

Einführung in die Grundlagen der Numerik (WS 22/23)

Manuel Hinz

24. November 2022

Inhaltsverzeichnis

1	Orthogonalität	3
1.1	Grundlegende Definitionen	3
1.2	Bestapproximationseigenschaft	4
1.3	Orthonormalbasen	5
2	Das lineare Ausgleichsproblem	7
2.1	Problemstellung und Normalengleichung	7
2.2	Methode der Orthogonalisierung	9
2.3	Grundüberlegungen zu Orthogonalisierungsverfahren	10
2.4	QR -Zerlegung mittels Givens-Rotationen	11
2.5	QR -Zerlegung mittels Householder-Transformationen	13
2.6	Pseudoinverse	14
3	Iterative Verfahren für große, dünn besetzte, Gleichungssysteme	18
3.1	Motivation	18
3.2	Grundidee von Projektionsmethoden	19
3.3	Verfahren des steilsten Abstiegs	21
3.4	Krylovräume	22
3.5	Arnoldi-Verfahren	23
3.6	Verfahren der vollständigen Orthogonalisierung	24
3.7	Das GMRES-Verfahren	26
3.8	Der symmetrische Lanczos-Prozess	28
3.9	Das Verfahren der konjugierten Gradienten	30
4	Lineare Eigenwertprobleme	35
4.1	Problemstellung und Beispiele	35
4.2	Gerschgorin-Kreise (Abschätzungen für EW)	37
4.3	Kondition des Eigenwertproblems	38
4.4	Vektoriteration (Potenzmethode, etc)	38

Vorwort

Diese Mitschrift von der Vorlesung Einführung in die Grundlagen der Numerik (Dölz, WS 2022/2023) wird von mir neben der Vorlesung geschrieben und ist dementsprechend Fehleranfällig. Fehler gerne an mh@mssh.dev!

Kapitel 1

Orthogonalität

1.1 Grundlegende Definitionen

Definition 1.1. Sei X ein \mathbb{R} -Vektorraum und $\langle \cdot, \cdot \rangle : X \times X \rightarrow \mathbb{R}$ eine Abbildung. $\langle \cdot, \cdot \rangle$ heißt **Skalarprodukt** oder inneres Produkt, falls

$$\forall f \in X \setminus \{0\} : \langle f, f \rangle > 0 \quad (\text{Positivität})$$

$$\forall f, g \in X : \langle f, g \rangle = \langle g, f \rangle \quad (\text{Symmetrie})$$

$$\forall \alpha, \beta \in \mathbb{R}, f, g, h \in X : \langle \alpha f + \beta g, h \rangle = \alpha \langle f, h \rangle + \beta \langle g, h \rangle \quad (\text{Linearität im ersten Argument})$$

Bemerkung 1.2. Symmetrie und Linearität im ersten Argument implizieren, dass $\langle \cdot, \cdot \rangle$ eine bilineare Abbildung ist.

Definition 1.3. Sei X ein \mathbb{R} -Vektorraum mit Skalarprodukt $\langle \cdot, \cdot \rangle$. Wir bezeichnen die zugehörige **Norm** (in Abhängigkeit von einem Vektor $f \in X$) mit

$$\|f\| = \sqrt{\langle f, f \rangle}.$$

Lemma 1.4. Sei X ein \mathbb{R} -Vektorraum mit Skalarprodukt $\langle \cdot, \cdot \rangle$. Dann gilt die Cauchy-Schwarz-Ungleichung:

$$\forall f, g \in X : \langle f, g \rangle \leq \|f\| \cdot \|g\| \quad (\text{C.S.})$$

mit Gleichheit genau dann, wenn f und g linear abhängig sind.

Beweis. O.B.d.A. $f, g \neq 0$, da sonst offensichtlich Gleichheit gilt. Sei $\alpha \neq 0$, dann gilt mit $f, g \in X$ und $\alpha \in \mathbb{R}$:

$$0 \leq \|f - \alpha g\|^2 = \langle f - \alpha g, f - \alpha g \rangle = \|f\|^2 - 2\alpha \langle f, g \rangle + \alpha^2 \|g\|^2$$

Wählen wir jetzt $\alpha = \frac{\langle f, g \rangle}{\|g\|^2}$ folgt:

$$\begin{aligned} 0 &\leq \|f\|^2 - \frac{2\langle f, g \rangle^2}{\|g\|^2} + \frac{\langle f, g \rangle^2}{\|g\|^2} \\ &\implies \langle f, g \rangle^2 \leq \|f\|^2 \cdot \|g\|^2. \end{aligned}$$

□

Eingefügte Bemerkung. Rechnung zur Begründung von $\langle f - \alpha g, f - \alpha g \rangle = \|f\|^2 - 2\alpha \langle f, g \rangle + \alpha^2 \|g\|^2$:

$$\begin{aligned} &\langle f - \alpha g, f - \alpha g \rangle \\ &= \langle f, f - \alpha g \rangle - \alpha \langle g, f - \alpha g \rangle \\ &= \langle f, f \rangle - \alpha \langle f, g \rangle - \alpha \langle g, f \rangle + \alpha^2 \langle g, g \rangle \\ &= \|f\|^2 - 2\alpha \langle f, g \rangle + \alpha^2 \|g\|^2 \end{aligned}$$

Beispiel 1.5. 1. $X = \mathbb{R}^n$ und $\langle x, y \rangle = \sum_{i=1}^n x_i y_i$ (Euklidisches Skalarprodukt)

2. $X = \mathbb{R}^n$, $\langle x, y \rangle = x^\top A y$, wobei A positiv definit und symmetrisch ist

3. $I = [a, b]$, $w : I \rightarrow \mathbb{R}$ beschränkt und strikt positiv:

$$X = \left\{ f : I \rightarrow \mathbb{R} : \int_a^b f(x)^2 w(t) dt < \infty \right\} = L^2(I, w)$$

mit

$$\langle f, g \rangle = \int_a^b f(t)g(t)w(t)dt$$

Eingefügte Bemerkung. Die Definition von $L^2(I, w)$ ist hier nicht ganz richtig, man müsste natürlich noch Äquivalenzklassen, bzgl. Gleichheit bis auf Nullmengen, bilden. Dies wird hier, da Analysis 3 / Wtheo. nicht nicht vorausgesetzt wird, ignoriert.

Definition 1.6. Sei X ein \mathbb{R} -VR mit Skalarprodukt $\langle \cdot, \cdot \rangle$. $f, g \in X$ heißen **orthogonal**, falls $\langle f, g \rangle = 0$.

Bemerkung 1.7. Im \mathbb{R}^n mit dem euklidischen Skalarprodukt stimmt Definition 1.6, wegen

$$\langle x, y \rangle = \|x\| \|y\| \cos(\theta), \theta = \angle(x, y),$$

mit unserem bisherigen Verständnis überein.

1.2 Bestapproximationseigenschaft

Definition 1.8. Sei V ein \mathbb{R} -VR mit Skalarprodukt $\langle \cdot, \cdot \rangle$ und U ein Unterraum.

$$U^\perp = \{v \in V : \langle v, u \rangle = 0, \forall u \in U\}$$

heißt das **orthogonale Komplement** von U .

Satz 1.9. Unter den Annahmen von Definition 1.8 und der zusätzlichen Annahme, dass U endlich dimensional ist, gilt folgendes für $v \in V$:

$$\|v - u\| = \min_{w \in U} \|v - w\|$$

genau dann, wenn $v - u \in U^\perp$.

Beispiel 1.10. $V = \mathbb{R}^2$, $U = \text{span} \left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}$ mit euklidischem Skalarprodukt $\langle \cdot, \cdot \rangle$. Dann ist $U^\perp = \text{span} \left\{ \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right\}$.

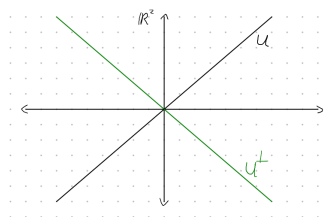


Abbildung 1.1: U und U^\perp

Beweis von Satz 1.9. Sei $v \in V$ und seien $u, w \in U$. Dann gilt:

$$\begin{aligned} \|v - w\|^2 &= \langle v - w, v - w \rangle = \langle (v - u) + (u - w), (v - u) + (u - w) \rangle \\ &= \|v - u\|^2 + 2 \underbrace{\langle v - u, u - w \rangle}_{\in U} + \|u - w\|^2 \geq \|v - u\|^2 \end{aligned}$$

mit Gleichheit genau dann, wenn $w - u = 0$ (da dann der $\|u - w\|$ Term verschwindet). □

Bemerkung 1.11. Der Satz sagt, dass es zu jedem $v \in V$ ein eindeutiges, bestmögliches $u \in U$ gibt.

Definition 1.12. Die Lösung aus Satz 1.9 heißt **orthogonale Projektion** von v auf U . Die Abbildung

$$P : V \rightarrow U, v \mapsto P(v) \text{ mit } \|v - Pv\| = \min_{w \in U} \|v - w\|$$

ist linear und wird **orthogonale Projektion** genannt.

Eingefügte Bemerkung (Beweis der Linearität). Für $v_1, v_2 \in V$ und $\alpha \in \mathbb{R}$ gilt:

$$\begin{aligned} v_1 - Pv_1 &\in U^\perp \\ v_2 - Pv_2 &\in U^\perp \end{aligned}$$

Daher

$$\alpha(v_1 - Pv_1) + (v_2 - Pv_2) = (\alpha v_1 + v_2) - (\alpha Pv_1 + Pv_2) \in U^\perp.$$

Aber dann muss $\alpha Pv_1 + Pv_2$ schon, wegen der Eindeutigkeit, $P(\alpha v_1 + v_2)$ sein.

Bemerkung 1.13. Satz 1.9 gilt auch, wenn U durch $W = w_0 + U$ ersetzt wird. Die orthogonale Projektion ist analog definiert..

Frage: Die Orthogonale Projektion hat offenbar gute Eigenschaften. Aber: wie berechnen wir sie? Wie wählen wir U ?

- Berechnung ist leicht
- U wählen schwierig

1.3 Orthonormalbasen

Definition 1.14. Sei X ein \mathbb{R} -VR mit Skalarprodukt $\langle \cdot, \cdot \rangle$ und $X_n \subset X$ ein endlich dimensionaler Teilraum mit Basis $\{\varphi_1, \dots, \varphi_n\}$. Die Basis heißt **Orthogonalbasis**, falls

$$\forall i \neq j : \langle \varphi_i, \varphi_j \rangle = 0$$

gilt und Orthonormalbasis (ONB), falls zusätzlich $\|\varphi_i\| = 1$ gilt. Das impliziert:

$$\langle \varphi_i, \varphi_j \rangle = \delta_{i,j}.$$

Beispiel 1.15. 1. \mathbb{R}^n mit euklidischem Skalarprodukt und kanonischer Basis

2. $X = L^2(I, 1)$ mit entsprechendem Skalarprodukt und X_n der Raum der trigonometrischen Polynome bis Grad n . Dann ist folgendes eine ONB:

$$\left\{ \frac{1}{\sqrt{2\pi}}, \frac{\sin(x)}{\sqrt{\pi}}, \frac{\cos(x)}{\sqrt{\pi}}, \dots, \frac{\sin(nx)}{\sqrt{\pi}}, \frac{\cos(nx)}{\sqrt{\pi}} \right\}$$

Eingefügte Bemerkung. Trigonometrische Polynome sind Funktionen der Form

$$f(t) = \sum_{k=1}^n a_k \cos(kx) + b_k \sin(kx).$$

Die größte Faktor vor dem x ist der Grad eines trigonometrischen Polynoms.

Satz 1.16. Sei $\{\varphi_1, \dots, \varphi_n\}$ eine ONB von $X_n \subset X$. Dann gilt

$$1. f = \sum_{i=1}^n \langle \varphi_i, f \rangle \varphi_i$$

2. $\|f\|^2 = \sum_{i=1}^n \langle \varphi_i, f \rangle^2$

3. Die orthogonale Projektion f_n von $f \in X \setminus X_n$ ist gegeben durch

$$f_n = \sum_{i=1}^n \langle \varphi_i, f \rangle \varphi_i$$

4. im Fall von 3.:

$$\|f_n\|^2 = \sum_{i=1}^n \langle \varphi_i, f \rangle^2 \leq \|f\|^2$$

Beweis. 1.:

$$\begin{aligned} f \in X_n &\implies \exists \alpha_i \in \mathbb{R} : f = \sum_{i=1}^n \alpha_i \varphi_i \\ \implies \langle \varphi_i, f \rangle &= \langle \varphi_i, \sum_{j=1}^n \alpha_j \varphi_j \rangle = \sum_{j=1}^n \alpha_j \langle \varphi_i, \varphi_j \rangle = \alpha_i \end{aligned}$$

2.:

$$\begin{aligned} \|f\|^2 &= \langle f, f \rangle \\ &= \left\langle \sum_{i=1}^n \alpha_i \varphi_i, \sum_{j=1}^n \alpha_j \varphi_j \right\rangle = \sum_{i,j=1}^n \alpha_i \alpha_j \delta_{i,j} = \sum_{i=1}^n \alpha_i^2 \end{aligned}$$

3.:

$f \in X \setminus X_n$:

$$\begin{aligned} \|f - \underbrace{\tilde{f}_n}_{\in X_n}\| &= \left\langle f - \sum_{i=1}^n \tilde{\alpha}_i \varphi_i, f - \sum_{i=1}^n \tilde{\alpha}_i \varphi_i \right\rangle \\ &= \|f\|^2 - 2 \sum_{i=1}^n \tilde{\alpha}_i \underbrace{\langle \varphi_i, f \rangle}_{=\alpha_i} + \sum_{i,j=1}^n \alpha_i \alpha_j \langle \varphi_i, \varphi_j \rangle \\ &= \|f\|^2 - \sum_{i=1}^n \tilde{\alpha}_i \alpha_i + \sum_{i=1}^n \tilde{\alpha}_i^2 \stackrel{\text{Quadratische Ergänzung}}{=} \|f\|^2 - \sum_{i=1}^n \alpha_i^2 + \sum_{i=1}^n \underbrace{(\alpha_i - \tilde{\alpha}_i)^2}_{\geq 0} \end{aligned} \quad (1.1)$$

Dies wird minimiert, wenn $\tilde{\alpha}_i = \alpha_i$ ist.

4.:

$f \in X_n$ wurde in 2. gezeigt. Sonst:

$$f \notin x_n \implies \text{mit } \alpha_i = \tilde{\alpha}_i \text{ in (1.1) :}$$

$$0 \leq \|f - f_n\|^2 = \|f\|^2 - \sum_{i=1}^n \underbrace{\alpha_i^2}_{\langle \varphi_i, f \rangle^2}$$

Es folgt die Behauptung. □

Vorteile von Orthogonalität:

- Bestapproximation
- Einfache Basisdarstellung

Kapitel 2

Das lineare Ausgleichsproblem

2.1 Problemstellung und Normalengleichung

Gegeben seien Punkte $(t_i, b_i) \in \mathbb{R}^2$ mit $i = 1, \dots, m$. Wir nehmen an, dass es eine Gestzmäßigkeit im Sinne eines parameterabhängigen Modelles

$$b_i = b(t_i) = b(t_i; \underbrace{x_1, \dots, x_n}_{\text{Parameter}}),$$

wobei die Parameter x_1, \dots, x_n unbekannt seien, gibt. In der Praxis sind die Messungen zusätzlich mit Fehlern behaftet und das Modell gilt nur approximativ. Zusätzlich gibt es oft mehr Messungen als Parameter, d.h. $m > n$.

Frage: Gegeben die Messungen, können wir zugehörige Parameter bestimmen?

Annahme: b ist linear in den Parametern, d.h. es gibt Funktionen

$$a_i : \mathbb{R} \rightarrow \mathbb{R}$$

s.d.

$$b(t; x_1, \dots, x_n) = a_1(t)x_1 + \dots + a_n(t)x_n.$$

Idee: Formuliere ein lineares Gleichungssystem:

$$b_i \approx b(t_i; x_1, \dots, x_n) = a_1(t_i)x_1 + \dots + a_n(t_i)x_n, i = 1, \dots, m$$

kurz $Ax \approx b$ mit $A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n, b \in \mathbb{R}^m$.

Problem: Durch Modell- und Messfehler gilt das Gleichungssystem nur ungefähr, und wir mehr Gleichungen als Unbekannte ("das Gleichungssystem ist überbestimmt"). Wir können unser Gleichungssystem also im Allgemeinen nicht lösen.

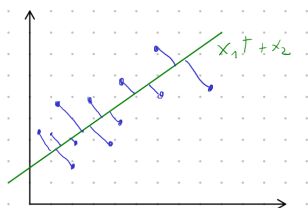


Abbildung 2.1: Datenpunkte und approximierte Gerade

Beispiel 2.1.

Idee: Finde Parameter, sodass das Modell "bestmöglich" mit den Messpunkten übereinstimmt, d.h. finde $(x_1, \dots, x_n)^t = x \in \mathbb{R}^n$ s.d.:

$$\|Ax - b\| = \min_{y \in \mathbb{R}^n} \|Ay - b\| \quad (2.1)$$

Definition 2.2. Die Gleichung (2.1) heißt **lineares Ausgleichsproblem**. Der Term $Ax - b$ heißt **Residuum**.

Bemerkung: $V = \mathbb{R}^m, U = \text{Bild}(A) \subset V, \dim(\text{Bild}(A)) \leq n \leq m$
Grundannahme

Statt V mit euklidischem Skalarprodukt aus.

$\xRightarrow{\text{Satz 1.9}}$ Es gibt genau ein $Ax \in \text{Bild}(A)$ so, dass

$$\|Ax - b\| = \min_{w \in U} \|w - b\|$$

gilt.

Aber: Wie berechnen wir x ?

Satz 2.3. Sei $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, m \geq n, x \in \mathbb{R}^n$ ist genau dann eine Lösung von (2.1) bezüglich der euklidischen Norm, falls

$$A^t Ax = A^t b. \quad (2.2)$$

Insbesondere ist das lineare Ausgleichsproblem genau dann lösbar, falls $\text{rang}(A) = n$.

Beweis.

$$\begin{aligned} \|Ax - b\| &= \min_{y \in \mathbb{R}^n} \|Ay - b\| \\ &\xLeftrightarrow{\text{Satz (1.9)}} Ax - b \in U^\perp = \text{Bild}(A)^\perp \\ &\iff \forall y \in \mathbb{R}^n : \langle Ax - b, Ay \rangle = 0 \\ &\iff \forall y \in \mathbb{R}^n : \langle A^t Ax - A^t b, y \rangle = 0 \\ &\iff A^t Ax = A^t b \end{aligned}$$

Die letzte Gleichung ist genau dann invertierbar, wenn $A^t A$ vollen Rang hat, also wenn A vollen Rang (n) hat. \square

Bemerkung 2.4. Im Beweis verwenden wir, dass $Ax - b$ orthogonal zu $U = \text{Bild}(A)$,

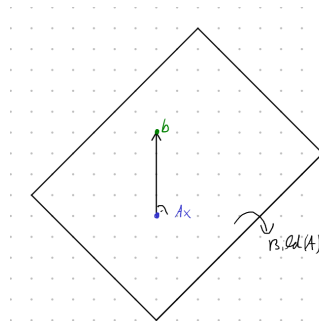


Abbildung 2.2: Hyperebene und Projektion

d.h. eine Normale zur Hyperebene $\text{Bild}(A)$ im \mathbb{R}^m , ist. Deshalb heißt (2.2) auch **Normalengleichung**.

Bemerkung 2.5. Für $m = n$ und $\text{rang}(A) = n$ ist die Lösung des linearen Ausgleichsproblems exakt (im mathematischen Sinne).

Satz 2.6. Für $A \in \mathbb{R}^{m \times n}$ ist $A^t A$ symmetrisch und positiv semidefinit. Falls $m \geq n$ ist $A^t A$ genau dann positiv definit, wenn $\text{rang}(A) = n$.

Beweis. • Symmetrisch: klar

• positiv semidefinit:

$$\forall x \in \mathbb{R}^n : x^t (A^t A) x = (Ax^t)(Ax) = \|Ax\|_2^2 \geq 0$$

- positiv definit: $\text{rang}(A) = n \implies Ax = 0 \iff x = 0 \implies \|Ax\|_2 = 0 \iff x = 0 \implies \text{Behauptung.}$

□

Einfachste Möglichkeit zur Lösung von (2.2): Berechne $A^t A$, $A^t b$, löse LGS mittels Cholesky. Kosten sind ungefähr:

$$\frac{n^2 m}{2} + m \cdot n + \frac{n^3}{6} + \frac{n^2}{2} + \frac{n^2}{2} \approx \frac{mn^2}{2} \text{ für } m \gg n.$$

Eingefügte Bemerkung. Anmerkung vom Dozent: $A^t A$ eig. immer schlecht zu berechnen.

Aber: Dieser Vorgang ist schlechter konditioniert als das lineare Ausgleichsproblem:

Eingeschobene Definition / Wiederholung

$$\begin{aligned} \text{cond}(A) &= \|A\| \|A^{-1}\| \\ \|A\| &= \max_{\|x\|=1} \|Ax\| \end{aligned}$$

Falls $A \in \mathbb{R}^{n \times n}$ spd (symmetrisch, positiv definit) gilt $\text{cond}_2((A^t A)) = \text{cond}_2(A)^2$.
Für $A \in \mathbb{R}^{m \times n}$ gelten ähnliche Überlegungen, siehe Deuffhard & Hohmann.

Beispiel 2.7. Sei $A = \begin{bmatrix} 1 & 1 \\ \epsilon & 0 \\ 0 & \epsilon \end{bmatrix}$ mit $\epsilon > \underbrace{\epsilon}_{\text{Maschinenengenauigkeit}}, \epsilon^2 < \epsilon$.

$$\implies A^t A = \begin{bmatrix} 1 + \epsilon^2 & 1 \\ 1 & 1 + \epsilon^2 \end{bmatrix} \stackrel{\text{im Computer}}{=} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$\implies A^t A$ ist im Computer singular, obwohl A vollen Rang hat!

Idee / Wunsch: Gebe einen Algorithmus an, der das lineare Ausgleichsproblem löst und nur auf A arbeitet.

2.2 Methode der Orthogonalisierung

Definition 2.8. Eine Matrix $Q \in \mathbb{R}^{n \times n}$ heißt **orthogonal**, wenn $Q^t Q = I$, d.h. falls die Spalten von Q eine ONB bzgl. des euklidischen Skalarprodukts bilden. Schreibe $Q \in O(n)$.

Notation: $\langle \cdot, \cdot \rangle_2, \|\cdot\|_2$ für das euklidische Skalarprodukt / die euklidische Norm.

Lemma 2.9. Für alle $Q \in O(n)$ gilt

1. $\|Qx\|_2 = \|x\|_2$ (Invarianz der Norm bzgl. orthogonaler Projektionen)
2. $\text{cond}_2(Q) = 1$

Beweis. 1.: $\|Qx\|_2^2 = \langle Qx, Qx \rangle_2 = \langle Q^t Qx, x \rangle_2 = \langle x, x \rangle_2 = \|x\|_2^2$

2.: $\|Q\|_2 = \max_{\|x\|_2=1} \|Qx\|_2 = 1$ und auch $\|Q^{-1}\|_2 = 1 \implies \text{Behauptung.}$

□

Satz 2.10. $A \in \mathbb{R}^{m \times n}, m \geq n, \text{rang}(A) = n$. Dann hat A eine QR-Zerlegung:

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$$

wobei $Q \in O(m), R \in \mathbb{R}^{n \times n}$ eine obere Dreiecksmatrix ist.

Beweis. Schreibe das Gram-Schmidt-Orthogonalisierungsverfahren in Matrixform:

$$Q = \underbrace{\begin{bmatrix} A_n & \dots & A_2 & A_1 \end{bmatrix}}_{[B_n \dots B_1]} \underbrace{\begin{bmatrix} 1 & \dots & \dots & \dots & -\frac{\langle A_n, A_1 \rangle_2}{\|A_1\|_2^2} \\ & \ddots & & & \vdots \\ & & 1 & -\frac{\langle A_3, A_2 \rangle_2}{\|A_2\|_2^2} & -\frac{\langle A_3, A_1 \rangle_2}{\|A_1\|_2^2} \\ & & & 1 & -\frac{\langle A_2, A_1 \rangle_2}{\|A_1\|_2^2} \\ \mathbf{0} & & & & 1 \end{bmatrix}}_{R'} \underbrace{\begin{bmatrix} \frac{1}{\|B_1\|_2} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\|B_n\|_2} \end{bmatrix}}_{R''}$$

$\Rightarrow Q \in \mathbb{R}^{m \times n}, R'R''$ ist obere Dreiecksmatrix mit nicht-null Diagonaleinträgen

\Rightarrow invertierbar: $R = (R'R'')^{-1}$

$\Rightarrow QR = A$, wenn wir Q zu einer ONB von \mathbb{R}^m erweitern. □

— Ende von Vorlesung 02 am 13.10.2022 —

Satz 2.11. Sei $A \in \mathbb{R}^{m \times n}, m \geq n, \text{rang}(A) = n, b \in \mathbb{R}^n$. Sei $A = QR$ eine QR -Zerlegung von A und

$$\underbrace{Q^t A}_{=R} = Q^t b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \begin{matrix} \in \mathbb{R}^n \\ \in \mathbb{R}^{m-n} \end{matrix}.$$

Dann ist $x = R_1^{-1}b_1$ die Lösung des linearen Ausgleichsproblems, wobei $R_1 \in \mathbb{R}^{n \times n}$ der obere Teil von R ist.

Beweis.

$$\begin{aligned} \|Ax - b\|_2^2 &\stackrel{\text{Lemma 2.9}}{=} \|Q^t(Ax - b)\|_2^2 \\ &= \left\| \begin{bmatrix} R_1 x - b \\ b_2 \end{bmatrix} \right\|_2^2 = \|R_1 x - b_1\|_2^2 + \|b_2\|_2^2 \\ &\geq \|b_2\|_2^2 \end{aligned}$$

$n = \text{rang}(A) = \text{rang}(R) = \text{rang}(R_1) \Rightarrow R_1$ invertierbar \Rightarrow Behauptung □

Problem:

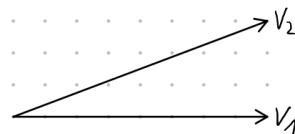


Abbildung 2.3: Problemstellung

$w_2 = v_2 - \frac{\langle v_2, v_1 \rangle_2}{\langle v_1, v_1 \rangle_2} v_1$ ist problematisch, falls $v_1 \approx v_2$ (Auslöschung). Beim Gram-Schmidt-Verfahren können Rundungsfehler auftreten. Es ist instabil.

Ziel: Stabiler Algorithmus um QR -Zerlegungen zu berechnen.

2.3 Grundüberlegungen zu Orthogonalisierungsverfahren

Problemstellung: Gegeben $v_1 = \alpha e_1 \in \mathbb{R}^2, v_2 \in \mathbb{R}^2$ transformiere v_2 auf $\tilde{w}_2 = \beta e_2$, gebe β an.

Gram-Schmidt: $\beta = \|w\|_2$

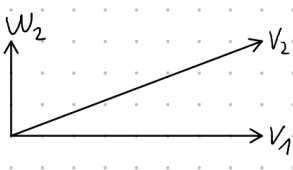


Abbildung 2.4: Gram-Schmidt

Drehungen: $\tilde{w}_2 = Qv_2$

$$Q = \begin{bmatrix} \cos(-\theta) & \sin(-\theta) \\ -\sin(-\theta) & \cos(-\theta) \end{bmatrix}$$

$$\beta = \|v_2\|_2$$

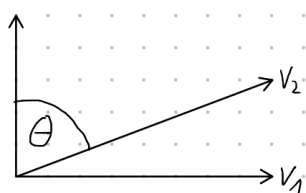


Abbildung 2.5: Drehungsansatz

Spiegelungen: $\tilde{w}_2 = Qv_2$, $Q = I - 2\frac{vv^t}{v^t v}$ und $\beta = \|v_2\|_2$

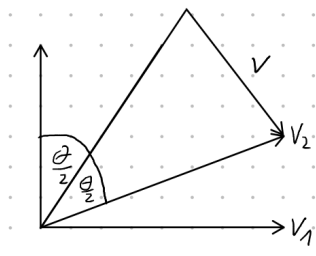


Abbildung 2.6: Spiegelungsansatz

Idee: Benutze orthogonale Transformationen Q_1, \dots, Q_n um $A \in \mathbb{R}^{m \times n}$, $\text{rang}(A) = n$, sukzessive zu reduzieren.

$$A \rightsquigarrow Q_1 A \rightsquigarrow Q_2 Q_1 A \rightsquigarrow \dots \rightsquigarrow \begin{bmatrix} R_1 \\ 0 \\ 0 \end{bmatrix}$$

Weil $\text{cond}_2(Q) = 1$ ist die Vorgehensweise stabil, bzw. gut konditioniert.

Aber: Wie wählen wir Q_1, \dots, Q_n ?

2.4 QR-Zerlegung mittels Givens-Rotationen

Definition 2.12. Eine Matrix der Form

$$\delta_{k,l} = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & c & & s & \\ & & & 1 & & \\ & & -s & & c & \\ & & & & & 1 & \\ & & & & & & \ddots & \\ & & & & & & & 1 \end{bmatrix}$$

, wobei die s, c Einträge in der k, l ten Zeile / Spalte sind, heißen Givens-Rotationen.

Bemerkung: Für $c = \cos(\theta)$, $s = \sin(\theta)$ ist $\delta_{k,l}$ eine Drehung um θ in in der Koordinaten (k, l) . $\delta_{k,l}$ ist Orthogonal.

Frage: Wie wählen wir c, s ?

Gegeben $x \in \mathbb{R}^n$, eliminiere l te Koordinate zu 0.

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} x_k \\ x_l \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}$$

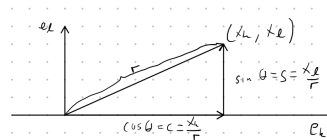


Abbildung 2.7: Trigonometriesetting

$$r^2 = x_k^2 + x_l^2 \implies \pm \sqrt{x_k^2 + x_l^2}$$

Aber: Diese Berechnungsweise ist nicht unbedingt stabil ($x_k \gg x_l$)

Stabile Variante:

$$\begin{aligned} \text{Falls } |x_l| > |x_k| &\implies \tau = \frac{x_k}{x_l}, s = \frac{1}{\sqrt{1+\tau^2}}, c = s\tau \\ \text{Sonst: } \tau = \frac{x_l}{x_k}, c &= \frac{1}{\sqrt{1+\tau^2}}, s = c\tau \end{aligned} \quad (2.3)$$

Beispielprozess:

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \rightsquigarrow \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ 0 & * & * \end{bmatrix} \rightsquigarrow \begin{bmatrix} * & * & * \\ * & * & * \\ 0 & * & * \\ 0 & * & * \end{bmatrix} \rightsquigarrow \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & 0 \end{bmatrix}$$

Algorithm 2.13

Input: $A \in \mathbb{R}^{m \times n}$, $m \geq n$

Output: R von der QR -Zerlegung (A wird zerstört "in place")

for $j = 1, \dots, n$ **do**

for $i = m, m-1, \dots, j+1$ **do**

 Berechne c, s wie in (2.3)

$$A[i-1:j, j:n] = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^t A[i-1:j, j:n]$$

end for

end for

$m \approx n$:

c, s : In jedem Eintrag einmal Wurzeln ziehen: $\implies \frac{n^2}{2}$ Quadratwurzeln und $\frac{4n^3}{3}$ Multiplikationen

$m \gg n$: $m \cdot n$ Quadratwurzeln und $2m \cdot n^2$ Multiplikationen

Bemerkung 2.14. Der Algorithmus 2.13 berechnet nur R von der QR-Zerlegung. Zur Berechnung von Q müssten zusätzliche Operationen investiert werden um die Givens-Rotation auf I anzuwenden. Für das lineare Ausgleichsproblem benötigen wir $Q^t b$, weshalb wir den Algorithmus auf $\begin{bmatrix} A & | & b \end{bmatrix}$ anwenden können (da $R = Q^t A$).

Bemerkung 2.15. Für $m = n$ ist die QR-Zerlegung eine (teure) Alternative zur LR-Zerlegung.

2.5 QR-Zerlegung mittels Householder-Transformationen

Definition 2.16. Für $v \in \mathbb{R}^n, v \neq 0$, heißt

$$Q = I - 2 \frac{\overbrace{vv^t}^{\in \mathbb{R}^{n \times n}}}{\underbrace{v^t v}_{\in \mathbb{R}}}$$

Householder-Transformation / Reflexion / Spiegelung.

Wichtig!

Nicht vv^t berechnen, das ist sehr ineffizient!

Für $a, v \in \mathbb{R}^n, v \neq 0$ ist $Qa = \left(I - 2 \frac{vv^t}{v^t v}\right) a = a - 2 \frac{\langle v, a \rangle_2}{\langle v, v \rangle_2} v$

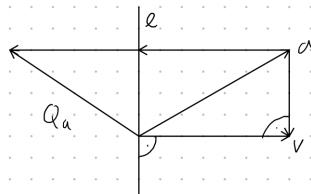


Abbildung 2.8: Householder-Transformationssetting

Qa ist a an l gespiegelt.

Lemma 2.17. Für eine Householder-Transformation $Q \in \mathbb{R}^{n \times n}$ gilt:

1. Q ist symmetrisch
2. Q ist orthogonal
3. Q ist involutionisch (eine Involution), d.h. $Q^2 = I$

Beweis. Nachrechnen. □

Frage: Gegeben $a \in \mathbb{R}^n$, wie müssen wir v wählen, so dass $Qa = \alpha e_1$ für $\alpha \in \mathbb{R}$?

Beobachte:

1. $|\alpha| = \|\alpha e_1\|_2 = \|Qa\|_2 = \|a\|_2$

$$2. \underbrace{a - 2 \frac{\langle v, a \rangle}{\langle v, v \rangle}}_{\in \mathbb{R}} v = Qa$$

$$\implies v \in \text{span}(\alpha e_1 - a) \implies \alpha = \pm \|a\|_2$$

$$\text{Vermeide Auslöschung} \implies \alpha = -\text{sign}(a_1) \cdot \|a\|_2$$

Effiziente Berechnung: Beobachte:

$$\begin{aligned} \|v\|_2^2 &= \langle v, v \rangle_2 = \langle a - \alpha e_1, a - \alpha e_1 \rangle_2 \\ &= \|a\|_2^2 - 2\alpha a_1 + \alpha^2 \\ &= -2\alpha(a_1 - \alpha) \\ \implies Qa &= a - 2 \frac{\langle v, a \rangle_2}{\|v\|_2^2} a = a + \frac{\langle v, a \rangle_2}{\alpha(a_1 - \alpha)} v \end{aligned}$$

— Ende von Vorlesung 03 am 18.10.2022 —

Lemma 2.18. Sie $\alpha \in \mathbb{R}^n, a \neq 0, a \notin \text{span}\{e_1\}$. Sei

$$v = a - \alpha e_1, \alpha = -\text{sign}(a_1) \cdot \|a\|_2 \quad (2.4)$$

Dann ist

$$\left(I - 2 \frac{vv^t}{v^t v} \right) a = a + \frac{v^t a}{\alpha(a_1 - \alpha)} v = \alpha e_1. \quad (2.5)$$

Beweis. Siehe oben. □

Algorithm 2.19

Input: $A \in \mathbb{R}^{m \times n}, m \geq n$ “Mehr Zeilen als Spalten”

Output: $A \in \mathbb{R}^{m \times n}$, obere rechte Dreiecksmatrix R , Rest Householder-Transformationen

for $j = 1, \dots, n$ **do**

Berechne v, α wie in (2.4), mit $a = A[j : m, j] \in \mathbb{R}^{m-j+1}$

$v = \frac{1}{v_1} v$

Berechne $A[j : m, j : n] = \left(I - 2 \frac{vv^t}{v^t v} \right) A[j : m, j : n]$ wie in (2.5)

if $j < m$ **then**

$A[j+1:m, j] = v[2:m-j+1]$

end if

end for

▷ Iterieren über die Spalten

▷ Erster Eintrag wird nicht gespeichert, daher normalisieren wir

▷ Index startet von 1

Bemerkung 2.20. Die Skalierung $v = \frac{1}{v_1} v$ stellt sicher, dass die der erste Eintrag von v nicht gespeichert werden muss.

Aufwand: $m \sim n \rightsquigarrow \frac{2}{3}n^3$ Multiplikationen

$m \gg n \rightsquigarrow 2n^2m$ Multiplikationen

Schneller als Givensrotationen, stabiler als Normalengleichungen

2.6 Pseudoinverse

Ausgangspunkt: Wir wollen ein stabiles numerisches Verfahren, dass

$$Ax = b, A \in \mathbb{R}^{m \times n}, m \geq n, \text{rang}(A) = n, b \in \mathbb{R}^n$$

“lösen” kann, d.h. es gilt

$$\|Ax - b\|_2 = \min_{y \in \mathbb{R}^n} \|Ay - b\|_2$$

Mathematisch können wir die Abbildung $b \mapsto x$, wegen der Normalengleichung (2.2), schreiben als

$$x = \underbrace{(A^t A)^{-1} A^t}_{:= A^\dagger} b = A^\dagger b$$

$A^\dagger \in \mathbb{R}^{n \times m}$. Wegen $A^\dagger A = I$ heißt A^\dagger auch **Pseudoinverse**.

Frage: Können wir den Begriff der Inversen noch weiter verallgemeinern? Auf beliebige Matrizen?

Satz 1.9: $A \in \mathbb{R}^{m \times n}, U = \text{Bild}(A)$

$$\begin{aligned} \implies \|Ax - b\|_2 &= \min_{y \in \mathbb{R}^n} \|Ay - b\|_2 \xLeftrightarrow{\text{Satz 1.9}} Ax - b \in \text{Bild}(A)^\perp \\ \iff Ax - Pb - \underbrace{(b - Pb)}_{\in U^\perp: \text{Satz 1.9}} &\in \text{Bild}(A)^\perp, Pb \text{ ist die orthogonale Projektion von } b \text{ auf } U \\ \iff \underbrace{Ax}_{\in U} - \underbrace{Pb}_{\in U} &\in \text{Bild}(A)^\perp \\ \iff Ax &= Pb \end{aligned}$$

Falls $\text{rang}(A) < n$ (z.B., falls $m < n$) ist $Ax = Pb$ nicht eindeutig lösbar (aber es existiert immer eine Lösung).

Für $\tilde{x} \in \mathbb{R}^n$ mit $A\tilde{x} = Pb, x' \in \ker(A)$ ist $A(\tilde{x} + x') = Pb$.

$$\begin{aligned} L(b) &= \left\{ x \in \mathbb{R}^n : \|Ax - b\|_2 = \min_{y \in \mathbb{R}^n} \|Ay - b\|_2 \right\} \\ &= \{x \in \mathbb{R}^n : Ax = Pb\} \\ &= \tilde{x} + \ker(A) \end{aligned}$$

Sind gewisse Lösungen sinnvoller als andere?

Wähle: $x \in \tilde{x} + \ker(A)$ mit minimaler Norm als "eindeutige" Lösung von $Ax = b$.

$$\begin{aligned} \xRightarrow{\text{Bem. 1.13}} \|x - 0\|_2 &= \min_{y \in \tilde{x} + \ker(A)} \|y - 0\|_2 \iff x \in (\tilde{x} + \ker(A))^\perp \\ &\iff x \in \ker(A)^\perp \end{aligned}$$

Anmerkung

Hier ist nicht ganz klar, was mit $(\tilde{x} + \ker(A))^\perp$ gemeint ist, da dies z.B. für $\ker(A) = \text{span}\{(0, 1)^t\}$ und $\tilde{x} = (1, 0)^t$ nur $\{0\}$ ist, was natürlich nicht der Intuition entspricht!

Statt der ursprünglichen Definition müssen wir hier wieder zurück schieben ($-\tilde{x}$ rechnen), was kein Problem ist, da wir o.B.d.A. $\tilde{x} \perp \ker(A)$ voraussetzen dürfen, bevor wir das Skalarprodukt berechnen!

Zum Beispiel ist also $v = (1, 0)^t$ im obigen Beispiel doch im orthogonalen Komplement, da $\langle v, \tilde{x} + u - \tilde{x} \rangle_2 = 0$ für $u \in \ker(A)$

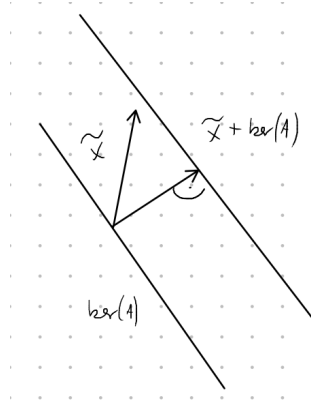


Abbildung 2.9: Setting

Bemerkung 2.21. Diese Wahl von x für $b \mapsto x$ ist linear: Für $b_1, b_2 \in \mathbb{R}^m$ ist:

$$\left. \begin{array}{l} Ax_1 = b_1 \quad x_1 \in \ker(A)^\perp \\ Ax_2 = b_2 \quad x_2 \in \ker(A)^\perp \end{array} \right\} \implies P(x_1 + x_2) = P(x_1) + P(x_2) = Ax_1 + Ax_2 = A(x_1 + x_2), x_1 + x_2 \in \ker(A)^\perp$$

Definition 2.22. Sei $A \in \mathbb{R}^{m \times n}$. Die Abbildungsmatrix $A^\dagger \in \mathbb{R}^{n \times m}$ von $b \mapsto x$ heißt **Pseudoinverse** oder **Moore-Pensore-Inverse** von A . D.h. gegeben $b \in \mathbb{R}^n$, dann ist $x = A^\dagger b$ die eindeutige Lösung von

$$\min_{y \in \ker(A)^\perp} \|Ay - b\|_2 = \|Ax - b\|_2.$$

Satz 2.23. $A \in \mathbb{R}^{m \times n}$. Dann ist $A^\dagger \in \mathbb{R}^{n \times m}$ eindeutig über die Moore-Penrose-Axiome definiert:

1. $(A^\dagger A)^t = AA^\dagger$
2. $(AA^\dagger)^t = A^\dagger A$
3. $A^\dagger AA^\dagger = A^\dagger$
4. $AA^\dagger A = A$

Beweis. Siehe Literatur oder später

□

Frage: Wie berechnen wir $x = A^\dagger b$?

Sei $A \in \mathbb{R}^{m \times n}$, $\text{rang}(A) = p \leq \min(m, n)$. Bringe A mittels orthogonaler Transformationen (z.B. Householder) auf obere Dreiecksgestalt, d.h.:

$$Q^t A = \begin{bmatrix} R & S \\ * & 0 & 0 \end{bmatrix} \quad (2.6)$$

wobei $S \in \mathbb{R}^{p \times (n-p)}$. Setze Analog $x = \begin{bmatrix} x_1 \in \mathbb{R}^p \\ x_2 \in \mathbb{R}^{n-p} \end{bmatrix}$, $Q^t b = \begin{bmatrix} b_1 \in \mathbb{R}^p \\ b_2 \in \mathbb{R}^{m-p} \end{bmatrix}$

Lemma 2.24. Mit obigen Bezeichnungen ist $x = A^\dagger b$ genau dann, wenn

$$x_1 = R^{-1}b_1 - R^{-1}Sx_2.$$

Beweis.

$$\begin{aligned}\|Ax - b\|_2^2 &= \|Q^t(Ax - b)\|_2^2 \\ &= \left\| \begin{pmatrix} Rx_1 + Sx_2 - b \\ -b_2 \end{pmatrix} \right\|_2^2 \\ &= \|Rx_1 + Sx_2 - b_1\|_2^2 + \|b_2\|_2^2\end{aligned}$$

ist minimal, falls $Rx_1 = b_1 - Sx_2$. □

Wir sehen $p = \text{rang}(A) = n \implies$ wie vorher, lineares Ausgleichsproblem!

Sonst: $x_2 = ?$

Lemma 2.25. Sei $p < n$, $V = R^{-1}S \in \mathbb{R}^{n \times (n-p)}$ und $u = R^{-1}b_1 \in \mathbb{R}^p$. Dann ist

$$\begin{aligned}x &= A^\dagger b \\ \iff (I + V^t V)x_2 &= V^t u \\ x_1 &= u - Vx_2\end{aligned}$$

Beweis.

$$\begin{aligned}\|x\|_2^2 &= \|x_1\|_2^2 + \|x_2\|_2^2 \\ &\stackrel{\text{Lemma 2.24}}{=} \|u - Vx_2\|_2^2 + \|x_2\|_2^2 \\ &= \|u\|_2^2 - 2\langle u, Vx_2 \rangle_2 + \langle Vx_2, Vx_2 \rangle_2 + \langle x_2, x_2 \rangle_2 \\ &= \|u\|_2^2 + \langle x_2, (I + V^t V)x_2 - 2V^t u \rangle_2 = \varphi(x_2)\end{aligned}$$

Minimiere $\varphi(x_2)$:

$$\begin{aligned}\varphi'(x_2) &= -2V^t u + 2(I + V^t V)x_2 \\ \varphi'(x_2) &= 2(I + V^t V) \implies \text{spd}\end{aligned}$$

φ minimal $\iff \varphi'(x_2) = 0 \implies$ Behauptung. □

Algorithm 2.26

Input: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$

Output: $x = A^\dagger b$

Berechne QR -Zerlegung (2.6) von A

$$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = Q^t b$$

$V = R^{-1}S$ mittels Rückwertssubstitution

$u = R^{-1}b_1$ mittels Rückwertssubstitution

Löse $(I + V^t V)x_2 = V^t u$ mittels Cholesky-Zerlegung

$$x_1 = u - Vx_2$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

—Ende von Vorlesung 04 am 20.10.2022—

Kapitel 3

Iterative Verfahren für große, dünn besetzte, Gleichungssysteme

3.1 Motivation

Sei $\Omega \subset \mathbb{R}^d, d \in \mathbb{N}$. Betrachte die stationäre Wärmeleitungsgleichung, eine partielle Differenzialgleichung

$$\begin{cases} -\Delta u(x) = f(x) & x \in \Omega \\ u(x) = 0 & x \in \partial\Omega \end{cases} \quad (3.1)$$

mit Wärmequelle $f \in C(\Omega)$ und dem Laplace-Operator:

$$\Delta u = \sum_{i=1}^n \frac{\partial^2 u(x)}{\partial x_i^2}. \quad (3.2)$$

Die Lösung $u \in C^2(\Omega)$, falls existent, beschreibt die Temperaturverteilung im Raum Ω .

Diese Gleichung ist i.A. nicht von Hand lösbar!

Idee: Berechne approximative Lösung im Computer.

Ansatz: Für $g \in C^2(\mathbb{R})$ ist

$$\begin{aligned} g''(x) &= \lim_{h \searrow 0} \frac{g'(x+h) - g'(x)}{h} \approx \frac{g'(x+h) - g'(x)}{h} \\ &\approx \frac{\frac{g(x+h) - g(x)}{h} - \frac{g(x) - g(x-h)}{h}}{h} \\ &\approx \frac{g(x+h) - 2g(x) + g(x-h)}{h^2} \end{aligned}$$

\rightsquigarrow Ersetze $\frac{\partial^2 u}{\partial x_i^2}$ in (3.2)

\rightsquigarrow Überziehe Ω mit einem regelmäßigen Gitter mit Maschenweite $h = \frac{1}{n}, n \in \mathbb{N}$.

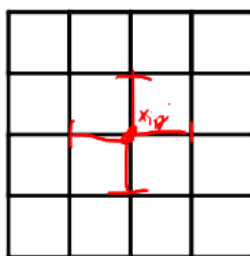


Abbildung 3.1: Gitter

Bezeichne die Gitterpunkte mit x_{ij} und $u_{ij} = u(x_{ij})$.

$$\stackrel{d=2}{\implies} \frac{1}{h^2} (4u_{ij} - u_{i+1j} - u_{i-1j} - u_{ij+1} - u_{ij-1}) = f_{ij} : i, j \in 1, \dots, n-1$$

$$u_{ij} = 0, i \in \{0, n\} \text{ oder } j \in \{0, n\}$$

Wir erhalten ein lineares Gleichungssystem mit $N = (n-1)^2$ Unbekannten und $O(1)$ Einträgen pro Zeile.
 $\implies A \in \mathbb{R}^{n \times n}$ hat $O(N)$ Einträge. Wir haben das Lösen einer (linearen) partiellen Differentialgleichung durch das Lösen eines linearen Gleichungssystems ersetzt.

Beispiel 3.1. $\Omega = (0, 1)^2, n = 4 \implies h = \frac{1}{4}$. Erhalte:

$$\left[\begin{array}{ccc|cc} 4 & -1 & & -1 & & \\ -1 & 4 & -1 & & -1 & \\ & -1 & 4 & & & -1 \\ \hline -1 & & & 4 & -1 & \\ & -1 & & -1 & 4 & -1 \\ & & -1 & & -1 & 4 \\ \hline & & & -1 & & \\ & & & & -1 & \\ & & & & & -1 \end{array} \right] \begin{bmatrix} u_{11} \\ u_{12} \\ u_{13} \\ u_{21} \\ u_{22} \\ u_{23} \\ u_{31} \\ u_{32} \\ u_{33} \end{bmatrix} = \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix}$$

Aber: Um die Lösung von (3.1) gut zu approximieren ist oft $N \gg 1$ erforderlich. Für kleine bis mittlere N , d.h. in 2022 je nach Modell ~ 10 Millionen, sind graphenbasierte Löser eine Option.

Was tun für große N ?

Beobachtung: Matrix-Vektor-Multiplikation sind für dünn besetzte Matrizen in $O(N)$ berechenbar.

Frage: Wie bauen wir gute Löser für LGS (lineare Gleichungssysteme) nur unter Anwendung von Matrix-Vektor-Multiplikationen?

Idee: Benutze Orthogonalität um eine Bestapproximationseigenschaft zu erhalten.

3.2 Grundidee von Projektionsmethoden

Sei $A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n$ und K, L Unterräume vom \mathbb{R}^n .

Idee: Finde eine approximative Lösung \tilde{x} zu $Ax = b$ mit

$$\tilde{x} \in K \text{ und } b - A\tilde{x} \perp_2 L$$

Kanonische Wahl: $L = AK$.

Falls wir eine Startnäherung x_0 zu x kennen, können wir \tilde{x} in $x_0 + K$ suchen:

Finde $\tilde{x} \in x_0 + K$ mit $b - A\tilde{x} \perp_2 L$

Beobachtung: $\tilde{x} \in x_0 + K \implies \exists d \in K : \tilde{x} = x_0 + d$

$$\implies \underbrace{b - A(x_0 + d)}_r \perp_2 L$$

$$\iff r_0 - Ad \perp_2 L$$

Eine approximative Lösung $\tilde{x} = x_0 + d$ muss also erfüllen:

$$\begin{cases} \tilde{x} = x_0 + d \\ \langle r_0 - Ad, w \rangle_2 = 0 \quad \forall w \in L \end{cases} \quad (3.3)$$

Idee: Wähle x_0, K, L , berechne $d \in K$ durch Lösen eines Unterproblems. Setze $x_1 = x_0 + d$, wähle neue Unterräume, beginne von vorne.

Wie implementieren wir diese Idee im Computer?

Sei $K = \text{span}\{v_1, \dots, v_n\}$, $L = \{w_1, \dots, w_n\}$

$V = [v_1 | \dots | v_n]$ und $W = [w_1 | \dots | w_n]$

(3.3) ist äquivalent zu

$$\begin{cases} \tilde{x} = x_0 + Vy & y \in \mathbb{R}^m \\ W_i^t AVy = W_i^t r_0 & i = 1, \dots, n \end{cases} \iff \underbrace{W^t AV}_{m \times m} y = W^t r_0 \quad (3.4)$$

$$\implies \tilde{x} = x_0 + V(W^t AV)^{-1} W^t r_0$$

Algorithm 3.2 Prototyp einer iterativen Projektionsmethode

Input: $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$, Fehlertoleranz α

Output: Näherung $x_{i+1} \approx x$

i=0

while Fehlertoleranz noch nicht erreicht **do**

 Wähle K_i, L_i

 Wähle Basen V, W von K_i, L_i

$r_1 = Ax_i$

$y = (W^t AV)^{-1} W^t r_i$

$x_{i+1} = x_i + Vy_i$

$i = i + 1$

end while

Aber: $W^t AV$ ist nicht notwendigerweise invertierbar:

Beispiel 3.3.

$$A = \left[\begin{array}{c|c} 0 & I \\ \hline I & I \end{array} \right] \in \mathbb{R}^{2m \times 2m}$$

$$K = L = \text{span}\{e_1, \dots, e_m\} \implies V = W = \left[\begin{array}{c} I_m \\ 0 \end{array} \right] \in \mathbb{R}^{2m \times m}$$

$$\implies W^t AV = 0 \text{ ist nicht invertierbar.}$$

Lemma 3.4. Sei einer der folgenden Bedingungen erfüllt:

1. A ist spd, $K = L$

2. A invertierbar, $L = AK$

Dann ist $W^t AV$ für alle Basen von K, L invertierbar.

Beweis. 1.: $L = K \implies W = V\delta$ mit $\delta \in \mathbb{R}^{m \times m}$ invertierbar.
 $\implies B = W^t AV = \delta^t V^t AV$

$$0 < \underbrace{y^t Ay}_{\substack{\text{spd, invertierbar}}}, y = Vx \\ = \underbrace{x^t V^t AV x}_{\substack{\text{spd, invertierbar}}}$$

2.: $L = AK \implies W = AV\delta, \delta \in \mathbb{R}^{m \times m}$ invertierbar

$$\implies B = W^t AV = \delta^t \underbrace{V^t A^t AV}_{\text{spd}} \implies \text{invertierbar} \implies \text{Beh.}$$

□

— Ende von Vorlesung 05 am 25.10.2022 —

3.3 Verfahren des steilsten Abstiegs

Idee: Wähle $K = L = \text{span}\{r_i\} = \text{span}\{b - Ax_i\}$

$$\implies x_{i+1} = x_i + \underbrace{\alpha_i r_i}_{d_i \in K} \\ \implies \alpha = \frac{r_i^t r_i}{r_i^t A r_i}$$

Algorithm 3.5 Verfahren des steilsten Abstiegs

Input: A, b , Startvektor x_0 , Fehlertoleranz

Output: Näherung $x_{i+1} \approx x$

while Fehlertoleranz noch nicht erreicht **do**

▷ Praxis $\epsilon = 10^{-8}$, $\|r_i\|_2 < \epsilon$

$r_i = b - Ax_i$

$\alpha_i = \frac{r_i^t r_i}{r_i^t A r_i}$

$x_{i+1} = x_i + \alpha_i r_i$

end while

Bemerkung 3.6. Wegen (3.3) gilt:

$$\begin{aligned} 0 &= \langle r_i - Ad_i, r_i \rangle_2 \\ &= \langle b - Ax_i - Ad_i, r_i \rangle_2 \\ &= \langle b - Ax_{i+1}, r_i \rangle_2 \\ &= \langle Ax - Ax_{i+1}, r_i \rangle_2 \\ &= \langle x - x_{i+1}, r_i \rangle_A = (\star) \end{aligned}$$

Mit

$$\langle \cdot, \cdot \rangle_A = \langle A \cdot, \cdot \rangle_2$$

Aus (\star) folgt

$$\begin{aligned} 0 = (\star) &\iff x - x_{i+1} \perp_A r_i \\ &\iff x - x_{i+1} \perp_A x_i + \text{span}\{r_i\} \\ &\stackrel{\text{Satz 1.9}}{\iff} \|x - x_{i+1}\|_A = \min_{y \in x_i + \text{span}\{r_i\}} \|x - y\|_A \\ &\iff \frac{1}{2} \|x - x_{i+1}\|_A^2 = \min_{\alpha \in \mathbb{R}} \frac{1}{2} \|x - x_i - \alpha r_i\|_A^2 \end{aligned}$$

Betrachte $f(x_i) = \frac{1}{2} \|x - x_i\|_A^2 = \frac{1}{2} \langle A(x - x_i), x - x_i \rangle_2$

$$f'(x_i) = - \underbrace{Ax}_{=b} + Ax_i = -r_i$$

D.h. am Punkt x_i gehen wir in Richtung des steilsten Abstiegs,

Satz 3.7. Sei $A \in \mathbb{R}^{n \times n}$ spd. Dann gilt für die Iterierung des Verfahrens des steilsten Abstiegs dass:

$$\begin{aligned} \|x - x_{i+1}\|_A &\leq \frac{\lambda_{\max}(A) - \lambda_{\min}(A)}{\lambda_{\max}(A) + \lambda_{\min}(A)} \|x - x_i\|_A \\ &\stackrel{\frac{1}{\lambda_{\min}}}{=} \frac{\text{cond}_2(A) - 1}{\text{cond}_2(A) + 1} \|x - x_i\|_A \end{aligned}$$

wobei $\lambda_{\max}, \lambda_{\min}$ die größten, kleinsten Eigenwerte sind.

Beweis. Übung □

Bemerkung 3.8. Im Prinzip lassen sich mit Hilfe der Normalengleichung auch allgemeinere (invertierbare) Matrizen behandeln. Hierbei wird die Kondition verschlechtert d.h. die Konvergenz verschlechtert sich.

3.4 Krylovräume

Beobachte: Im Verfahren des steilsten Abstiegs gilt:

$$\begin{aligned} x_i &= x_0 + \alpha_0 r_0 + \dots + \alpha_{i-1} r_{i-1} = x_0 + \alpha_0 r_0 + \dots + (\alpha_{i-2} I + \alpha_{i-1} (I - \alpha_{i-2})) r_{i-2} \\ &= x_0 q_{i-1}(A) r_0, q_{i-1} \in \Pi_{i-1} \end{aligned}$$

Idee: Finde eine bessere Approximation von x_i in $x_0 + K_{i-1}(A, r_0)$.

Definition 3.9. Sei $A \in \mathbb{R}^{n \times n}, v \in \mathbb{R}^n, n \geq 1$. Der Raum

$$K_m = \text{span}(v, Av, A^2v, \dots, A^{m-1}v)$$

heißt **Krylovraum** von A zu v .

Es gilt $K_m(A, v) \subseteq K_{m+1}(A, v)$

Lemma 3.10. Sei $\mathbb{R}^{n \times n}, v \in \mathbb{R}^n$. Dann gilt:

1. $\dim K_m(A, v) \leq \min\{m, n\}$
2. $\dim(K_m(A, v)) = \dim(K_{m+1}(A, v)) = m \implies \dim(K_{m+i}(A, v)) = m, i = 0, 1, \dots$
3. Für m wie im 2. gilt

$$\{Ax : x \in K_m(A, v)\} \subseteq K_m(A, v)$$

D.h. $K_m(A, v)$ ist invariant unter A .

Beweis. Übung □

Bemerkung 3.11. Betrachte $Ax = b$ mit Startnäherung x_0 , Residuum $r_0 = b - Ax_0$. Für $i = 0, \dots$ Wähle

$$\begin{aligned} x_{i+1} &\in x_0 + K_{i+1}(A, r_0) \\ \implies x_{i+1} x_0 &= q_i(A) r_0 \\ \implies r_{i+1} &= b - Ax_{i+1} = \underbrace{b - Ax_0}_{r_0} - Aq_i(A) r_0 \\ &= q_{i+1}(A) r_0 \in K_{i+2}(A, r_0) \end{aligned}$$

Das bedeutet: Sind K, L geeignete Krylovräume in einer Projektionsmethode, dann können wir immer garantieren, dass das Ergebnis in einem Krylovraum ist.

Aber: Die Vektoren $v, Av, \dots, A^n v$ sind numerisch keine guten Basen der Krylovräume, da sie zunehmend in eine ähnliche Richtung zeigen.

3.5 Arnoldi-Verfahren

Gesucht: Numerisch gutartige Basis von $K_m(A, v)$, welche einfach zu einer gutartigen Basis von $K_{m+1}(A, v)$ erweitert werden kann.

Idee: Arrangiere die Vektoren v, Av, \dots in einer "wachsenden" Matrix

$$[v | Av | A^2 v | \dots]$$

und wende auf jede Spalte ein Orthogonalisierungsverfahren (Gram-Schmidt, Householder, Givens) an.

Algorithm 3.12 Arnoldi-Verfahren (Gram-Schmidt Variante)

Input: $A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n, m \in \mathbb{N}$

Output: $V_m = [v_1 | \dots | v_m]$ ONB von $K_m(A, v)$, $v_{m+1} \in \mathbb{R}^n, H_{m+1,m} \in \mathbb{R}^{(m+1) \times m}$

$$v_1 = \frac{v}{\|v\|_2}$$

for $j = 1, m$ **do**

$$z = Av_j$$

▷ $\neq A^{j-1}r_0$

$$h_{ij} = \langle z, v_i \rangle_2, i = 1, \dots, j$$

$$w_j = z - \sum_{i=1}^j h_{ij} v_i$$

$$h_{j+1,j} = \|w_j\|_2$$

if $h_{j+1,j} = 0$ **then**

stop

▷ Krylovraum stagniert

end if

$$v_{j+1} = \frac{w_j}{h_{j+1,j}}$$

end for

—Ende von Vorlesung 06 am 27.10.2022—

Lemma 3.13. Falls das Arnoldi-Verfahren nicht vorzeitig abbricht, ist v_1, \dots, v_m eine ONB von $K_m(A, r_0)$.

Beweis. Orthogonalität: Ok

Orthonormal: Ok

Basis von $K_m(A, r_0)$: $j = 1$ ok

$j \implies j+1$

$h_{j+1,j} v_j = w_j$ (folgt aus der letzten Zeile von Algorithmus 3.12).

$$\begin{aligned} w_j &= \underbrace{Av_j}_{\in K_j(A, r_0) \implies v_j = q_{j-1}(A)v} - \underbrace{\sum_{i=1}^j h_{i,j} v_i}_{\tilde{q}_{j-1}(A)v, \tilde{q}_{j-1} \in \Pi_{j-1}} = (\star) \\ (\star) &= Aq_{j-1}(A)v_j - \tilde{q}_{j-1}(A)v_j = \underbrace{q_j(A)v}_{\in K_{j+1}(A, r_0)} \end{aligned}$$

□

Bemerkung: Diese Matrix $H_{h+1,j} \in \mathbb{R}^{(j+1) \times j}$ hat eine bestimmte Struktur, die Hessenberg-Struktur genannt wird.

Vorteile dieser Struktur

Zum Beispiel kann man eine QR-Zerlegung finden, in dem man Pro Spalte eine Givensrotation anwendet.

Lemma 3.14. Seien $V_m \in \mathbb{R}^{n \times n}, H_{m+1,m} \in \mathbb{R}^{(m+1) \times m}$, wie im Arnoldi-Verfahren erzeugt. Sei $H_{m,m} \in \mathbb{R}^{m \times m}$ wie $H_{m+1,m}$, aber ohne die letzte Zeile. Dann gilt:

$$\underbrace{A}_{\in \mathbb{R}^{n \times n}} \underbrace{V_m}_{\in \mathbb{R}^{n \times m}} = \underbrace{V_m}_{\in \mathbb{R}^{n \times m}} \underbrace{H_{m,m}}_{\in \mathbb{R}^{m \times m}} + \underbrace{w_m e_m^t}_{\in \mathbb{R}^{m \times m}} = \underbrace{V_{m+1}}_{\in \mathbb{R}^{n \times m+1}} \underbrace{H_{m+1,m}}_{\in \mathbb{R}^{m+1 \times m}} \quad (3.5)$$

$$V_m^t A V_m = H_{m,m} \quad (3.6)$$

Beweis. Gemäß Algorithmus haben wir

$$Av_j = z = \underbrace{w_j}_{h_{j+1,j}v_j} + \sum_{i=1}^j h_{ij}v_i = \sum_{i=1}^{j+1} h_{ij}v_i : j = 1, \dots, m$$

Daher gilt (3.5) (folgt aus Matrix Schreibweise).

Für (3.6):

$$V_m^t AV_m = \underbrace{V_m^t V_m}_{=I} H_{m,m} + \underbrace{V_m^t w_m e_m^t}_{=0} = H_{m,m}$$

□

Lemma 3.15. Sei j der Iterationsindex, bei dem das Arnoldi-Verfahren das erste Mal abbricht. Dann gilt:

$$K_j(A, r_0) = \dots = K_m(A, r_0)$$

$$AV_m = V_m H_{m,m}$$

Beweis. Übung.

□

3.6 Verfahren der vollständigen Orthogonalisierung

Ziel: Kombinieren von unserem Wissen über Projektionsmethoden mit demjenigen Wissen über Krylovräume.

Zutaten:

- Finde $\tilde{x} \in x_0 + K$ s.d. $bA\tilde{x} \perp L$
- Wähle K, L als Krylovräume in jeder Iteration
- Wähle Basen V, W von K, L in jeder Iteration

$$(3.4) \implies \begin{cases} \tilde{x} = x_0 + Vy \\ W^t AVy = w^t r_0 \end{cases}$$

Idee: Setze $r_0 = b - Ax_0$, $K = L = K_m(A, r_0)$ im m -ter Iteration, $V = W$ ONB von $K_m(A, r_0)$, berechnet mittels Arnoldi-Verfahren.

Zutaten: aktualisiert:

- $r_0 = b - Ax_0$
- $\beta = \|r_0\|_2, v_1 = \frac{r_0}{\beta}$
-

$$\begin{cases} x_m = x_0 + V_m y_m \\ \underbrace{V_m^t AV_m}_{\stackrel{3.6}{=} H_{m,m}} \underbrace{y_m}_{\beta v_1} = V_m^t r_0 \iff H_{m,m} y_m = \beta e_1 \end{cases}$$

Algorithm 3.16 Verfahren der vollständigen Orthogonalisierung**Input:** $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$, $x_0 \in \mathbb{R}^n$, $m \in \mathbb{N}$ **Output:** $x_m \in x_0 + K_m(A, r_0)$, $b - Ax_m \perp K_m(A, r_0)$, $x_m \approx A^{-1}b$

$$r_0 = b - Ax_0, \beta = \|r_0\|_2, v_1 = \frac{r_0}{\beta}$$

for $j = 1, m$ **do**Bestimme $V_j, H_{j,j}$ mit Arnoldi-Verfahren (Stop beim Abbruch)Löse $\underbrace{H_{j,j}}_{\in \mathbb{R}^{j \times j}} y_j = \beta e_1$

$$x_j = x_0 + V_j y_j$$

Konvergenztest

end for

Was ist ein geeigneter Konvergenztest?

Lemma 3.17. *Im Algorithmus 3.16 gilt*

$$r_m = b - Ax_m = -h_{m+1,m}(e_m^t y_m) V_{m+1}$$

d.h. es gilt auch

$$\|r_m\|_2 = \underbrace{|h_{m+1,m}(e_m^t y_m)|}_{>0} = h_{m+1,m} |e_m^t y_m|$$

Beweis.

$$\begin{aligned} b - Ax_m &= b - A(x_0 + V_m y_m) \\ &= r_0 - AV_m y_m = (\star) \end{aligned}$$

$$\begin{aligned} (\star) &= \underbrace{r_0}_{\beta v_1} - \underbrace{AV_m}_{\stackrel{3.5}{=} V_m H_{m,m} - w_m e_m^t} \\ &= \beta v_1 - \underbrace{V_m \underbrace{H_{m,m} y_m}_{=\beta e_1}}_{\beta v_1} - \underbrace{w_m}_{=h_{m+1,m}} (e_m^t y_m) \\ &= -h_{m+1,m}(e_m^t y_m) v_{m+1} \end{aligned}$$

□

Bemerkung:

- Hauptkosten (Alles mit Vektoren der Länge n)
 - Matrix-Vektor-Multiplikation (1, Mal, sehr teuer)
 - Skalarprodukte
 - Vektor-Updates
- \implies Berechnen des Residuums wie in Lemma (3.17) lohnt sich.
- Speicherbedarf und Aufwand per Iteration werden in jeder Iteration teuer!

Umgehen von großen m Wir können Neustarten, um m wieder auf 1 zu setzen und hohe Kosten von großen m zu verhindern.

3.7 Das GMRES-Verfahren

Idee: Lemma (3.17) gibt uns eine explizite Darstellung von $\|r_j\|_2$. Können wir die Idee dahinter benutzen, um $\|r_j\|_2$ in jeder Iteration zu minimieren?

$$\begin{aligned} r_j &= b - A \underbrace{x_j}_{=x_0 + V_j y_j} \\ &= \underbrace{b - Ax_0}_{=r_0 = \beta v_1 = V_{m+1} \beta e_1}_{\in \mathbb{R}^{m+1}} - \underbrace{AV_j}_{\stackrel{3.5}{=} V_{j+1} H_{j+1,j}} y_j \\ &= V_{j+1}(\beta e_1 - H_{j+1,j} y_j) \end{aligned}$$

$$\Rightarrow \|r_j\|_2 = \|V_{j+1}(\beta e_1 - H_{j+1,j} y_j)\|_2$$

$$\text{Da } \|V_{j+1} z_{j+1}\|_2^2 = z_{j+1}^t z_{j+1} = \|z_{j+1}\|_2^2$$

$$\Rightarrow \|r_j\|_2 = \left\| \underbrace{\beta e_1}_{\in \mathbb{R}^{j+1}} - \underbrace{H_{j+1,j}}_{\in \mathbb{R}^{j+1 \times j}} \underbrace{y_j}_{\in \mathbb{R}^j} \right\| \quad (3.7)$$

$\Rightarrow \|r_j\|_2$ wird minimal, falls y_j als Lösung des linearen Ausgleichsproblems $H_{j+1,j} y_j = \beta e_1$ gewählt wird.

Algorithm 3.18 GMRES-Verfahren, Generalized Minimal Residual Method, Prototyp

Input: $A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n, x_0 \in \mathbb{R}^n, m \in \mathbb{N}$

Output: $x_m \in x_0 + K_m(A, r_0), \|r_m\|_2 = \|b - Ax_m\|_2$ minimal

$$r_0 = b - Ax_0, \beta = \|r_0\|_2, v_1 = \frac{r_0}{\beta}$$

for $j = 1, m$ **do**

 Bestimme $V_{j+1}, H_{j+1,j}$ mit Arnoldi-Verfahren (Stop beim Abbruch)

 Löse $H_{j+1,j} y_j = \beta e_1$

$$x_j = x_0 + V_j y_j$$

 Konvergenztest

end for

Bemerkung 3.19. 1. Bis auf die Minimalitätseigenschaft von $\|r_j\|_2$ stimmen die Algorithmen 3.16 und 3.18 weitgehend überein. D.h. sie haben auch ähnliche Nachteile.

2. Algorithmus 3.18 kann auch als Projektionsmethode mit $K = K_m(A, r_0)$ und $L = AK$ hergeleitet werden. (2 Zeiler, wenn man die richtige Formel sieht)

Frage: wie können wir den Algorithmus 3.18 praxistauglich machen. D.h.: Was ist ein geeigneter Konvergenztest und wie lösen wir das lineare Ausgleichsproblem?

Beobachte: (3.7) sagt: $\|r_j\|_2$ ist genau der Fehler des linearen Ausgleichsproblems. Diesen Fehler können wir explizit berechnen!

Analog zum Beweis von Satz 2.11:

$$\text{Sei } H_{j+1,j} = Q_j R_j = \underbrace{Q_j}_{\in \mathbb{R}^{j+1, j+1}} \begin{bmatrix} R_j \\ 0 \\ \underbrace{\quad}_{\in \mathbb{R}} \end{bmatrix} \text{ eine QR-Zerlegung.}$$

Sei

$$Q^t \beta e_1 = \begin{bmatrix} \underbrace{\quad}_{\in \mathbb{R}^j} \\ b_1 \\ b_2 \\ \underbrace{\quad}_{\in \mathbb{R}} \end{bmatrix}$$

$$\|r_j\|_2^2 \stackrel{(3.7)}{=} \|\beta e_1 - H_{j+1,j} y_j\|_2^2 = \|Q^t(\beta e_1 - H_{j+1,j} y_j)\|_2^2 = (\star)$$

$$\begin{aligned} (\star) &= \left\| \begin{pmatrix} b_1 - \tilde{R}_j y_j \\ b_2 \end{pmatrix} \right\|_2^2 \\ &= \underbrace{\|b_1 - \tilde{R}_j y_j\|_2^2}_{=0 \text{ falls } H_{j+1,j} \text{ bzw. } \tilde{R}_j \text{ vollen Rang hat}} + \underbrace{\|b_2\|_2^2}_{=|b|^2} \\ &\implies \|r_j\|_2 = |b_2| \end{aligned}$$

Wir berechnen b_2 als Nebenprodukt beim Lösen des linearen Ausgleichsproblems.

Aber: In jeder Iteration eine QR-Zerlegung zu berechnen ist teuer. Wie geht es besser?

Beobachte:

- $H_{j+1,j}$ hat Hessenbergstruktur, d.h. $H_{j+1,j}$ ist eine rechte obere Dreiecksmatrix, wo zusätzlich die untere Diagonale nicht notwendigerweise 0 ist. Dafür kann mit j Givensrotationen eine QR-Zerlegung berechnet werden.
- Beim Iterationsschritt $j \implies j+1$ werden lediglich eine neue Spalte und eine neue Zeile an $H_{j+1,j}$ drangehängt um $H_{j+2,j+1}$ zu erhalten. Die restlichen Einträge bleiben unverändert.

Sei $H_{j+1,j} = Q_j R_j$ eine QR-Zerlegung.

$$\begin{bmatrix} Q_j^t & 0 \\ 0 & 1 \end{bmatrix} H_{j+2,j+1} = \begin{bmatrix} Q_j^t & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} H_{j+1,j} & \star \\ 0 & \star \end{bmatrix} =$$

Die letzte Matrix kann mittels einer einzigen Givensrotation in obere Dreiecksgestalt gebracht werden.

\implies Erweitere QR-Zerlegung in jedem Schritt.

— Ende von Vorlesung 08 am 08.11.2022 —

Algorithm 3.20 GMRES

Input: $A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n, x_0 \in \mathbb{R}^n, m \in \mathbb{N}$

Output: $x_m \in x_0 + K_m(A, r_0), \|r_m\|_2 = \|b - Ax_m\|_2$ minimal

$$r_0 = b - Ax_0, \beta = \|r_0\|_2, v_1 = \frac{r_0}{\beta}$$

$$\hat{b} = \beta e_1$$

for $j = 1, m$ **do**

Bestimme $V_{j+1}, H_{j+1,j}$ mit Arnoldi-Verfahren ▷ Durch Anhängen von Spalten/ Zeilen, Stop beim Abbruch

Wende $G_{12}, G_{23}, \dots, G_{j-1,j}$ auf die letzte Spalte von $H_{j+1,j}$ an

$$\text{Bestimme } G_{j,j+1} \text{ so, dass } H_{j+1,j} = G_{j,j+1} H_{j+1,j} = \begin{bmatrix} \tilde{R}_{j+1} \\ 0 \end{bmatrix}$$

$$\hat{b} = G_{j,j+1} \hat{b}$$

if \hat{b} klein **then**

$$\text{Löse } \tilde{R}_j y_j = [\hat{b}_i]_{i=1,\dots,j}$$

$$\text{Gebe } x_j = x_0 + V_j y_j \text{ zurück}$$

end if

end for

Berechne x_m wie oben, beginne von vorne mit $x_0 = x_m$

Bemerkung 3.21. 1. Außer, dass GMRES für invertierte Matrizen für $m = n$ konvergiert, ist bis heute wenig über Konvergenzaussagen bekannt.

2. Außer der Matrix-Vektor-Multiplikation im Arnoldi-Verfahren sind an der j -ten Iteration nur Vektor/Matrizen der Größe j , bzw. $j \times j$ beteiligt. Der Lösungsvektor wird erst nach erfülltem Konvergenzkriterium zusammengesetzt.
3. Für $m \ll n$ sprechen wir von einem **Restarted-GMRES**. Oft ist z.B. $m = 20$ ausreichend.
4. Das Verfahren der vollständigen Orthogonalisierung kann analog abgeändert werden.
5. Falls das Arnoldi-Verfahren abbricht, ist die Näherungslösung exakt ("Lucky Breakdown", Übung).
6. Das GMRES-Verfahren ist heutzutage (2022) eines der beliebtesten Verfahren zum Lösen LGS ohne weitere, besondere Eigenschaften.

3.8 Der symmetrische Lanczos-Prozess

Frage: Können wir ein besseres Verfahren herleiten, wenn wir zusätzliche Annahmen zu unserer Matrix treffen? z.B. Symmetrie oder spd?

Beobachte: Für A symmetrisch ist

$$H_{m,m} \stackrel{(3.6)}{=} V_m^t A V_m$$

symmetrisch und hat Hessenberg-Struktur

$$H_{m,m} = \begin{bmatrix} \alpha_1 & \beta_1 & & & & \\ \beta_1 & \alpha_2 & & & & \\ & \beta_2 & \ddots & & & \\ & & \ddots & \ddots & & \\ & & & \beta_{n-1} & \alpha_{n-1} & \beta_{n-1} \\ & & & & \beta_n & \alpha_n \end{bmatrix}$$

Die vom Arnoldi-Verfahren generierte Matrix $T_m = H_{m,m}$ ist tridiagonal und symmetrisch. **Idee:**

- Die null-Einträge $h_{ij} = \langle v_j, v_i \rangle, i = 1, \dots, j-2$ müssen gar nicht erst berechnet werden
- Wir können die Symmetrie im Algorithmus explizit ausnutzen.

Algorithm 3.22 Lanczos-Verfahren

Input: $A \in \mathbb{R}^{n \times n}$ symmetrisch, $v \in \mathbb{R}^n, x_0 \in \mathbb{R}^n, m \in \mathbb{N}$

Output: $V_m = [v_1 \dots v_m]$ von $K_m(A, v), v_{m+1} \in \mathbb{R}^n, \beta \in \mathbb{R}, T_m \in \mathbb{R}^{m \times m}$

$\beta_1 = 0, v_0 = 0$

for $j = 1, m$ **do**

$h_{ij} = 0, i = 1, \dots, j-2$

$h_{j-1,j} = \beta_j$

$w_j = z - \sum_{i=1}^{j-1} h_{ij} v_i = Av_j - \beta_j v_{j-1}$

▷ In der Praxis hat w keinen Index

$\alpha_j = \langle w_j, v_j \rangle_2 = h_{jj}$

$w_j = w_j - \alpha_j v_j$

▷ $w = Av_j - \alpha_j v_j - \beta_j v_{j-1}$

$\beta_{j+1} = \|w_j\|_2$

if $\beta_{j+1} = 0$ **then**

▷ in der Praxis: testen ob Betrag klein

Stop

end if $v_{j+1} = \frac{w_j}{\beta_{j+1}}$

end for

Berechne x_m wie oben, beginne von vorne mit $x_0 = x_m$

Bemerkung 3.23. Das Lanczos-Verfahren implementiert die Berechnung der ONB als Drei-Term-Rekursion, die für höhere Iterationszahlen instabil werden kann.

Idee: Wende unser neues Verfahren auf die Methode der vollständigen Orthogonalisierung an.

Algorithm 3.24 Lanczos-Verfahren für lineare Gleichungssysteme

Input: $A \in \mathbb{R}^{n \times n}$ symmetrisch, $b \in \mathbb{R}^n, x_0 \in \mathbb{R}^n, m \in \mathbb{N}$

Output: $x_m \in x_0 + K_m(A, r_0), x_m \approx A^{-1}b$

$$r_0 = b - Ax_0, \beta = \|r_0\|_2, v_1 = \frac{r_0}{\beta}$$

for $j = 1, m$ **do**

Bestimme V_j, T_j mittel Lanczos-Verfahren (Stop bei Abbruch)

Löse $T_j y_j = \beta e_1$

$x_j = x_0 + V_j y_j$

Konvergenztest

end for

Berechne x_m wie oben, beginne von vorne mit $x_0 = x_m$

Bemerkung 3.25. 1. Lemma 3.17, d.h.

$$\|r_j\|_2 = \|b - Ax_j\|_2 = \beta_{j+1} |e_j^t y_j|$$

gilt weiterhin. Wir können den Konvergenztest also ohne Berechnung von x_j durchführen.

2. Da T_j tridiagonal ist, kann $T_j y_j = \beta e_1$ in $O(j)$ gelöst werden.

3. Speicherbedarf und Rechenaufwand wachsen immer noch mit j .

—Ende von Vorlesung 09 am 10.11.2022—

Frage: Können wir es vermeiden die Basis V_j vom Krylovraum zu speichern?

Beobachte: Betrachte LR -zerlegung

$$T_m = L_m R_m = \begin{pmatrix} 1 & & & \\ \lambda_2 & \ddots & & \\ & \ddots & \ddots & \\ & & \lambda_m & 1 \end{pmatrix} \begin{pmatrix} \eta_1 & \beta_2 & & \\ & \ddots & \ddots & \\ & & \ddots & \beta_m \\ & & & \eta_m \end{pmatrix}$$

Zeile und Spalte ergänzen:

$$\lambda_m = \frac{\beta_m}{\eta_{m+1}}, \eta_m = \alpha_m - \lambda_m \beta_m$$

Wobei α_i die Einträge auf der Diagonalen und β_j die Einträge der Nebendiagonalen von T_j sind.

$$\implies x_m = x_0 + \underbrace{V_m y_m}_{T_m^{-1} \beta e_1 = R_m^{-1} L_m^{-1} \beta e_1}$$

$$x_0 = \underbrace{V_m R_m^{-1}}_{P_m} \underbrace{L_m^{-1} \beta e_1}_{z_m}$$

$$P_m R_m = V_m \implies v_m = \beta_m p_{m-1} + \eta_m p_m$$

$$\implies p_m = \frac{1}{2}(v_m - \beta_m p_{m-1})$$

D.h. P_m kann mit wachsendem m einfach und effizient berechnet werden.

$$z_m = L_m^{-1} \beta e_1 = \begin{pmatrix} & & & \mathbf{0} \\ & L_{m-1}^{-1} & & \vdots \\ & & & \mathbf{0} \\ 0 & \dots & 0 & -\lambda_m & 1 \end{pmatrix} \begin{pmatrix} \beta \\ \\ \\ 0 \end{pmatrix} = \begin{pmatrix} \\ z_{m-1} \\ \zeta_m \end{pmatrix}$$

$$\begin{bmatrix} & & L_{m-1}^{-1} & & \\ & & & & \\ 0 & \dots & 0 & -\lambda_m & 1 \end{bmatrix} \begin{bmatrix} z_{m-1} \\ \zeta_m \end{bmatrix} = L_m z_m = \beta e_1 = \begin{bmatrix} \beta \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Mit $\zeta_m = -\lambda_m \zeta_{m-1}$

$$\begin{aligned} \implies x_m &= x_0 + p_m z_m \\ &= x_0 + \begin{bmatrix} p_{m-1} & p_m \end{bmatrix} \begin{bmatrix} z_{m-1} \\ \zeta_m \end{bmatrix} \\ &= x_0 + \underbrace{P_{m-1} z_{m-1}}_{x_{m-1}} + \zeta_m p_m \\ &= x_{m-1} + \zeta_m p_m \end{aligned}$$

Algorithm 3.26 Direktes Lanczos-Verfahren für LGS

Input: $A \in \mathbb{R}^{n \times n}$ symmetrisch, $b \in \mathbb{R}^n, x_0 \in \mathbb{R}^n, m \in \mathbb{N}$

Output: $x_m \in x_0 + K_m(A, r_0), x_m \approx A^{-1}b$

$$r_0 = b - Ax_0, \zeta_1 = \beta = \|r_0\|_2, v_1 = \frac{r_0}{\beta_1}$$

$$\beta_1 = 0 \in \mathbb{R}, \eta_0 = 1 \in \mathbb{R}$$

$$p_0 = 0 \in \mathbb{R}^n$$

for $j = 1, m$ **do**

$$W_j = Av_j - \beta_j v_{j-1}$$

$$\alpha_j = \langle w_j, v_j \rangle$$

$$\frac{\beta_j}{\eta_{j-1}}, \eta_j = \alpha_j - \lambda_j \beta_j$$

$$\zeta_j = -\lambda_j \zeta_{j-1}$$

$$p_j = \frac{1}{2j}(v_j - \beta_j p_{j-1})$$

$$x_j = x_{j-1} + \zeta_j p_{j-1}$$

Konvergenztest

$$w_j = w_j - \alpha_j v_j$$

$$\beta_{j+1} = \|w_j\|_2, \text{ stop, falls } \beta_j = 0$$

$$v_{j+1} = \frac{1}{\beta_{j+1}} w_j$$

end for

Berechne x_m wie oben, beginne von vorne mit $x_0 = x_m$

▷ Wie beim Lanczos

▷ Wie beim Lanczos

▷ : Für $j > 1$

Bemerkung 3.27. 1. Wir müssen nur v_j, v_{j-1}, w_j, p_j und x_j aktiv im Speicher behalten. Der Speicherbedarf und Rechenaufwand pro Iteration ist unabhängig von j .

2. Algorithmus 3.24 und 3.26 sind mathematisch äquivalent, d.h. bei exakter Arithmetik erhalten wir die gleichen Iterierten. Wir müssen aber keine Vektoren vom Krylovraum speichern.

3. Bisher haben wir nur eine schnelle Version des Verfahrens der vollständigen Orthogonalisierung (Algorithmus 3.16) für symmetrische Matrizen hergeleitet. Unser Verständnis der Fehleranalyse hat sich nicht verbessert.

3.9 Das Verfahren der konjugierten Gradienten

Auch: Method of conjugate gradients / CG-Verfahren

Lemma 3.28. Seien $v_j, r_j = b - Ax_j$ und $p_j, j = 0, 1, \dots$ wie in den Algorithmen 3.24 und 3.26 generiert. Dann gilt:

1. $r_j = \sigma_{j+1} v_{j+1}$ mit $\sigma_{j+1} \in \mathbb{R}$. D.h. insbesondere, dass die Residuen r_j orthogonal sind.

2. Die p_j sind A -orthogonal, d.h. es gilt

$$\langle p_i, p_j \rangle_A = \langle Ap_i, p_j \rangle_2$$

für $i \neq j$.

Beweis. 1. folgt aus Lemma 3.17:

$$r_j = \underbrace{-h_{j+1,j}(e_j^t y_j)}_{\in \mathbb{R}} v_{j+1}$$

2.: Es reicht zu zeigen, dass $P_m^t A \underbrace{P_m}_{V_m R_m^{-1}} = R_m^{-t} \underbrace{V_m^t A V_m}_{=T_m=L_m R_m} R_m^{-1} = R_m^{-t} L_m$

$P_m^t A P_m$ ist symmetrisch und $R_m^{-t} L_m$ ist untere Dreiecksmatrix $\implies P_m^t A P_m$ ist diagonal. □

Idee: Vereinfache Algorithmus 3.26 zu einem Verfahren der Form

$$x_{j+1} = x_j + \tilde{\alpha}_j \tilde{p}_j, \tilde{p}_{j+1} = r_{j+1} + \tilde{\beta}_j \tilde{p}_j \quad (3.8)$$

benutze dazu die Eigenschaften aus Lemma 3.28.

Wie sieht r_{j+1} aus?

$$\begin{aligned} r_{j+1} &= b - A x_{j+1} \\ &\stackrel{3.8}{=} b - A x_j - \tilde{\alpha}_j A \tilde{p}_j \\ &= r_j - \tilde{\alpha}_j A \tilde{p}_j \end{aligned} \quad (3.9)$$

Außerdem

$$\begin{aligned} 0 &= \langle r_{j+1}, r_j \rangle_2 = \langle r_j - \tilde{\alpha}_j A \tilde{p}_j, r_j \rangle_2 = \langle r_j, r_j \rangle_2 - \tilde{\alpha}_j^2 \langle A \tilde{p}_j, r_j \rangle_2 \\ &\implies \tilde{\alpha}_j = \frac{\langle r_j, r_j \rangle_2}{\langle A \tilde{p}_j, r_j \rangle_2} = \frac{\langle r_j, r_j \rangle}{\langle A \tilde{p}_j, \tilde{p}_j \rangle_2} \end{aligned}$$

$$\begin{aligned} 0 &\stackrel{!}{=} \langle A \tilde{p}_j, r_j \rangle_2 = \langle A r_{j+1}, \tilde{p}_j \rangle_2 + \tilde{\beta}_j \langle A \tilde{p}_j, \tilde{p}_j \rangle_2 \\ &\implies \tilde{\beta}_j = -\frac{\langle r_{j+1}, A \tilde{p}_j \rangle_2}{\langle A \tilde{p}_j, \tilde{p}_j \rangle_2} \\ &\stackrel{(3.9)}{=} \frac{1}{\alpha_j} \frac{\langle r_{j+1}, r_{j+1} \rangle}{\langle A \tilde{p}_j, \tilde{p}_j \rangle_2} = \frac{\langle r_{j+1}, r_j \rangle_2}{\langle r_j, r_j \rangle_2} \end{aligned}$$

Algorithm 3.29 CG-Verfahren

Input: $A \in \mathbb{R}^{n \times n}$ spd, $b \in \mathbb{R}^n, x_0 \in \mathbb{R}^n$

Output: $x_m \approx A^{-1}b$

```

 $r_0 = b - A x_0, \tilde{p}_0 = r_0$ 
for  $j = 1, m$  do
   $\tilde{\alpha}_j = \frac{\langle r_j, r_j \rangle_2}{\langle A \tilde{p}_j, \tilde{p}_j \rangle_2}$ 
   $x_{j+1} = x_j + \tilde{\alpha}_j \tilde{p}_j$ 
   $r_{j+1} = r_j - \tilde{\alpha}_j A \tilde{p}_j$ 
   $\tilde{\beta}_j = \frac{\langle r_{j+1}, r_j \rangle_2}{\langle r_j, r_j \rangle_2}$ 
   $\tilde{p}_{j+1} = r_{j+1} + \tilde{\beta}_j \tilde{p}_j$ 
  Konvergenztest
end for
```

— Ende von Vorlesung 10 am 15.11.2022 —

Nachtrag:

Beim direkten Lanczosverfahren:

$$\rho_j = \frac{1}{2j}(v_j - \beta_j p_{j-1})$$

$$x_j = x_{j-1} + \zeta_j p_j$$

Es folgt mit $r_j = \sigma_{j+1} v_{j+1}$, $\sigma_{j+1} \in \mathbb{R}$, dass die r_j orthogonal sind. Dann gilt

$$\langle A p_j, p_i \rangle_2 = 0, i \neq j$$

$$\implies x_{j+1} = x_j + \tilde{\alpha}_j \tilde{p}_j$$

mit

$$\tilde{p}_{j+1} = r_{j+1} + \tilde{\beta}_j \tilde{p}_j$$

wobei es von \tilde{p} zu p einen Indexshift gibt!

Bemerkung 3.30. 1. Man kann per Induktion zeigen, dass die Iterierten x_j (und damit die Residuen r_j) des CG-Verfahrens stimmen mit denen des direkten Lanczos-Verfahrens überein. Die \tilde{p}_i sind vielfache der p_{j+1} .

2. Da CG-Verfahren muss nur $x_j, r_j \tilde{p}_j$ und (zur Beschleunigung) $A \tilde{p}_j$ speichern.

3. Mit $\langle r_j, r_j \rangle_2 = \|r_j\|_2^2$ wird $\|r_j\|_2$ bis auf Kosten einer Quadratwurzel berechnet.

Was ist neu? A spd $\implies \langle A \cdot, \cdot \rangle_2 = \langle \cdot, \cdot \rangle_A$ ist ein Skalarprodukt und $\|\cdot\|_A = \sqrt{\langle \cdot, \cdot \rangle_A}$ ist eine Norm.

Die Iterierten von CG und Lanczos sind gleich \implies CG-Verfahren ist ein Projektionsverfahren.

Finde $x_j \in x_0 + K_j(A, r_0)$ mit $r_j = b - A x_j \perp R_j(A, r_0)$.

$$\begin{aligned} \implies 0 &= \langle r_j, y \rangle_2, y \in K_j(A, r_0) \\ &= \langle b - A x_j, y \rangle_2 \\ &= (\star) \end{aligned}$$

$$\begin{aligned} (\star) &= \langle A x - A x_j, y \rangle_2 \\ &= \langle A(x - x_j), y \rangle_2 \\ &= \langle x - x_j, y \rangle_A \end{aligned}$$

$$\implies x - x_j \perp_A K_j(A, r_0)$$

$$\stackrel{\text{Satz 1.9}}{\implies} \|x - x_j\|_A = \min_{y \in K_j(A, r_0)} \|x - y\|_A \quad (3.10)$$

Bemerkung 3.31. (3.10) ist eine Aussage über den Fehler, nicht das Residuum!

Lemma 3.32. Sei x_j die j -te Iterierte des CG-Verfahrens. Dann ist $x_j = x_0 + q_{j-1}(A)r_0$, wobei $q_{j-1} \in \Pi_{j-1}$ mit

$$\left\| \underbrace{(I - q_{j-1}(A)A)(x - x_0)}_{=x-x_j} \right\|_A = \min_{p \in \Pi_{j-1}} \|(I - p(A)A)(x - x_0)\| \quad (3.11)$$

Beweis. $x_j = x_0 + K_j(A, r_0) \implies x_j = x_0 + p_{j-1}(A) \underbrace{r_0}_{=b-Ax_0=A(x-x_0)}, p_{j-1} \in \Pi_{j-1}$. Mit (3.10) folgt die

Behauptung. □

Lemma 3.33. Für die Iterierten x_j des CG-Verfahrens gilt

$$\frac{\|x - x_j\|_A}{\|x - x_0\|_A} \leq \min_{p \in \Pi_j, p(0)=1} \max_{\lambda \text{ Ew von } A} |p(\lambda)| \quad (3.12)$$

Beweis. Seien $(\lambda_i, w_i)_{i=1}^n$ die Eigenpaare von A . $x - x_0 = \sum_{i=1}^n \xi_i w_i$.

$$\begin{aligned} \|x - x_j\|_A^2 &\stackrel{\text{Lemma 3.32}}{=} \min_{p \in \Pi_j, p(0)=1} \|p(A)(x - x_0)\|_A^2 \\ &= \left\| p(A) \sum_{i=1}^n \xi_i w_i \right\|_A^2 \\ &= \left\| \sum_{i=1}^n \xi_i p(\lambda_i) w_i \right\|_A^2 \\ &= \left\langle A \sum_{i=1}^n \xi_i p(\lambda_i) w_i, \sum_{i=1}^n \xi_i p(\lambda_i) w_i \right\rangle_2 \\ &= \left\langle \sum_{i=1}^n \lambda_i \xi_i p(\lambda_i) w_i, \sum_{i=1}^n \xi_i p(\lambda_i) w_i \right\rangle_2 \\ &= \sum_{i=1}^n \lambda_i \xi_i^2 p(\lambda_i)^2 \\ &\leq \max_{i=1, \dots, n} |p(\lambda_i)| \underbrace{\sum_{i=1}^n \lambda_i^2 \xi_i^2}_{\|x - x_0\|_A^2} \end{aligned}$$

□

Korollar 3.34. Falls A spd l verschiedene Eigenwerte hat, dann konvergiert das Verfahren in l Schritten.

Beweis. Wähle

$$p(x) = \left(1 - \frac{x}{\lambda_1}\right) \cdots \left(1 - \frac{x}{\lambda_l}\right)$$

in Lemma 3.33.

□

Ziel: Finde eine ähnliche Konvergenzaussage für Eigenwerte $0 < \lambda_{\min} = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n = \lambda_{\max} < \infty$ von spd Matrizen.

Idee: Beschränke die rechte Seite von (3.12) geeignet:

$$\min_{p \in \Pi_j, p(0)=1} \max_{\lambda \text{ Ew von } A} |p(\lambda)| \leq \min_{p \in \Pi_j, p(0)=1} \underbrace{\max_{\lambda \in [\lambda_{\min}, \lambda_{\max}]} |p(\lambda)|}_{\|p\|_C([\lambda_{\min}, \lambda_{\max}])} \quad (3.13)$$

Frage: Für welche oder welches Polynom ist dieser Ausdruck minimal?

Definition 3.35 (Alma 2: Definition 15.10). Für $n \in \mathbb{N}_0$ sind die **Tschebyscheff-Polynome** definiert als

$$T_n(x) = \cos(n \cos^{-1}(x)) \in \Pi_n, x \in [-1, 1]$$

Satz 3.36 (Alma 2: Satz 15.12). Unter allen Polynomen $p_n(x) = x^n + \cdots \in \Pi_n$ ist $\|p_n\|_{C([-1,1])}$ für $p_n = 2^{1-n} T_n$ minimal.

Idee: Modifiziere diese Aussage um (3.13) zu beschränken.

Lemma 3.37. Sei $[a, b] \subset \mathbb{R}$ und $t_0 \in \mathbb{R} \setminus [a, b]$. Dann minimiert

$$\hat{T}_n(t) = \frac{T_n(x(t))}{T_n(x(t_0))}, x(t) = 2 \frac{t-a}{b-a} - 1$$

den Ausdruck $\|\cdot\|_{C([a,b])}$ unter allen Polynomen $p \in \Pi_n$ mit $p(t_0) = 1$

Beweis. Übung. □

$$\Rightarrow \min_{p \in \Pi_j, p(0)=1} \max_{\lambda \text{ Ew von } A} |p(\lambda)| \stackrel{(3.13), \text{Lemma 3.37}}{\leq} \left\| \hat{T}_j \right\|_{C([\lambda_{\min}, \lambda_{\max}])} = \frac{1}{|T_j(x(0))|} \quad (3.14)$$

Satz 3.38. Für den Approximationsfehler $x - x_0$ in der j -ten Iteration des CJ-Verfahrens gilt

$$\|x - x_j\|_A \leq 2 \left(\frac{\sqrt{\text{cond}_2(A)} - 1}{\sqrt{\text{cond}_2(A)} + 1} \right)^j \|x - x_0\|_A$$

Beweis. In (3.14) ist

$$\begin{aligned} x_0 &= 2 \frac{0 - \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}} - 1 \\ &= - \frac{\lambda_{\max} + \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}} \\ &= - \frac{\text{cond}_2(A) + 1}{\text{cond}_2(A) - 1} \end{aligned}$$

Außerdem:

$$\begin{aligned} T_j(x(0)) &= T_j \left(- \frac{\text{cond}_2(A) + 1}{\text{cond}_2(A) - 1} \right) \\ &\stackrel{\text{Übung}}{=} \frac{1}{2} \left(\left(\frac{\sqrt{\text{cond}_2(A)} + 1}{\sqrt{\text{cond}_2(A)} - 1} \right)^j + \left(\frac{\sqrt{\text{cond}_2(A)} - 1}{\sqrt{\text{cond}_2(A)} + 1} \right)^j \right) \\ &\geq \frac{1}{2} \left(\frac{\sqrt{\text{cond}_2(A)} + 1}{\sqrt{\text{cond}_2(A)} - 1} \right)^j \end{aligned}$$

Dann folgt mit (3.14) die Behauptung. □

Bemerkung 3.39. 1. Das Verfahren des steilsten Absties hat eine ähnliche Abschätzung aber mit $\text{cond}_2(A)$ statt $\sqrt{\text{cond}_2(A)}$.

2. Für spd Matrizen führen die Algorithmen 3.16, 3.24, 3.26 auf die gleiche Abschätzung, sind aber teurer. Das CG-Verfahren ist das wichtigste aller Verfahren (auch in der Prüfung!) zum Lösen von LGS.

$$\underbrace{M}_{\approx A^{-1}} Ax = Mb$$

Ende von Vorlesung 11 am 17.11.2022

Kapitel 4

Lineare Eigenwertprobleme

4.1 Problemstellung und Beispiele

Definition 4.1. Gegeben sei eine Matrix $A \in \mathbb{C}^{n \times n}$ mit $n \in \mathbb{N}$. $(\lambda, x) \in \mathbb{C} \times \mathbb{C}^n \setminus 0$ heißt **Eigenpaar (EP)** von A , falls

$$Ax = \lambda x \quad (4.1)$$

λ heißt *Eigenwert (Ew)* und x *Eigenvektor (EV)*. (4.1) heißt **spezielles Matrixeigenwertproblem**.

Bemerkung 4.2. 1. Gegeben $A, B \in \mathbb{C}^{n \times n}$ heißt

$$Ax = \lambda Bx \quad (4.2)$$

das **allgemeine Matrixeigenwertproblem**. Falls B invertierbar ist, dann ist (4.2) äquivalent zu (4.1), d.h.

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

Falls B eine Cholesky-Zerlegung hat, ist (4.2) äquivalent zu (4.1), d.h.

$$L^{-1}Ax = \lambda \underbrace{L^t x}_{:=y}$$

$$L^{-1}AL^{-t}y = \lambda y$$

2. Wir betrachten nur spezielle Matrixeigenwertprobleme und sprechen nur von **Eigenwertproblemen**. Es gibt aber auch Algorithmen, die (4.2) in der allgemeinsten Form lösen.

3. Ist $A = A^t \in \mathbb{R}^{n \times n}$ in (4.1) sprechen wir vom **symmetrischen Eigenwertproblem**.

Beispiel 4.3. Viele Räuber-Beute Modelle in Biologie, Epidemiologie und Wirtschaft können als gewöhnliche Differentialgleichungen geschrieben werden

$$y'(t) = Ay(t), y(0) = y_0 \in \mathbb{R}^n, A \in \mathbb{R}^{n \times n}$$

geschrieben werden.

Sei $A = V\Lambda V^*$ eine Diagonalform von A mit $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$. Dann ist

$$y(t) = e^{At}y_0, y_0 = Ve^{\Lambda t}V^*y_0 = V \begin{bmatrix} e^{\lambda_1 t} & & \\ & \ddots & \\ & & e^{\lambda_n t} \end{bmatrix}$$

Mit

$$e^{At} = \sum_{i=0}^{\infty} \frac{(At)^i}{i!} = V \sum_{i=0}^{\infty} \frac{(\Lambda t)^i}{i!} V^*$$

Beispiel 4.4. Gegeben einer Reihe von Messwerten oder Datenpunkten $x_i \in \mathbb{R}^n, i = 1, \dots, m$, welche in einer Datenmatrix angeordnet sind:

$$X = [x_1 | \dots | x_n] \in \mathbb{R}^{n \times m}$$

Der Einfachheit halber sei

$$\frac{1}{m} \sum_{i=1}^m x_i = 0$$

Die Kovarianzmatrix

$$C = \frac{1}{m} X X^t = \frac{1}{m} \sum_{i=1}^m x_i x_i^t$$

Seien $(\lambda_i, v_i), i = 1, \dots, n$ die EP von C , mit $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq 0$

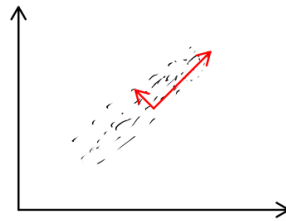


Abbildung 4.1: PCA Beispiel

Dann geben die Eigenwerte λ_i die Varianzen der x_i in Richtung v_i an. D.h. die x_i haben in Richtung v_i die größte Varianz, etc.

Beispiel 4.5. Das Schwingen einer Trommel der Form $\Omega \subset \mathbb{R}^2$, offen, mit Frequenz λ erfüllt

$$\begin{aligned} \Delta u(x) &= \lambda u(x), & x \in \Omega \\ u(x) &= 0, & x \in \partial\Omega \end{aligned}$$

Mit einem ähnlichen Vorgehen wie in Kapitel 3.1 erhalten wir ein Matrixeigenwertproblem

$$Ax = \lambda x$$

Ein Beispiel für A ist Beispiel 3.1. A ist immer spd. Alternative Diskretisierungsmethoden führen auf

$$Ax = \lambda Bx$$

mit A, B spd, sehr groß und dünn besetzt.

Bemerkung 4.6. Sei $A \in \mathbb{C}^{n \times n}$.

1. Die EW von A sind die Nullstellen des charakteristischen Polynoms

$$p_A(\lambda) = \det(A - \lambda I)$$

$$p_A \in \Pi_n \implies p_A \text{ hat } n \text{ komplexe Nullstellen}$$

$\implies A$ hat n (möglicherweise gleiche) Eigenwerte.

2. Für A hermitisch $A = A^*$ sind die EW reell. Die zugehörigen EV können als ONB genutzt werden. Falls A zusätzlich reell ist, können die EV reell und orthogonal gewählt werden.
3. $A \in \mathbb{R}^{n \times n}, Ax = \lambda x$ ($\lambda \in \mathbb{C}$ möglich), so folgt

$$A\bar{x} = \overline{Ax} = \overline{\lambda x} = \bar{\lambda} \bar{x}$$

$\implies (\bar{\lambda}, \bar{x})$ ist auch ein EP.

4. Die EW zu einem Gegeben EW sind nicht eindeutig:

$$Ax = \lambda x \implies Ay = \lambda y, y = \alpha x, \alpha \in \mathbb{C} \setminus \{0\}$$

$$Ax_1 = \lambda x_1$$

$$Ax_2 = \lambda x_2$$

$$\implies A\alpha x_1 + \beta x_2 = \lambda(\alpha x_1 + \beta x_2)$$

5. hnliche Matrizen, d.h. Matrizen der Form $B = CAC^{-1}$, C regulär, haben die gleichen EW.

Frage: Was können wir sonst noch sagen?

4.2 Gerschgorin-Kreise (Abschätzungen für EW)

Sei (λ, x) ein EP von $A \in \mathbb{C}^{n \times n}$. Betrachte die l te Zeile von $Ax = \lambda x$:

$$\lambda x_l = \sum_{j=1}^n a_{lj} x_j = a_{ll} x_l + \sum_{j=1, j \neq l}^n a_{lj} x_j \quad (4.3)$$

Sei i so, dass $|x_i| = \max_{j=1, \dots, n} |x_j| > 0$.

Dann gilt:

$$\begin{aligned} |\lambda - a_{ii}| |x_i| &= |(\lambda - a_{ii}) x_i| \\ &\stackrel{(4.3)}{=} \left| \sum_{j=1, j \neq i}^n a_{ij} x_j \right| \\ &\leq \sum_{j=1, j \neq i}^n |a_{ij}| |x_i| \\ &\leq |x_i| \sum_{j=1, j \neq i}^n |a_{ij}| := r_i \\ \implies |\lambda a_{ii}| &\leq \sum_{j=1, j \neq i}^n |a_{ij}| = r_i \end{aligned}$$

$\implies \lambda$ liegt in einem Kres mit Radius r_i .

Aber: x ist unbekannt $\implies i$ ist unbekannt

$\implies \lambda$ liegt in der Vereinigung aller solche Kreise

—Ende von Vorlesung 12 am 22.11.2022—

Satz 4.7. [Satz von Gerschgorin] Alle Eigenwerte einer Matrix $A \in \mathbb{C}^{n \times n}$ liegen in

$$\bigcup_{i=1}^n K_i$$

mit

$$K_i = \{z \in \mathbb{C} : |z - a_{ii}| \leq r_i\}, r_i = \sum_{j=1, j \neq i}^n |a_{ij}|, i = 1, \dots, n$$

Beweis. Oben. □

Bemerkung 4.8. 1. Bilden k Geschkorin-Kreise eine von den restlichen Kreisen disjunkte Punktmenge, so liegen in dieser Punktmenge genau k EW.

2. Ist $A \in \mathbb{R}^{n \times n}$, dann haben A und A^t die gleichen Eigenwerte. \implies EW von A liegen im Durchschnitt der jeweiligen Kreise.

3.

4.3 Kondition des Eigenwertproblems

Wiederholung: Die Kondition beschreibt, wie sich Änderungen in den Eingabedaten auf die Werte der Ausgabe auswirken.

$$\kappa_{\text{abs}} = |f'(x)| = \frac{\Delta y}{\Delta x}$$

Frage: Ist das Eigenwertproblem gut- oder Schlechtkonditioniert?

Kondition des Problems

In Matrix & Computations ist das ausführlich beschrieben.

Beispiel 4.9. Sei $\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \Rightarrow$ EW sind $\lambda_{1/2} = 0$.

Sei $\tilde{A} = \begin{bmatrix} 0 & 1 \\ \delta & 0 \end{bmatrix}, \delta > 0 \Rightarrow$ EW sind $\lambda_{1/2} = \pm\sqrt{\delta}$.

$$\kappa_{\text{abs}} = \frac{|0 - \pm\sqrt{\delta}|}{\left\|A - \tilde{A}\right\|_2} = \frac{|\sqrt{\delta}|}{|\delta|} \xrightarrow{\delta \rightarrow 0} \infty$$

Daher ist das Eigenwertproblem in diesem Fall schlechtkonditioniert.

Bemerkung 4.10. Allgemeiner kann man zeigen, dass (DAH Satz 5.2) die absolute Kondition eines einfachen Eigenwerts λ von $A \in \mathbb{C}^{n \times n}$ bzgl. $\|\cdot\|_2$ gegeben ist durch

$$\kappa_{\text{abs}} = \frac{\|x\| \|y\|}{\langle x, y \rangle_2} = \frac{1}{|\cos(\angle(x, y))|}$$

wo bei $Ax = \lambda x$ (Rechteigenvektor) und $A^*y = \bar{\lambda}y$ (Linkseigenvektor). D.h. einfache Eigenwerte hermitescher Matrizen sind gutkonditioniert, da x parallel zu y ist.

Bei mehrfachen oder nahe zusammenliegenden EW ist die Berechnung einzelner EV-Suche schlechtkonditioniert. Die Berechnung einer ONB des zugehörigen Eigenpaars ist aber gutkonditioniert.

Bemerkung 4.11. Die kanonische Idee die Eigenwerte als Nullstellen des charakteristischen Polynoms zu berechnen ist schlechtkonditioniert. Tatsächlich wird die Nullstellenberechnung in der Praxis als Matrixeigenwertproblem gelöst.

Berechnung der EW

$(x-1)(x-2)\cdots(x-20)$ ist ein gutes Beispiel, warum das Berechnen von Nst (hier Eigenwerten) schwer ist, weil die Koeffizienten sehr groß werden. Außerdem ist das Berechnen des Polynoms, durch Berechnung der Determinante, teuer.

4.4 Vektoriteration (Potenzmethode, etc)

Idee: Multipliziere die Matrix mit einem Vektor.

Iterationsfolge: Gegeben $A \in \mathbb{C}^{n \times n}, x_0 \in \mathbb{C}^n$, setze

$$x_{k+1} = Ax_k$$

Satz 4.12. Sei $A \in \mathbb{R}^{n \times n}, x \in \mathbb{R}^n$ mit $A = A^t$ symmetrisch, mit einfachem betragsmäßig größtem EW λ_1 , d.h.

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n| \geq 0.$$

Sei $x_0^t u_1 \neq 0, u_1$ EV zu λ_1 . Dann konvergiert

$$y_k = \frac{x_k}{\|x_k\|}, x_{k+1} = Ax_k, k = 0, 1, \dots$$

zu u_1 . Die Konvergenz ist alternierend, falls $\lambda_1 < 0$.

Außerdem gilt

$$y_k^t A y_k \xrightarrow{k \rightarrow \infty} \lambda_1 \quad (4.4)$$

Beweis. Sei u_1, \dots, u_n eine ONB von \mathbb{R}^n aus EV von A , mit u_i EV von λ_i . Sei

$$x_0 = \sum_{i=1}^n \alpha_i u_i \quad (4.5)$$

$$\Rightarrow x_k = A^k x_0 \stackrel{4.5}{=} \sum_{i=1}^n \alpha_i A^k u_i = (\star)$$

$$\begin{aligned} (\star) &= \sum_{i=1}^n \alpha_i \lambda_i^k u_i \\ &= \alpha_1 \lambda_1^k u_1 + \sum_{i=2}^n \alpha_i \lambda_i^k u_i \\ &= \alpha_1 \lambda_1^k \left(u_1 + \underbrace{\sum_{i=2}^n \frac{\alpha_i}{\alpha_1} \frac{\lambda_i^k}{\lambda_1^k} u_i}_{= z_k \xrightarrow{k \rightarrow \infty} u_1} \right) \end{aligned}$$

$$\begin{aligned} y_k &= \frac{x_k}{\|x_k\|} \\ &= \frac{\alpha_1 \lambda_1^k z_k}{\|\alpha_1 \lambda_1^k z_k\|} = \begin{cases} = \pm \frac{z_k}{\|z_k\|_2} \rightarrow \pm u_1 & \lambda_1 > 0 \\ = \pm (-1)^k \frac{z_k}{\|z_k\|_2} \rightarrow \pm u_1 & \lambda < 0 \end{cases} \end{aligned}$$

z.z.: (4.4)

$$\begin{aligned} |y_k^t A y_k - \lambda_1| &= |y_k^t A y_k - \lambda_1 y_k^t y_k| \\ &= \left| \frac{x_k^t (A - \lambda_1 I) x_k}{x_k^t x_k} \right| \\ &= \left| \frac{x_0^t A^{2k} (A - \lambda_1 I) x_0}{x_0^t A^{2k} x_0} \right| \\ &\stackrel{4.5}{=} \left| \frac{\sum_{i=1}^n \alpha_i^2 \lambda_i^{2n} (\lambda_i - \lambda_1)}{\sum_{i=1}^n \alpha_i^2 \lambda_i^{2k}} \right| = (\star\star) \end{aligned}$$

$$\begin{aligned} (\star\star) &= \max_{i=2, \dots, n} |\lambda_i - \lambda_1| \underbrace{\left| \frac{\sum_{i=2}^n \alpha_i^2 \lambda_i^{2k}}{\alpha_1^2 \lambda_1^{2k}} \right|}_{= \frac{1}{\alpha_1^2} \sum_{i=2}^n \alpha_i^2 \left(\frac{\lambda_i}{\lambda_1} \right)^{2k}} \\ &= \max_{i=2, \dots, n} |\lambda_i - \lambda_1| \left(\frac{\max_{i=2, \dots, n} |\lambda_i|}{\lambda_1} \right)^{2k} \end{aligned}$$

□

Bemerkung 4.13. 1. Um Overflow/Underflow zu vermeiden, muss die Iteration angepasst werden:

$$y_0 = \frac{x_0}{\|x_0\|_2}, y_1 = Ay_0, y_k = \frac{x_k}{\|x_k\|_2}, k = 0, 1, \dots$$

Die Eigenschaften bleiben unverändert.

2. Der Satz 4.12 gilt auch für mehrfache EW λ_1 .

3. In der Praxis ist $x_0^t u_1 \neq 0$ wegen Rundungsfehlern nicht von Bedeutung, muss also nicht überprüft werden.

4. Die Konvergenzgeschwindigkeit hängt von $\left|\frac{\lambda_2}{\lambda_1}\right|$ ab. Falls $|\lambda_1| \approx |\lambda_2|$ konvergiert das Verfahren sehr langsam.

— Ende von Vorlesung 13 am 24.11.2022 —