_1| > |\lambda_2|$ und $\langle x^0, v_1 \rangle \neq 0$, so folgt für die Iterierten der Vektoriteration:*

$$\begin{aligned} \|\pm x^k - v_1\|_2 &= \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right), \\ \|\lambda^k - \lambda_1\|_2 &= \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right). \end{aligned}$$

Ist A symmetrisch, gilt sogar

$$\|\lambda^k - \lambda_1\|_2 = \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}\right).$$

³John William Strutt, dritter Baron Rayleigh (1842-1919), ist in der zweiten Hälfte des 19. Jahrhunderts durch seine Beiträge zu fast allen Bereichen der Physik, insbesondere zu Schwingungsphänomenen, berühmt geworden.

⁴Vorgeschlagen 1929 durch Richard von Mises, der in der ersten Hälfte des 20. Jahrhunderts zur angewandten Mathematik (etwa im Bereich der Flugzeugkonstruktion) viel beigetragen hat.

⁵Tatsächlich genügt für die Konvergenz, dass es nur einen betragsgrößten Eigenwert gibt, dessen algebraische Vielfachheit gleich der geometrischen Vielfachheit ist. Dies kann man mit Hilfe der Jordan-Normalform zeigen.

10.2 INVERSE VEKTORITERATION

Die Vektoriteration kann auch zur Berechnung weiterer Eigenwerte und Eigenvektoren verwendet werden. Ist nämlich $\lambda \in \sigma(A)$ und $\mu \notin \sigma(A)$, so gilt

$$Ax = \lambda x \Leftrightarrow (A - \mu I)x = (\lambda - \mu)x \Leftrightarrow (A - \mu I)^{-1}x = (\lambda - \mu)^{-1}x.$$

Ist $\lambda - \mu$ also klein genug, dann ist $\frac{1}{\lambda - \mu}$ der betragsgrößte Eigenwert von $(A - \mu I)^{-1}$. Kennen wir also eine gute Schätzung von λ , so können wir die Vektoriteration zur Berechnung des zugehörigen Eigenvektors (und einer verbesserten Näherung des Eigenwerts) einsetzen. Diesen Ansatz nennt man *inverse Vektoriteration mit Spektralverschiebung*⁶. Natürlich wird die Matrix $(A - \mu I)^{-1}$ dabei nicht explizit gebildet, sondern das zugehörige lineare Gleichungssystem gelöst:

Algorithmus 10.2 Inverse Vektoriteration

Input: $\mu, A, x^0 \neq 0, \varepsilon, k = 0, \lambda^0 = 0$

- 1: berechne LR-Zerlegung von $A - \mu I$
- 2: **while** $\|\lambda^k x^k - Ax^k\|_2 > \varepsilon$ **do**
- 3: $k \leftarrow k + 1$
- 4: löse $(A - \mu I)w = x^{k-1}$ durch Vorwärts-/Rückwärtssubstitution
- 5: $x^k \leftarrow w / \|w\|_2$
- 6: $\lambda^k \leftarrow (\langle w, x^{k-1} \rangle)^{-1} + \mu$

Output: x^k, λ^k

Die Konvergenz der Iteration hängt vom Abstand von μ zum nächsten benachbarten Eigenwert ab. Je näher μ am gewünschten Eigenwert λ_i , und je weiter weg vom nächstgelegenen Eigenwert λ_j , desto schneller konvergiert die Iteration. Präziser wird der Fehler in jeder Iteration um den Faktor

$$\max_{j \neq i} \frac{|\lambda_i - \mu|}{|\lambda_j - \mu|}$$

reduziert. Die Konvergenzgeschwindigkeit kann wesentlich gesteigert werden, indem in jedem Iterationsschritt der Rayleigh-Quotient als Verschiebung $\mu = \lambda^{k-1}$ verwendet wird (*Rayleigh-Iteration*). Allerdings steigt dadurch der Rechenaufwand stark an, da die LR-Zerlegung von $(A - \lambda^{k-1}I)$ in jedem Schritt neu durchgeführt werden muss.

Um weitere Eigenwerte und Eigenvektoren zu bestimmen, kann man eine *Deflation* durchführen. Sind $\lambda_1, \dots, \lambda_n$ die Eigenwerte von A mit zugehörigen Eigenvektoren v_1, \dots, v_n , so

⁶Helmut Wielandt – eigentlich ein reiner Mathematiker – entwickelte diese Methode 1944 während seiner Arbeit an der Aerodynamischen Versuchsanstalt Göttingen.

hat die Matrix

$$\tilde{A} = A - \lambda_1 \frac{v_1 v_1^*}{\langle v_1, v_1 \rangle}$$

die Eigenwerte $0, \lambda_2, \dots, \lambda_n$ mit (falls A normal ist, d. h. die v_i orthogonal sind), zugehörigen Eigenvektoren v_1, v_2, \dots, v_n . Man kann also, beginnend mit dem betragsgrößten (oder betragskleinsten) sämtliche Eigenwerte und Eigenvektoren durch Vektoriteration (bzw. inverse Iteration) der Reihe nach berechnen. In der Praxis ist dieses Verfahren jedoch zu aufwendig, falls mehr als die ersten (bzw. letzten) paar Eigenwerte gesucht sind.

10.3 DAS QR-VERFAHREN

Ein sehr elegantes Verfahren, sämtliche Eigenwerte und Eigenvektoren (bzw. allgemeiner, invarianter Unterräume) einer Matrix zu berechnen, ist das *QR-Verfahren*.⁷ Dabei wird im Wesentlichen mit Hilfe der Vektoriteration und der QR-Zerlegung die Schur-Normalform einer Matrix (näherungsweise) konstruiert. Der Algorithmus selber ist äußerst einfach:

Algorithmus 10.3 QR-Verfahren

Input: A, m^*

1: $A^{(0)} \leftarrow A$

2: **for** $m = 1$ to m^* **do**

3: $Q^{(m)}, R^{(m)} \leftarrow A^{(m-1)}$

▷ QR-Zerlegung von $A^{(m-1)}$

4: $A^{(m)} \leftarrow R^{(m)} Q^{(m)}$

Output: $A^{(m)}, Q^{(m)}$

Die Iterierten $A^{(m)}$ im QR-Verfahren konvergieren (unter bestimmten Voraussetzungen) gegen eine obere Dreiecksmatrix, deren Diagonalelemente die Eigenwerte von A sind, und das Produkt der $Q^{(m)}$ konvergiert gegen eine unitäre Matrix, deren Spalten die zugehörigen Schurvektoren (für normale Matrizen also die Eigenvektoren) sind.

Wir nähern uns der Konstruktion dieses Verfahrens in mehreren Schritten. Wir nehmen dabei der Bequemlichkeit halber an, dass A normal ist und

$$(10.5) \quad |\lambda_1| > |\lambda_2| > \dots > |\lambda_n|$$

⁷In der heute verwendeten Form geht das Verfahren auf die unabhängigen Veröffentlichungen von [Wera Nikolajewna Kublanowskaja](#) (1961) und [John G. F. Francis](#) (1961/62, inklusive impliziter Shifts) zurück. Der Kerngedanke, eine Matrixzerlegung für die Eigenwertberechnung einzusetzen, stammt von [Heinz Rutishauser](#), der 1954 eine Rekursion für die Eigenwertberechnung basierend auf Nullstellen charakteristischer Polynome (den *qd-Algorithmus*) herleitete, und erkannte, dass diese in Matrixschreibweise genau dem *LR-Verfahren* $L^{(k)} R^{(k)} = A^{(k)}, A^{(k+1)} = R^{(k)} L^{(k)}$ entspricht. Die Details sind nachzulesen in [[Gutknecht und Parlett 2011](#)].

gilt, d. h. die zugehörigen Eigenvektoren v_1, v_2, \dots, v_n eine orthonormale Basis des \mathbb{C}^n bilden. (Ansonsten ist überall „Eigenvektor“ durch „Schurvektor“ und „Spann der Eigenvektoren“ durch „invarianter Unterraum“ zu ersetzen.)

SCHRITT 1: (Unterraumiteration)

Die Vektoriteration liefert für ein x mit $\langle x, v_1 \rangle \neq 0$ eine Folge $A^k x$, die gegen cv_1 für $c \in \mathbb{C}$ konvergiert, d.h. ein Element aus dem von v_1 aufgespannten Unterraum. Ein naheliegender Ansatz für die Berechnung mehrerer Eigenvektoren ist also, die Vektoriteration für einen Unterraum mit einer größeren Dimension $1 \leq k \leq n$ durchzuführen. Dafür wählen wir k linear unabhängige Vektoren x_1, \dots, x_k und setzen

$$\begin{aligned} S_k &= \text{span}\{x_1, \dots, x_k\}, \\ T_k &= \text{span}\{v_1, \dots, v_k\}, \\ U_k &= \text{span}\{v_{k+1}, \dots, v_n\}. \end{aligned}$$

Mit $A^m S_k = \{A^m x : x \in S_k\}$ kann man dann zeigen, dass

$$A^m S_k \longrightarrow T_k \quad \text{für } m \rightarrow \infty,$$

falls $S_k \cap U_k = \{0\}$ gilt. Sei dafür $x \in S_k$ beliebig, und schreibe.

$$x = (\xi_1 v_1 + \dots + \xi_k v_k) + (\xi_{k+1} v_{k+1} + \dots + \xi_n v_n)$$

Dann gilt

$$\begin{aligned} A^m x &= (\xi_1 \lambda_1^m v_1 + \dots + \xi_k \lambda_k^m v_k) + (\xi_{k+1} \lambda_{k+1}^m v_{k+1} + \dots + \xi_n \lambda_n^m v_n) \\ &= \lambda_k^m \left[\left(\xi_1 \left(\frac{\lambda_1}{\lambda_k} \right)^m v_1 + \dots + \xi_k v_k \right) \right. \\ &\quad \left. + \left(\xi_{k+1} \left(\frac{\lambda_{k+1}}{\lambda_k} \right)^m v_{k+1} + \dots + \xi_n \left(\frac{\lambda_n}{\lambda_k} \right)^m v_n \right) \right]. \end{aligned}$$

Wegen (10.5) sind sämtliche Brüche in der ersten Klammer größer als Eins, und sämtliche Brüche in der zweiten Klammer kleiner als Eins. Für $m \rightarrow \infty$ konvergiert die zweite Klammer also gegen 0. Es gilt also

$$A^m x \longrightarrow v \in T_k.$$

Da S_k und T_k die gleiche Dimension haben und wegen $x \notin U_k$ die erste Klammer nicht verschwindet, muss also $A^m S_k$ gegen T_k konvergieren.

SCHRITT 2: (*Basisiteration*)

Für die praktische Durchführung der Unterraumiteration wählt man eine Basis $\{x_1, \dots, x_k\}$ und berechnet $\{A^m x_1, \dots, A^m x_k\}$. Allerdings gilt wegen (10.3), dass $A^m x_i$ für (fast) alle x_i gegen v_1 konvergiert;⁸ diese Basis wird also mit fortschreitender Iteration immer schlechter konditioniert sein. Außerdem ist eine Normalisierung nötig, damit die Iterierten beschränkt bleiben. Deshalb wird in jedem Schritt eine neue Orthonormalbasis $\{q_1^{(m)}, \dots, q_k^{(m)}\}$ von $A^m S_k$ bestimmt. Der Name des QR-Verfahrens stammt nun daher, dass dies durch eine QR-Zerlegung geschehen kann. Seien a_1, \dots, a_n die Spalten von A , und q_1, \dots, q_n die (orthonormalen) Spalten von Q , so folgt nämlich aus $A = QR$:

$$\begin{aligned} a_1 &= q_1 r_{11} && \Rightarrow && \text{span}\{a_1\} = \text{span}\{q_1\}, \\ a_2 &= q_1 r_{12} + q_2 r_{22} && \Rightarrow && \text{span}\{a_1, a_2\} = \text{span}\{q_1, q_2\}, \\ &\vdots && && \vdots \\ a_n &= q_1 r_{1n} + \dots + q_n r_{nn} && \Rightarrow && \text{span}\{a_1, a_2, \dots, a_n\} = \text{span}\{q_1, q_2, \dots, q_n\}. \end{aligned}$$

Um alle Eigenvektoren zu bestimmen, führen wir also jeweils einen Basisiterationsschritt auf den n orthonormalen Vektoren $q_1^{(0)}, \dots, q_n^{(0)}$ durch, und berechnen dann durch QR-Zerlegung eine neue orthonormale Basis. Dann gilt für alle $1 \leq k \leq n$:

$$S_k^{(m)} := \text{span}\{q_1^{(m)}, \dots, q_k^{(m)}\} = A^m S_k \longrightarrow T_k.$$

Da die Spalten von Q orthogonal sind, ist $q_k^{(m)} \in A^m S_k$ orthogonal zu $A^m S_{k-1}$ für alle $2 \leq k \leq n$. Somit konvergiert $q_1^{(m)} \in A^m S_1$ gegen den (einzig) Basisvektor v_1 von T_1 ; damit konvergiert $q_2^{(m)} \in A^m S_2$ gegen den zu v_1 orthogonalen Basisvektor v_2 von T_2 ; und so weiter. In anderen Worten, die Spalten $q_i^{(m)}$, $1 \leq i \leq n$, konvergieren simultan gegen die entsprechenden Eigenvektoren von A .

Wählen wir also eine Matrix $Q^{(0)}$ mit orthonormalen Spalten (etwa die Identität), so lassen sich die obigen Überlegungen mit Hilfe der QR-Zerlegung in Matrixform schreiben:

$$(10.6) \quad \begin{cases} S^{(m)} = A Q^{(m-1)} \\ Q^{(m)} R^{(m)} = S^{(m)} \end{cases}$$

Die Spalten der Matrizen $Q^{(m)}$ konvergieren dann gegen eine orthonormale Basis aus Eigenvektoren.

⁸wenn auch nicht mit der gleichen Geschwindigkeit

SCHRITT 3: (Schur-Normalform)

Wir wollen nun aus den näherungsweise Eigenvektoren in $Q^{(m)}$ die gesuchte Näherung der Eigenwerte von A konstruieren. Dazu verwenden wir eine iterative Approximation der Schur-Normalform (3.3):

$$(10.7) \quad A^{(m)} := (Q^{(m)})^* A Q^{(m)}$$

konvergiert gegen eine obere Dreiecksmatrix, auf deren Diagonalen die Eigenwerte von A stehen. Um das zu zeigen, betrachten wir (10.7) spaltenweise. Bezeichne die j -te Spalte $q_j^{(m)}$ von $Q^{(m)}$. Da $q_j^{(m)}$ im Unterraum $A^m S_j$ liegt, welcher gegen T_j konvergiert, können wir diesen Vektor schreiben als

$$q_j^{(m)} = \sum_{k=1}^j c_k v_k + \varepsilon_1^{(m)},$$

mit einer Folge $\varepsilon_1^{(m)} \rightarrow 0 \in \mathbb{C}^n$ für $m \rightarrow \infty$. Dann gilt:

$$A q_j^{(m)} = \sum_{k=1}^j c_k \lambda_k v_k + \varepsilon_2^{(m)},$$

mit einer neuen Nullfolge $\varepsilon_2^{(m)} \in \mathbb{C}^n$. Da $A^m S_j$ gegen T_j konvergiert, konvergieren die $q_i^{(m)}$, $1 \leq i \leq j$, gegen eine Basis von T_j . Es lässt sich also umgekehrt jeder Eigenvektor $v_k \in T_j$ darstellen als

$$v_k = \sum_{i=1}^j d_{k,i}^{(m)} q_i^{(m)} + \varepsilon_3^{(m)},$$

wobei wieder $\varepsilon_3^{(m)} \rightarrow 0$ mit $m \rightarrow \infty$ konvergiert. Zusammen gilt also

$$A q_j^{(m)} = \sum_{k=1}^j \alpha_k^{(m)} q_k^{(m)} + \varepsilon^{(m)},$$

wobei wir alle auftretenden Nullfolgen in $\varepsilon^{(m)}$ und alle skalaren Koeffizienten in $\alpha_k^{(m)}$ gesammelt haben. Da die $q_k^{(m)}$ nach Konstruktion orthogonal sind, folgt für $i > j$ aus der Stetigkeit des Skalarprodukts, dass

$$\langle A q_j^{(m)}, q_i^{(m)} \rangle = \sum_{k=1}^j \alpha_k^{(m)} \underbrace{\langle q_k^{(m)}, q_i^{(m)} \rangle}_{=0} + \langle \varepsilon^{(m)}, q_i^{(m)} \rangle = \langle \varepsilon^{(m)}, q_i^{(m)} \rangle \rightarrow 0.$$

Die Subdiagonaleinträge von $(Q^{(m)})^* A Q^{(m)}$ konvergieren also gegen Null. Weiterhin sind die Diagonalelemente genau die Rayleigh-Quotienten zu den $q^{(m)}$, die wie in der Vektoriteration gegen die entsprechenden Eigenwerte von A konvergieren.

SCHRITT 4: (QR-Algorithmus)

Der Algorithmus 10.3 konstruiert genau die in (10.7) definierten Matrizen $A^{(m)}$ für $Q^{(0)} = I$. Dies zeigt man durch Induktion nach m : Der Induktionsanfang für $m = 0$ ist klar. Bezeichne nun $\tilde{R}^{(m)}, \tilde{Q}^{(m)}, \tilde{A}^{(m)}$ die Matrizen aus dem QR-Verfahren, und $R^{(m)}, Q^{(m)}, A^{(m)}$ die entsprechenden Matrizen aus (10.6) und (10.7). Einsetzen von (10.6) für $m + 1$ in (10.7) ergibt dann

$$A^{(m)} = (Q^{(m)})^* A Q^{(m)} = [(Q^{(m)})^* Q^{(m+1)}] R^{(m+1)} =: QR.$$

Da die QR-Zerlegung (nach Festlegen des Vorzeichens) eindeutig ist, muss nach Induktionsvoraussetzung QR auch die QR-Zerlegung von $\tilde{A}^{(m)}$ sein. Es folgt also aus Schritt 4 im QR-Algorithmus 10.3:

$$\begin{aligned} \tilde{A}^{(m+1)} &= RQ = R^{(m+1)} [(Q^{(m)})^* Q^{(m+1)}] \\ &= ((Q^{(m+1)})^* A Q^{(m)}) (Q^{(m)})^* Q^{(m+1)} \\ &= (Q^{(m+1)})^* A Q^{(m+1)} = A^{(m+1)}, \end{aligned}$$

wobei wir in der letzten Zeile die Gleichung $Q^{(m)} R^{(m)} = A Q^{(m-1)}$ aus (10.6), aufgelöst nach $R^{(m)}$, verwendet haben.

Wir fassen zusammen:

Satz 10.2. Sei $A \in \mathbb{C}^{n \times n}$ normal, gelte für die Eigenwerte von A , dass $|\lambda_1| > |\lambda_2| > \dots > |\lambda_n|$ und für die Eigenvektoren

$$(10.8) \quad \text{span}\{e_1, \dots, e_k\} \cap \text{span}\{v_{k+1}, \dots, v_n\} = \{0\} \quad \text{für alle } 1 \leq k < n,$$

so konvergieren die Iterierten $A^{(m)}$ im QR-Verfahren gegen eine obere Dreiecksmatrix, deren Diagonalelemente die Eigenwerte von A sind, und die Spalten von $Q^{(m)}$ konvergieren gegen die Eigenvektoren von A .

Bedingung (10.8) entspricht dabei der Voraussetzung in der Unterraumiteration, dass $S_k \cap U_k = \{0\}$ gilt. Sie ist in der Praxis so gut wie immer erfüllt. Aus der Herleitung ist auch ersichtlich, dass die Einträge auf der Diagonalen von $A^{(m)}$ nach ihrem Betrag sortiert sind.

Bemerkung. Für $A \in \mathbb{R}^{n \times n}$ sind auch die Näherungen $A^{(m)}$ reell; Satz 10.2 gilt aber auch in diesem Fall, da wegen der Annahme (10.5) die Matrix A keine komplex konjugierten Eigenwerte hat. Andernfalls kann man analog über die reelle Schur-Normalform zeigen, dass $A^{(m)}$ gegen die obere Block-Dreiecksmatrix in (3.3) konvergiert. Ist $A \in \mathbb{C}^{n \times n}$ selbstadjungiert, konvergiert $A^{(m)}$ gegen eine (reelle) Diagonalmatrix.

10.4 PRAKTISCHE DURCHFÜHRUNG DES QR-VERFAHRENS

Das QR-Verfahren in der Form von Algorithmus 10.3 hat folgende Nachteile:

1. In jedem Schritt ist eine QR-Zerlegung notwendig, die $\mathcal{O}(n^3)$ Operationen benötigt.
2. Die Konvergenz hängt von dem Maximum über alle $\frac{|\lambda_{j+1}|}{|\lambda_j|}$, $1 \leq j \leq n-1$, ab. Liegen also zwei Eigenwerte sehr nahe beieinander, kann die Konvergenz (für alle) beliebig langsam sein.

In der Praxis bedient man sich deshalb verschiedener Techniken, um das Verfahren zu beschleunigen. Wir betrachten hier stellvertretend jeweils die einfachsten Varianten.

1. Transformation auf Hessenbergform

Zur Erinnerung: Eine Matrix A hat *obere Hessenbergform*, falls $a_{ij} = 0$ für $i > j + 1$ gilt. In diesem Fall kann man A durch Givens-Rotationen in $\mathcal{O}(n^2)$ Schritten auf obere Dreiecksform bringen (da ja pro Zeile nur ein Element „wegrotiert“ werden muss):

$$G_{n,n-1} \cdots G_{21} A^{(m)} = R^{(m)}.$$

Geht man dabei wie angedeutet von oben nach unten (statt umgekehrt) vor, bleibt im QR-Verfahren

$$A^{(m+1)} = R^{(m)} G_{21}^* \cdots G_{n,n-1}^*$$

die obere Hessenbergform erhalten. Wir betrachten dies anhand eines Beispiels $A \in \mathbb{C}^{4 \times 4}$:

$$\begin{aligned}
 A^{(m)} = \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{pmatrix} &\xrightarrow{G_{43} G_{32} G_{21}} \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{pmatrix} \xrightarrow{\cdot G_{21}^*} \begin{pmatrix} \circledast & \circledast & * & * \\ \circledast & \circledast & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{pmatrix} \\
 &\xrightarrow{\cdot G_{32}^*} \begin{pmatrix} * & \circledast & \circledast & * \\ * & \circledast & \circledast & * \\ 0 & \circledast & \circledast & * \\ 0 & 0 & 0 & * \end{pmatrix} \xrightarrow{\cdot G_{43}^*} \begin{pmatrix} * & * & \circledast & \circledast \\ * & * & \circledast & \circledast \\ 0 & * & \circledast & \circledast \\ 0 & 0 & \circledast & \circledast \end{pmatrix} = A^{(m+1)}.
 \end{aligned}$$

Nutzt man die Assoziativität der Matrixmultiplikation, kann man die berechneten Givens-Rotationen sofort anwenden, ohne sie zwischenzuspeichern:

$$A^{(m+1)} = (G_{n,n-1} (\cdots (G_{21} A^{(m)}) G_{21}^*) \cdots) G_{n,n-1}^*$$

(Beachten Sie, dass die Givens-Rotation $G_{k,k-1}$ weiterhin so berechnet wird, dass der $(k, k-1)$ -te Eintrag von $(G_{k-1,k-2} \cdots G_{21})A^{(m)}$ annulliert wird.)⁹

Um nun eine beliebige Matrix A auf obere Hessenbergform zu bringen (ohne dass sich die Eigenwerte ändern), kann man wieder Givens-Rotationen verwenden:¹⁰ Analog zur QR-Zerlegung eliminiert man Einträge unterhalb der Diagonalen von unten nach oben, wobei man jede Givens-Rotation sofort wieder von rechts anwendet und die nächste Rotation für die so geänderte Matrix berechnet. Ein Beispiel soll dies wieder verdeutlichen:

$$\begin{aligned}
 A = \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{pmatrix} &\xrightarrow{G_{34}} \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ \otimes & \otimes & \otimes & \otimes \\ 0 & \otimes & \otimes & \otimes \end{pmatrix} \xrightarrow{\cdot G_{34}^*} \begin{pmatrix} * & * & \otimes & \otimes \\ * & * & \otimes & \otimes \\ * & * & \otimes & \otimes \\ 0 & * & \otimes & \otimes \end{pmatrix} \\
 &\xrightarrow{G_{23}} \begin{pmatrix} * & * & * & * \\ \otimes & \otimes & \otimes & \otimes \\ 0 & \otimes & \otimes & \otimes \\ 0 & * & * & * \end{pmatrix} \xrightarrow{\cdot G_{23}^*} \begin{pmatrix} * & \otimes & \otimes & * \\ * & \otimes & \otimes & * \\ 0 & \otimes & \otimes & * \\ 0 & \otimes & \otimes & * \end{pmatrix} \\
 &\xrightarrow{G_{34}} \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ 0 & \otimes & \otimes & \otimes \\ 0 & 0 & \otimes & \otimes \end{pmatrix} \xrightarrow{\cdot G_{34}^*} \begin{pmatrix} * & * & \otimes & \otimes \\ * & * & \otimes & \otimes \\ 0 & * & \otimes & \otimes \\ 0 & 0 & \otimes & \otimes \end{pmatrix} := \tilde{A}.
 \end{aligned}$$

Wegen

$$\tilde{A} = (G_{21}(\cdots (G_{n,n-1}A)G_{n,n-1}^* \cdots)G_{21}^* =: Q^*AQ$$

ist \tilde{A} ähnlich zu A , hat also die selben Eigenwerte.

2. Spektralverschiebung

Die Konvergenzgeschwindigkeit des QR-Verfahrens hängt ab vom (kleinsten) Verhältnis der Eigenwerte $\frac{|\lambda_{j+1}|}{|\lambda_j|}$. Haben wir also eine Näherung μ für einen Eigenwert λ_i so, dass

$$|\mu - \lambda_i| \ll |\mu - \lambda_j| \quad \text{für alle } j \neq i$$

gilt, und wenden das QR-Verfahren auf $A - \mu I$ an, dann wird der betragskleinste Eigenwert $\lambda_i - \mu$ von $A - \mu I$ sehr rasch angenähert. Macht man die Spektralverschiebung wieder

⁹Tatsächlich kann $A^{(m)}$ sofort bei Anwendung der berechneten Givens-Rotation von links und rechts überschrieben werden. Dass das funktioniert, ist nicht offensichtlich, und folgt aus Satz 6.4 sowie der linearen Abhängigkeit von Vektoren der Form $G(a, 0)^T$ und $G(b, 0)^T$ für jede Rotation G . Letzteres sorgt dafür, dass die zusätzlich erzeugten Subdiagonaleinträge bei der nächsten Rotation wieder verschwinden.

¹⁰Eine Alternative ist der [Arnoldi-Prozess](#), der ebenfalls eine orthogonale Matrix Q liefert, so dass Q^*AQ obere Hessenbergform hat. (Tatsächlich war dies die ursprüngliche Motivation für den Arnoldi-Prozess.)

rückgängig, erhält man eine bessere Näherung $A^{(m+1)}$:

$$\begin{aligned} Q^{(m)}R^{(m)} &= A^{(m)} - \mu I, \\ A^{(m+1)} &= R^{(m)}Q^{(m)} + \mu I. \end{aligned}$$

Wegen

$$A^{(m+1)} = (Q^{(m)})^*(A^{(m)} - \mu I)Q^{(m)} + \mu I = (Q^{(m)})^*A^{(m)}Q^{(m)}$$

gilt immer noch $\sigma(A^{(m+1)}) = \sigma(A)$. (Die obere Hessenbergform bleibt durch Addition einer Diagonalmatrix natürlich erhalten.)

Wie bei der inversen Vektoriteration kann man nun in jedem Schritt einen neuen Parameter $\mu^{(m)}$ wählen, um die Konvergenzgeschwindigkeit weiter zu verbessern. Da die Diagonaleinträge von $A^{(m)}$ gegen die Eigenwerte von A konvergieren, liegt die Wahl eines Diagonaleintrags für $\mu^{(m)}$ nahe. Nun ist die ursprüngliche Spektralverschiebung so gewählt, dass $\lambda_i - \mu$ der betragskleinste Eigenwert von $A - \mu I$ ist. Weil die Diagonalelemente im QR-Algorithmus betragsmäßig abfallend sortiert werden, strebt der Eintrag $a_{nn}^{(m)} - \mu$ schnell gegen $\lambda_i - \mu$. Man wählt daher sinnvollerweise den *Rayleigh-Shift* $\mu^{(m)} = a_{nn}^{(m)}$.

Diese Verschiebung versagt allerdings, falls der Block

$$\begin{pmatrix} a_{n-1,n-1}^{(m)} & a_{n-1,n}^{(m)} \\ a_{n,n-1}^{(m)} & a_{nn}^{(m)} \end{pmatrix}$$

ein Paar komplex konjugierter (oder allgemeiner, mehrere betragsgleiche) Eigenwerte hat. In diesem Fall berechnet man zum Beispiel die Eigenwerte μ_1, μ_2 dieser 2×2 -Matrix und wählt als Verschiebung denjenigen, der näher an a_{nn} liegt (*Wilkinson-Shift*) – dadurch wird $A^{(m+1)}$ jedoch selbst für reelle A in der Regel komplex. Alternativ führt man hintereinander zwei Schritte des QR-Verfahrens mit Verschiebung μ_1 sowie Verschiebung μ_2 aus. Sind Q_1, R_1 und Q_2, R_2 die im QR-Verfahren berechneten Matrizen zur Verschiebung μ_1 beziehungsweise μ_2 , so erkennt man durch Umformungen der Iterationsvorschriften, dass

$$QR := (Q_1 Q_2)(R_2 R_1) = (A^{(m)} - \mu_1 I)(A^{(m)} - \mu_2 I) =: M$$

die QR-Zerlegung einer reellen (wegen $\mu_2 = \bar{\mu}_1$) Matrix ergibt, und dass

$$A^{(m+2)} = Q^* A^{(m)} Q$$

gilt. Da sowohl Berechnung als auch QR-Zerlegung von M jeweils $\mathcal{O}(n^3)$ Operation benötigen, ist dieser *doppelte Shift* in der Praxis nicht sinnvoll. Es ist aber möglich, $A^{(m+2)}$ zu berechnen, ohne M explizit aufzustellen. Dazu verwendet man Satz 6.4: Solange die erste Spalte von Q' der von Q entspricht, und das Ergebnis obere Hessenbergform hat, ergibt jede unitäre Ähnlichkeitstransformation mit Q' eine Matrix, die (bis auf Vorzeichen der Subdiagonalelemente) gleich $A^{(m+2)}$ ist. Wir gehen also wie folgt vor:

1. Berechne die erste Spalte m_1 von M . Da $A^{(m)}$ echte¹¹ obere Hessenbergform hat, hat m_1 (als erste Spalte des Produkts zweier Hessenbergmatrizen) nur drei Einträge, die man aus den Elementen $a_{ij}^{(m)}$, $1 \leq i \leq j + 1 \leq 3$ sowie μ_1, μ_2 explizit berechnen kann (ohne M aufzustellen).
2. Berechne eine unitäre Matrix Q_0 , so dass $Q_0 m_1 = \sigma e_1$ für $\sigma \in \mathbb{R}$ ist. Für Q_0 kann zum Beispiel ein Produkt von geeigneten Givens-Rotationen oder eine Householder-Reflexion verwendet werden.
3. Bilde $A_0 := Q_0 A^{(m)} Q_0^*$. Da Q_0 nur das zweite und dritte Element von m_1 annulliert, erzeugt dies von Null verschiedene Einträge in der ersten, zweiten und dritten Spalte beziehungsweise Zeile von A_0 . Für $n = 6$ erhält man zum Beispiel

$$A^{(m)} \xrightarrow{Q_0} \begin{pmatrix} \circledast & \circledast & \circledast & \circledast & \circledast & \circledast \\ \circledast & \circledast & \circledast & \circledast & \circledast & \circledast \\ \circledast & \circledast & \circledast & \circledast & \circledast & \circledast \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{pmatrix} \xrightarrow{\cdot Q_0^*} \begin{pmatrix} \circledast & \circledast & \circledast & * & * & * \\ \circledast & \circledast & \circledast & * & * & * \\ \circledast & \circledast & \circledast & * & * & * \\ \circledast & \circledast & \circledast & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{pmatrix} = A_0.$$

(Kreise deuten wieder durch Q_0 geänderte Einträge an.) Beachten Sie, dass hier keine explizite Verschiebung auf $A^{(m)}$ angewendet wird – die gesamte Information über die Shifts μ_1 und μ_2 steckt in Q_0 .

4. Die restliche Aufgabe besteht darin, A_0 durch unitäre Ähnlichkeitstransformation wieder auf obere Hessenbergform zu bringen. Dafür berechnen wir der Reihe nach Rotationen oder Reflexionen Q_k , $1 \leq k \leq n - 2$, die die Einträge $a_{k+2,k}$ und $a_{k+3,k}$ (falls $k < n - 2$) von A_{k-1} annullieren, und setzen $A_k = Q_k A_{k-1} Q_k^*$. Für $n = 6$ und $k = 1$ ergibt das

$$A_0 \xrightarrow{Q_1} \begin{pmatrix} * & * & * & * & * & * \\ \circledast & \circledast & \circledast & \circledast & \circledast & \circledast \\ 0 & \circledast & \circledast & \circledast & \circledast & \circledast \\ 0 & 0 & \circledast & \circledast & \circledast & \circledast \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{pmatrix} \xrightarrow{\cdot Q_1^*} \begin{pmatrix} * & \circledast & \circledast & \circledast & * & * \\ * & \circledast & \circledast & \circledast & * & * \\ 0 & \circledast & \circledast & \circledast & * & * \\ 0 & \circledast & \circledast & \circledast & * & * \\ 0 & \circledast & \circledast & \circledast & * & * \\ 0 & 0 & 0 & 0 & * & * \end{pmatrix} = A_1.$$

Für $k = 2$ gehen wir analog vor, wobei Q_2 nur auf die zweite bis vierte Spalte beziehungsweise Zeile von A_1 wirkt und daher die erste Spalte erhalten bleibt. Nach $k = 4$ Schritten liegt daher A_4 in oberer Hessenbergform vor.

Die Ähnlichkeitstransformation mit Q_0 hat also eine „Beule“ unterhalb der Diagonalen von $A^{(m)}$ erzeugt, die wir mit den weiteren Transformationen Q_1, \dots, Q_{n-2} entlang der

¹¹Ist dies nicht der Fall, so ist $a_{21}^{(m)} = 0$ und damit $a_{11}^{(m)}$ ein Eigenwert von A , der durch Deflation entfernt werden sollte; siehe unten

Diagonalen schieben, bis sie die Matrix verlässt.¹² Die Matrix $Q' := (Q_{n-2} \dots Q_0)^*$ ist dabei als Produkt unitärer Matrizen unitär, und nach Konstruktion ist die erste Spalte von Q' identisch mit der ersten Spalte von Q (da die Q_i nur auf die Zeilen $i+1, i+2, i+3$ wirken, ist wegen ihrer Reihenfolge $Q'e_1 = Q_0^*e_1 = Qe_1$). Nach Satz 6.4 ist also (bis auf Vorzeichen der Spalten, die hier keine Rolle spielen) $Q' = Q$ und $A_{n-2} = (Q')^*A^{(m)}Q' = Q^*A^{(m)}Q = A^{(m+2)}$. Dieses Vorgehen bezeichnet man als *impliziten Shift* oder *Francis-Shift*.¹³

Genauso können auch mehr als zwei Shifts gleichzeitig durchgeführt werden, wobei die „Beule“ (und damit der Aufwand für deren Entfernung) mit der Anzahl der Shifts wächst. Umgekehrt kann auch der Rayleigh-Shift implizit durchgeführt werden (dadurch kann die Matrix $A^{(m)}$ bei jeder Rotation überschrieben werden, was Speicherplatz spart).¹⁴

3. Deflation

Bei Anwendung der Spektralverschiebung wird der Eintrag $a_{n,n-1}^{(m)}$ in der Regel quadratisch gegen 0 gehen. Sobald $a_{n,n-1}^{(m)} = \varepsilon$ klein genug ist, hat $A^{(m)}$ die Form

$$A^{(m)} = \begin{pmatrix} & & & * \\ & \bar{A} & & * \\ & & & * \\ 0 & 0 & \varepsilon & \tilde{\lambda}_n \end{pmatrix},$$

wobei $\tilde{\lambda}_n$ eine gute Näherung von λ_n und $\bar{A} \in \mathbb{C}^{(n-1) \times (n-1)}$ eine obere Hessenbergmatrix ist, deren Eigenwerte ungefähr gleich den Eigenwerten $\lambda_j, j \neq i$, der Matrix A sind. Es liegt also nahe, die Iteration mit $A^{(m+1)} = \bar{A}$ fortzusetzen, und so mit jedem berechneten Eigenwert die Dimension der auftretenden Matrizen zu reduzieren.

Algorithmus 10.4 beschreibt das QR-Verfahren inklusive der oben beschriebenen Modifikationen (mit expliziten Rayleigh-Shifts). Sind zusätzlich die Eigenvektoren von A gesucht, müssen zumindest im letzten Schritt die Givens-Rotationen aufmultipliziert werden, um die Matrix $Q^{(m)}$ zu erhalten. (Ist A nicht normal, erhält man die gesuchten Eigenvektoren v_i durch jeweils eine einzige inverse Iteration auf $A^{(0)}$ – unter Ausnutzung der oberen Hessenbergform – mit Verschiebung $\lambda_i^{(m)}$ und anschließender Multiplikation mit Q .)

¹²Dieser Vorgang wird daher im Englischen als „*bulge chasing*“ bezeichnet.

¹³Auch der Francis-Shift kann versagen, wenn es zusätzliche betragsgleiche Eigenwerte gibt. Um diesen Fall zu behandeln, wird in der Praxis nach z. B. 10 Schritten ohne Deflation ein zufälliger *exceptional shift* durchgeführt.

¹⁴Die QR-Iteration mit impliziten Rayleigh- und Francis-Shifts ist auch die Grundlage von MATLABs „*eig*“-Funktion für die Berechnung sämtlicher Eigenwerte einer vollbesetzten Matrix.

Algorithmus 10.4 QR-Verfahren mit expliziten Rayleigh-Shifts und Deflation**Input:** $A \in \mathbb{R}^{n \times n}$, $\varepsilon > 0$

- 1: Transformiere A auf obere Hessenbergform $\rightarrow A^{(0)}$, setze $m = 0$
- 2: **repeat**
- 3: $\mu^{(m)} \leftarrow a_{nn}^{(m)}$
- 4: $A^{(m)} \leftarrow A^{(m)} - \mu^{(m)}I$, $A^{(m+1)} \leftarrow A^{(m)}$
- 5: **for** $k = 2$ to n **do**
- 6: Berechne Givens-Rotation $G_{k,k-1}$ um $a_{k,k-1}^{(m)}$ zu annullieren
- 7: $A^{(m)} \leftarrow G_{k,k-1}A^{(m)}$
- 8: $A^{(m+1)} \leftarrow G_{k,k-1}A^{(m+1)}G_{k,k-1}^*$
- 9: $A^{(m+1)} \leftarrow A^{(m+1)} + \mu^{(m)}I$
- 10: $m \leftarrow m + 1$
- 11: **if** $|a_{n,n-1}^{(m)}| < \varepsilon$ **then**
- 12: $\lambda_n \leftarrow a_{nn}^{(m)}$
- 13: $A^{(m)} \leftarrow (a_{ij}^{(m)})_{i,j=1}^{n-1}$
- 14: $n \leftarrow n - 1$
- 15: **until** $n = 0$

Output: λ_i

PROJEKTIONSVERFAHREN

Auch für Eigenwertprobleme basieren die modernen Verfahren für große, dünn besetzte Matrizen auf einem Projektionsansatz: Statt das volle Problem zu lösen, geht man zu einem (viel) kleineren Problem in einem Unterraum über. Dabei besteht – wie auch bei der Vektor- und QR-Iteration – die Hauptaufgabe in der Berechnung von *Eigenvektoren*, aus denen die zugehörigen Eigenwerte leicht mit Hilfe des Rayleigh-Quotienten (10.4) berechnet werden können. Üblicherweise ist man dabei nur an wenigen Eigenwerten und Eigenvektoren, meist den betragsgrößten oder -kleinsten, interessiert.

Der Grundansatz sind wieder die Galerkin-Bedingungen: Zu einer gegebenen Matrix $A \in \mathbb{C}^{n \times n}$ sucht man $\tilde{\lambda} \in \mathbb{C}$ und $\tilde{v} \in \mathbb{C}^n \setminus \{0\}$, so dass

$$\begin{cases} \tilde{v} \in \mathcal{K} \\ A\tilde{v} - \tilde{\lambda}\tilde{v} \perp \mathcal{L} \end{cases}$$

für geeignete Unterräume $\mathcal{K}, \mathcal{L} \subset \mathbb{C}^n$ der Dimension $m \leq n$ gilt. Haben wir eine Basis v_1, \dots, v_m von \mathcal{K} und eine Basis w_1, \dots, w_m von \mathcal{L} zur Verfügung, so können wir mit $V = [v_1, \dots, v_m]$ und $W = [w_1, \dots, w_m]$ sowie $\xi \in \mathbb{C}^m \setminus \{0\}$ mit $V\xi = \tilde{v}$ die Galerkin-Bedingungen in Matrixform schreiben als

$$W^*(AV\xi - \tilde{\lambda}V\xi) = 0.$$

Sind die Basen biorthogonal, so gilt $W^*V = I$, und $\tilde{\lambda}$ und ξ müssen

$$(W^*AV)\xi = \tilde{\lambda}\xi$$

erfüllen. Für $m \ll n$ kann dieses Eigenwertproblem für $A_m := W^*AV \in \mathbb{C}^{m \times m}$ mit Hilfe der QR-Iteration gelöst werden. Analog zu Algorithmus 5.1 erhält man das folgende allgemeine Verfahren.

Algorithmus 11.1 Projektionsverfahren für Eigenwertprobleme

Input: $A \in \mathbb{C}^{n \times n}$, $\varepsilon > 0$, $k = 0$, $r_i^0 = 1$,

- 1: **while** $\|r_i^k\|_2 > \varepsilon$ für ein i **do**
- 2: $k \leftarrow k + 1$
- 3: Wähle Unterräume \mathcal{K}, \mathcal{L}
- 4: Berechne biorthogonale Basen V, W von \mathcal{K} und \mathcal{L}
- 5: Berechne Eigenwerte λ_i^k und Eigenvektoren y_i^k von $A_k = W^*AV$
- 6: $v_i^k = Vy_i^k$
- 7: $r_i^k = Av_i^k - \lambda_i^k v_i^k$

Output: v_i^k, λ_i^k

Eine wesentliche Frage ist hier, ob ein kleines Residuum r_i^k auch bedeutet, dass der zugehörige Eigenwert λ_i^k eine gute Approximation eines Eigenwerts von A darstellt. Die Antwort gibt der folgende Störungssatz:

Satz 11.1 (Bauer–Fike). Sei $A \in \mathbb{C}^{n \times n}$ diagonalisierbar mit $A = XDX^{-1}$, und seien $\tilde{\lambda} \in \mathbb{C}$ und $\tilde{v} \in \mathbb{C}^n$ mit $\|\tilde{v}\|_2 = 1$ gegeben. Dann existiert ein Eigenwert λ von A mit

$$|\lambda - \tilde{\lambda}| \leq \kappa_2(X) \|A\tilde{v} - \tilde{\lambda}\tilde{v}\|_2.$$

Beweis. Angenommen, $\tilde{\lambda}$ ist kein Eigenwert von A (sonst ist die Ungleichung trivial). Dann ist $A - \tilde{\lambda}I$ invertierbar, und es gilt wegen $A = XDX^{-1}$ für $r := A\tilde{v} - \tilde{\lambda}\tilde{v}$:

$$\tilde{v} = (A - \tilde{\lambda}I)^{-1}r = X(D - \tilde{\lambda}I)^{-1}X^{-1}r.$$

Da \tilde{v} normiert ist, folgt daraus

$$\begin{aligned} 1 &= \|X(D - \tilde{\lambda}I)^{-1}X^{-1}r\|_2 \\ &\leq \|X\|_2 \|X^{-1}\|_2 \|(D - \tilde{\lambda}I)^{-1}\|_2 \|r\|_2. \end{aligned}$$

Nun ist die Spektralnorm einer Diagonalmatrix gegeben durch den betragsgrößten Diagonaleintrag (der ja gleich dem betragsgrößten Eigenwert ist). Daher gilt

$$1 \leq \kappa(X) \|r\|_2 \max_{\lambda \in \sigma(A)} |\lambda - \tilde{\lambda}|^{-1},$$

und wir erhalten die gewünschte Abschätzung für denjenigen Eigenwert, für den das Maximum angenommen wird (d. h. den, der am nächsten an $\tilde{\lambda}$ liegt). \square

Daraus folgt sofort eine wichtige Erkenntnis: Das Eigenwertproblem für selbstadjungierte Matrizen (und allgemeiner, für normale Matrizen, für die X unitär ist) ist stabil. Dagegen kann für nicht-selbstadjungierte Matrizen die Konditionszahl von X beliebig groß sein. Tatsächlich spielen für nicht-selbstadjungierte Matrizen auch die *Links-Eigenvektoren* eine wichtige Rolle, was auf die folgende Definition führt:

Definition 11.2. Sei $A \in \mathbb{C}^{n \times n}$ diagonalisierbar, $\lambda_i \in \mathbb{C}$ ein Eigenwert von A , und $v_i, w_i \in \mathbb{C}^n$ mit $\|v_i\| = \|w_i\| = 1$ die zugehörigen Rechts- bzw. Links-Eigenvektoren. Dann sind die Konditionszahlen von λ_i bzw. v_i bezüglich Störungen von A definiert als

$$\begin{aligned}\kappa(\lambda_i) &= \frac{1}{|\langle v_i, w_i \rangle|}, \\ \kappa(v_i) &= \sum_{j \neq i} \frac{\kappa(\lambda_j)}{|\lambda_j - \lambda_i|}.\end{aligned}$$

Die Kondition eines Eigenvektors wird also zusätzlich durch die Separation des zugehörigen Eigenwerts vom Rest des Spektrums bestimmt. (Hat A mehrfache Eigenwerte, sind die damit verbundenen Eigenvektoren ja auch nicht eindeutig bestimmt.)

Auch beim Eigenwertproblem unterscheiden wir zwischen orthogonalen und schiefen Projektionsverfahren. Die wesentliche Frage wird dabei nach der Genauigkeit der Approximation der Eigenvektoren durch Vektoren in \mathcal{K} sein.

11.1 ORTHOGONALE PROJEKTIONSVERFAHREN

Der orthogonale Projektionsansatz mit $\mathcal{L} = \mathcal{K}$ wird als *Rayleigh-Ritz-Verfahren* bezeichnet. Führen wir die orthogonale Projektion $P_{\mathcal{K}}$ auf \mathcal{K} ein, so können wir die Galerkin-Bedingungen schreiben als

$$(11.1) \quad A_{\mathcal{K}} \tilde{v} := P_{\mathcal{K}} A P_{\mathcal{K}} \tilde{v} = \tilde{\lambda} \tilde{v},$$

wobei $\tilde{v} \in \mathbb{C}^n \setminus \{0\}$ ist. (Da die linke Seite wegen $P_{\mathcal{K}}$ in \mathcal{K} ist, muss auch $\tilde{v} \in \mathcal{K}$ sein.) Erfüllt $(\tilde{\lambda}, \tilde{v})$ die Bedingung (11.1), so nennt man $\tilde{\lambda}$ *Ritz-Wert* und \tilde{v} *Ritz-Vektor* von A .

Für die später betrachteten Krylovraum-Verfahren ist folgendes Resultat wichtig:

Satz 11.3. *Wenn \mathcal{K} invariant unter A ist, so sind die Ritz-Werte bzw. Ritz-Vektoren Eigenwerte bzw. Eigenvektoren von A .*

Beweis. Angenommen $\tilde{\lambda} \in \mathbb{C}$ und $\tilde{v} \in \mathcal{K} \setminus \{0\}$ erfüllen (11.1). Dann ist insbesondere $P_{\mathcal{K}} \tilde{v} = \tilde{v}$. Ist \mathcal{K} invariant unter A , so ist auch $A\tilde{v} \in \mathcal{K}$, und daher ist (11.1) äquivalent zu

$$\begin{aligned}0 &= A_{\mathcal{K}} \tilde{v} - \tilde{\lambda} \tilde{v} \\ &= P_{\mathcal{K}}(A\tilde{v} - \tilde{\lambda} \tilde{v}) \\ &= A\tilde{v} - \tilde{\lambda} \tilde{v},\end{aligned}$$

d. h. \tilde{v} ist Eigenvektor zum Eigenwert $\tilde{\lambda}$ von A . □

Ist \mathcal{K} nicht invariant, so wird die bestmögliche Näherung dadurch bestimmt, wie „nahe“ der gesuchte Eigenvektor v an \mathcal{K} liegt, gemessen durch den Abstand $\|(I - P_{\mathcal{K}})v\|_2$. Das nächste Lemma erlaubt die Abschätzung des Residuums des projizierten Problems für den *exakten* Eigenwert und Eigenvektor (der dafür auf \mathcal{K} projiziert werden muss) durch diese Distanz.

Lemma 11.4. *Sei $\lambda \in \mathbb{C}$ Eigenwert von A mit Eigenvektor $v \in \mathbb{C}^n$. Dann gilt die Abschätzung*

$$\|(A_{\mathcal{K}} - \lambda I)P_{\mathcal{K}}v\|_2 \leq \gamma \|(I - P_{\mathcal{K}})v\|_2$$

mit $\gamma = \|P_{\mathcal{K}}A(I - P_{\mathcal{K}})\|_2$.

Beweis. Da $P_{\mathcal{K}}$ und $(I - P_{\mathcal{K}})$ Projektionen sind, gilt

$$\begin{aligned} \|(A_{\mathcal{K}} - \lambda I)P_{\mathcal{K}}v\|_2 &= \|P_{\mathcal{K}}\lambda v - A_{\mathcal{K}}P_{\mathcal{K}}v\|_2 \\ &= \|P_{\mathcal{K}}Av - P_{\mathcal{K}}AP_{\mathcal{K}}v\|_2 \\ &= \|P_{\mathcal{K}}A(I - P_{\mathcal{K}})v\|_2 \\ &= \|P_{\mathcal{K}}A(I - P_{\mathcal{K}})(I - P_{\mathcal{K}})v\|_2 \\ &\leq \gamma \|(I - P_{\mathcal{K}})v\|_2, \end{aligned}$$

wobei wir die Definition von $A_{\mathcal{K}}$ und $Av = \lambda v$ verwendet haben. □

Dabei kann γ durch $\|A\|_2$ abgeschätzt werden, da die Norm orthogonaler Projektionen gleich 1 ist. Wenden wir nun den Satz von **Bauer–Fike** auf $A_{\mathcal{K}}$ und das Paar $(\lambda, P_{\mathcal{K}}v)$ an, erhalten wir eine Fehlerabschätzung für das orthogonale Projektionsverfahren.

Folgerung 11.5. *Sei $A_{\mathcal{K}}$ diagonalisierbar und λ ein Eigenwert von A mit Eigenvektor v . Dann existiert ein Ritz-Wert $\tilde{\lambda}$ mit*

$$|\lambda - \tilde{\lambda}| \leq \kappa(X)\gamma \|(I - P_{\mathcal{K}})v\|_2.$$

Ist v also „nahe“ an \mathcal{K} und A selbstadjungiert, so liefert das orthogonale Projektionsverfahren eine gute Näherung. Wie gut diese Näherung ist, hängt natürlich von der speziellen Wahl von \mathcal{K} ab; im nächsten Kapitel werden wir diesen Abstand für Krylovräume untersuchen.

Für selbstadjungierte Matrizen können wir auch Optimalitätsaussagen über die Ritz-Werte beweisen. Wir nehmen wieder an, dass die Eigenwerte von A absteigend angeordnet sind, d. h. es gelte

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n,$$

und ebenso gelte für die Ritz-Werte

$$\tilde{\lambda}_1 \geq \dots \geq \tilde{\lambda}_m.$$

Nach dem Satz von Courant–Fischer ist dann

$$\tilde{\lambda}_1 = \max_{x \in \mathcal{K}, x \neq 0} \frac{\langle x, A_{\mathcal{K}}x \rangle}{\langle x, x \rangle} = \max_{x \in \mathcal{K}, x \neq 0} \frac{\langle P_{\mathcal{K}}x, AP_{\mathcal{K}}x \rangle}{\langle x, x \rangle} = \max_{x \in \mathcal{K}, x \neq 0} \frac{\langle x, Ax \rangle}{\langle x, x \rangle}$$

und ebenso

$$\tilde{\lambda}_m = \min_{x \in \mathcal{K}, x \neq 0} \frac{\langle x, Ax \rangle}{\langle x, x \rangle}.$$

Allgemein gilt:

Satz 11.6. Der Ritz-Wert $\tilde{\lambda}_i$ und der zugehörige Ritz-Vektor \tilde{v}_i einer selbstadjungierten Matrix A über den Unterraum \mathcal{K} erfüllen

$$\tilde{\lambda}_i = \frac{\langle \tilde{v}_i, A\tilde{v}_i \rangle}{\langle \tilde{v}_i, \tilde{v}_i \rangle} = \max_{S \subset \mathcal{K}, \dim(S)=i} \min_{x \in S \setminus \{0\}} \frac{\langle x, Ax \rangle}{\langle x, x \rangle}.$$

Folgerung 11.7. Für die Eigenwerte λ_i und Ritzwerte $\tilde{\lambda}_i$, $1 \leq i \leq m$, von A gilt

$$\lambda_i \geq \tilde{\lambda}_i \geq \lambda_n.$$

Beweis. Dies folgt aus

$$\tilde{\lambda}_i = \max_{S \subset \mathcal{K}, \dim(S)=i} \min_{x \in S \setminus \{0\}} \frac{\langle x, Ax \rangle}{\langle x, x \rangle} \leq \max_{S \subset \mathbb{C}^n, \dim(S)=i} \min_{x \in S \setminus \{0\}} \frac{\langle x, Ax \rangle}{\langle x, x \rangle} = \lambda_i$$

sowie

$$\tilde{\lambda}_i \geq \tilde{\lambda}_m = \min_{x \in \mathcal{K} \setminus \{0\}} \frac{\langle x, Ax \rangle}{\langle x, x \rangle} \geq \min_{x \in \mathbb{C}^n \setminus \{0\}} \frac{\langle x, Ax \rangle}{\langle x, x \rangle} = \lambda_n. \quad \square$$

Dies bedeutet, dass die extremalen Ritz-Werte bei wachsender Dimension von \mathcal{K} die extremalen Eigenwerte von A monoton „von innen“ annähern. Orthogonale Projektionsverfahren eignen sich also insbesondere dafür, die größten bzw. kleinsten Eigenwerte einer (selbstadjungierten) Matrix zu berechnen.

11.2 SCHIEFE PROJEKTIONSVERFAHREN

Wir betrachten nun Projektionsverfahren mit $\mathcal{L} \neq \mathcal{K}$, d. h. wir suchen $\tilde{v} \in \mathcal{K}$ mit $A\tilde{v} - \tilde{\lambda}\tilde{v}$ orthogonal zu \mathcal{L} . Führen wir die (schiefe) Projektion $Q_{\mathcal{K}}^{\mathcal{L}}$ auf \mathcal{K} senkrecht zu \mathcal{L} ein, haben die Galerkin-Bedingungen die Form

$$A_{\mathcal{K}}^{\mathcal{L}}\tilde{v} := Q_{\mathcal{K}}^{\mathcal{L}}AP_{\mathcal{K}}\tilde{v} = \tilde{\lambda}\tilde{v},$$

wobei $P_{\mathcal{K}}$ wieder die orthogonale Projektion auf \mathcal{K} bezeichnet. Man vergewissert sich leicht, dass die Resultate in Abschnitt 11.1 bis einschließlich Folgerung 11.5 auch in diesem Fall gelten. Allerdings kann die Konstante $\gamma = \|Q_{\mathcal{K}}^{\mathcal{L}} A(I - P_{\mathcal{K}})\|_2$ nicht mehr durch $\|A\|_2$ beschränkt werden, und das projizierte Problem kann viel schlechter konditioniert sein als das ursprüngliche Eigenwertproblem.

Auch hier ist der Fall $\mathcal{L} = A\mathcal{K}$ besonders interessant. Sei wieder V eine Matrix, deren Spalten eine Basis von \mathcal{K} bilden. Dann kann man die Galerkin-Bedingungen in Matrixform schreiben als

$$[(AV)^*AV]\xi = \tilde{\lambda}[V^*A^*V]\xi.$$

Dies ist ein *verallgemeinertes Eigenwertproblem* für eine selbstadjungierte, positiv definite Matrix. Wählt man nun W so, dass die Spalten eine orthonormale Basis von $A\mathcal{K}$ bilden, so sind – für invertierbares A – die Spalten von $A^{-1}W$ eine Basis von \mathcal{K} . Setzt man diese in die Galerkin-Bedingungen ein, erhält man

$$\tilde{\lambda}^{-1}\xi = (W^*A^{-1}W)\xi,$$

d. h. ein orthogonal projiziertes Eigenwertproblem für A^{-1} . Diejenigen $\tilde{\lambda}$, die diese Eigenwertgleichung erfüllen, nennt man *harmonische Ritz-Werte* von A ; sie nähern die betragskleinsten Eigenwerte von A „von außen“ an. Ist man nur an diesen interessiert, so ist es oft günstiger, diese mit Hilfe von Projektionsverfahren für A^{-1} zu berechnen (wobei natürlich statt Multiplikation mit A^{-1} das entsprechende lineare Gleichungssystem für A gelöst wird; idealerweise mit Hilfe einer vorberechneten LU- oder Cholesky-Zerlegung von A).

KRYLOVRAUM-VERFAHREN

12

Wir kommen nun zu dem Standard-Verfahren für die Eigenwertbestimmung großer, dünnbesetzter Matrizen: dem *Arnoldi-* beziehungsweise *Lanczos-Verfahren*. Dabei handelt es sich um orthogonale Projektionsverfahren auf die Krylovräume

$$\mathcal{K}_m(A, v) = \text{span}\{v, Av, A^2v, \dots, A^{m-1}v\}.$$

Folgende Invarianz-Eigenschaften der Krylovräume werden dabei nützlich sein:

- $\mathcal{K}_m(\alpha A, \beta v) = \mathcal{K}_m(A, v)$ für alle $\alpha, \beta \neq 0$,
- $\mathcal{K}_m(A - \mu I, v) = \mathcal{K}_m(A, v)$ für alle $\mu \in \mathbb{C}$,
- $\mathcal{K}_m(X^{-1}AX, X^{-1}v) = X^{-1}\mathcal{K}_m(A, v)$ für alle invertierbaren $X \in \mathbb{C}^{n \times n}$.

12.1 ARNOLDI- UND LANCZOS-VERFAHREN

Wir kommen rasch zum Kern der Sache, denn wir sind gut vorbereitet. Seien $A \in \mathbb{C}^{n \times n}$ und $v \in \mathbb{C}^n \setminus \{0\}$ gegeben. Dann berechnet der [Arnoldi-Prozess](#) nach m Schritten eine Matrix $Q_m \in \mathbb{C}^{n \times m}$, deren Spalten q_1, \dots, q_m eine orthonormale Basis von $\mathcal{K}_m(A, v)$ bilden, und eine obere Hessenbergmatrix $H_m \in \mathbb{C}^{m \times m}$ mit

$$Q_m^* A Q_m = H_m.$$

Die Ritz-Werte von A bezüglich $\mathcal{K}_m(A, v)$ sind also die Eigenwerte von H_m , und die Ritz-Vektoren können aus den zugehörigen Eigenvektoren ξ_i von H_m durch $\hat{v}_i = Q_m \xi_i$ berechnet werden. Der Arnoldi-Prozess bricht nur ab, wenn $\mathcal{K}_m(A, v)$ invariant unter A ist; in diesem Fall sind die Ritz-Werte nach Satz 11.3 bereits Eigenwerte von A (und zwar die betragsgrößten – sowohl positive als auch negative wegen $\mathcal{K}_m(A, v) = \mathcal{K}_m(-A, v)$).

Bricht der Prozess nicht ab, existieren auch $h_{m+1,m} \neq 0$ und $q_{m+1} \in \mathbb{C}^n \setminus \{0\}$ mit

$$(12.1) \quad A Q_m = Q_m H_m + h_{m+1,m} q_{m+1} e_m^*.$$

Wie im GMRES-Verfahren können wir diese Darstellung verwenden, um ein Abbruch-Kriterium zu definieren.

Satz 12.1. *Angenommen, es wurden m Schritte im Arnoldi-Prozess durchgeführt, und seien $\tilde{\lambda}_i$ und $\tilde{v}_i = Q_m \xi_i$, $i = 1, \dots, m$, die Ritz-Werte und Ritz-Vektoren von $A \in \mathbb{C}^{n \times n}$ bezüglich $\mathcal{K}_m(A, v)$. Dann gilt*

$$\|A\tilde{v}_i - \tilde{\lambda}_i\tilde{v}_i\|_2 = |h_{m+1,m}\langle \xi_i, e_m \rangle|.$$

Beweis. Multiplizieren von (12.1) mit ξ_i , $1 \leq i \leq m$ ergibt

$$\begin{aligned} A Q_m \xi_i &= Q_m H_m \xi_i + h_{m+1,m} q_{m+1} \langle \xi_i, e_m \rangle \\ &= \tilde{\lambda}_i Q_m \xi_i + h_{m+1,m} q_{m+1} \langle \xi_i, e_m \rangle. \end{aligned}$$

Daraus folgt

$$\begin{aligned} A\tilde{v}_i - \tilde{\lambda}_i\tilde{v}_i &= A Q_m \xi_i - \tilde{\lambda}_i Q_m \xi_i \\ &= h_{m+1,m} q_{m+1} \langle \xi_i, e_m \rangle \end{aligned}$$

und damit die Behauptung. □

Die letzte Komponente eines berechneten Eigenvektors von H_m , multipliziert mit $h_{m+1,m}$, liefert also das Residuum der zugehörigen Eigenwertgleichung.

Ist A selbstadjungiert, so liefert stattdessen der **Lanczos-Prozess** eine tridiagonale Matrix T_m , deren Eigenwerte und Eigenvektoren die Ritz-Werte und Ritz-Vektoren darstellen. Wieder bricht der Prozess nur ab, wenn die Ritz-Werte exakt sind; ansonsten gilt Satz 12.1 (mit β_{m+1} anstelle von $h_{m+1,m}$).

Das Arnoldi-Verfahren (und – *mutatis mutandis* – das Lanczos-Verfahren) zur (näherungsweisen) Berechnung von k Eigenwerten der Matrix A kann also wie folgt zusammengefasst werden:

Beginne mit Zufallsvektor v_1

for $m = 1, \dots$ **do**

1. Führe einen Schritt im Arnoldi-Prozess durch $\rightsquigarrow H_m, Q_m, h_{m+1,m}$
 2. Bestimme Eigenwerte $\tilde{\lambda}_i$ und Eigenvektoren ξ_i von H_m
 3. Falls $|h_{m+1,m}\langle \xi_i, e_m \rangle| < \varepsilon$, berechne Ritz-Vektoren $\tilde{v}_i = Q_m \xi_i$
-

Da die Matrix H_m bereits obere Hessenbergform hat, bietet sich das QR-Verfahren in Schritt 2 an. In der Praxis wird man – insbesondere beim Arnoldi-Verfahren – m nicht beliebig groß werden lassen, sondern Neustarts durchführen. Ebenso möchte man bereits berechnete gute Näherungen an gesuchte Eigenvektoren behalten. Entsprechende Modifikation werden in Abschnitt 12.3 besprochen.

12.2 KONVERGENZ

Um das Konvergenzverhalten der Krylovraum-Verfahren zu untersuchen, müssen wir nach Lemma 11.4 den Abstand $\|(I - P_{\mathcal{K}})v_i\|$ der Eigenvektoren v_i von A zum Krylovraum $\mathcal{K} = \mathcal{K}_m(A, v)$ untersuchen. Dabei nutzen wir wieder aus, dass jeder Vektor $x \in \mathcal{K}_m(A, v)$ geschrieben werden kann als $x = p_{m-1}(A)v$ für ein Polynom p_{m-1} vom Höchstgrad $m - 1$.

Satz 12.2. Sei $A \in \mathbb{C}^{n \times n}$ diagonalisierbar mit Eigenwerten λ_i und normierten Eigenvektoren v_i , $i = 1, \dots, n$, und sei

$$v = \sum_{k=1}^n \alpha_k v_k$$

gegeben. Bezeichne $P_{\mathcal{K}}$ die orthogonale Projektion auf $\mathcal{K}_m(A, v)$. Dann gilt für alle $1 \leq i \leq n$ mit $\alpha_i \neq 0$ die Abschätzung

$$\|(I - P_{\mathcal{K}})v_i\|_2 \leq \left(\min_{\substack{p \in \mathcal{P}_{m-1} \\ p(\lambda_i) = 1}} \max_{\lambda \in \sigma(A) \setminus \{\lambda_i\}} |p(\lambda)| \right) \sum_{\substack{j=1 \\ j \neq i}}^n \frac{|\alpha_j|}{|\alpha_i|}.$$

Beweis. Wir betrachten statt v_i den skalierten Eigenvektor $\alpha_i v_i$. Nach Satz 2.5 erfüllt die orthogonale Projektion auf $\mathcal{K}_m(A, v)$ die Eigenschaft

$$\begin{aligned} \|\alpha_i v_i - P_{\mathcal{K}} \alpha_i v_i\|_2 &= \min_{x \in \mathcal{K}_m(A, v)} \|\alpha_i v_i - x\|_2 = \min_{p \in \mathcal{P}_{m-1}} \|\alpha_i v_i - p(A)v\|_2 \\ &\leq \min_{\substack{p \in \mathcal{P}_{m-1} \\ p(\lambda_i) = 1}} \|\alpha_i v_i - p(A)v\|_2. \end{aligned}$$

Sei $p \in \mathcal{P}_{m-1}$ mit $p(\lambda_i) = 1$ beliebig. Dann gilt

$$\|\alpha_i v_i - p(A)v\|_2 = \|p(\lambda_i)\alpha_i v_i - \sum_{j=1}^n p(\lambda_j)\alpha_j v_j\|_2 \leq \max_{\substack{j \neq i \\ j=1 \\ j \neq i}} |p(\lambda_j)| \sum_{\substack{j=1 \\ j \neq i}}^n |\alpha_j|,$$

da die v_i normierte Eigenvektoren sind. Division durch $|\alpha_i| \neq 0$ liefert die gewünschte Abschätzung. \square

Für selbstadjungierte Matrizen können wir wieder konkretere Schranken angeben, indem wir das minimierende Polynom nach oben durch Tschebyschow-Polynome abschätzen.

Satz 12.3. Sei A selbstadjungiert mit absteigend sortierten Eigenwerten λ_i und zugehörigen Eigenvektoren v_i . Dann gilt

$$\min_{\substack{p \in \mathcal{P}_{m-1} \\ p(\lambda_i) = 1}} \max_{\lambda \in \sigma(A) \setminus \{\lambda_i\}} |p(\lambda)| \leq \frac{2\kappa_i}{\left(\gamma_i + \sqrt{\gamma_i^2 - 1}\right)^{m-i}}$$

mit

$$\kappa_1 = 1, \quad \kappa_i = \prod_{j=1}^{i-1} \frac{\lambda_j - \lambda_n}{\lambda_j - \lambda_i} \quad \text{für } i > 1$$

und

$$\gamma_i = 1 + 2 \frac{\lambda_i - \lambda_{i+1}}{\lambda_{i+1} - \lambda_n}.$$

Beweis. Wir betrachten zuerst den Fall $i = 1$. Setze

$$p(\lambda) = \frac{T_{m-1}(r_1(\lambda))}{T_{m-1}(\gamma_1)}, \quad r_1(\lambda) = 1 + 2 \frac{\lambda - \lambda_2}{\lambda_2 - \lambda_n}.$$

Dann ist $p \in \mathcal{P}_{m-1}$ und $p(\lambda_1) = 1$. Außerdem gilt $r_1(\lambda) \in [-1, 1]$ für $\lambda \in \{\lambda_2, \dots, \lambda_n\}$ und damit $T_{m-1}(r_1(\lambda)) \leq 1$. Weiterhin zeigt man mit Induktion, dass

$$T_k(z) = \frac{1}{2} \left[\left(z + \sqrt{z^2 - 1} \right)^k + \left(z - \sqrt{z^2 - 1} \right)^k \right]$$

für $|z| \geq 1$ gilt. Damit ist

$$p(\lambda) \leq \frac{1}{T_{m-1}(\gamma_i)} \leq \frac{2}{\left(\gamma_1 - \sqrt{\gamma_1^2 - 1} \right)^{m-1}},$$

woraus die Abschätzung für $i = 1$ folgt.

Für $i > 1$ wählen wir

$$p(\lambda) = \frac{(\lambda_1 - \lambda) \cdots (\lambda_{i-1} - \lambda)}{(\lambda_1 - \lambda_i) \cdots (\lambda_{i-1} - \lambda_i)} q(\lambda)$$

für ein Polynom $q \in \mathcal{P}_{m-i}$ mit $q(\lambda_i) = 1$. Dann ist $p(\lambda_j) = 0$ für $j < i$ und daher

$$\begin{aligned} \max_{\lambda \in \sigma(A) \setminus \{\lambda_i\}} |p(\lambda)| &= \max_{\lambda \in \{\lambda_{i+1}, \dots, \lambda_n\}} \left| \frac{(\lambda_1 - \lambda) \cdots (\lambda_{i-1} - \lambda)}{(\lambda_1 - \lambda_i) \cdots (\lambda_{i-1} - \lambda_i)} q(\lambda) \right| \\ &\leq \kappa_i \max_{\lambda \in \{\lambda_{i+1}, \dots, \lambda_n\}} |q(\lambda)|. \end{aligned}$$

Die Aussage folgt nun, indem wir für q wie im Fall $i = 1$ das Polynom

$$q(\lambda) = \frac{T_{m-i}(r_i(\lambda))}{T_{m-i}(\gamma_i)}, \quad r_i(\lambda) = 1 + 2 \frac{\lambda - \lambda_{i+1}}{\lambda_{i+1} - \lambda_n},$$

wählen. □

Zusammen mit Folgerung 11.5 liefert dies die Konvergenz des Lanczos-Verfahrens. Für das Arnoldi-Verfahren bei nicht-selbstadjungierten Matrizen kann man ähnlich wie für GMRES mit Hilfe komplexer Tschebyschow-Polynome ähnliche Abschätzungen herleiten, wenn das Spektrum von A in bestimmten Ellipsen in der komplexen Ebene enthalten ist.

12.3 NEUSTARTS UND DEFLATION

Da die Schritte im Arnoldi-Prozess mit wachsender Dimension des Krylovraums immer teurer werden, bedient man sich in der Praxis wie im GMRES-Verfahren regelmäßiger Neustarts: nach m Iterationen setzt man $\mathcal{K}_{m+1} = \mathcal{K}_1(A, \tilde{v})$ mit einem neuen Startvektor \tilde{v} . Dabei soll natürlich der bisherige Fortschritt in der Berechnung der Ritz-Vektoren nicht verlorengehen. Nach Satz 12.2 hängt die Konvergenzgeschwindigkeit wesentlich vom Verhältnis des Winkels α_i zwischen \tilde{v} und eines gesuchten Eigenvektors v_i zu den übrigen Winkeln ab. Es liegt daher nahe, den neuen Vektor als Linearkombination der bisher gefundenen Ritz-Vektoren zu wählen: Angenommen, $\tilde{\lambda}_1^{(m)}, \dots, \tilde{\lambda}_k^{(m)}$ sind Ritz-Werte, die in der Nähe der gesuchten Eigenwerte (z. B. der k betragsgrößten) liegen, und $\tilde{v}_1^{(m)}, \dots, \tilde{v}_k^{(m)}$ sind die zugehörigen Ritz-Vektoren. Dann wählen wir

$$\tilde{v} = \sum_{i=1}^k \gamma_i \tilde{v}_i^{(m)} \in \mathcal{K}_m(A, v)$$

für $\gamma_i \in \mathbb{R}$. Da jeder Vektor $x \in \mathcal{K}_m(A, v)$ als $x = p(A)v$ für ein Polynom $p \in \mathcal{P}_{m-1}$ geschrieben werden kann, existiert ein Polynom $q \in \mathcal{P}_{m-1}$ mit $\tilde{v} = q(A)v$. Wir wenden also einen *polynomiellen Filter* auf v an, um unseren neuen Start-Vektor zu erhalten. Dabei bleibt die Frage offen, wie die γ_i zu wählen sind. Günstiger ist eine alternative Vorgehensweise: Statt die gewünschten Eigenwerte zu verstärken, werden die unerwünschten Eigenwerte unterdrückt. Ähnlich wie im Beweis von Satz 12.2 wählt man dafür das *Filterpolynom*

$$(12.2) \quad q(z) = (z - \tilde{\lambda}_{k+1}^{(m)}) \cdots (z - \tilde{\lambda}_m^{(m)}).$$

Dann gilt

$$q(A)\tilde{v} = \sum_{i=1}^n \alpha_i q(\lambda_i) v_i,$$

so dass die (relativen) Beiträge der unerwünschten Eigenvektoren klein und die der gesuchten Eigenvektoren groß sind. Durch diese *exakte Verschiebung* werden also die unerwünschten Eigenwerte in die Mitte des Spektrums verschoben, so dass die gesuchten Eigenwerte nun die extremalen sind. Die zugehörigen Ritz-Vektoren werden daher immer schneller konvergieren.

Sind erwünschte Eigenvektoren hinreichend genau berechnet worden, ist es sinnvoll, die weitere Iteration auf das orthogonale Komplement des von diesen aufgespannten Unterraums zu beschränken. Zum einen verbessert dies die Stabilität und erlaubt es, mehrere linear unabhängige Eigenvektoren zu einem mehrfachen Eigenwert zu berechnen. Andererseits wird dadurch Aufwand gespart. Angenommen, v_1, \dots, v_s sind bereits berechnete erwünschte (orthonormale) Eigenvektoren mit Eigenwerten $\lambda_1, \dots, \lambda_s$. Dann hat

$$\tilde{A} = A - VDV^*, \quad V = [v_1, \dots, v_s], \quad D_{ij} = \begin{cases} \lambda_j & \text{für } 1 \leq i = j \leq s, \\ 0 & \text{sonst,} \end{cases}$$

die Eigenwerte $0, \dots, 0, \lambda_{s+1}, \dots, \lambda_n$. Alternativ kann man die Deflation in den Arnoldi-Prozess integrieren: Als Start-Vektor wählt man v_{s+1} orthogonal zu v_1, \dots, v_s , und berechnet die weiteren Vektoren v_{s+j} , indem man gegen *alle* $v_i, i = 1, \dots, s + j - 1$, orthogonalisiert. Dadurch erhält man eine orthonormale Basis q_1, \dots, q_m von

$$\mathcal{K} = \text{span}\{v_1, \dots, v_s, v_{s+1}, Av_{s+1}, \dots, A^{m-s-1}v_{s+1}\}.$$

Die so erzeugte Matrix $H_m = Q_m^* A Q_m$ hat dann partielle Schur-Normalform. Für $m = 5$ und $s = 2$ ist etwa

$$(12.3) \quad H_m = \left(\begin{array}{cc|ccc} * & * & * & * & * \\ & * & * & * & * \\ \hline & & * & * & * \\ & & * & * & * \\ & & & * & * \end{array} \right).$$

Da der obere $(s \times s)$ -Block bereits in oberer Dreiecksform ist, genügt es, die Eigenwerte des unteren $(m - s) \times (m - s)$ -Blocks zu berechnen. Dieses Vorgehen wird als „locking“ bezeichnet. (Unter dem Namen „purging“ existieren auch Verfahren, um berechnete *unerwünschte* Eigenvektoren zu entfernen.) Algorithmus 12.1 fasst das Arnoldi-Verfahren mit exakten Verschiebungen und Deflation zusammen.

Algorithmus 12.1 Arnoldi-Verfahren mit expliziten Neustarts

Input: $A \in \mathbb{C}^{n \times n}$, $m < n$, $k < m$, $\varepsilon > 0$

- 1: Wähle zufälligen Vektor v_1 mit $\|v_1\|_2 = 1$, setze $l = 1$
- 2: **while** $l < k$ **do**
- 3: **for** $j = l, \dots, m$ **do** ▷ Arnoldi-Prozess
- 4: Berechne $w = Av_j$
- 5: **for** $i = 1, \dots, j$ **do**
- 6: $h_{ij} = \langle v_i, w \rangle$
- 7: $w \leftarrow w - h_{ij}v_i$
- 8: $h_{j+1,j} = \|w\|_2, v_{j+1} = w/h_{j+1,j}$
- 9: Berechne Eigenwerte $\tilde{\lambda}_i$ und Eigenvektoren ξ_i von H_m
- 10: Wähle gesuchten Eigenvektor ξ_s und berechne Ritz-Vektor $\tilde{v} = Q_m \xi_s$ ▷ Filtern
- 11: Wähle unerwünschte Eigenwerte $\tilde{\lambda}_{k+1}, \dots, \tilde{\lambda}_m$ und setze
 $z = (A - \tilde{\lambda}_{k+1}I) \cdots (A - \tilde{\lambda}_m I) \tilde{v}$
- 12: Orthonormalisiere z gegen v_1, \dots, v_l und ersetze v_l ▷ Neustart
- 13: **if** $|h_{m+1,m} \langle \xi_s, e_m \rangle| < \varepsilon$ **then** ▷ Locking
- 14: Berechne $h_{i,l} = \langle v_i, Av_l \rangle, i = 1, \dots, l$
- 15: Setze $\lambda_l = \tilde{\lambda}_s$
- 16: Berechne v_{l+1} mit Arnoldi-Prozess, $l \leftarrow l + 1$

Output: $\lambda_j, v_j, j = 1, \dots, k$

Es gibt eine elegante Möglichkeit, dieses Verfahren deutlich effizienter durchzuführen (denn bei jedem Neustart sind $m - k$ Multiplikationen mit einer $(n \times n)$ -Matrix im Schritt 11 nötig). Wegen

$$\begin{aligned}\mathcal{K}_k(A, q(A)v) &= \text{span}\{q(A), Aq(A)v, \dots, A^{m-1}q(A)v\} \\ &= \text{span}\{q(A), q(A)(Av), \dots, q(A)(A^{m-1}v)\} \\ &= q(A)\mathcal{K}_k(A, v)\end{aligned}$$

wird dieser Filter gleichzeitig auf alle Vektoren im Krylovraum angewendet. Anstatt eine neue Basis für den gefilterten Startvektor \tilde{v} zu berechnen, können wir also genauso gut die bereits berechnete Basis des Krylovraums $\mathcal{K}_m(A, v)$ filtern, um eine Basis von $\mathcal{K}_m(A, \tilde{v})$ zu erhalten. In der Tat ist es möglich, die Basis Q_m und die projizierte Matrix H_m zu filtern, ohne das Filterpolynom explizit anwenden zu müssen. Dazu verwenden wir, dass nach Satz 6.4 die Ergebnisse des Arnoldi-Prozess durch den Startvektor eindeutig bestimmt sind.

Angenommen, wir haben m Schritte im Arnoldi-Prozess mit $v_1 = v$ durchgeführt und haben dadurch orthonormale Vektoren v_1, \dots, v_{m+1} und h_{ij} , $1 \leq i \leq j + 1 \leq m + 1$, mit

$$(12.4) \quad AV_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^*.$$

Um den ersten Faktor des Filterpolynoms (12.2) mit Verschiebung $\mu_1 = \tilde{\lambda}_{k+1}$ auf die Vektoren v_1, \dots, v_m anzuwenden, verwenden wir (12.4) und erhalten

$$(12.5) \quad (A - \mu_1 I)V_m = V_m(H_m - \mu_1 I) + h_{m+1,m} v_{m+1} e_m^*.$$

Die Idee ist nun, auf H_m eine QR-Iteration mit Spektralverschiebung anzuwenden:

$$H_m - \mu_1 I =: Q_1 R_1, \quad H_m^{(1)} := R_1 Q_1 + \mu_1 I.$$

Durch Einsetzen und Multiplikation von rechts mit Q_1 erhalten wir dann aus (12.5)

$$(12.6) \quad A(V_m Q_1) = (V_m Q_1)H_m^{(1)} + h_{m+1,m} v_{m+1} (e_m^* Q_1).$$

Diese Gleichung soll nun in die Form (12.4) gebracht werden. Da H_m in oberer Hessenbergform vorlag und V_m unitär ist, folgt aus den Eigenschaften der QR-Iteration:

1. $H_m^{(1)}$ ist auch in oberer Hessenbergform,
2. $V_m^{(1)} := V_m Q_1$ ist als Produkt unitärer Matrizen auch unitär,
3. Q_1 ist das Produkt von $m - 1$ Givens-Rotationen $G_{1,2}, \dots, G_{m-1,m}$.

Der letzte Punkt bedeutet, dass $e_m^* Q_1$ nur von der letzten Rotation abhängt:

$$e_m^* Q_1 = e_m^* G_{m-1,m} = s_m e_{m-1}^* + c_m e_m^*.$$

Damit sind lediglich die letzten beiden Einträge in $e_m^* Q_1$ von Null verschieden. Zwar hat deshalb (12.6) selbst nicht die Form (12.4), wir können sie aber durch Weglassen der letzten

Zeile und Spalte wiederherstellen. Bezeichnet $\tilde{H}_{m-1}^{(1)}$ die Teil-Matrix der ersten $(m-1) \times (m-1)$ Einträge von $H_m^{(1)}$ und $V_{m-1}^{(1)}$ die Matrix der ersten $m-1$ Spalten von $V_m^{(1)}$, dann gilt wegen der Hessenbergform von $H_m^{(1)}$

$$AV_{m-1}^{(1)} = V_{m-1}^{(1)} \tilde{H}_{m-1}^{(1)} + h_{m,m-1}^{(1)} v_m^{(1)} e_{m-1}^* + h_{m+1,m} s_m v_{m+1} e_{m-1}^*,$$

wobei der zweite Summand von dem m -ten Eintrag der (in $\tilde{H}_{m-1}^{(1)}$ gestrichenen) $(m-1)$ -ten Spalte von $V_m^{(1)} H_m^{(1)}$ stammt. Da $v_m^{(1)}$ und v_{m+1} orthogonal zu $v_1^{(1)}, \dots, v_{m-1}^{(1)}$ sind, können wir durch

$$\hat{h}_{m,m-1}^{(1)} \hat{v}_m^{(1)} := h_{m,m-1}^{(1)} v_m^{(1)} + h_{m+1,m} s_m v_{m+1}, \quad \|\hat{v}_m^{(1)}\|_2 = 1,$$

einen Vektor $\hat{v}_m^{(1)}$ definieren, so dass $v_1^{(1)}, \dots, v_{m-1}^{(1)}, \hat{v}_m^{(1)}$ orthonormal sind, $\hat{h}_{m,m-1}^{(1)} > 0$ ist, und

$$(12.7) \quad AV_{m-1}^{(1)} = V_{m-1}^{(1)} \tilde{H}_{m-1}^{(1)} + \hat{h}_{m,m-1}^{(1)} \hat{v}_m^{(1)} e_{m-1}^*$$

gilt – wir erhalten also wieder eine Gleichung der Form (12.1). Betrachten wir die erste Spalte von (12.5) und setzen die QR-Zerlegung ein, ergibt dies wegen der Dreiecksform von R_1 :

$$\begin{aligned} (A - \mu_1 I)v_1 &= (A - \mu_1 I)V_m e_1 = V_m (H_m - \mu_1 I)e_1 = V_m (Q_1 R_1)e_1 \\ &= V_m^{(1)}(r_{11}e_1) = r_{11}v_1^{(1)}. \end{aligned}$$

Der Vektor $v_1^{(1)}$ ist also ein Vielfaches des gefilterten Startvektors \tilde{v}_1 . Aus der Eindeutigkeit von (12.1) und der Invarianz von Krylovräumen unter Multiplikation mit Skalaren folgt nun, dass (12.7) identisch ist mit dem Ergebnis, wenn $m-1$ Schritte im Arnoldi-Prozess für den Start-Vektor $v_1^{(1)}$ durchgeführt werden. Ein weiterer Schritt im Arnoldi-Prozess liefert also in beiden Fällen eine Basis des Krylovraums $\mathcal{K}_m(A, q(A)v)$ sowie die darauf projizierte Matrix H_m in oberer Hessenbergform.

Alternativ können zuerst weitere implizite Verschiebungen durchgeführt werden: eine QR-Iteration mit Spektralverschiebung $\mu_2 = \tilde{\lambda}_{k+2}$ angewendet auf $H_m^{(1)}$ liefert eine Hessenbergmatrix $H_m^{(2)}$ sowie $V_m^{(2)} := V_m^{(1)} Q_2$ mit

$$(12.8) \quad AV_m^{(2)} = V_m^{(2)} H_m^{(2)} + h_{m+1,m} v_{m+1} (e_m^* Q_1 Q_2),$$

so dass die erste Spalte $v_1^{(2)}$ von $V_m^{(2)}$ ein skalares Vielfaches von $(A - \mu_2 I)(A - \mu_1 I)v$ ist. Diesmal wirken nur die letzten beiden Rotationen von Q_2 auf $e_m^* Q_1$, so dass

$$e_m^* Q_1 Q_2 = (0, \dots, 0, s_1, s_2, s_3)$$

gilt. Betrachten wir nur die ersten $(m-2)$ Spalten von (12.8), erhalten wir

$$AV_{m-2}^{(2)} = V_{m-2}^{(2)} \tilde{H}_{m-2}^{(2)} + \hat{h}_{m-1,m-2}^{(2)} \hat{v}_{m-1}^{(2)} e_{m-2}^*$$

mit

$$\hat{h}_{m-1,m-2}^{(2)} \hat{v}_{m-1}^{(2)} := h_{m-1,m-2}^{(2)} v_{m-1}^{(2)} + h_{m+1,m} s_1 v_{m+1}.$$

Beachten Sie, dass hier $\hat{h}_{m,m-1}^{(1)}$ und $\hat{v}_m^{(1)}$ nicht benötigt wurden. Ebenso verfährt man mit den restlichen Verschiebungen, wobei jedesmal eine Spalte von V_m und H_m „verlorengehen“. Wir können also zuerst alle $m - k$ Verschiebungen durch (am günstigsten implizite) QR-Iterationen durchführen, die Rotationen der Reihe nach auf V_m anwenden, den modifizierten Vektor \hat{v}_{k+1} berechnen, und danach diese $k + 1$ Vektoren mit $m - k$ Schritten im Arnoldi-Prozess zu einer Basis von $\mathcal{K}_m(A, q(A)v)$ ergänzen.

Die wesentliche Ersparnis gegenüber den expliziten Neustarts liegt darin, dass die QR-Zerlegungen nur für eine $(m \times m)$ -Matrix durchgeführt werden müssen, und dass k Schritte im Arnoldi-Prozess gespart werden. Falls $k < m \ll n$ gilt (üblicherweise wählt man $m = 2k$), ist dies wesentlich günstiger als Multiplikationen mit einer $(n \times n)$ -Matrix. Algorithmus 12.2 fasst die wesentlichen Schritte zusammen.¹ Die Deflation durch „locking“ kann dabei in der QR-Iteration berücksichtigt werden, indem Blöcke übersprungen werden, die wie in (12.3) bereits in oberer Dreiecksform vorliegen. (Dies entspricht einer Deflation der entsprechenden Einträge, die nach der QR-Iteration wieder rückgängig gemacht wird.)

Algorithmus 12.2 Arnoldi-Verfahren mit impliziten Neustarts

- 1: Wähle zufälligen Vektor v_1 mit $\|v_1\|_2 = 1$
 - 2: **repeat**
 - 3: Ergänze auf m Schritte im Arnoldi-Prozess $\rightsquigarrow H_m, V_m, v_{m+1}, h_{m+1,m}$
 - 4: Berechne Eigenwerte $\tilde{\lambda}_i$ und Eigenvektoren ξ_i von H_m
 - 5: Wähle unerwünschte Eigenwerte $\mu_i = \tilde{\lambda}_{k+i}$, $i = 1, \dots, m - k$, setze $w \leftarrow e_m^*$
 - 6: **for** $j = 1, \dots, m - k$ **do**
 - 7: $Q_j R_j \leftarrow H_m - \mu_j I$
 - 8: $H_m \leftarrow R_j Q_j + \mu_j I$
 - 9: $V_m \leftarrow V_m Q_j$, $w \leftarrow w Q_j$
 - 10: $v_{k+1} \leftarrow h_{k+1,k} v_{k+1} + w_k h_{m+1,m} v_{m+1}$
 - 11: $h_{k+1,k} \leftarrow \|v_{k+1}\|_2$, $v_{k+1} \leftarrow v_{k+1} / h_{k+1,k}$
 - 12: **until** Konvergenz
-

¹Dieses Verfahren ist die Grundlage für `ARPACK` („ARNoldi PACKage“), eines der führenden Softwarepakete für die numerische Berechnung von Eigenwerten großer, dünn besetzter Matrizen. Auf `ARPACK` basiert auch MATLABs „eigs“-Funktion.

Teil IV

AUSGLEICHSPROBLEME

KRYLOVRAUM-VERFAHREN

13

Wir kommen nun zu Verfahren für nichtquadratische Matrizen $A \in \mathbb{C}^{m \times n}$ mit $m \neq n$. In diesem Fall ist das lineare Gleichungssystem $Ax = b$ im Allgemeinen nicht eindeutig lösbar. Wir unterscheiden nun die folgenden Fälle:

- (i) $m > n$: Das Gleichungssystem ist *überbestimmt*; für gegebenes $b \in \mathbb{C}^m$ muss kein $x \in \mathbb{C}^n$ existieren mit $Ax = b$.
- (ii) $m < n$: Das Gleichungssystem ist *unterbestimmt*; zwar existiert für jedes $b \in \mathbb{C}^m$ ein $x \in \mathbb{C}^n$ mit $Ax = b$, dieses ist jedoch nicht eindeutig.

In beiden Fällen muss man den Lösungsbegriff daher erweitern. In Folge nehmen wir der Einfachheit halber an, dass $\text{rank } A = \min\{m, n\}$ gilt, und damit genau einer der beiden Fälle eintritt. Für diese leiten wir nun Krylovraum-Verfahren her.

13.1 DAS CGLS-VERFAHREN

Ist das Gleichungssystem $Ax = b$ überbestimmt, so sucht man anstelle der nicht-existent Lösung ein $x^* \in \mathbb{C}^n$, welches das Residuum minimiert, d. h. eine Lösung ist des *Ausgleichs-* oder *Kleinste-Quadrate-Problems*

$$(13.1) \quad \min_{x \in \mathbb{C}^n} \|Ax - b\|_2^2.$$

Aus Satz 5.2 (mit $\mathcal{K} = \mathbb{C}^n$ und $x^0 = 0$) wissen wir, dass eine Ausgleichslösung die Galerkin-Bedingungen

$$\begin{cases} \tilde{x} \in \mathbb{C}^n \\ b - A\tilde{x} \perp \text{Im}(A) \end{cases}$$

erfüllen muss. Wegen $\text{Im}(A)^\perp = \text{Ker}(A^*)$ ist dies äquivalent zu $A^*(b - Ax) = 0$, woraus wir die *Normalengleichungen*

$$A^*Ax = A^*b$$

erhalten. Offensichtlich ist A^*A selbstadjungiert; da wir $\text{rank } A = n$ angenommen haben, ist A^*A auch positiv definit: Für alle $x \in \mathbb{C}^n$ gilt

$$\langle x, A^*Ax \rangle = \langle Ax, Ax \rangle = \|Ax\|_2^2 \geq 0.$$

Angenommen, es gäbe ein $x \in \mathbb{C}^n$ mit $\langle x, A^*Ax \rangle = 0$ und damit $Ax = 0$. Wegen $\text{rank } A = n$ ist A aber injektiv, und deshalb muss $x = 0$ sein. Also hat (13.1) eine eindeutige Lösung, die man mit Hilfe des CG-Verfahrens berechnen kann.

Aus Stabilitätsgründen formuliert man Algorithmus (7.1) leicht um:

- (i) Das explizite Aufstellen von A^*A wird vermieden; stattdessen multipliziert man separat mit A und A^* .
- (ii) Man führt die Rekursion für das ursprüngliche Residuum $r^k = b - Ax^k$ durch statt für $s^k := A^*r^k = A^*b - A^*Ax^k$.

Dies führt auf das *CGLS-Verfahren* (für „least squares“).¹

Algorithmus 13.1 CGLS-Verfahren

Input: $A, b, x^0, \varepsilon > 0$

- 1: $r^0 = b - Ax^0, p_0 = s^0 = A^*r^0, k = 0$
- 2: **while** $\|s^k\|_2 > \varepsilon$ **do**
- 3: $k \leftarrow k + 1$
- 4: $q^k = Ap_{k-1}$
- 5: $\alpha_k = \|s^{k-1}\|_2^2 / \|q^k\|_2^2$
- 6: $x^k = x^{k-1} + \alpha_k p_{k-1}$
- 7: $r^k = r^{k-1} - \alpha_k q^k$
- 8: $s^k = A^*r^k$
- 9: $\beta_k = \|r^k\|_2^2 / \|r^{k-1}\|_2^2$
- 10: $p_k = s^k + \beta_k p_{k-1}$

Output: x^k

Aus der Minimierungseigenschaft des CG-Verfahrens (Satz 5.1) folgt sofort, dass die Iterierten x^k die Norm

$$\|x^* - x^k\|_{A^*A}^2 = \|(Ax^* - b) - (Ax^k - b)\|_2^2 = \|r^* - r^k\|_2^2$$

über den Krylovraum $x^0 + \mathcal{K}_k(A^*A, A^*b)$ minimieren. Auch die Konvergenz folgt direkt aus dem entsprechenden Konvergenzsatz 7.3:

$$\|r^* - r^k\|_2 \leq 2 \left(\frac{\kappa(A) - 1}{\kappa(A) + 1} \right)^k \|r^* - r^0\|_2,$$

wobei wir $\kappa(A^*A) = \kappa(A)^2$ verwendet haben.

¹zusammen mit dem CG-Verfahren publiziert durch Magnus Hestenes und Eduard Stiefel in [Hestenes und Stiefel 1952]

13.2 DAS CGME-VERFAHREN

Ist das Gleichungssystem $Ax = b$ dagegen unterbestimmt, so existiert eine Lösung, die jedoch nicht eindeutig ist. Wir können aber unter allen Lösungen diejenige Lösung $x^* \in \mathbb{C}^n$ aussuchen, die minimale Norm $\|x^*\|_2$ besitzt. Sei dafür $\tilde{x} \in \mathbb{C}^n$ mit $A\tilde{x} = b$ beliebig. Mit Hilfe der orthogonalen Zerlegung $\mathbb{C}^n = \text{Ker}(A) \oplus \text{Ker}(A)^\perp$ können wir schreiben $\tilde{x} = \tilde{x}_1 + \tilde{x}_2$ mit $\tilde{x}_1 \in \text{Ker}(A)$ und $\tilde{x}_2 \in \text{Ker}(A)^\perp$. Dann gilt $b = A\tilde{x} = A\tilde{x}_2$ sowie

$$\|\tilde{x}\|_2^2 = \|\tilde{x}_1\|_2^2 + \|\tilde{x}_2\|_2^2,$$

was offensichtlich minimal ist für $\tilde{x}_1 = 0$. Eine Lösung minimaler Norm liegt also notwendig in $\text{Ker}(A)^\perp = \text{Im}(A^*)$, d. h. es existiert ein $y \in \mathbb{C}^m$ mit

$$\begin{cases} Ax = b, \\ x = A^*y. \end{cases}$$

Einsetzen der zweiten Gleichung in die erste ergibt dann die Normalgleichungen

$$(13.2) \quad AA^*y = b.$$

Wieder ist $AA^* \in \mathbb{C}^{m \times m}$ selbstadjungiert und wegen $\text{rank } A = m$ positiv definit. Also hat (13.2) ebenfalls eine eindeutige Lösung, die man mit Hilfe des CG-Verfahrens berechnen kann.

Eliminiert man in Algorithmus (7.1) die Rekursion für y^k durch Einführen von $x^k := A^*y^k$, so erhält man daraus das *CGME-Verfahren* (für „minimal error“).² Hierbei wird üblicherweise mit $y^0 = x^0 = 0$ gestartet.

Algorithmus 13.2 CGME-Verfahren

Input: $A, b, \varepsilon > 0$

1: $x^0 = 0, p^0 = r^0 = b - Ax^0, k = 0$

2: **while** $\|r^k\|_2 > \varepsilon$ **do**

3: $k \leftarrow k + 1$

4: $q^k = A^*p_{k-1}$

5: $\alpha_k = \|r^{k-1}\|_2^2 / \|q^k\|_2^2$

6: $x^k = x^{k-1} + \alpha_k q^k$

7: $r^k = r^{k-1} - \alpha_k Aq^k$

8: $\beta_k = \|r^k\|_2^2 / \|r^{k-1}\|_2^2$

9: $p_k = r^k + \beta_k p_{k-1}$

Output: x^k

²publiziert durch Edward J. Craig in [Craig 1955]

Wieder folgt aus der Minimierungseigenschaft des CG-Verfahrens die Optimalität von

$$\|y^* - y^k\|_{AA^*}^2 = \|A^*y^* - A^*y^k\|_2^2 = \|x^* - x^k\|_2^2$$

mit $x^* := A^*y^*$, wobei $x^k \in A^*\mathcal{K}_k(AA^*, b) = \mathcal{K}_k(A^*A, A^*b)$ liegt. Aus Satz 7.3 folgt wiederum die Fehlerabschätzung

$$\|x^* - x^k\|_2 \leq 2 \left(\frac{\kappa(A) - 1}{\kappa(A) + 1} \right)^k \|x^* - x^0\|_2.$$

DAS LSQR-VERFAHREN

14

Ein alternatives Verfahren für überstimmte Gleichungssysteme basiert auf einer Kombination der Ideen hinter dem Bi-Lanczos- und dem GMRES-Verfahren.

14.1 DER PAIGE–SAUNDERS–LANCZOS-PROZESS

Wir beginnen mit einer weiteren Variante des Lanczos-Prozesses: Diesmal versuchen wir, eine Matrix $A \in \mathbb{C}^{m \times n}$ mit $m > n$ durch unitäre Transformationen auf *Bidiagonalform* zu bringen. (Im Vergleich zum Lanczos-Prozess geben wir also die Ähnlichkeit auf; im Vergleich zum Bi-Lanczos-Prozess fordern wir Unitarität im Gegensatz zur Biorthogonalität.) Wir suchen also zwei unitäre Matrizen $V_n \in \mathbb{C}^{n \times n}$ und $W_m \in \mathbb{C}^{m \times m}$ so, dass gilt

$$W_m^* A V_n = \begin{pmatrix} \bar{B}_n \\ 0 \end{pmatrix} =: B_n \quad \text{mit} \quad \bar{B}_n = \begin{pmatrix} \alpha_1 & & & & \\ \beta_2 & \alpha_2 & & & \\ & \beta_3 & \ddots & & \\ & & \ddots & \alpha_n & \\ & & & \beta_{n+1} & \end{pmatrix} \in \mathbb{R}^{(n+1) \times n},$$

wobei wir wieder der Einfachheit halber angenommen haben, dass $\text{rank}(A) = n$ ist. Genau wie für das Bi-Lanczos-Verfahren leiten wir aus diesen Bedingungen eine Rekursion sowohl für die Spalten von V_n und W_m als auch die Einträge von \bar{B}_n her. Dazu betrachten wir wieder $AV_n = W_m B_n$ sowie $A^* W_m = V_n B_n^*$ spaltenweise. Wir erhalten dann für alle $1 \leq k \leq n$

$$(14.1) \quad Av_k = \alpha_k w_k + \beta_{k+1} w_{k+1},$$

$$(14.2) \quad A^* w_{k+1} = \beta_{k+1} v_k + \alpha_{k+1} v_{k+1}.$$

Wählen wir $w_1 \in \mathbb{C}^m$ beliebig und setzen $\tilde{v}_1 := \alpha_1 v_1 = A^* w_1$, so erhalten wir für $\tilde{v}_k := \alpha_k v_k$ und $w_k := \beta_k w_k$, $1 \leq k \leq n$ die Rekursion

$$\begin{aligned} \tilde{w}_{k+1} &= Av_k - \alpha_k w_k, \\ \tilde{v}_{k+1} &= A^* w_{k+1} - \beta_{k+1} v_k, \end{aligned}$$

wobei wir α_k und β_{k+1} (reell und positiv) jeweils so wählen, dass $\|v_{k+1}\|_2 = \|w_{k+1}\|_2 = 1$ gilt. Wie für den Bi-Lanczos-Prozess zeigt man durch starke Induktion und Einsetzen der Rekursionsvorschrift, dass die so erzeugten \tilde{v}_k bzw. \tilde{w}_k (und damit v_k bzw. w_k) orthogonal sind.

Wir erhalten dadurch den *Paige–Saunders–Lanczos-Prozess*.¹

Algorithmus 14.1 Paige–Saunders–Lanczos-Prozess

Input: $A \in \mathbb{C}^{m \times n}$, $w_1 \in \mathbb{C}^n \setminus \{0\}$

- 1: $v_0 = 0$, $\beta_1 = \|w_1\|_2$
- 2: **for** $j = 1, \dots, k$ **do**
- 3: $w_j \leftarrow w_j / \beta_j$
- 4: $v_j = A^* w_j - \beta_j v_{j-1}$
- 5: $\alpha_j = \|v_j\|_2$
- 6: **if** $\alpha_j > 0$ **then**
- 7: $v_j \leftarrow v_j / \alpha_j$
- 8: $w_{j+1} = A v_j - \alpha_j w_j$
- 9: $\beta_{j+1} = \|w_{j+1}\|_2$
- 10: **if** $\beta_j = 0$ **then** Stop

Output: $v_j, \alpha_j, 1 \leq j \leq k, \beta_j, w_j, 1 \leq j \leq k+1$

Wieder können wir (14.1) und (14.2) für $1 \leq k \neq n$ in Matrixform schreiben als

$$(14.3) \quad AV_k = W_{k+1} \bar{B}_k,$$

$$(14.4) \quad A^* W_{k+1} = V_k \bar{B}_k^* + \alpha_{k+1} v_{k+1} e_{k+1}^T.$$

Ebenso folgt aus (14.1) und (14.2) per Induktion, dass gilt

$$\begin{aligned} \text{span}\{v_1, \dots, v_k\} &= \mathcal{K}_k(A^*A, A^*w_1), \\ \text{span}\{w_1, \dots, w_k\} &= \mathcal{K}_k(AA^*, w_1). \end{aligned}$$

14.2 LSQR-VERFAHREN

Wir gehen nun analog zur Herleitung des GMRES-Verfahrens vor: Als Approximation x^k der gesuchten Ausgleichslösung verwenden wir die Lösung von

$$(14.5) \quad \min_{x \in \mathcal{K}_k(A^*A, A^*b)} \|Ax - b\|_2.$$

¹publiziert durch Christopher C. Paige und Michael A. Saunders in [Paige und Saunders 1982]

(Die Iterierten x^k sind also in exakter Arithmetik identisch mit denen des CGLS-Verfahrens; der folgende Algorithmus ist aber in der Praxis insbesondere für schlechtkonditionierte Matrizen stabiler.) Starten wir den Paige–Saunders–Lanczos-Prozess mit $w_1 = b$, so erhalten wir nach k Schritten eine unitäre Matrix V_k , deren Spalten eine Orthonormalbasis von $\mathcal{K}_k(A^*A, A^*b)$ bilden, sowie eine unitäre Matrix W_{k+1} mit (14.3). Also ist die Lösung von (14.5) gegeben durch $\bar{x} = V_k \bar{\xi}$, wobei $\bar{\xi} \in \mathbb{R}^k$ die Lösung ist von

$$\min_{\xi \in \mathbb{R}^k} \|\beta_1 e_1 - \bar{B}_k \xi\|_2$$

mit $\beta_1 = \|b\|_2$. Wieder lösen wir dieses Problem mit Hilfe einer QR-Zerlegung, die wir für $k = 1, \dots$ spaltenweise mit Givens-Rotationen aufbauen können; da \bar{B}_k bidiagonal ist, vereinfacht sich das Verfahren aber wesentlich.

Angenommen, wir haben im Schritt $k = 1$ die neuen Koeffizienten α_2 und β_2 berechnet. Wir berechnen nun eine Givens-Rotation $G_{2,1}$, um das Element β_2 zu eliminieren, und wenden diese auf die rechte Seite $\beta_1 e_1$ an:

$$G_{2,1} \begin{pmatrix} \alpha_1 & 0 \\ \beta_2 & \alpha_2 \end{pmatrix} = \begin{pmatrix} r_{1,1} & r_{1,2} \\ 0 & r_{2,2} \end{pmatrix}, \quad G_{2,1} \begin{pmatrix} \beta_1 \\ 0 \end{pmatrix} = \begin{pmatrix} d_1 \\ \rho_2 \end{pmatrix}.$$

(Für die Elemente c_1, s_1 von $G_{2,1}$ und die neuen Einträge $r_{1,1}, r_{1,2}, r_{2,2}$ kann man geschlossene Formeln angeben.) Da die folgenden Rotationen nicht auf die erste Spalte wirken, sind $r_{1,1}, r_{1,2}$ und d_1 bereits final; die anderen Einträge werden in der nächsten Iteration überschrieben. Im nächsten Schritt $k = 2$ erzeugt nun der Paige–Saunders–Lanczos-Prozess eine weitere Spalte und Zeile von \bar{B}_2 ; dabei enthält die neue Spalte jedoch nur Nullen in den ersten beiden Zeilen. Im Gegensatz zum GMRES-Verfahren muss daher die Rotation $G_{2,1}$ gar nicht mehr auf die neue Spalte angewendet werden und kann nach der ersten Iteration gleich verworfen werden. Es muss also nur noch die neue Rotation $G_{3,2}$ berechnet und angewendet werden. Nach k Schritten erhalten wir also für $G_k = G_{k+1,k} \circ \dots \circ G_{2,1}$

$$G_k \bar{B}_k = \begin{pmatrix} R_k \\ 0 \end{pmatrix} \quad \text{mit} \quad R_k = \begin{pmatrix} r_{1,1} & r_{1,2} & & & \\ & r_{2,2} & \ddots & & \\ & & \ddots & r_{k-1,k} & \\ & & & & r_{k,k} \end{pmatrix}, \quad G_k \begin{pmatrix} \beta_1 \\ 0 \end{pmatrix} = \begin{pmatrix} d_k \\ \rho_k \end{pmatrix}.$$

Damit ist wieder $\xi^k = R_k^{-1} d_k$ und $\|\beta_1 e_1 - \bar{B}_k \xi^k\|_2 = |\rho_k|$, wobei das Gleichungssystem für die obere Dreiecksmatrix R_k durch Rückwärtssubstitution gelöst werden kann.

Tatsächlich können wir aufgrund der Bidiagonalstruktur von R_k sogar eine Rekursion für $x^k = V_k \xi^k = V_k R_k^{-1} d_k =: H_k d_k$ angeben. Schreiben wir $H_k = [h_1, \dots, h_k]$ und $d_k = (\theta_1, \dots, \theta_k)^T$, so gilt zunächst mit $x^0 = 0$

$$x^k = H_k d_k = \sum_{j=1}^k \theta_j h_j = \sum_{j=1}^{k-1} \theta_j h_j + \theta_k h_k = x^{k-1} + \theta_k h_k.$$

Da die θ_k bereits berechnet sind, brauchen wir nur noch eine Rekursion für die Spalten h_k . Dafür betrachten wir $H_k R_k = V_k$ spaltenweise:

$$[H_{k-1}, h_j] \begin{pmatrix} R_{k-1} & r_{k-1,k} e_k \\ 0 & r_{k,k} \end{pmatrix} = [V_{k-1}, v_k].$$

Aus der ersten (Block-)Spalte erhält man durch Ausmultiplizieren $H_{k-1} R_{k-1} = V_{k-1}$; aus der letzten Spalte durch Auflösen nach h_k erhält man daher rekursiv

$$h_1 = v_1 r_{1,1}^{-1}, \quad h_k = (v_k - r_{k-1,k} h_{k-1}) r_{k,k}^{-1} \quad \text{für } k > 1.$$

Damit ist in jedem Schritt nur noch die Information aus dem jeweils letzten Schritt benötigt. Die Rekursion kann noch etwas vereinfacht werden durch Einführen von $u_k = r_{k,k} h_k$: Dann gilt

$$u^1 = v_1, \quad x^k = x^{k-1} + \frac{\theta_k}{r_{k,k}} u^k, \quad u^{k+1} = v_{k+1} - \frac{r_{k,k+1}}{r_{k,k}} u^k.$$

Bleibt nur noch das Abbruchkriterium. Zwar ist die Residuumsnorm $\|b - Ax^k\|_2$ als Teil der Rekursion berechnet worden; für die Ausgleichsrechnung benötigen wir aber das Residuum der Normalengleichung $s^k := A^* b - A^* A x^k = A^* r^k$. Zunächst folgt aus der obigen Konstruktion

$$\begin{aligned} r^k &= b - Ax^k = W_{k+1} (\beta_1 e_1 - \bar{B}_k \xi^k) \\ &= W_{k+1} G_k^* \left(\begin{pmatrix} d_k \\ \rho_k \end{pmatrix} - \begin{pmatrix} R_k \\ 0 \end{pmatrix} \right) \\ &= \rho_k W_{k+1} G_k^* e_{k+1}, \end{aligned}$$

da G_k unitär ist und $R_k \xi^k = d_k$ gilt. Aus (14.4) folgt nun

$$\begin{aligned} s^k &:= A^* r^k = \rho_k (V_k \bar{B}_k^* + \alpha_{k+1} v_{k+1} e_{k+1}^T) G_k^* e_{k+1} \\ &= \rho_k V_k (G_k \bar{B}_k)^* e_{k+1} + \rho_k \alpha_{k+1} (e_{k+1}^T G_k^* e_{k+1}) v_{k+1} \\ &= -\rho_k \alpha_{k+1} c_k v_{k+1}, \end{aligned}$$

da das Produkt $(G_k \bar{B}_k)^* e_{k+1} = [R_k^*, 0] e_{k+1}$ verschwindet und der $(k+1)$ te Diagonaleintrag von G_k genau $-c_k$ ist. Da die v_k eine Orthonormalbasis bilden, ist also

$$\|s^k\|_2 = \alpha_{k+1} |c_k \rho_k|.$$

Zusammen erhalten wir das *LSQR-Verfahren*.²

²ebenfalls publiziert durch Christopher C. Paige und Michael A. Saunders in [Paige und Saunders 1982]

Algorithmus 14.2 LSQR-Verfahren**Input:** $A, b, \varepsilon > 0$

- 1: $x^0 = 0, \beta_1 = \|b\|_2$
- 2: $w_1 = b/\beta_1, v_1 = A^*w_1, \alpha_1 = \|v_1\|_2, v_1 \leftarrow v_1/\alpha_1$ ▷ Erster Schritt im PSL-Prozess
- 3: $u^1 = v_1, r_{1,1} = \alpha_1, \rho_1 = \beta_1$
- 4: **for** $k = 1, \dots$ **do**
- 5: $w_{k+1} = Av_k - \alpha_k w_k$ ▷ k-ter Schritt im PSL-Prozess
- 6: $\beta_{k+1} = \|w_{k+1}\|_2, w_{k+1} \leftarrow w_{k+1}/\beta_{k+1}$
- 7: $v_{k+1} = A^*w_{k+1} - \beta_{k+1}v_k$
- 8: $\alpha_{k+1} = \|v_{k+1}\|_2, v_{k+1} \leftarrow v_{k+1}/\alpha_{k+1}$
- 9: $\sigma = \|(r_{k,k}, \beta_{k+1})\|_2$ ▷ Berechne Givens-Rotation $G_{k+1,k}$
- 10: $c_k = r_{k-1,k-1}/\sigma, s_k = \beta_{k+1}/\sigma$
- 11: $r_{k,k+1} = s_k \alpha_{k+1}, r_{k+1,k+1} = -c_k \alpha_{k+1}$ ▷ Anwenden auf \bar{B}_k
- 12: $\theta_k = c_k \rho_k, \rho_{k+1} = s_k \rho_k$ ▷ Anwenden auf $\beta_1 e_1$
- 13: $x^{k+1} = x^k + (\theta_k/r_{k,k})u^k$ ▷ Näherung aktualisieren
- 14: $u^{k+1} = v^k - (r_{k,k+1}/r_{k,k})u^k$
- 15: **if** $\alpha_{k+1}|c_k \rho_k| \leq \varepsilon$ **then Stop**

Output: x^k

LITERATUR

- Å. Björck (2015). *Numerical Methods in Matrix Computations*. Bd. 59. Texts in Applied Mathematics. Springer, Cham. DOI: [10.1007/978-3-319-05089-8](https://doi.org/10.1007/978-3-319-05089-8).
- J. Boyd (2013). *Finding the Zeros of a Univariate Equation: Proxy Rootfinders, Chebyshev Interpolation, and the Companion Matrix*. SIAM Review 55.2, S. 375–396. DOI: [10.1137/110838297](https://doi.org/10.1137/110838297).
- E. J. Craig (1955). *The N-step iteration procedures*. J. Math. and Phys. 34, S. 64–73. DOI: [10.1002/sapm195534164](https://doi.org/10.1002/sapm195534164).
- W. Gautschi (2012). *Numerical Analysis*. 2. Aufl. Birkhäuser, Boston. DOI: [10.1007/978-0-8176-8259-0](https://doi.org/10.1007/978-0-8176-8259-0).
- G. H. Golub und C. F. Van Loan (2013). *Matrix Computations*. 4. Aufl. Johns Hopkins University Press, Baltimore, MD.
- M. H. Gutknecht und B. N. Parlett (2011). *From qd to LR, or, how were the qd and LR algorithms discovered?* IMA Journal of Numerical Analysis 31.3, S. 741–754. DOI: [10.1093/imanum/drq003](https://doi.org/10.1093/imanum/drq003).
- M. R. Hestenes und E. Stiefel (1952). *Methods of conjugate gradients for solving linear systems*. J. Research Nat. Bur. Standards 49, 409–436 (1953). DOI: [10.6028/jres.049.044](https://doi.org/10.6028/jres.049.044).
- C. C. Paige und M. A. Saunders (1982). *LSQR: an algorithm for sparse linear equations and sparse least squares*. ACM Trans. Math. Software 8.1, S. 43–71. DOI: [10.1145/355984.355989](https://doi.org/10.1145/355984.355989).
- Y. Saad und M. H. Schultz (1986). *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*. SIAM J. Sci. Statist. Comput. 7.3, S. 856–869. DOI: [10.1137/0907058](https://doi.org/10.1137/0907058).
- Y. Saad (2003). *Iterative Methods For Sparse Linear Systems*. 2. Aufl. SIAM, Philadelphia, PA. URL: http://www-users.cs.umn.edu/~saad/IterMethBook_2ndEd.pdf.
- Y. Saad (2011). *Numerical Methods for Large Eigenvalue Problems*. 2. Aufl. SIAM, Philadelphia, PA. URL: http://www-users.cs.umn.edu/~saad/eig_book_2ndEd.pdf.
- H. A. van der Vorst (1992). *Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems*. SIAM Journal on Scientific and Statistical Computing 13.2, S. 631–644. DOI: [10.1137/0913035](https://doi.org/10.1137/0913035).

- H. A. van der Vorst (2009). *Iterative Krylov Methods for Large Linear Systems*. Cambridge University Press, Cambridge.
- D. S. Watkins (2007). *The Matrix Eigenvalue Problem*. GR and Krylov subspace methods. SIAM, Philadelphia, PA. DOI: [10.1137/1.9780898717808](https://doi.org/10.1137/1.9780898717808).
- D. S. Watkins (2008). *The QR algorithm revisited*. SIAM Review 50.1, S. 133–145. DOI: [10.1137/060659454](https://doi.org/10.1137/060659454).