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

Manuel Hinz

24. Januar 2023

Inhaltsverzeichnis

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 $\langle \cdot, \cdot \rangle : X \times X \to \mathbb{R}$ eine Abbildung. $\langle \cdot, \cdot \rangle$ heißt* ***Skalarprodukt*** *oder inneres Produkt, falls*

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

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

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

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 Abhänigkeit 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 gil die Cauchy-Schwarz-Ungleichung:*

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

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

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

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

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

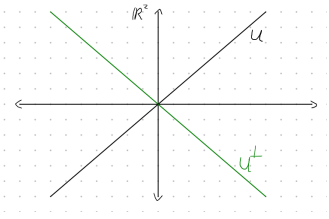

Abbildung 1.1: U und $U^\perp$

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

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

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*

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

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:*

$$v_1 - Pv_1 \in U^\perp$$
$$v_2 - Pv_2 \in U^\perp$$

Daher

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

Aber dann muss $\alpha P v_1 + P v_2$ 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 $\{\varphi_1, \ldots, \varphi_n\}$. 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:*

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

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 eine trigonometrischen Polynoms.

Satz 1.16. *Sei $\{\varphi_1, \ldots, \varphi_n\}$ 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.:*

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

Beweis. 1.:

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

2.:

$$\|f\|^2 = \langle f, f \rangle$$
$$= \langle \sum_{i=1}^{n} \alpha_i \varphi_i, \sum_{j=1}^{n} \alpha_j \varphi_j \rangle = \sum_{i,j=1}^{n} \alpha_i \alpha_j \delta_{i,j} = \sum_{i=1}^{n} \alpha_i^2$$

3.:

$f \in X \setminus X_n$:

$$\|f - \underbrace{\tilde{f}_n}_{\in X_n}\| = \langle f - \sum_{i=1}^{n} \tilde{\alpha}_i \varphi_i, f - \sum_{i=1}^{n} \tilde{\alpha}_i \varphi_i \rangle$$
$$= \|f\|^2 - 2 \sum_{i=1}^{n} \tilde{\alpha}_i \underbrace{\langle \varphi_i, f \rangle}_{=:\alpha_i} + \sum_{i,j=1}^{n} \alpha_i \alpha_j \langle \varphi_i, \varphi_j \rangle$$
$$= \|f\|^2 - \sum_{i=1}^{n} \tilde{\alpha}_i \alpha_i + \sum_{i=1}^{n} \tilde{\alpha}_i^2 \overset{\text{Quadratische Ergänzung}}{=} \|f\|^2 - \sum_{i=1}^{n} \alpha_i^2 + \sum_{i=1}^{n} \underbrace{(\alpha_i - \tilde{\alpha}_i)^2}_{\geq 0} \tag{1.1}$$

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

Ende von Vorlesung 01 am 11.10.2022

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

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

s.d.

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

Idee: Formuliere ein lineares Gleichungssystem:

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

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.

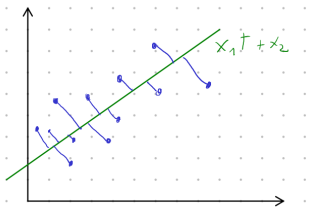

Abbildung 2.1: Datenpunkte und approximierte Gerade

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.:

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

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

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

Statte V mit euklidischem Skalarprodukt aus.
$\overset{Satz 1.9}{\Longrightarrow}$ 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*

$$A^t Ax = A^t b. \tag{2.2}$$

Insbesondere ist das lineare Ausgleichproblem genau dann lösbar, falls rang$(A) = n$.

Beweis.

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

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

Bemerkung 2.4. *Im beweis verwenden wir, dass $Ax - b$ orthogonal zu $U = Bild(A)$,*

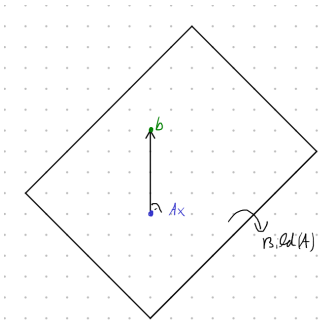

Abbildung 2.2: Hyperebene und Projektion

*d.h. eine Normale zur Hyperebene Bild(A) im R^m, ist. Deshalb heißt (2.2) auch **Normalengleichung**.*

Bemerkung 2.5. *Für $m = n$ und rang$(A) = n$ ist die Lösung des linearen Ausgleichproblems 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 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: $\text{rang}(A) = n \implies Ax = 0 \iff x = 0 \implies \|Ax\|_2 = 0 \iff x = 0 \implies$ Behauptung.

□

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

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

Eingefügte Bemerkung. *Anmerkung vom Donzent: A^tA eig. immer schlecht zu berechnen.*

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

Eingeschobene Definition / Wiederholung

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

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

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

$$\implies A^tA = \begin{bmatrix} 1+\epsilon^2 & 1 \\ 1 & 1+\epsilon^2 \end{bmatrix} \overset{im\ Computer}{=} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$\implies$ *A^tA ist im Computer singulär, 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 $Q^tQ = I$, 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. $cond_2(Q) = 1$

Beweis. 1.: $\|Qx\|_2^2 = \langle Qx, Qx\rangle_2 = \langle Q^tQx, x\rangle_2 = \langle x, x\rangle_2 = \|x\|_2^2$
2.: $\|Q\|_2 = \max_{\|x\|_2=1} \|Qx\| = 1$ und auch $\|Q^-1\|_2 = 1 \implies$ Behauptung. □

Satz 2.10. *$A \in \mathbb{R}^{m\times n}, m \geq n, rang(A) = n$. 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:

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

$\implies Q \in R^{m\times n}, R'R''$ ist obere Dreiecksmatrix mit nicht-null Diagonaleinträgen
$\implies$ invertierbar: $R = (R'R'')^{-1}$
$\implies QR = A$, wenn wir Q zu einer ONB von R^m erweitern. □

Ende von Vorlesung 02 am 13.10.2022

Satz 2.11. *Sei $A \in \mathbb{R}^{m\times n}, m \geq n, rang(A) = n, b \in \mathbb{R}^n$. 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.

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

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

□

Problem:

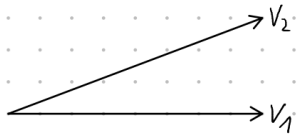

Abbildung 2.3: Problemstellung

$w_2 = v_2 - \frac{\langle v_2, v_1\rangle_2}{\langle v_1, v_1\rangle_2} v_1$ 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: $\beta = \|w\|_2$

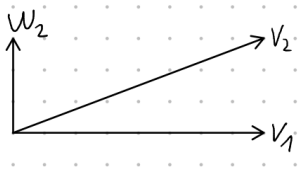

Abbildung 2.4: Gram-Schmidt

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

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

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

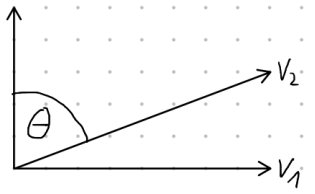

Abbildung 2.5: Drehungsansatz

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

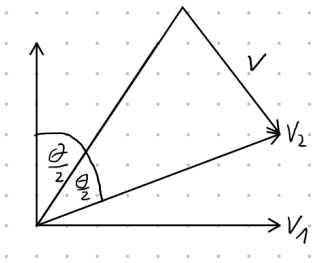

Abbildung 2.6: Spiegelungsansatz

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

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

Weil $\text{cond}_2(Q) = 1$ ist die Vorgehensweise stabil, bzw. gut konditioniert.
Aber: Wie wählen wir $Q_1, \ldots, Q_n$?

2.4 QR-Zerlegung mittels Givens-Rotationen

Definition 2.12. *Eine Matrix der Form*

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

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

Bemerke: 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 $x \in \mathbb{R}^n$, elemeniere $l$te Koordinate zu 0.

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

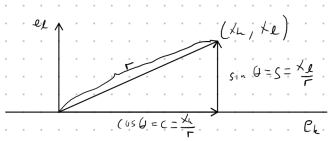

Abbildung 2.7: Trigonometriesetting

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

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

$$\begin{array}{c} \text{Falls } |x_l| > |x_k| \implies \tau = \frac{x_k}{x_l}, s = \frac{1}{\sqrt{1+\tau^2}}, c = s\tau \\ \text{Sonst: } \tau = \frac{x_l}{x_k}, c = \frac{1}{\sqrt{1+\tau^2}}, s = c\tau \end{array} \tag{2.3}$$

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, ..., n do
    for i = m, m-1, ..., j+1 do
        Berechne c, s wie in (2.3)
        A[i-1 : i, j : n] = [c  s; -s  c]^t A[i-1 : i, j : n]
    end for
end for
```

$\underline{m \approx n}$**:**
c, s: In jedem Eintrag einmal Wurzeln ziehen: $\implies \frac{n^2}{2}$ Quadratwurzeln und $\frac{4n^3}{3}$ Multiplikationen
$\underline{m \gg n}$**:** $m \cdot n$ 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 $\left[\begin{array}{c|c} A & b \end{array}\right]$ 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 uneffizient!

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

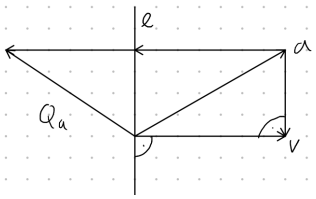

Abbildung 2.8: Householder-Transformationssetting

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

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

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

Effiziente Berechnung: Beobachte:

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

Ende von Vorlesung 03 am 18.10.2022

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

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

Dann ist

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

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 $j = 1, \ldots, n$ **do** ▷ Iterieren über die Spalten
  Berechne v, α wie in (2.4) ,mit $a = A[j : m, j] \in \mathbb{R}^{m-j+1}$
  $v = \frac{1}{v_1} v$ ▷ Erster Eintrag wird nicht gespeichert, daher normalisieren wir
  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] ▷ Index startet von 1
  end if
end for

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
$m \gg n \rightsquigarrow 2n^2 m$ 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: $A \in \mathbb{R}^{m \times n}, U = \text{Bild}(A)$

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

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 $Ax = b$.

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

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 $\tilde{x} \perp \ker(A)$ vorraussetzen dürfen, bevor wir das Skalarprodukt berechnen!
Zum Beispiel ist also $v = (1,0)^t$ im obigen Beipspiel doch im orthogonalen Komplement, da $\langle v, \tilde{x} + u - \tilde{x} \rangle_2 = 0$ für $u \in \ker(A)$

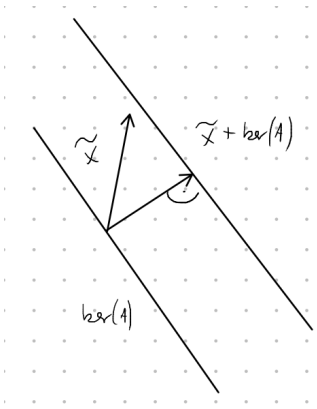

Abbildung 2.9: Setting

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

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

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 $A^\dagger \in \mathbb{R}^{m\times n}$ 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.:

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

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

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

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

Beweis.

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

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

Wir sehen $p = \operatorname{rang}(A) = n \implies$ wie vorher, lineares Ausgleichsproblem!
Sonst: $x_2 =?$

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

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

Beweis.

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

Minimiere $\varphi(x_2)$:

$$\begin{aligned}\varphi'(x_2) &= -2V^tu + 2(I+V^tV)x_2 \\ \varphi'(x_2) &= 2(I+V^tV) \implies \text{spd}\end{aligned}$$

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
$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = Q^tb$
$V = R^{-1}S$ mittels Rückwertssubstitution
$u = R^{-1}b_1$ mittels Rückwertssubstitution
Löse $(I+V^tV)x_2 = V^tu$ 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, Gleichungsysteme

3.1 Motivation

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

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

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

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

Die Lösung $u \in C^2(\Omega)$, falls existent, beschreit 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 $h = \frac{1}{n}, n \in \mathbb{N}$.

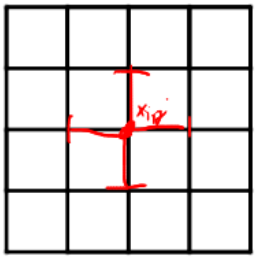

Abbildung 3.1: Gitter

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

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

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

Wir erhalten ein lineares Gleichungssystem mit $N = (n-1)^2$ Unbekannten und $O(1)$ Einträgen pro Zeile.
$\Longrightarrow$ $A \in \mathbb{R}^{n \times n}$ hat $O(N)$ Einträge. Wir haben das Lösen eines (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:*

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

Aber: Um die Lösung von (3.1) gut zu approximierenm ist oft $N \gg 1$ erforderlich. Für kleine bis mittlere N, d.h. in 2022 je nach Modell $\sim$ 10 Millionen, sind graphenbasierte Löser eine Option.
Was tun für große N?
Beobachtung: Matrix-Vektor-Multiplikation sind für dünn besetze 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 Startnährung 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: $\tilde{x} \in x_0 + K \implies \exists d \in K : \tilde{x} = x_0 + d$

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

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

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

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

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 $K = \text{span}\{v_1, \ldots, v_n\}, L = \{w_1, \ldots, w_n\}$
$V = [v_1 | \ldots | v_n]$ und $W = [w_1 | \ldots | w_n]$
(3.3) ist äquivalent zu

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

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

Algorithm 3.2 Prototyp einer interativen Projektionsmethode

Input: $A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n$, Fehlertoleranz α
Output: Nährung $x_{i+1} \approx x$

```
i=0
while Fehlertoleranz noch nicht erreicht do
    Wähle K_i, L_i
    Wähle Basen V, W von K_i, L_i
    r_1 = A x_i
    y = (W^t A V)^{-1} W^t r_i
    x_{i+1} = x_i + V y_i
    i = i + 1
end while
```

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

Beispiel 3.3.

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

$$K = L = span\{e_1, \ldots, e_m\} \implies V = W = \left[\frac{I_m}{0}\right] \in K^{2m \times m}$$

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

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 $\delta \in R^{m\times m}$ invertierbar.
$\implies B = W^tAV = \delta^tV^tAV$

$$0 < \underbrace{y^tAy}_{} \quad , y = Vx$$
$$= \underbrace{x^tV^tAVx}_{\text{spd, invertierbar}}$$

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

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

□

Ende von Vorlesung 05 am 25.10.2022

3.3 Verfahren des steilsten Abstiegs

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

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

Algorithm 3.5 Verfahren des steilsten Abstiegs

Input: A, b, Startvektor x_0, Fehlertoleranz
Output: Nährung $x_{i+1} \approx x$

```
while Fehlertoleranz noch nicht erreicht do        ▷ Praxis ϵ = 10^{-8}, ‖r_i‖_2 < ϵ
    r_i = b − Ax_i
    α_i = r_i^t r_i / (r_i^t A r_i)
    x_{i+1} = x_i + α_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

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

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 Abstiegsm dass:*

$$\|x - x_{i+1}\|_A \leq \frac{\lambda_{\max}(A) - \lambda_{\min}(A)}{\lambda_{\max}(A) + \lambda_{\min}(A)} \|x - x_i\|_A$$
$$\overset{\frac{1}{\lambda_{\min}}}{=} \frac{cond_2(A) - 1}{cond_2(A) + 1} \|x - x_i\|_A$$

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 verschlechtetertm d.h. die Konvergenz verschlechtert sich.*

3.4 Krylovräume

Beobachte: Im Verfahren des steilsten Abstiegs gilt:

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

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*

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

*heißt **Kylovraum** 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. $\dim(K_m(A, v)) = \dim(K_{m+1}(A, v)) = m \implies \dim(K_{m+i}(A, v)) = m, i = 0, 1, \ldots$
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 Startnährung x_0, Residuum $r_0 = b - Ax_0$. Für $i = 0, \ldots$. Wähle*

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

Das bedeutet: Sind K, L geeignete Krylovräume in einer Projektionsmethode, dann können wwir immer garantieren, dass das Ergebnis in einem Krylovraum ist.
Aber: Die Vektoren $v, Av, \ldots, A^n v$ 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

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

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 = v / ||v||_2
for j = 1, m do
    z = A v_j                                   ▷ = A^{j-1} r_0
    h_ij = ⟨z, v_i⟩_2, i = 1, ... j
    w_j = z − Σ_{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} = 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).

$$w_j = \underbrace{Av_j}_{\in K_j(A,r_0) \implies v_j = q_{j-1}(A)v} - \underbrace{\sum_{i=1}^{2} h_{i,j}v_i}_{\tilde{q}_{j-1}(A)v, \tilde{q}_{j-1}\in\Pi_{j-1}} = (\star)$$

$$(\star) = Aq_{j-1}(A)v_j - \tilde{q}_{j-1}(A)v_j = \underbrace{q_j(A)v}_{\in K_{j+1}(A,r_0)}$$

□

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

Vorteile dieser Struktur

Zum Beipspiel kann man eine QR-Zerlegung finden, in dem man Pro Spalte eome 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:*

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

$$V_m^t A V_m = H_{m,m} \tag{3.6}$$

Beweis. Gemäß Algorithmus haben wir

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

Daher gilt (3.5) (folgt aus Matrix Schreibweise).
Für (3.6):

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

□

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

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

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

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:

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

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

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 Algorihtmus 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

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

Beweis.

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

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

□

Bemerke:

- Hauptkosten (Alles mit Vektoren der Länge n)
  - Matrix-Vektor-Multiplikation (1, Mal, sehr teuer)
  - Skalarprodukte
  - Vektor-Updates

  $\Longrightarrow$ 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.

Ende von Vorlesung 07 am 03.11.2022

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?

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

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

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

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

$\implies \|r_j\|_2$ wird minimal, falls y_j als Lösung des linaren 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 Algotihmen 3.16 und 3.18 weitgehen 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 Algortihmus 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:
Sei $H_{j+1,j} = Q_j R_j = \underbrace{Q_j}_{\in\mathbb{R}^{j+1,j+1}} \begin{bmatrix} R_j \\ \underbrace{0}_{\in\mathbb{R}} \end{bmatrix}$ eine QR-Zerlegung.
Sei

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

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

$$(\star) = \left\| \begin{pmatrix} b_1 - \tilde{R}_j y_j \\ b_2 \end{pmatrix} \right\|_2^2$$

$$= \underbrace{\left\|b_1 - \tilde{R}_j y_j\right\|_2^2}_{=0 \text{ falls } H_{j+1,j} \text{ bzw. } \tilde{R}_j \text{ vollen Rang hat}} + \underbrace{\|b_2\|_2^2}_{=|b|^2}$$

$$\implies \|r_j\|_2 = |b_2|$$

Wir berechnen b_2 als Nebenprodukt beim Lösen des linearen Ausgleichsrpoblems.
Aber: In jeder Iteration eine QR-Zerlegung zu berechnen ist teuer. Wie geht es besser?
Beobachte:

- $H_{j+1,j}$ hat Hessenbergstruktur, d.h. $H_{j+1,j}$ hat Ist eine rechte obere Dreiecksmatrix, wo zusätzlich die utere Diagonale nicht notwendigerweise 0 ist. Dafür kann mit j Givensrotaionen eine QR-Zerlegung berechnet werden.
- Beim Iterationsschritt $j \implies j+1$ werden jediglich eine neue Spalte und eine neue Zeile an $H_{j+1,j}$ drangehängt um 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 iin jedem Schrit.

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, β = ‖r_0‖_2 , v_1 = r_0/β
b̂ = βe_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}, ..., G_{j−1,j} auf die letzte Spalte von H_{j+1,j} an
    Bestimme G_{j,j+1} so, dass H_{h+1,j} = G_{j,j+1} H_{j+1,j} = [R̃_{j+1}; 0]
    b̂ = G_{j,j+1} b̂
    if b̂ klein then
        Löse R̃_j y_j = [b̂_i]_{i=1,...,j}
        Gebe x_j = x_0 + V_j y_j 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 Nährungslösung 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 zusäztliche Annahmen zu unserer Matrix treffen? z.B. Symmetrie oder spd?
Beobachte: Für A symmetrisch ist

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

symmetrisch und hat Hessenberg-Struktur

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

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 Synmmetrie 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: $V_m = \begin{bmatrix} v_1 & \dots v_m \end{bmatrix}$ 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**
  $h_{ij} = 0, i = 1, \dots, j_2$
  $h_{j-1,j} = \beta_j$
  $w_j = z - \sum_{i=1}^{j-1} h_{ij} v_i = A v_j - \beta_j v_{j-1}$ ▷ In der Praxis hat w keinen Index
  $\alpha_j = \langle w_j, v_j \rangle_2 = h_{jj}$
  $w_j = w_j - \alpha_j v_j$ ▷ $w = A v_j - \alpha_j v_j - \beta_j v_{j-1}$
  $\beta_{j+1} = \|w_j\|_2$
  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: $x_m \in x_0 + K_m(A, r_0), x_m \approx A^{-1}b$

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

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

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.

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

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

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

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

Algorithm 3.26 Direkes 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**
  $W_j = Av_j - \beta_j v_{j-1}$ ▷ Wie beim Lanczos
  $\alpha_j = \langle w_j, v_j \rangle$ ▷ Wie beim Lanczos
  $\frac{\beta_j}{\eta_{j-1}}, \eta_j = \alpha_j - \lambda_j \beta_j$
  $\zeta_j = -\lambda_j \zeta_{j-1}$ ▷ : Für $j > 1$
  $p_j = \frac{1}{2j}(v_j - \beta_j p_{j-1})$
  $x_j = x_{j-1} + \zeta_j p_{j-1}$
  Konvergenztest
  $w_j = w_j - \alpha_j v_j$
  $\beta_{j+1} = \|w_j\|_2$, 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 $x_0 = x_m$

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 matemathisch äquivalent, d.h.bei exater 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 dem 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 Residuuen* 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 $\implies$ $P_m^t A P_m$ is diagonal.

□

Idee: Vereinfache Algorithmus 3.26 zu einem Verfahren der Form
$$x_{j+1} = x_j + \tilde{\alpha}_j \tilde{p}_j, \tilde{p}_{j+1} = r_{j+1} + \tilde{\beta}_j \tilde{p}_j \tag{3.8}$$
benutze dazu de Eigenschaften aus Lemma 3.28.
Wie sieht r_{j+1} aus?
$$\begin{aligned} r_{j+1} &= b - Ax_{j+1} \\ &\overset{3.8}{=} b - Ax_j - \tilde{\alpha}_j A\tilde{p}_j \\ &= r_j - \tilde{\alpha}_j A\tilde{p}_j \end{aligned} \tag{3.9}$$
Außerdem
$$\begin{aligned} 0 = \langle r_{j+1}, r_j \rangle_2 = \langle r_j - \tilde{\alpha}_j A\tilde{p}_j, r_j \rangle_2 = \langle r_j, r_j \rangle_2 - \tilde{\alpha}_j^2 \langle A\tilde{p}_j, r_j \rangle_2 \\ \implies \tilde{\alpha_j} = \frac{\langle r_j, r_j \rangle_2}{\langle A\tilde{p}_j, r_j \rangle_2} = \frac{\langle r_j, r_j \rangle}{\langle A\tilde{p}_j, \tilde{p}_j \rangle_2} \end{aligned}$$

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

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

$r_0 = b - Ax_0, \tilde{p}_0 = r_0$
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 direken 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

$$\langle Ap_j, p_i \rangle_2 = 0, i \neq j$$

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

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 $r_j = b - Ax_j \perp R_j(A, r_0)$.

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

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

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

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

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*

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

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*

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

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

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

□

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

Beweis. Wähle

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

in Lemma 3.33. □

Ziel: Finde eine ähnliche Konvergenzaussage für Eigenwerte $0 < \lambda_{\min} = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n = \lambda_{\max} < \infty$ von spd Matrizen.
Idee: Beschränke die rechte Seite von (3.12) geeignet:

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

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 $p_n(x) = x^n + \dots \in \Pi_n$ ist $\|p_n\|_{C([-1,1])}$ für $p_n = 2^{1-n} T$ 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. □

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

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

$$\|x - x_j\|_A \leq 2\left(\frac{\sqrt{(cond_2(A))} - 1}{\sqrt{cond_2(A)} + 1}\right)^j \|x - x_0\|_A$$

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:

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

Dann folgt mit (3.14) die Behauptung.

□

Bemerkung 3.39.
1. *Das Verfahren des steilsten Absties hat eine ähnliche Abschätzung aber mit $cond_2(A)$ statt $\sqrt{cond_2(A)}$.*
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*

$$Ax = \lambda x \tag{4.1}$$

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

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

$$Ax = \lambda Bx \tag{4.2}$$

*das **allgemeine Matrixeigenwetrproblem**. 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, doe (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 $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$. 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 Messewerten oder Datenpunkten $x_i \in \mathbb{R}^n, i = 1, \dots m$, welche in einer Datenmatrix angeordnet sind:*

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

Der Einfachheit halber sei

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

Die Kovarianzmatrix

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

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

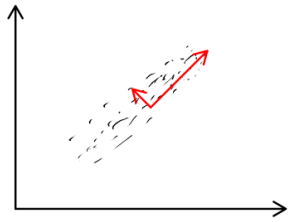

Abbildung 4.1: PCA Beispiel

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*

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

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 füren auf

$$Ax = \lambda B x$$

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 de 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 reel. Die zugehörigen EV können als ONB genutz 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*

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

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

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

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

$$\begin{aligned} Ax_1 &= \lambda x_1 \\ Ax_2 &= \lambda x_2 \\ \implies A\alpha x_1 + \beta x_2 &= \lambda(\alpha x_1 + \beta x_2) \end{aligned}$$

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 $Ax = \lambda x$:

$$\lambda x_l = \sum_{j=1}^{n} a_{l_j} x_j = a_{ll} x_l + \sum_{j=1, j\neq l}^{n} a_{l_j} x_j \tag{4.3}$$

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

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

$$\implies |\lambda a_{ii}| \leq \sum_{j=1, j\neq i}^{n} |a_{ij}| = r_i$$

$\implies$ λ liebt in einem Kres mit Radius r_i.
Aber: x ist unbekannt $\implies$ i ist unbekannt
$\implies$ λ liebt in der Vereinigun 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 deschreibt, 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* $\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \implies$ *EW sind* $\lambda_{1/2} = 0$.

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

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

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*

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

wo bei $Ax = \lambda x$ *(Rechtseigenvektor) und* $A^*y = \bar{\lambda}y$ *(Linkseigenvektor). D.h. einfache Eigenwerte hermitischer 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 Eingenpaars ist aber gutkonditioniert.

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

Berechnung der EW

$(x-1)(x-2)\cdot\cdots\cdot(x-20)$ ist ein gutes Beipsiel, 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

$$y_k^t A y_k \overset{k\to\infty}{\to} \lambda_1 \tag{4.4}$$

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

$$x_0 = \sum_{i=1}^{n} \alpha_i u_i \tag{4.5}$$

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

$$\begin{aligned}(\star) &= \sum_{i=1}^{n} \alpha_i \lambda_i^k u_i \\ &= a_1 \lambda_1^k u_1 + \sum_{i=2}^{n} \alpha_i \lambda_i^k u_i \\ &= a_1 \lambda_1^k \underbrace{\left(u_1 + \sum_{i=2}^{n} \frac{\alpha_i}{\alpha_1} \frac{\lambda_i^k}{\lambda_1^k} u_i \right)}_{= z_k \overset{k\to\infty}{\to} u_1}\end{aligned}$$

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

<u>z.z.:</u> (4.4)

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

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

□

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

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

Die Eigenschaften bleiben unverändert.

2. *Der Satz 4.12 gilt aich 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 übeprüft werden.*
4. *Die Konvergenzgeschwindigkeit hängt von $\left|\frac{\lambda_2}{\lambda_1}\right|$ ab. Falls $|\lambda_1| \approx |\lambda_2|$ konvergietr das Verfahren sehr langsam.*

Ende von Vorlesung 13 am 24.11.2022

Algorithm 4.14 Vektoriteration

Input: $A \in \mathbb{R}n \times n, x_0 \in \mathbb{R}^n$ **Output:** $\lambda_1^{(1)} \approx \lambda_1, y_k \approx \pm u_1$

$y_0 = \frac{x_0}{\|x_0\|_2}$
for $j = 0, 1, 2, \dots$ **do**
  $x_{k+1} = Ay_k$
  $y_{k+1} = \frac{x_k}{\|x_k\|_2}$
  $\lambda_1^{(k+1)} = y_{k+1}^t A y_{k+1}$
  Kovergenztest
end for

Frage: Wie können wir beliebige EW und EV berechnen?
Idee: Verwende Nährung $\tilde{\lambda}$ an λ_i, welche näher an λ_i als allen anderen EW ist.
$\rightsquigarrow$ die Vektoriteration auf

$$\left(A - \tilde{\lambda}\right)^{-1}$$

an (shift & invert). $\implies \left(A - \tilde{\lambda}\right)^{-1}$ hat $(\lambda_k - \tilde{\lambda})^{-1}, j = 1, \dots n$ als EW. $\implies$ gößter Eigenwert von $\left(A - \tilde{\lambda}\right)^{-1}$ ist $(\lambda_i - \tilde{\lambda})^{-1}$. Daraus bekommen wir λ_i.
Dieses Verfahren heißt **inverse Vektoriteration**.
Aber: Wir brauchen $\tilde{\lambda}$ und müssen in jeder Iteration ein lineares Gleichungsystem lösen.

4.5 Das QR-Verfahren

Ziel: Algorithmus zum Berechnen aller EW einer Matrix $A \in \mathbb{C}^{n \times n}$.
Idee: Formuliere Iterationsvorschrift

$$A_0 = A \tag{4.6}$$

Berechne QR Zerlegung von $A_k - \mu_k I = Q_k R_k$ für $\mu_k \in \mathbb{C}$
Setze $A_{k+1} = R_k Q_k + \mu_k I$.

Lemma 4.15. *Seien μ_k, A_k, Q_k, R_k wie in (4.6). Dann gilt*

1. $A_{k+1} = Q_k^* A_k Q_k$
2. $A_{k+1} = (Q_0 Q_1 \dots Q_k)^* A (Q_0 Q_1 \dots Q_k)$
3. $$\prod_{j=0}^{k} (A - \mu_k I) = \underbrace{(Q_0 Q_1 \dots Q_k)}_{\underline{Q_k}} \underbrace{(R_k R_{k-1} \dots R_0)}_{\underline{R_k}}$$

Beweis. Übung :) □

Bemerkung 4.16. *Lemma 4.15 sagt, dass alle A_k die gleichen EW haben.*

Wie kommen wir auf (4.6)? Betrachte $\mu_k = 0$

Beobachtung 1

$$\underbrace{A^{k+1}e_1}_{\text{Vektoriteration!}} \overset{\text{Lemma 4.15}}{=} \underline{Q_k R_k} e_1$$

$$= \begin{bmatrix} & & & \\ & q_1^{(k)} & \dots & q_n^{(k)} \\ & & & \end{bmatrix} \begin{bmatrix} r_{11}^{(k)} & \dots & r_{1n}^{(k)} \\ & \ddots & \vdots \\ 0 & & r_{nn}^{(k)} \end{bmatrix} e_1$$

$$= r_{11}^{(k)} q_1^{(k)}$$

$\implies q_1^{(k)}$ ist approximativer Eigenvekor zum betragsmäßig größten EW λ_1.

$$\implies A_{k+1}e_1 \overset{\text{Lemma 4.15}}{=} \underline{Q_k^*} \underbrace{A \underbrace{\underline{Q_k} e_1}_{q_1^{(k)}}}_{\approx \lambda_1 q_1^{(k)}}$$

$$\approx \lambda_1 \underline{Q_k}^* q_1^{(k)} = \lambda_1 e_1$$

$$\implies A_{k+1} = \begin{bmatrix} \lambda_1 & \star & \dots & \star \\ 0 & \star & \dots & \star \\ \vdots & \vdots & & \vdots \\ 0 & \star & \dots & \star \end{bmatrix}$$

Beobachtung 2

$$\left(q_n^{(k)}\right)^* = e_n^t \underline{Q_k}^*$$

$$= \underbrace{e_n^t \underline{R_k}}_{r_n n^{(k)} e_n^t} A^{-(k+1)} = r_{nn}^{(k)} e_n^t A^{-(k+1)}$$

$q_n^{(k)}$ ist ein appriximativer Eigenvektor zum betragsmäßig kleinsten EW λ_n von A.

$$\implies e_n^t A_{k+1} \overset{\text{Lemma 4.15}}{=} \underbrace{e_n^t \underline{Q_k}^*}_{=\left(q_n^{(k)}\right)^*} A \underline{Q_k}$$

$$\approx \lambda_n \left(q_n^{(k)}\right)^* \underline{Q_k}$$

$$= \lambda_n e_n^t$$

$$A_{k+1} = \begin{bmatrix} \lambda_1 & \star & \dots & \star \\ 0 & \star & \dots & \star \\ \vdots & \vdots & & \vdots \\ 0 & \star & \dots & \star \\ 0 & 0 & \dots & \lambda_n \end{bmatrix}$$

Satz 4.17. *Sei $A \in \mathbb{C}^{n\times n}$ mit paarweise verschiedenen Eigenwerten $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n| > 0$.*
$\Lambda = diag(\lambda_1, \ldots, \lambda_n)$ und $X = \begin{bmatrix} x_1 & \ldots & x_n \end{bmatrix}$
die zugehörige EV Matrix ($A = X\Lambda X^{-1}$). Existiert die LR-Zerlegung $X^{-1} = LR$, dann sind die Matrizen A_k aus (4.6) mit $\mu_k = 0$ asymptoisch obere Dreiecksmatrizen. Außerdem konvergiert $diag(diag(A_k))$ mindestens linear gegen Λ

Idee: Leite zwei erschiedene QR-Zerlegungen von A^k mit verschiedenen Eigenschaften her.
QR-Zerlegung 1: Lemma 4.15.3 $\implies A^k = \underline{Q_{k-1}R_{k-1}}$.
QR-Zerlegung 2:

$$\begin{aligned} A^k &= X\Lambda^k \underbrace{X^{-1}}_{LR} \\ &= X\Lambda^k LR \\ &= \underbrace{(X\Lambda^k L\Lambda^{-k})}_{X_k \text{ mit QR-Zerlegung } X_k = P_k U_k} (\Lambda^k R) \\ &= \underbrace{P_k}_{\underline{P_k}} \underbrace{(U_k \Lambda^k R)}_{\underline{U_k}} \end{aligned}$$

Ende von Vorlesung 14 am 29.11.2022

Beweis. **Bemerke:**

$$\Lambda^k L \Lambda^{-k} = \begin{bmatrix} \lambda_1^k & & \\ & \ddots & \\ & & \lambda_n^k \end{bmatrix} \begin{bmatrix} 1 & & \\ \star & \ddots & \\ \star & \star & 1 \end{bmatrix} \begin{bmatrix} \lambda_1^{-k} & & \\ & \ddots & \\ & & \lambda_n^{-k} \end{bmatrix}$$

$$\implies \left(\Lambda^k L \Lambda^{-k}\right)_{ij} = \begin{cases} 0 & j > 1 \\ 1 & i = j \\ I_{ij} \left(\frac{\lambda_i}{\lambda_j}\right)^k & i > j \end{cases}$$

$$\implies X_k = X(I + E_k), \|E_k\|_2 = O(q^k)$$

$q \in (0,1)$ hängt von $\frac{\lambda_i}{\lambda_j}, i > j$ ab.
Eindeutigkeit der QR-Zerlegung bis auf unitäre Skalierungen (Übung)

$$\implies \underline{Q_{k-1}} = \underline{P_k} S_k^*, \underline{R_{k-1}} = \underline{U_k} S_k$$

und S_k unitäre Diagonalmatrix (d.h. die diagonaleinträge haben Betrag 1).

$$\begin{aligned} \implies A_k &\overset{4.15.2}{=} \underline{Q_{k-1}}^* A \underline{Q_{k-1}} \\ &= S_k \underline{P_k}^* A \underline{P_k} S_k^* \\ &= S_k U_k \underbrace{U_k^{-1} \underline{P_k}^*}_{=X_k^{-1}} \underbrace{A \underline{P_k}}_{=A^{k+1}\underline{U_k}^{-1} = \underbrace{X\Lambda^{k+1}L\Lambda^{-(k+1)}}_{=X_{k+1}} \Lambda^{(k+1)} R} S_k^* (\star) \underbrace{U_k^{-1}}_{R^{-1}\Lambda^{-k}U_k^{-1}} \\ &= S_k U_k \underbrace{X_k^{-1} X_{k+1}}_{=(I+E_k)^{-1}(I+E_k) = I + F_k, \|F_k\|_2 = O(q^k)} \Lambda U_k^{-1} S_k^* \\ &= \underbrace{S_k U_k \Lambda U_k^{-1} S_k^*}_{\text{obere Dreiecksmatrix}} + \underbrace{S_k U_k F_k \Lambda U_k^{-1} S_k^*}_{(+)} \end{aligned}$$

Mit Nebenrechnung

$$\|U_k\|_2 = \|P_k^* X_k\|_2 = \|X_k\|_2$$

$$\begin{aligned}\|(+)\|_2 &\leq \underbrace{\|S_k\|_2}_{=1} \|U_k\|_2 \|F_k\|_2 \left\|U_k^{-1}\right\|_2 \underbrace{\|S_k\|_2}_{=1} \underbrace{\|\Lambda\|_2}_{=|\lambda_1|} \\ &\leq \underbrace{\mathrm{cond}_2(X_k)}_{\leq 2\mathrm{cond}_2(X)} |\lambda_1| \|F_k\|_2 = O(q^k)\end{aligned}$$

$\implies$ A_k konvergiert gegen eine obere Dreiecksamtrix.
z.z. $\mathrm{diag}(\mathrm{diag}(A_k)) \overset{k\to\infty}{\to} \Lambda$

$$\begin{aligned}X_k \overset{k\to\infty}{\to} X &\implies P_k, U_k \overset{k\to\infty}{\to} I \\ &\implies \mathrm{diag}(\mathrm{diag}(A_k)) = \mathrm{diag}(\mathrm{diag}(S_k U_k \Lambda U_k^{-1} S_k^*)) \\ &= S_k \mathrm{diag}(\mathrm{diag}(U_k \Lambda U_k^{-1})) S_k^* \overset{k\to\infty}{\to} \Lambda\end{aligned}$$

□

Bemerkung 4.18. *1. Der Fehler im k-ten Schritt hängt von* $\left|\frac{\lambda_i}{\lambda_k}\right|^k, i > j$ *ab. D.h. falls* $|\lambda_i| \approx |\lambda_j|$*, dann konvergiert das Verfahren sehr langsam.*

2. Wir müssen pro Iteration eine QR-Zerlegung mit Aufwand $O(n^3)$ *berechnen.*

4.6 Implementierung des QR-Verfahrens

Problem 1: Der Aufwand pro Iteration ist zu hoch.
Idee: Transformiere $A \in \mathbb{C}^{n\times n}$ zuerst mittels Ähnlichkeitstransformationen auf Hessenberg-Form.

$$\begin{bmatrix}\star&\star&\star&\star\\ \star&\star&\star&\star\\ \star&\star&\star&\star\\ \star&\star&\star&\star\end{bmatrix} \overset{P_1\cdot}{\rightsquigarrow} \begin{bmatrix}\star&\star&\star&\star\\ \star&\star&\star&\star\\ 0&\star&\star&\star\\ 0&\star&\star&\star\end{bmatrix} \overset{\cdot P_1}{\rightsquigarrow} \begin{bmatrix}\star&\star&\star&\star\\ \star&\star&\star&\star\\ 0&\star&\star&\star\\ 0&\star&\star&\star\end{bmatrix} \overset{P_2\cdot}{\rightsquigarrow} \begin{bmatrix}\star&\star&\star&\star\\ \star&\star&\star&\star\\ 0&\star&\star&\star\\ 0&0&\star&\star\end{bmatrix} \overset{P_2\cdot}{\rightsquigarrow} \dots$$

$\implies$ mit geeigneten orthogonalen Ähnlichkeitstransformationen $P_1, \dots, P_{n-1}$ hat
$A = (P_{n-2} \cdot \dots \cdot P_1) A (P_{n-2} \cdot \dots \cdot P_1)^*$ hat obere Hessenberg-Form.
Gesamtaufwand:

$$\sum_{k=2}^{n-1} 2((k+1)k) + nk \leq 3n^3$$

Vorteil: Die QR-Zerlegung einer Hessenberg-Matrix kann in $(2n^2)$ Multiplikationen statt $O(n^3)$ berechnet werden.
$\rightsquigarrow \underbrace{(G_{n-1,n} \dots G_{2,3} G_{1,2})}_{=Q_0^*}(A_0 - \mu_0 I) = R_0$
Aber: hat $R_0 Q_0 + \mu_k I$ wieder Hessenberg-Form?

$$\begin{bmatrix}\star&\star&\star&\star\\ &\star&\star&\star\\ &&\star&\star\\ &&&\star\end{bmatrix} \overset{\cdot G_{1,2}^*}{\to} \begin{bmatrix}\star&\star&\star&\star\\ \star&\star&\star&\star\\ &&\star&\star\\ &&&\star\end{bmatrix} \overset{\cdot G_{1,2}^*}{\to} \begin{bmatrix}\star&\star&\star&\star\\ \star&\star&\star&\star\\ &\star&\star&\star\\ &&&\star\end{bmatrix}$$

$\implies$ Ja! Gesamtaufwand pro Iteration: $4n^2$.
Problem 2 Die Konvergenz des QR-Verfahrens ohne Shift, d.h. $\mu_k = 0$, kann sehr langsam sein.
Frage: Wie müssen wir μ_k in (4.6) wählen?

Korollar 4.19. *Unter den gleichen Voraussetzungen wie in 4.17, aber mit $\mu_k \neq 0$, hängt der Fehler des QR-Verfahrens im kten Schritt von*

$$\left|\frac{\lambda_i - \mu_0}{\lambda_j - \mu_0}\right| \cdot \ldots \cdot \left|\frac{\lambda_i - \mu_{k-1}}{\lambda_j - \mu_{k-1}}\right|, i > j$$

ab.

Beweis. Analog zu Satz 4.17. □

Idee 1: Wähle $\mu_k = a_{nn}^{(k)}$
Idee 2: Wähle μ_k als EW von $\begin{bmatrix} a_{n-1n-1}^{(k)} & a_{n-1n}^{(k)} \\ a_{nn-1}^{(k)} & a_{nn}^{(k)} \end{bmatrix}$ der näher an $a_{nn}^{(k)}$ liegt.
In beiden Fällen kann man zeigen, dass nach wenigen Schritten

$$A_k \approx \begin{bmatrix} \star & \ldots & \star & \star \\ \star & \ddots & & \vdots \\ & \ddots & \star & \vdots \\ & & \star & \star \\ 0 & \ldots & 0 & \lambda_n \end{bmatrix}$$

Diesen Verfahren heißt Deflation.y
In der Praxis benötigt das Verfahren nur $O(n)$ Iterationen. $\implies$ Gesamtaufwand $P(n^3)$.
Problem 3: Wie berechnen wir Eigenvektoren?
$A = Q^*RQ$ betrachte R (siehe Notizen)
$A = PA_0P^*$ Die obere Hessenbergmarix $A_0 = P^*AP$ hat die gleichen EW wie A.
$\rightsquigarrow$ Berechne die EV von A_0 mittels inverser Vektoriteration, Sift-Parameter Eigenwetrnährung aus dem QR-Verfahren.
Benutze QR-zerlegung um LGS zu lösen. $\rightsquigarrow O(n^2)$ prob EV, da die inverse Vektoriteration meistens in einem Schritt konvergiert.
$\implies$ Zusatzaufwand $O(n^3)$ um EV zu berechnen.
Ende von Vorlesung 15 am 01.12.2022

4.7 Singulärwertzerlegung

Definition 4.20. *Sei $A \in \mathbb{C}^{m\times n}, m \geq n$. Eine **Singulärwertzerlegung von** A (SVD) ist eine Faktorisierung*

$$A = U\Sigma V^*$$

mit unitären Matrizen $U = \begin{bmatrix} u_1 & \ldots & v_m \end{bmatrix} \in \mathbb{C}^{m\times m}, V = \begin{bmatrix} v_1 & \ldots & v_n \in \mathbb{C}^{n\times n} \end{bmatrix}$ mit

$$\Sigma = \begin{bmatrix} \hat{\sigma} = diag(\sigma_1, \ldots, \sigma_n) \in \mathbb{R}^{n\times n} \\ 0 \end{bmatrix}$$

mit $\sigma_i \geq 0$. Die u_i heißen linke Singulärvektoren, die v_i rechte Singulärvektoren und die σ_i Singulärwerte.

Satz 4.21. *Für jede Matrix $A \in \mathbb{C}^{m\times n}, m \geq n$ existiert eine SVD.*

Beweis. **Idee:** Zeige, dass es unitäre Matrizen U, V gibt, mit:

$$U^*AV = \begin{bmatrix} \sigma & 0 \\ 0 & B \end{bmatrix}, 0 \leq \sigma \in \mathbb{R} \tag{4.7}$$

Der Rest folgt dann durch Induktion.

Sei

$$\sigma = \|A\|_2 = \max_{x\in\mathbb{C}^n, \|x\|_2=1} \|Ax\|_2 \geq 0$$

Das Maximum wird angenommen $\implies \exists x \in \mathbb{C}^n, \|x\|_2 = 1$ mit

$$\sigma^2 = \|Ax\|_2^2 = x^* A^* A x = (\star)$$

Setze

$$\begin{aligned} u = \frac{1}{\sigma} Ax &\overset{(\star)}{\implies} \|u\|_2 = 1 \\ v = x &\implies \|v\|_2 = 1 \\ &\implies Av = \sigma u = (\star\star) \end{aligned}$$

Ergänze zur ONB:

$$U = \begin{bmatrix} u_1 & \dots & u_m \end{bmatrix}$$

$$V = \begin{bmatrix} v_1 & \dots & v_m \end{bmatrix}$$

$$\implies U^* A V = \begin{bmatrix} u^* \\ u_2^* \\ \vdots \\ u_m^* \end{bmatrix} A \begin{bmatrix} v & v_2 & \dots & v_n \end{bmatrix}$$

$$= \begin{bmatrix} \sigma & w^* \\ 0 & \\ \vdots & B \\ 0 & \end{bmatrix} = A_1, w \in \mathbb{C}^{n-1}$$

$$\begin{aligned} \sigma^2 &= \|A\|_2^2 \\ &\overset{\text{Invarianz unter oth. Proj.}}{=} \|A_1\|_2^2 \\ &= \sup_{0\neq x\in\mathbb{C}^n} \frac{\|A_1 x\|_2^2}{\|x\|_2^2} \end{aligned}$$

$$\begin{aligned} &\overset{x=\begin{pmatrix}\sigma\\ w\end{pmatrix}}{\geq} \frac{1}{\sigma^2 + \|w\|_2^2} \left\| \begin{bmatrix} \sigma & w^* \\ 0 & \\ \vdots & B \\ 0 & \end{bmatrix} \begin{pmatrix} \sigma \\ w \end{pmatrix} \right\| \\ &= \frac{1}{\sigma^2 + \|w\|_2^2} \left\| \begin{pmatrix} \sigma^2 + \|w\|_2^2 \\ Bw \end{pmatrix} \right\| \\ &\geq \sigma^2 + \|w\|_2^2 \implies w = 0 \implies (4.7) \end{aligned}$$

□

Korollar 4.22. *Sei $U^+ A V = \Sigma$ eine SVD von $A \in \mathbb{C}^{m\times n}$. Dann gilt:*

1. *$Av_i = \sigma_i u_i$ und $A^* u_i = \sigma_i v_i$*
2. *Falls $\sigma_1 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_n = 0$ gilt*

   (a) *Rang*$(A) = r$
   (b) *Kern*$(A) = span(v_{r+1}, \dots, v_n)$
   (c) *Bild*$(A) = span(u_1, \dots, u_r)$

3. $\|A\|_2 = \sigma_1$
4. $\|A\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}^2|} = \sqrt{\sum i = 1^n \sigma_i^2}$
5. $cond_2(A) = \frac{\sigma_1}{\sigma_n}$
6. *Für* $\lambda_i = \sigma_i^2$ *gilt* λ_i *sind EW von* AA^* *und* A^*A *zu den Eigenvektoren* $u_1, \dots, u_n$ *und* $v_1, \dots, v_n$

Beweis. Übungen. □

Wozu brauchen wir eine SVD?
z.B. als theoretisches Hilfmittel:

Lemma 4.23. *Sei* $A \in \mathbb{C}^{m \times n}, m \geq n$ *mit* $rang(A) = r$. *Sei* $U\Sigma V^*$ *eine SVD von* A. *Dann ist*

$$A^\dagger = V\Sigma^\dagger U^*, \Sigma^\dagger = \begin{bmatrix} \sigma_1^{-1} & & & & & \\ & \ddots & & & & \\ & & \sigma_r^{-1} & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{bmatrix}$$

Beweis. Übungen. □

Bemerkung 4.24. *Die charakterisierung von Lemma 4.23 bietet eine einfache Möglichkeit Satz 2.23 zu beweisen.*

z.B. als Hilfsmittel zur Datenkompression:

Satz 4.25 (Eckhart-Young)**.** *Sei* $A \in \mathbb{C}^{m \times n}, m \geq n$, *und* $k < rang(A) = r$. *Sei* $A = U\Sigma V^*$ *eine SVD und*

$$A_k = \sum_{j=1}^{k} \sigma_k u_j v_j^*$$

Dann ist

$$\min_{rang(B)=k} \|A - B\|_2 \overset{1.}{=} \|A - A_k\|_2 \overset{2.}{=} \sigma_{k+1}$$

Beweis. $\operatorname{rang}(A_k) = k$ ist klar.

$$\|A - A_k\|_2 = \left\| U \left(\begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ & 0 & \end{pmatrix} - \begin{pmatrix} \sigma_1 & & & & \\ & \ddots & & & \\ & & \sigma_k & & \\ & & & 0 & \\ & & & & \ddots \\ & & 0 & & \end{pmatrix} \right) \right\|_2^2 = \sigma_{k+1} \implies 2.$$

z.z. 1.:
Sei

$$\begin{aligned} & B \in \mathbb{C}^{m\times n}, \operatorname{rang}(B) = k \\ \implies & \dim(\ker(B)) = n-k \\ \implies & \underbrace{\ker(B) \cap \operatorname{span}(v_1, \dots v_{k+1})}_{Z} \neq \{0\} \end{aligned}$$

Sei $z \in Z, z \neq 0, \|z\|_2 = 1$

$$\begin{aligned} \implies Bz &= 0 \\ \implies Az &= U\Sigma V^* z \\ &= \begin{bmatrix} u_1 & \dots & u_m \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ & 0 & \end{bmatrix} \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} z \\ &= \sum_{j=1}^{k+1} \sigma_i (v_i^* z) u_i \end{aligned}$$

$$\begin{aligned} \|A-B\|_2^2 &= \max_{\|x\|_2=1} \|(A-B)x\|_2^2 \\ &\overset{x=z}{\geq} \|(A-B)z\|_2^2 \\ &= \|Az\|_2^2 \\ &= z^* A^* A z \\ &= \sum_{j=1}^{k+1} \sigma_j^2 (V_j^* z)^2 \\ \geq \sigma_{k+1}^2 \underbrace{\sum_{j=1}^{k+1} (V_j^* z)^2}_{\overset{\text{Satz1.16}}{=} \|z\|_2^2 = 1} &\geq \sigma_{k+1}^2 \end{aligned}$$

□

Bemerkung 4.26. *1. Satz 4.25 ist eine Bestapüroximationsaussage zur Approximation von belibiegen Matrizen durch Matrizen von niedriegerem Rang.*

2. Die Aussage gilt ähnlich auch für die Frobeniusnorm.

Wie berechnen wir eine SVD?

Möglichkeit 1.

Berechne EW $\lambda_i = \sigma_i^2$ und zugeh EV von A^*A oder AA^*. Die Matrizen sind hermitisch, d.h. die Berechnung der EW & EV-Berecnung gutgestellt/ gutkonditioniert **falls** AA^*, A^*A nicht berechnet werden müssen.

Beispiel 4.27. *Rundungsgenauigkeit* 4 *Ziffern.*

$$A = A^t = \begin{bmatrix} 1.005 & 0.995 \\ 0.995 & 1.005 \end{bmatrix} \implies \sigma_1 = 2, \sigma_2 = 0.01$$

Aber:

$$\begin{aligned} round(A^*A) &= \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} \\ \implies \lambda_1 = 4, \lambda_2 &= 0 \\ \implies \sigma_1 = \sqrt{(\lambda_1)}, \sigma_2 &= \sqrt{\lambda_2} \end{aligned}$$

Ende von Vorlesung 16 am 06.12.2022

Möglichkeit 2: SVD-Algorithmus

4.8 SVD-Algorithmus

Idee: Führe QR-Algorithmus auf A^*A durch, ohne A^*A zu berechnen.
Beobachtung: Sei $\mathbb{C}^{m\times n} \ni A = U\Sigma V^*$ eine SVD. Seien $\tilde{U} \in \mathbb{C}^{m\times m}, \tilde{V} \in \mathbb{C}^{n\times n}$ unitäre Matrizen.

$$\implies \tilde{A} = \tilde{U}A\tilde{V}^* = \underbrace{\tilde{U}U}_{\hat{U}}\Sigma\underbrace{V^*\tilde{V}^*}_{V^*}$$

ist eine SVD von $\tilde{A}$ und $\hat{U}, \hat{V}$ sind unitär.
Idee: Analog zum QR-Verfahren:
Schritt 1: Transformieren $A \in \mathbb{C}^{m\times n}$ mittels unitärer Ähnlichkeitstransformationen auf obere Dreiecksgestalt $B \in \mathbb{C}^{m\times n}$

$$B = \begin{bmatrix} \text{diagonal Matirix mit einer Diagonalen über ihr besetzt} \\ 0 \end{bmatrix}$$

$\implies A, B$ haben die gleichen Singulärwerte
$\implies B^*B$ ist hermitisch und tridiagonal, hat also Hessenbergform.
Schritt 2: Wende modifizierten QR-Algorithmus an (auf B^*B, ohne dieses Produkt zu berechnen).
Zu Stufe 1:

Lemma 4.28. *Zur Matrix $A \in \mathbb{C}^{m\times n}, m \geq n$, existieren unitäre Matrizen $P \in \mathbb{C}^{m\times m}, Q \in \mathbb{C}^{n\times n}$ s.d.*

$$B = PAQ = \begin{bmatrix} C \\ 0 \end{bmatrix}$$

mit $C \in \mathbb{C}^{n\times n}$ eine Bidiagonalform.

Beweis. z.B. mit Householder-Transformationen:

$$\begin{bmatrix} \star & \dots & \star \\ \vdots & \ddots & \vdots \\ \star & \dots & \star \end{bmatrix} \overset{P_1\cdot}{=} \begin{bmatrix} \star & \star & \dots & \star \\ 0 & \star & \dots & \star \\ 0 & \star & \dots & \star \\ 0 & \vdots & \ddots & \vdots \\ 0 & \star & \dots & \star \end{bmatrix} \overset{\cdot Q_1}{=} \begin{bmatrix} \star & \star & 0 & \dots & 0 \\ 0 & \star & \star & \dots & \star \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & \star & \dots & \dots & \star \end{bmatrix} = \dots = B = PAQ$$

□

Zu Stufe 2. Bemerke A^*A, B^*B, C^*C haben die gleichen EW. D.h. A, B, C haben die gleichen Singulärwerte.
C ist am "einfachsten" $\implies$ Modifiziere QR-Algorithmus für C^*C
Beobachte:

$$C^*C = \begin{bmatrix} c_{11} & & & \\ c_{12} & \ddots & & \\ & \ddots & \ddots & \\ & & c_{n-1,n} & c_{nn} \end{bmatrix} \begin{bmatrix} c_{11} & c_{21} & & \\ & \ddots & \ddots & \\ & & \ddots & c_{n-1,n} \\ & & & c_{nn} \end{bmatrix} = \begin{bmatrix} |c_{11}|^2 \end{bmatrix}$$

$$G_{12}C^*C = \begin{bmatrix} \star & \star & \star & \\ 0 & \ddots & \ddots & \\ & & \star & \end{bmatrix}$$

$$\underbrace{\tilde{Q}G_{12}}_{Q^*} C^*C = R = \begin{bmatrix} \star & \dots & \star \\ & \ddots & \vdots \\ & & \star \end{bmatrix}$$

für passendes $\tilde{Q}$.

$$RQ = \underbrace{Q^*C^*C}\, Q \tag{4.8}$$

Ausserdem: G_{12} hängt nur von $\begin{bmatrix} |c_{11}|^2 \\ c_{11}\overline{c_{12}} \end{bmatrix}$ ab:

$$\begin{bmatrix} \star \\ 0 \end{bmatrix} = G_{12} \begin{bmatrix} \star \\ \star \end{bmatrix}$$

d.h. G_{12} ist unabhängig von der Skalierung dieses Vektors.

$\implies$ G_{12} hängt nur von $\begin{bmatrix} \overline{c_{11}} \\ \overline{c_{12}} \end{bmatrix}$ ab.

Frage: Wie berechnen wir $\tilde{Q}$?

Satz 4.29 (Implizietes Q Theorem)**.** *Sei $\in \mathbb{C}^{n\times n}$, $Q \in \mathbb{C}^{n\times m}$ mit orthonormalen Spalten so, dass $H = Q^*AQ$ obere Hessenberg-Form hat. Falls die untere Nebendiagonaleinträge alle nicht Null sind, dann sind diese Matrizen Q und H (bis auf Skalierung der Spalten von Q mit $\zeta_i \in \mathbb{C}, |\zeta_i| = 1$) eindeutig durch die erste oder die letzte Spalte von Q bestimmt.*

Beweis. Ähnlich zur QR-zerlegung, siehe z.B. Golub & van Loan. □

Das bedeutet: Die erste Spalte von Q hängt nur von G_{12} ab. D.h. de erste Spalte von Q hängt nur von C ab.
Beobachte: C ist eine obere Bidiagonalmatrix. Wir wenden dann G_{12} von rechts an und eleminieren den ersten Eintrag der oberen Nebendiagonale (bekommen aber ggf. einen Eintrag in der ersten Spalte unter dem ersten Eintrag). Mit einer weiteren Givenrotation U_2^* können wir dann diesen Eintrag elemenieren, bekommen aber einen neuen Eintrag usw. Am Schluss sollten wir dann eine Bidiagonalform $\tilde{C}$ haben.
Im englischen sagt man "chasing down the diagonal".

$$\tilde{C} = \underbrace{(U_n^* \cdot \dots \cdot U_2^*)}_{\tilde{U}} C \underbrace{(G_{12}^* V_3 \cdot \dots \cdot V_n)}_{\tilde{V}}$$

ist bidiagonal und $\tilde{U}, \tilde{V}$ unitär.
$\implies$ $\tilde{C}^*\tilde{C} = \tilde{V}^*C^*\tilde{U}^*\tilde{U}C\tilde{V} = \tilde{V}^*C^*C\tilde{V}$ ist tridiagonal, d.h. hat auch obere Hessenberg-Form. Die erste Spalte von $\tilde{V}$ ist durch G_{12}, also C, bestimmt.
$\implies$ $RQ \overset{(4.8)}{=} Q^*C^*CQ \overset{\text{Satz } 4.29}{=} \tilde{V}^*C^*C\tilde{V}\tilde{C}^*\tilde{C}$ unter der Annahme, dass die untere Nebendiagonale von C^*C tatsächlich nur nicht-Null-Einträge hat
und $Q = \tilde{V}\text{diag}(\zeta_1, \dots \zeta_n), |\zeta_i| = 1$.

Algorithm 4.30 SVD-Algorithmus

Input: $A \in \mathbb{C}^{m\times n}$

Output: Singulärwerte von A

Stufe 1: Transformiere A wie im Beweis von Satz **??** auf Bidiagonalgestalt

$$PAQ = B = \begin{bmatrix} C \\ 0 \end{bmatrix} \text{ C bidiagonal}$$

Stufe 2: $C_0 = C$

for $k = 0, \dots$ **do**

"chasing down the diagonal": Berechne

$$C_{k+1}(U_n^* \cdot \dots \cdot U_2^*)C(G_{12}V_3 \cdot \dots \cdot V_n)$$

Konvergenzkriterium

end for

Bemerkung 4.31. *1. Konvergenzkriterien, Shifts, etc. gilt alles analog zum QR-Algorithmus.*

2. Aufwand (für $m = n$, pro Iteration):

Stufe 1: $O(n^3)$

Stufe 2: $O(n)$

Ende von Vorlesung 17 am 08.12.2022

4.9 Große, dünnbesetzte lineare Eigenwertprobleme

Projektionsmethoden!

Erinnerung: Projektionsmethode zum Lösen von $Ax = b$, $A \in \mathbb{R}^{n\times n}, x, b \in \mathbb{R}^n$:

$$\text{Finde } \tilde{x} \in K \text{ so, dass } b - A\tilde{x} \perp K, \tag{4.9}$$

wobei $K \subset \mathbb{R}^n$ ein endlichdimensionaler Unterraum sei.

Idee:

Setzte $b = \lambda x$, d.h. für $A \in \mathbb{C}^{n\times n}$:

Finde $(\tilde{\lambda}, \tilde{u}) \in \mathbb{C} \times \mathbb{C}^n$ so, dass $A\tilde{u} - \tilde{\lambda}\tilde{u} \perp K$, wobei $K \in \mathbb{C}^n$ ein endlichdimensionaler Unterraum sei.

Sei $V = [v_1, \dots, v_m]$, $m < n$, eine ONB von K.

$\implies \tilde{u} \in K \iff \tilde{u} = Vy, y \in \mathbb{C}^m$

Außerdem:

$$\begin{aligned}
(4.9) &\iff AVy - \tilde{\lambda}Vy \perp K \\
&\iff \langle AVy - \tilde{\lambda}Vy, v_i\rangle_2 = 0, i = 1, \dots, m \\
&\iff V^*AV^*y - \tilde{\lambda}\underbrace{V^*V}_{=I}y = 0 \\
&\iff V^*AVy = \tilde{\lambda}y
\end{aligned} \tag{4.10}$$

Das ist ein EW-Problem der Größe $m \times m$ und kannn für m nicht zu groß z.B. mit dem QR-Verfahren gelöst werden.

Algorithm 4.32 Rayleigh-Ritz-Verfahren

Input: $A \in \mathbb{C}^{n\times n}$, Unterraum $K \in \mathbb{C}^n$, m-dimensional
Output: m approximierte Eigenpaare $(\tilde{\lambda}_i, \tilde{u}_i)_i, i = 1, \dots, m$

Berechne OBN von $K \rightsquigarrow V$
Berechne Matrix V^*AV
Berechne EP $(\tilde{\lambda}_i, \tilde{y}_i)\ i = 1, \dots, m$ von V^*AV
Berechne $(\tilde{\lambda}_i, \tilde{u}_i) = \tilde{\lambda}_i, \tilde{Vy}_i, i = 1, \dots, m$

Frage: Wie genau sind die Approximationen der EP?

Lemma 4.33. *Sei K invariant unter A, d.h. $AK \subset K$. Dann sind die mit dem Rayleigh-Ritz-Verfahren berechneten EP exakt.*

Beweis.

$$\tilde{\lambda}\tilde{u} = \tilde{\lambda}Vy = V\tilde{\lambda}y \overset{4.10}{=} VV^* \underbrace{A \underbrace{Vy}_{\tilde{u}}}_{A\tilde{u}\in K}$$

$$\begin{aligned} A\tilde{u} \in K \implies A\tilde{u} &= \sum_{i=1}^{m} \alpha_i v_i, \alpha_i \in \mathbb{C} \\ \implies VV^*A\tilde{u} &= VV^* \sum_{i=1}^{m} \alpha_i v_i \\ &= \sum_{i=1}^{m} \alpha_i \underbrace{V \underbrace{V^* v_i}_{e_i}}_{v_i} \\ &= A\tilde{u} \end{aligned}$$

□

Bemerkung 4.34. *Es können auch allgemeinere Projektionsmethoden*
Fine $(\tilde{\lambda}, \tilde{u}) \in \mathbb{C} \times K$ so, dass

$$A\tilde{u} - \tilde{\lambda}\tilde{u} \perp L$$

für endlichdimensionale Unterräume K, L verwendet werden.
Solche Methoden können relativ allgemein analysiert werden, bringen aber bis auf gewisse Spezialfälle wenig konkrete Aussagen.

In der Praxis werden eigentlich fast immer Krylovraumverfahren verwendet.
Idee: Das Arnoldi-Verfahren erzeugt iterativ eine ONB $v_1, \dots, v_m$ von $K_(A, v)$, $m = 1, \dots,$. Während der Iteration wid

$$V_m^* A V_m = H_{m,m}, V = [v_1, \dots, v_m] \text{ bexrechnet.}$$

$H_{m,m}$ hat obere Hessenberg-Form.

Algorithm 4.35 Arnoldi-Verfahren für lineare Eigenwertprobleme

Input: $A \in \mathbb{C}^{n\times n}, 0 \neq v \in \mathbb{C}^n, m \in N$
Output: Approximative Eigenpaare $(\tilde{\lambda_i}, \tilde{u_i})_i, , i = 1, \dots, m$

$v_1 = \frac{v}{\|v\|_2}$
for $i = 1, 2, \dots, m$ **do**
  Bestimme $V_j, H_{j,j}$ mittels Arnoldi-Verfahren
  (Stop bei Abbruch)
  Löse $H_{j,j}y_j = \tilde{\lambda} y_j$ mittels QR-Algoritmus
  $\rightsquigarrow$ EP $(\tilde{\lambda_i}, \tilde{y_{ji}}), i = 1, \dots, j$
  EP von A sind approximativ $(\tilde{\lambda_i}, V\tilde{y_i}), i = 1, \dots, j$
  Konvergenztest
end for

Bemerkung 4.36. *1. $H_{j,j} = V_j^* A V_j$ gemäß Lemma 3.14*

2. $H_{j,j}$ hat obere Hessenberg-Form, wir können also direkt mit der zweite Stufe des QR-Algorithmus anfangen.

3. Die Größe des Eigenwertproblems wächst mit jedem Iterationsschritt, ebenso der Speicherbedarf.

4. Lemma 4.33 gilt weiterhin. Das bedeutet: Falls das Arnoldi-Verfahren abbricht, dann ist der Krylovraum invariant unter A (Lemma 3.15) und die bisher berechnten EP sind exakt.

Frage: Wie sieht ein geeigneter Konvergenztest aus?

Lemma 4.37. *Sei $(\tilde{\lambda_i}^{(m)}, y_i^{(m)})$ ein EP von $H_{m,m}$ und $\tilde{u_i}^{(m)} = V y_i^{(m)}$. Dann gilt:*

$$(A - \tilde{\lambda_i}^{(m)} I) u_i^{(m)} = h_{m+1,m} e_m^* y_i^{(m)} v_{m+1}$$

und

$$\left\| (A - \tilde{\lambda_i}^{(m)} I) u_i^{(m)} \right\|_2 = h_{m+1,m} \left| e_m^* y_i^{(m)} \right|$$

Bemerkung 4.38. *Dieses Lemma ist das Analogon zu Lemma 3.17, mit $x_m = \tilde{u_i}^{(m)}$, $b = \tilde{\lambda_i}\tilde{u_i}^{(m)}$.*

Beweis. Analog zu Lemma 3.17. □

Frage: Wie viele Schritte brauchen wir bis zur Konvergenz? Wie gut sind unsere Approxumationen?

Bemerkung 4.39. *1. Eine Konvergenanalyse des Arnoldi-Verfahrens für den betragsmäßig größten EW ist schwierig, und für alle anderen EW noch schwieriger. In der Praxis werden die größten EW am besten approximiert.*

2. In der Praxis wird das Arnoldi-Verfahren mit einem Restart-Mechanismus angewendet um Speicherplatz zu sparen.

Ende von Vorlesung 18 am 13.12.2022

Frage: Was können wir über die Approximation der Eigenwerte von Algorithmus 4.35 für hermitische Matrizen sagen?

Bemerkung 4.40. *Für hermitische Matrizen kann Algorithmus 4.35 zum Lanczos-Verfahren vereinfacht werden. Falls die EV von A nicht berechnet werden sollen, spart dieser Schritt Speicherplatz und Rechenzeit.*

Algorithm 4.41 Lanczos-Verfahren für lineare Eigenwertprobleme

Input: $A \in \mathbb{C}^{n\times n}$ hermitsch, $v \in \mathbb{C}^n, v \neq 0, m \in N$
Output: Approximierte Eigenwerte von A

$v_1 = \frac{v}{\|v\|_2}$
for $j = 1, 2, \ldots, m$ **do**
  Bestimme T_j mittels Lanczos-Verfahren
  (Stop bei Abbruch)
  Löse $T_j y_j = \tilde{\lambda}^{(j)} y_j$
  $\rightsquigarrow$ EP $(\tilde{\lambda_i}, \tilde{y_{ji}}), i = 1, \ldots, j$
  Konvergenztest
end for

Satz 4.42. *Sei $A \in \mathbb{C}^{n\times n}$ hermitsch mit $\lambda_1 \geq \cdots \geq \lambda_n$ und orthonormalen EV $\eta_1, \ldots, \eta_n$. Seien $\tilde{\lambda}^{(j)}$ die EW von T_j in Algorithmus 4.41 und $v_1 = \frac{v}{\|v\|_2}$ der erste ONB-Basisvektor von $K_j(A, v)$.*
Dann gilt

$$\lambda_1 \geq \tilde{\lambda_1}^{(j)} \geq \lambda_1 - (\lambda_1 - \lambda_n)\frac{\tan^2(\angle(v_1, \eta_1))}{T_{j-1}^2(1 + 2\rho_1)}$$

mit $\rho_1 = \frac{\lambda_1 - \lambda_2}{\lambda_2 - \lambda_n}$

Beweis. Bemerke: **a)** $K_j(A, v) = \operatorname{span}(v_1, Av_1, \ldots, A^{j-1}v_1) = \{P(A)v_1, P \in \Pi_{j-1}\}$
b) $v_1 = \sum_{j=1}^n \xi_i \eta_i, \xi_i \overset{\text{Satz } 1.16}{=} \langle v_1, \eta_i \rangle_2 = \cos(\angle(v_1, \eta_1))$
c) Der größte Eigenwert ζ einer hermitsche Matrix $B \in \mathbb{C}^{n\times n}$ ist gegeben durch

$$\zeta = \max_{x \in C^n, x \neq 0} \frac{x^* B x}{x^* x} \text{ Blatt 6, A4}$$

$$\begin{aligned}
\tilde{\lambda_1}^{(j)} &= \max_{x \in C^n, x \neq 0} \frac{x^* T_j x}{x^* x} \\
&= \max_{x \in C^n, x \neq 0} \frac{x^* V_j^* A V_j x}{x^* V_j^* V_j x} \\
&= \max_{x \in K_j(A,v), x \neq 0} \frac{x^* A x}{x^* x} \leq \max_{x \in C^n, x \neq 0} \frac{x^* A x}{x^* x} \\
&= \max_{P \in \Pi_{j-1}, P \neq 0} \underbrace{\frac{v_1^* P(A) A P(A) v_1}{v_1^* P(A)^* P(A) v_1}}_{=(\star)} \\
(\star) &= \frac{(\sum_{i=1}^n \xi_i P(A)\eta_i)^* A (\sum_{i=1}^n \xi_i P(A)\eta_i)}{(\sum_{i=1}^n \xi_i P(A)\eta_i)^* (\sum_{i=1}^n \xi_i P(A)\eta_i)} \\
&= \frac{\sum_{i=1}^n |\xi_i|^2 P(\lambda_i)^2 \lambda_i}{\sum_{i=1}^n |\xi_i|^2 P(\lambda_i)^2} \\
&= \lambda_1 + \frac{\sum_{i=1}^n |\xi_i|^2 P(\lambda_i)^2 (\lambda_i - \lambda_1)}{\sum_{i=1}^n |\xi_i|^2 P(\lambda_i)^2} \\
&= (\star\star)
\end{aligned}$$

$$(\star\star) \geq \lambda_1 + (\lambda_n - \lambda_i)\frac{\sum_{i=2}^n |\xi_i|^2 P(\lambda_i)^2}{|\xi_1|^2 P(\lambda_1)^2 + \sum_{i=2}^n |\xi_i|^2 P(\lambda_i)^2}$$

$$\implies \max_{P\in\Pi_{j-1}, P\neq 0} \frac{v_1^* P(A) A P(A) v_1}{v_1^* P(A)^* P(A) v_1} \geq \lambda_1 + \underbrace{(\lambda_n - \lambda_1)}_{\leq 0} \min_{P\in\Pi_{j-1}, P\neq 0} \frac{\sum_{i=2}^n |\xi_i|^2 P(\lambda_i)^2}{|\xi_1|^2 P(\lambda_1)^2 + \sum_{i=2}^n |\xi_i|^2 P(\lambda_i)^2}$$

Idee: Ersetze das Maximum durch ein speziell gewähltes Polynom $P \in \Pi_{j-1}$ mit $|P(t)|$ möglichst klein auf $[\lambda_n, \lambda_2]$. $\overset{\text{Satz 3.36}}{\implies}$ Wähle $P(t) = T_{j-1}(x(t))$ mit

$$x(t) = 1 + 2\frac{t - \lambda_2}{\lambda_2 - \lambda_n}$$

und Tschebyscheff-Polynom

$$T_{j-1}(t) = \cos((j-1)\cos^{-1}(z)), t \in [-1, 1]$$
$$|P(x(t))| \leq 1 \forall t \in [\lambda_n, \lambda_2]$$

$$\implies \max_{P\in\Pi_{j-1}, P\neq 0} \frac{v_1^* P(A) A P(A) v_1}{v_1^* P(A)^* P(A) v_1} \geq \lambda_1 + (\lambda_n - \lambda_1)\frac{\sum_{i=2}^n |\xi_i|^2}{|\xi_1|^2 P(\lambda_1)^2}$$

$$\geq \lambda_1 + (\lambda_n - \lambda_1) \underbrace{\frac{\sum_{j=2}^n |\xi_i|^2}{|\xi_1|^2}}_{=\frac{1-|\xi_1|^2}{|\xi_1 vert^2}} \frac{1}{P(\lambda_1)^2}$$

$$\implies \frac{\sum_{j=2}^n |\xi_i|^2}{|\xi_1|^2} = \frac{1 - \cos^2(\angle(v_1, \eta_1))}{cos^2(\angle(v_1, \eta_1))} = \tan(\angle(v_1, \eta_1))$$

□

Bemerkung 4.43. *1. Der Satz 4.42 gilt bei exakter Arithmetik und unter der Annahme, das die EW von T_j exakt berechnet werden.*

2. $\rho_1 = \frac{\lambda_1 - \lambda_2}{\lambda_2 - \lambda_n} \geq 0$ ist möglich, d.h. stenggenommen ist $T_{j-1}(1 + 2\rho_1)$ über die Cosinus-Darstellung nicht definiert. Hier benutzt man die alternative Darstellung

$$T_{j-1}(t) = \frac{1}{2}\left(\left(t + \sqrt{t^2 - 1}\right)^{j-1} + \left(t - \sqrt{t^2 - 1}\right)^{j-1}\right)$$

Wir sehen, dass $T_{j-1}(1 + 2\rho_1)$ exponentiell steigt, somit fällt der Kehrwert exponentiell.

3. Die Aussage des Satzes gilt für den größten EW. Analoge Aussagen für die übrigen EW sind ebenfalls möglich und können dann mit Hilfe der Conrant-Fischer Darstellung gezeigt werden.

Ende von Vorlesung 19 am 20.12.2022

Kapitel 5

Numerische Integration, Revisited

5.1 Definition, Wiederholung und Fragestellung

Aufgabenstellung: Berechne $\int_a^b f(x)w(x)dx, w(x) > 0.$ mit Hilfe von Punktauswertungen von f, w.

Motivation

Wir können dieses Integral (nach Skalierung) als Erwartungswert von f interpretieren.

Definition 5.1. *Eine Quadraturformel zur Approximation von $\int_a^b f(x)w(x)dx$ mit Stützstellen (x_i) und Gewichten (w_i) ist ein gewichteter Mittelwert.*

$$Q[f] = \sum_{i=1}^{m} w_i f(x_i) \tag{5.1}$$

Wir sprechen von einer zusammengesetzten Quadraturformel $Q_n[f]$ auf $[a, b]$, wenn $[a, b]$ in n gleich große Teilintrvalle aufgeteilt, und das Integral auf jedem Teilintervall mit $Q[f]$ berechnet wird.

Definition 5.2. *Eine Quadraturformel $Q[f]$ auf $[a, b]$ hat Exaktheitsgrad q, falls*

$$Q[p] = \int_a^b p(x)w(x)dx, p \in \Pi_q.$$

Eine zusammengesetzte Quadraturformel $Q_n[f]$ konvergiert mit Ordnung s, falls

$$\left| Q_n[f] - \int_a^b f(x)w(x)dx \right| = O(n^{-s}), n \to \infty$$

Idee: Konstruktion von QUadraturformeln mit Exaktheitsgrad q: Ersetzte f durch ein Lagrange-Interplolationplynom

$$f(x) \approx \sum_{i=0}^{q} f(x_i)L_i(x)$$

von Grad q an Stützstellen $x_0, \ldots, x_q$

$$\implies \int_a^b f(x)w(x) \approx \int_a^b \sum_{i=0}^{q} f(x)w(x)dx$$

$$\sum_{i=0}^{q} f(x_i) \underbrace{\int_a^b L_i(x) w(x) dx}_{:=w_i} \tag{5.2}$$

Für $x_0 = a, x_q = b$, dazwischen equidistant, heißen diese Quadraurformeln Newton-Cotes-Formeln.

Satz 5.3. *Sei $f \in C^{(q+1)}([a,b])$. Die Quadraturformel mit $w = 1$ hat Ordnung q und es gilt:*

$$\left| \int_a^b f(x)dx - Q[f] \right| \leq \frac{\left\| f^{(q+1)} \right\|_{C([a,b])}}{(q+1)!} \int_a^b v(x)dx,$$

wobei $v \in \Pi_{q+1}$ das Knotenplynom

$$v(x) = (x - x_0) \cdot \cdots \cdot (x - x_q)$$

zu $x_0, \ldots, x_q$ ist.

Beweis. Siehe Alma 2. □

Frage: Oft müssen wir sehr viele Integrale numerisch ausgerechnet werden, oder die Auswertung von f ist sehr teuer. Können wir bessere Quadraturformeln finden, d.h. mit höhrem Exaktheitsgrad / höherer Ordnung, bei gleicher Anzahl von Funktionsauswertungen?

Satz 5.4. *Der Exaktheitsgrad der Quadraturformel (5.1) ist maximal $q = 2m - 1$.*

Beweis. Sei $v(x) = (x - x_1) \cdot \cdots \cdot (x - x_m)$ das Knotenpolynom zu $x_1, \ldots, x_m$.

$$p(x) = v(x)^2 = \prod_{i=1}^{m} (x - x_i)^2 \in \Pi_{2m}.$$

$$\implies Q[p] = \sum_{i=1}^{m} w_i \underbrace{p(x_i)}_{=0} = 0$$

$$\int_a^b p(x)w(x)dx = \int_a^b \prod_{i=1}^{m} (x - x_i)^2 \underbrace{w(x)}_{>0} dx \neq 0$$

□

Frage: Können wir Quadraturformeln (5.1) mit Exaktheitsgrad $q = 2m - 1$ finden und systematisch konstruieren?
Idee: Sei

$$v(x) = (x - x_1) \cdot \cdots \cdot (x - x_m)$$

das Knotenplynom und $p \in \Pi_{m-1}$ beliebig.

$$vp \in \Pi_{2m-1}, (vp)(x_i) = 0, i = 1, \ldots, m$$

Annahme: $Q[f]$ hat Exaktheitsgrad $2m - 1$

$$(\star) = Q[vp] = \sum_{i=1}^{m} w_i \underbrace{(vp)(x_i)}_{=0} = 0$$

$$(\star) = \underbrace{\int_a^b v(x)p(x)w(x)dx}_{=\langle v,p \rangle_w \text{ ist ein Skalarprodukt}} \quad \text{für alle } p \in \Pi_{m-1}$$

$$\implies v \perp \Pi_{m-1} \text{ bzgl. } \langle \cdot, \cdot \rangle_w$$

$$\implies x_1, \ldots, x_m \text{ müssen Nst eines Polynomes } v \in \Pi_m, v \perp \Pi_{m-1}$$

5.2 Orthogonalpolynome

Generalvoraussetzung: Sei $I \subset \mathbb{R}$ (ein Intervall), $w : I \to \mathbb{R}_{>0}$ eine Gewichtsfunktion mit

$$\int_I w(x)dx < \infty.$$

Wir schreiben

$$\langle f, g\rangle_w = \int_I f(x)g(x)w(x)dx,$$

$$\|f\|_w = \sqrt{\langle f, f\rangle_w},$$

und definieren

$$L^2_w(I) \coloneqq \{f : I \to \mathbb{R}, \|f\|_w < \infty\}$$

Satz 5.5. *Zu jeder Gewichtfunktion w und zugehörigem Skalarprodukt existiert eine eindeutige Folge $\{u_n\}_{n=0}^{\infty}$ von Polynomen $u_n \in \Pi_n$ mit*

$$u_n(x) = \gamma_n x^n + \ldots, \gamma_n > 0$$

und

$$\langle u_n, u_m\rangle_w = \delta_{nm}.$$

Außerdem gilt mit $u_{-1} \coloneqq 0$, dass

$$u_0 = \frac{1}{\|1\|_w}$$

$$a_{n+1}u_{n+1}(x) = (x - b_n)u_n(x) - a_n u_{n-1}(x), n \geq 0 \tag{5.3}$$

mit $a_n = \frac{\gamma_{n-1}}{\gamma_n}$ und $b_n = \langle xu_n, u_n\rangle_w$

Beweis. Per Induktion. **IV**: $n = 0$ klar.
IS: $n \implies n+1$ Seien $u_{-1}, u_0, \ldots, u_n$ wie im Satz, d.h. $u_0, \ldots, u_n$ ist eine ONB.

$$p_{n+1} = (x - b_n)u_n(x) - a_n u_{n-1} \tag{5.4}$$

$$\begin{aligned}\implies \langle p_{m+1}, u_n\rangle_w &= \langle xu_n, u_n\rangle_w - b_n\langle u_n, u_n\rangle_w - a_n\langle u_{n-1}, u_n\rangle_w \\ &= 0\end{aligned}$$

$$\begin{aligned}\langle p_{n+1}, u_{n-1}\rangle_w &= \langle xu_n, u_{n-1}\rangle_w - b_n\langle u_n, u_{n-1}\rangle_w - a_n\langle u_{n-1}, u_{n-1}\rangle_w \\ &= \langle u_n, xu_{n-1}, \rangle_w \\ &= \langle u_n, xu_{n-1}\rangle_w - a_n \\ &\overset{(??)}{=} \langle u_n, a_n u_n\rangle_w + \langle u_n, b_{n-1}u_{n-1}\rangle_+ \langle u_n, a_{n-1}u_{n-2}\rangle_w - a_n \\ &= 0\end{aligned}$$

Für $0 \leq k \leq n-2$:

$$p_{n+1}, u_k = \langle xu_n, u_k\rangle_w - b_n\langle u_n, u_k\rangle_w - a_n\langle u_{n-1}, u_k\rangle_w = \langle u_n, \underbrace{xu_k}_{\in\Pi_{k+1}=\mathrm{span}\{u_0,\ldots,u_{k+1}\}\perp u_n} \rangle_w = 0$$

Außerdem: $p_{n+1} = \gamma_n x^{n+1} + \ldots$

$$\implies u_{n+1}(x) = \frac{p_{n+1}(x)}{\|p_{n+1}\|_w}$$

$\implies \gamma_{n+1} > 0$
z.z. $p_{n+1} = a_{n+1}u_{n+1}$:

$$\frac{\gamma_n}{\|p_{n+1}\|_w} = \gamma_{n+1} \implies \|p_{n+1}\|_w = \frac{\gamma_n}{\gamma_{n+1}}$$

□

Ende von Vorlesung 20 am 10.01.2023

Bemerkung 5.6. *Die Drei-Term-Rekursion ist nicht notwendig für die Eindeutigkeit:*
Seien $u_n^{(1)}, u_n^{(2)} \in \Pi_n$ *mit* $u_n^{(i)} \perp \Pi_{n-1} = span\{u_0, \dots, u_{n-1}\}$.

$$\implies \Pi_n = span\{u_0, \dots, u_n^{(1)}\}$$
$$\implies u_n^{(2)} = \sum_{i=1}^{n-1} \langle u_n^{(2)}, u_i\rangle u_i + \langle u_n^{(2)}, u_n^{(1)} u_n^{(1)} = \alpha u_n^{(1)}\rangle$$

Durch die Normierung des ersten Koeffizienten folgt dann Gleichheit.

Beispiel 5.7. *1. Die Tschebyscheff-Polynome* $T_n(x) = \cos(n\cos^{-1}(x)), x \in [-1,1]$ *sind Orthogonalpolynome auf auf* $I = [-1,1]$ *mit* $w(x) = \frac{1}{\sqrt{1-x^2}}$. *Insb. ist hier*

$$b_n = \langle xu_n, u_n\rangle = \int_{-1}^{1} x\frac{u_n(x)^2}{\sqrt{1+x^2}} = 0.$$

2. Die Legendre-Polynome ergeben sich aus $I = [-1,1], w = 1$.

3. Die Hermite-Polynome ergeben sich aus $I = \mathbb{R}, w(x) = e^{-x^2}$.

Ausgangslage: (**??**) sagt, dass wir die Nullstellen $x_1, \dots, x_m$ des Polynoms $v \in \Pi_m, v \perp \Pi_{m-1}$, bestimmen müssen. Bemerkung 5.6 sagt, $v = \alpha u_m$, mit u_m aus Satz (5.5).
Frage: Hat $v = \alpha u_m$, d.h. u_m, überhaupt m verschiedene Nullstellen.

Satz 5.8. *Das Orthogonalpolynom* $u_m, m \in \mathbb{N}_0$, *aus Satz 5.5 hat genau* m *einfache Nullstellen in* I.

Beweis. $m = 0$**:** $u_0 = \text{const} \neq 0$, klar.
$m > 0$**:** Seien $\xi_1, \dots, \xi_k \in I$ die Pnkte in I an denen sich das Vorzeichen von u_m ändert. $\implies u_m(\xi_i) = 1, i = 1, \dots, k$.
z.z.: $k = m$
Definiere

$$q(x) = \begin{cases} 1 & \text{falls } k = 0 \\ (x-\xi_1)\cdot\dots\cdot(x-\xi_k), & \text{sonst} \end{cases}$$

$\implies q$ wechselt an den gleichen Stellen wie u_m das Vorzeichen in I.
$\implies q(x)u_m(x)w(x)$ wechselt das Vorzeichen nicht in I.
$\implies 0 \neq \int_i q(x)u_m(x)w(x)dx = \langle q, u_m\rangle_w$
$\implies$ Grad $q \geq m$
$\overset{u_m \in \Pi_m}{\implies}$ Grad $q = k = m$. □

Also: Falls es eine Quadraturformel mit $m \in \mathbb{N}$ Stützstellen und Exaktheitsgrad $q = 2m-1$ gibt, dann sind die Stützstellen $x_1, \dots, x_m$ die Nullstellen des Orthogonalpolynoms aus Satz 5.5.
Frage: Finden wir die entsprechenden Gewichte?

Satz 5.9. *Eine Quadraturformel (5.1) zu den Stützstellen $x_1, \dots, x_m, x_i \neq x_j$ habe den Exaktheitsgrad $q = m-1$. Dann sind die Gewichte gegeben durch*

$$w_i = \int_I L_i(x) w(x) dx, i = 1, \dots, m \tag{5.5}$$

mit den Lagrange-Polynomen $L_i \in \Pi_{m-1}$ zu den Stützstellen $x_1, \dots, x_m$.

Beweis. Q hat Exaktheitsgrad $q = m-1$. $\implies$ $L_i \in \Pi_{m-1}$ wird exakt integriert

$$\begin{aligned} \implies \int_I L_i(x) w(x) dx &= Q[L_i] \\ &= \sum_{j=1}^{m} w_j \underbrace{L_i(x_j)}_{=\delta_{ij}} = w_i \end{aligned}$$

□

5.3 Gauß-Quadratur

Definition 5.10. *Sei $I \subset \mathbb{R}$ ein Intervall und w eine Gewichtsfunktion $w : I \to \mathbb{R}, w > 0$. Die m-te **Gaus-Quadraturformel** $Q_w^m[f]$ ist definiert durch*

$$Q_w^m[f] = \sum_{i=1}^{m} w_i f(x_i) \approx \int_I f(x) w(x) dx,$$

mit $x_1, \dots, x_m$ den Nullstellen des m-ten Orthogonalpolynoms u_m aus Satz 5.5 und Gewichten w_i wie in (5.5).

Korollar 5.11. *Die m-te Gauss-Quadraturformel $Q_w^m[f]$ hat tatsächlich Exaktheitsgrad $q = 2m-1$.*

Beweis. Sei $p \in \Pi_{2m-1}$, setze

$$\tilde{p}(x) = \sum_{i=1}^{m} p(x_i) L_i(x) \in \Pi_{m-1}$$

Fall 1: $p \in \Pi_{m-1} \implies p = \tilde{p}$.

$$\int_I p(x) w(x) dx = \int_I \tilde{p}(x) w(x) dx = \sum_{i=1}^{m} p(x_i) \underbrace{\int_I L_i(x) w(x) dx}_{=w_i} = Q_w^m[p]$$

Fall 2: $p \in \Pi_{2m-1} \setminus \Pi_{m-1}$:
Definiere

$$\begin{aligned} & q(x) = p(x) - \tilde{p}(x) \\ & \implies q(x_i) = p(x_i) - \tilde{p}(x_i) = 0, i = 1, \dots, m \\ & \implies q(x) = v(x) r(x) \text{ mit } v(x) = (x_1) \cdot \dots \cdot (x - x_m) \text{ und} \\ & r \in \Pi_{m-1} \\ & \implies \int_I q(x) w(x) dx = \int_I v(x) r(x) w(x) dx = \langle v, r \rangle_w = \alpha \langle u_m, r \rangle_w = 0 \end{aligned}$$

$$\int_I p(x) w(x) dx = \underbrace{\int_I q(x) w(x) dx}_{=0} + \underbrace{\int_I \tilde{p}(x) w(x) dx}_{=Q_w^m[p]} = Q_w^m[p]$$

□

Beispiel 5.12. *Die Gaus-Quadraturformel auf $I = [-1, 1]$ mit $w = 1$ heißt auch* ***Gauss-Legendre-Quadratur****.*

Wir haben

m	x_i	w_i
1	0	2
2	$-\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}$	1,1
3	$-\frac{\sqrt{15}}{5}, 0, \frac{\sqrt{15}}{5}$	$\frac{5}{9}, \frac{8}{9}, \frac{5}{9}$

Ende von Vorlesung 21 am 12.01.2023

Satz 5.13. *Sei $f \in \mathbb{C}^{2m}(I)$. Dann gilt*

$$\left| \int_I f(x)w(x)dx - Q_w^m(f) \right| \leq \frac{\left\| f^{(2m)} \right\|_{C(I)}}{(2m)!\gamma_m^2}$$

mit $\gamma_m > 0$ aus Satz 5.5.

Beweis. **1.:** Finde "geeignete" Interpolation:
Sei $p \in \Pi_{2m-1}$, mit

$$p(x_i) = f(x_i), p'(x_i) = f'(x_i), i = 1, \ldots, m \tag{5.6}$$

Das heißt auch Hermit-Interpolation.
Z.z.: p existiert und ist eindeutig.

$$\begin{aligned} &\mu : \Pi_{2m-1} \to \mathbb{R}^{2m} \\ &p \mapsto (p(x_1), \ldots, p(x_n), p'(x_1), \ldots, p'(x_n)) \end{aligned}$$

- μ ist linear
- $\mu(p) = 0 \implies x_1, \ldots, x_n$ sind doppelte Nullstellen $\implies p$ hat $2m$ Nullstellen, d.h. es ist konstant 0. D.h. μ is injektiv.
- $\implies \mu$ ist bijektiv (Rang-Defekt Formel). D.h. p existiert und ist eindeutig.

2.: Zeige $f(x) - p(x) = \frac{f^{(2m)}(\xi)}{(2m)!} \underbrace{\prod_{i=1}^{m}(x - x_i)^2}_{=s(x)\in\Pi_{2m}}$.

Definiere:

$$h(t) = f(t) - p(t) - \frac{s(t)}{s(x)}(f(x) - p(x))$$

$\implies h$ hat doppelte Nullstellen $x_1, \ldots, x_n$ und eine zusätzliche Nst $t = x$
$\implies h$ hat $2m + 1$ Nullstellen
$\implies h'$ hat $2m$ Nullstellen
$\implies h''$ hat $2m - 1$ Nullstellen
$\vdots$
$\implies h^{(2m)}$ hat 1 Nullstelle $\xi \in I$
$\implies 0 = h^{(2m)}(\xi) = f^{(2m)}(\xi) - \underbrace{p^{(2m)}(\xi)}_{=0} - \frac{(2m)!}{s(x)}(f(x) - p(x))$

$$\implies f(x) - p(x) = \frac{f^(2m)(\xi)}{(2m)!}s(x) \tag{5.7}$$

3.: Fehlerabschätzung

$$\left| \int_I f(x)w(x)dx - \underbrace{Q_w^m(f)}_{\overset{(5.6)}{=} Q_w^m(p) = \int_I p(x)w(x)dx} \right| = (\star)$$

$$\begin{aligned}(*) &= \left| \int_I (f(x) - p(x))w(x)dx \right| \\ &\leq \frac{\left\| f^{(2m)} \right\|}{(2m)!} \int_I \underbrace{s(x)w(x)}_{\geq 0} dx\end{aligned}$$

s hat genau $2m$ Nst und u_m^2 hat genau $2m$ Nst, beide haben Grad $2m \implies s = \frac{1}{\gamma_n^2} u_m^2$:

$$\begin{aligned}&\int_I s(x)w(x)dx \\ &= \int_I \frac{u_m(x)^2}{\gamma_m^2} w(x)dx \\ &= \frac{1}{\gamma_m}^2 \underbrace{\langle u_m, u_m \rangle_w}_{= \|u_m\|_w^2 = 1} = \frac{1}{\gamma_n^2}\end{aligned}$$

□

Bemerkung 5.14. *Für die Gaus-Legendre-Formel (d.h. $I = [-1, 1], w = 1$) ist $\gamma_m^2 \approx \frac{1}{\pi} 4^m$*

Unabhängig von der Länge von I?

In der obigen Abschätzung kommt die Länge des Intervalles nicht direkt vor, was einem zunächst komisch vorkommen sollte. Allerdings ist die Abschätzung versteckt doch abhängig von der Länge L, da die Leitkoeffizieten γ_n kleiner werden müssen, wenn L größer wird damit die Polynome normiert bleiben!

5.4 Numerische Berechnung von Gauß-Quadraturformeln

Frage: Wie bestimmen wir Stützstellen x_i und Gewichte w_i von Gauß-Quadraturformeln numerisch?
$x_i \to$ Nullstellen der Orthgonal-Polynome
$w_i = \int_I L_i(x)w(x)dx$
Erinnerung: $u_i(x) = \gamma_i x^i + \cdots +$ erfüllen gemäß Satz 5.5:

$$\begin{aligned}&= u_{-1} = 0 \\ u_0 &= \frac{1}{\|1\|_w}\end{aligned}$$

$$a_{i+1}u_{i+1} = (x - b_i)u_i(x) - a_i u_{i-1}(x), i \geq 0$$

Satz 5.15. *Die Nulltellen von u_i stimmen mit den Eigenwerten von*

$$J_i = \begin{bmatrix} b_1 & a_1 & & & \\ a_1 & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & & a_{i-1} & b \end{bmatrix}$$

überein. Ist λ ein Eigenwert von J_i und $v = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}$ ein zugehöriger Eigenvektor mit $\|v\|_2 = 1$ und $v_1 \geq 0$.

Dann gilt:

$$v_k = \gamma u_{k-1}(\lambda), k = 1 \ldots, i$$

mit

$$\gamma = \left(\sum_{k=0}^{i-1} u_k(\lambda)^2 \right)^{-\frac{1}{2}}.$$

Beweis. Satz 5.8 sagt, dass die u_i i verschiedene, einfache Nullstellen hat.
Sei λ eine dieser Nullstellen und $v_k P \gamma u_{k-1}(\lambda)$ mit $\gamma \in \mathbb{R} \setminus \{0\}$ beliebig.
Überlegung:

$$\begin{aligned}(J_j v)_k &= b_k v_k + a_{k-1} v_{k-1} + a_k v_{k+1} \overset{!}{=} \\ &= \gamma(\underbrace{b_k u_{k-1}(\lambda) + a_{k-1} u_{k-2}(\lambda) + a_k u_k(\lambda)}_{\overset{(??)}{=} \lambda u_{k-1}(\lambda)}) \\ &= \lambda v_k, k = 2, \ldots, i-1\end{aligned}$$

$k = 1$: $v_0 = \gamma u_{-1}(\lambda) = 0$
$k = i$: $v_{i+1} = \gamma u_i(\lambda) = 0$ (da, Nst).
$\implies$ (λ, v) ist EP von J_i.
Z.z $\gamma = \left(\sum_{k=0}^{i-1} u_k(\lambda)^2 \right)^{-\frac{1}{2}}, v_1 \geq 0.$
$1 \overset{!}{=} \|v\|_2^2 = \sum_{k=1}^{i} (\gamma u_{k-1}(\lambda))^2 = \gamma^2 \sum_{k=1}^{i} u_{k-1}(\lambda)^2$
$\implies$ $\gamma = \left(\sum_{k=1}^{i-1} u_k(\lambda)^2 \right)^{-\frac{1}{2}}$
$v_1 = \gamma u_0(\lambda) > 0$

□

Ende von Vorlesung 22 am 17.01.2023

Satz 5.16. *Die Gewichte der Gauß-Quadratur $Q_w^m[f]$ sind positiv, mit*

$$w_j = \left(u_m k = 0^{m-1} u_k(x_j)^2 \right)^{-1}, j = 1, \ldots, m$$

Beweis. Sei $L_j \in \Pi_{m-1}$ das j-te Lagrange-Polynom zu $x_1, \ldots, x_m$. $\implies$ $L_j(x) = \sum_{k=0}^{m-1} \alpha_k u_k(x)$

$$\alpha_k = \langle L_j, u_k \rangle_w = \int_I \underbrace{\underbrace{L_j(x)}_{\in \Pi_{m-1}} \underbrace{u_k(x)}_{\in \Pi_{m-1}}}_{\in \Pi_{2m-2}} w(x)$$

$$= Q_w^m[f] = \sum_{l=0}^{m} w_l L_j(x_l) u_k(x_k) = w_j u_k(x_j)$$

Außerdem:

$$\begin{aligned}\|L_j\|_w^2 &= \sum_{k=0}^{m-1} a_k^2 = \sum_{k=0}^{m-1} w_j^2 u_k(x_j)^2 = \sum_{k=1}^{m-1} u_k(x_j)^2 \\ &= \int_I L_j(x)^2 w(x) dx = \sum_{k=1}^{m} w_k L_j(x_k)^2 = w_j\end{aligned}$$

□

Korollar 5.17. *Gegeben eine Gewichtsfunktion und die Koeffizienten der zugehörigen 3-Term-Rekursion der Orthogonalpolynome. Seien (λ_i, v_i) die EP von der Matrix ()??. Seien die v_i so, dass $\|v_i\|_2 = 1 (v_i)_1 \geq 0$. Dann sind:*

$$x_i = \lambda_i$$
$$w_i = (v_i)_1^2 \|1\|_w^2$$

Die Stützstellen und Gewichte von $Q_w^i[f]$.

Beweis. Stützstellen x_i: Satz 5.13

Gewiche w_i Satz 5.13 sagt $(v_i)_1 = \gamma u_0(x_i) = \left(\sum_{k=0}^{m-1} u_k(x_j)^2\right)^{-1/2} \frac{1}{\|1\|_w}$.

□

Bemerkung 5.18. *J_i ist symmetrisch mit einfachen EW. Das Lösen des EW-Problems ist somit gukonditioniert und kann mit dem QR-Algorithmus erfolgen.*

Bemerkung 5.19. *Das Berechnen von a_i und b_i (und $\|1\|_w$) ist je nach Gewichtsfunktion nichttrivial. Für die Gauss-Legendre-Quadratur ($I = [-1, 1], w = 1$) gilt:* $(*)$

$$a_n = \frac{n}{\sqrt{kn^2 - 1}}$$
$$b_n = 0$$

5.5 Tensorprodukt-Quadratur

Problem: Wie berechnen wir Integrale mit Integrationsbereich $D \subset \mathbb{R}^d, d > 1$?
Vereinfachung:

1. $D = [a_1, b_1] \times \cdots \times [a_d, b_d] \subset \mathbb{R}^d$ hat Tensorprodukt-Struktur.
2. $D = [0, 1]^2 = \square$

Neue Aufgabe:
Gegeben $f \in C(\square)$, berechne

$$(I \odot I)[f] \coloneqq \int_{\square} f(x)dx = \int_0^1 \int_0^1 f(y, z)dydz$$

Idee: Benutze für jedes Integral eine eigene Quadraturformel (z.B. Gauß, summierte Trapeze, etc.)
für y : $Q_y[g] = \sum_{i=1}^m \xi g(y_i)$
für z : $Q_z[g] = \sum_{i=1}^n \zeta g(z_i)$

$$(Q_y \otimes Q_z)[f] \coloneqq \sum_{i=1}^m \sum_{j=1}^m \underbrace{\xi_i \zeta_j}_{w_{ij}} f(y_i, z_j)$$

Satz 5.20. *Für Quadraturformel Q_y, Q_z mit Exaktheitsgrad mindestens 1 und positiven Gewichten gilt:*

$$\left|(I \otimes I)[f] - (Q_y \otimes Q_x)[f] \leq \sup_{y \in [0,1]} E_z[f(y_i, \cdot)] + \sup_{z \in [0,1]} E_y[f(\cdot, z)]\right|$$

mit

$$E_y(g) = |I[g] - Q_y[g]|$$
$$E_z(g) = |I[g] - Q_z[g]|$$

Beweis.

$$
\begin{aligned}
&(I \otimes I)[f] - (Q_y \otimes Q_x)[f] \\
&= \int_0^1 \int_0^1 f(y,z)dzdy - \sum_{i=1}^m \sum_{j=1}^n \xi_i \zeta_j f(y_i, z_i) \\
&= \underbrace{\int_0^1 \int_0^1 f(y,z)dz - \sum_{j=1}^n \zeta_j f(y,z)dy}_{(I)} + \underbrace{\sum_{j=1}^n \left(\int_0^1 f(y,z_i)dy - \sum_{i=1}^m \xi_i f(y_i, z_i) \right)}_{(II)}
\end{aligned}
$$

$$
\begin{aligned}
|(I)| &\leq \int_0^1 \left| \int_0^1 f(y,z)dz - \sum_{j=1}^n \xi_j f(y,z_i) \right| dy \leq 1 \cdot \sup_{y \in [0,1]} E_z(f(y,\cdot)) \\
|(II)| &\leq \sum_{j=1}^n \zeta_i \left| \int_0^1 f(y,z_i dy - \sum_{i=1}^m \zeta_i f(y,z_i)) \right| \leq \sup_{z \in [0,1] E_y[f(\cdot,z)]}
\end{aligned}
$$

□

Beispiel 5.21. *Berachte $Q_y = Q_z = T_n$, T_n summierte Trapezregel.*
$\implies E_y(g) = E_z(g) \leq \frac{h^2}{12} \|g''\|_{C([0,1])}$, $h = 1/n$.
Für $f \in \mathbb{C}^2(\square)$ gitl:

$$
|(I \otimes I)[f] - (T_N \otimes T_N)[f]| \leq \sup_{y \in [0,1] \frac{h^2}{12}} \left\| \frac{\partial^2 f(y,\cdot)}{\partial z^2} \right\|_{C^(}[0,1]) + \sup_{z \in [0,1] \frac{h^2}{12}} \left\| \frac{\partial^2 f(\cdot,z)}{\partial y^2} \right\|_{C^(}[0,1]) = (*)
$$

$$
(*) \leq \frac{h^2}{12} \left(\left\| \frac{\partial^2 f(y,\cdot)}{\partial z^2} \right\|_{C^(}[0,1]) + \left\| \frac{\partial^2 f(\cdot,z)}{\partial y^2} \right\|_{C^(}[0,1]) \right)
$$

Beobachte: *$T_N \otimes T_N$ hat somit die gleiche Ordnung wie T_N (d.h. Ordnung 2).*
Aber: *T_N benötigt $N+1$ Funktionsauswertungen, aber $T_N \otimes T_N$ benötigt $(N+1)^2$.*

Bemerkung 5.22. *Diese Tensorprodukt-Konstruktion kann man auf beliebige endlich Dimensionen mit ähnlichen Aussagen erweitert werden. Z.B. hat die Tensorprodukt summierte Trapezregel in d Dimensionen immer noch Ordnung 2, aber benötigt $(N+1)^d$ Auswertungen. Dies bezeichnen wir als den **Fluch der Dimension**.*

Ende von Vorlesung 23 am 19.01.2023

5.6 Monte-Carlo-Quadratur, revisited

Ziel: Wir wollen den Fluch der Dimension "umgehen".
Annahme: Analog zur Gauß-Quadratur:
Sei $D \subset \mathbb{R}^d$ und $w : D \to \mathbb{R}$ eine Gewichtsfunktion, d.h. $w > 0$. Sei

$$
z = \int_D w(x)dx.
$$

Beobachtung: Für $f : D \to \mathbb{R}$ ist

$$
\int_D f(x)w(x)dx = z \underbrace{\int_D f(x) \frac{w(x)}{z} dx}_{= \mathbb{E}[f(X)]} \tag{5.8}
$$

falls X eine "stetige" Zufallsvariable mit Dichtefunktion $\frac{w(x)}{z}$.

Definition 5.23. *Sei $(\Omega, \mathcal{A}, P)$ ein Wahrscheinlichkeitsraum und $X : \Omega \to D \subset \mathbb{R}^d$ eine Zufallsvariable. X heißt* ***stetig****, falls eine integrierbare Funktion $g : \mathbb{R}^d \to [0, \infty)$ mit*

$$\int_{\mathbb{R}^d} g(x)dx = 1$$

existiert sodass

$$P(\{X \in A\}) = \int_A g(x)dx, A \subset R^d$$

g heißt Dichtefunktion.

Idee: Analog zu Alma 2, approximiere $\mathbb{E}[P(x)]$ mittels Monte-Carlo-Verfahren.

Definition 5.24. *Seien $X, X_i : \Omega \to D_i$ i.i.d. stetige Zufallsvariablen mit Dichtefunktion g auf einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, P)$. Sei $f : D \to \mathbb{R}$ stetig. Dann wird*

$$E_n[f] = \frac{1}{n} \sum_{i=1}^{n} f(X_i)$$

der Monte-Carlo-Schätzer von $\mathbb{E}[f(X)]$ genannt.
O.B.d.A: *Sei $Z = 1 \implies \int_D f(x)w(x)dx = \mathbb{E}[f(X)]$.*

Frage: Ist $E_n[f]$ eine sinnvolle Quadraturformel?
Achtung: $E_n[f]$ ist eine Zufallsgröße.
Checks:

$$\mathbb{E}[E_n[f]] = \mathbb{E}[\frac{1}{n} \sum_{i=1}^{n} f(X_i)] = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}[f(X_i)] = \mathbb{E}[f(X)]$$

Varianz:

$$\mathbb{V}[E_n[f]] = \mathbb{V}[\frac{1}{n} \sum_{i=1}^{n} f(X_i)] = \frac{1}{n^2} \sum_{i=1}^{n} \mathbb{V}[f(X_i)] = \frac{\mathbb{V}[f(X)]}{n} \tag{5.9}$$

Satz 5.25. *Sei $f : D \to \mathbb{R}$ stetig mit*

$$|\int_D f(x)w(x)dx| < \infty$$

$$\int_D f^2(x)w(x)dx < \infty.$$

Dann gilt:

$$\mathbb{E}\left[\left|\int_D f(x)w(x)dx - E_n[f]\right|\right] \leq \sqrt{\frac{\mathbb{V}[f(X)]}{n}}$$

Beweis.

$$
\begin{aligned}
&\mathbb{E}\left[\left|\int_D f(x)w(x)dx - E_n[f]\right|\right]^2 \\
&= \left(\int_D \left|\int_D f(x)w(x)dx - E_n[f]\right| w(x)dx\right)^2 \\
&= \left(\left(\int_D \left|\int_D f(x)w(x)dx - E_n[f]\right| \sqrt{w(x)}dx\right)\sqrt{w(x)}\right)^2 = (*) \\
&\overset{C.S.}{\leq} \int_D \left|\int_D f(x)w(x)dx - E_n[f]\right|^2 w(x)dx \cdot \underbrace{\int_D w(x)dx}_{=1} \\
&= \int_D \left|\underbrace{\int_D f(x)w(x)dx}_{=\mathbb{E}[f(X)]=\mathbb{E}[E_n[f]]} - E_n[f]\right|^2 w(x)dx \\
&= \mathbb{V}[E_n[f]] \overset{(5.9)}{=} \frac{\mathbb{V}[f(X)]}{n}
\end{aligned}
$$

$\implies$ Abschätzung.
z.z.: $\mathbb{V}[f(X)] < \infty$

$$
0 \leq \mathbb{V}[f(X)] = \mathbb{E}[f(X)^2] \underbrace{-\mathbb{E}[f(X)]^2}_{\leq 0} < \infty
$$

nach Vorraussetzung.

□

Bemerkung 5.26. *1. Satz 5.25 ist eine Aussage über den zu erwartenden Fehler. Eine spezifische Realisierung kann beliebig falsch sein.*

2. Das generieren der nötien (Pseudo-)Zufallszahlen ist nicht trivial.

*3. Die Konvergenzrate des Monte-Carlo-Schätzers ist $O(N^{\frac{-1}{2}})$, wobei N die Anzahl der Funktionsauswertungstellen ist. Die Konvergenzrate ist **dimensionsunabhängig!***

Zum Vergleich: Die tensorierte, summierte Trapezregel

$$
T_n \otimes \ldots, \otimes T_n
$$

auf $[0,1]^d$ hat Konvergenzrate $O(h^2)$ bei $N = (n+1)^d$ Funktionsauswertungen. Es gilt $h = \frac{1}{n}$
$\implies O(h^2) = O(n^{-2}) = O(N^{-2/d})$
Die Konvergenzrate wird für große d also schlechter.

Top 500

Es gibt eine Liste der Top 500 Rechner, also die Top 500, welche z.B. auf top500.org erreichbar ist

5.7 Dünngitterquadratur

Frage: Falls wir den Fluch der Dimension nicht besiegen können, können wir ihn dann wenigstens eträglich machen?

Definition 5.27. *Sei*

$$\varphi(x) = \begin{cases} 1 - |x| & x \in [-1,1] \\ 0 & sonst \end{cases}.$$

$\Delta_j = \{0, 1, \ldots, 2^j\}$, *und*

$$\varphi_{j,k} = \varphi(2^j x - k)|_{[0,1]}, k \in \Delta_j, j \in \mathbb{N}_0$$

$\varphi_{j,k}, k \in \Delta_j$ *heißt* ***modale Basis*** *im Raum der linearen Splines / stückweise stetien Funktionen.*

$$V_j = \{f \in C([0,1]) : f|_{2^j, 2^j(k+1)} \in \Pi_1, k = 0, \ldots, 2^j - 1\}$$

Erinnerung: Wir können $f \in \mathbb{C}([0,1])$ mittels stückweiser linearer Interpolation durch Funktionen in V_j approximieren.

$$f(x) \approx f_j(x) = \sum_{k \in \Delta_j} f(2^{-j}k)\varphi_{j,k}(x)$$

$$\implies \int_0^1 f(x)dx \approx \int_0^1 f_j(x)dx = \sum_{k \in \Delta_j} f(2^{-j}k) \underbrace{\int_0^1 \varphi_{j,k}(x)dx}_{= \begin{cases} 2^{-(j+1)} & k \in \{0, 2^j\} \\ 2^{-j} & \text{sonst} \end{cases}\}}$$

$$= 2^{-(j+1)}(f(0) + f(1) + \sum_{k=1}^{2^j - 1}) f(2^{-j}k) = T_{2^j}[f]$$

Wie haben also die summierte Trapezregel auf komplizierte Art und Weise hergeleitet. Ende von Vorlesung 24 am 24.01.2023