Numerik Mitschrift WS 2017/18

zur Vorlesung von Prof. R.Schneider

Charlotte Reimers
Lotte Schnell
Phi Vo
Moritz Weckbecker
Tobias Winterhager

11. April 2018

Inhaltsverzeichnis

1 Rechnen mit Maschinenzahlen **2**
1.1 b-adische Darstellung . . . 2
1.2 Gleitkommadarstellung . . . 2
1.3 Rechnen mit Maschinenzahlen . . . 3
1.4 Kondition und Stabilität . . . 5
1.5 Orthogonale Projektion . . . 6

2 Lösung linearer Gleichungssysteme — Die LR-Zerlegung/Cholesky-Zerlegung **7**
2.1 Einführung . . . 7
2.2 LR-Zerlegung . . . 7
2.3 Spezialfall: Symmetrische positiv definite Matrizen . . . 10

3 Ausgleichsprobleme – Die QR-Zerlegung **12**
3.1 Einführung . . . 12
3.2 Berechnung der QR-Zerlegung mit Householder-Matrizen . . . 13
3.3 Berechnung der QR-Zerlegung mit Givens-Rotationen . . . 16

4 Matrixapproximation und Pseudoinverse – Die Singulärwertzerlegung **19**
4.1 Einführung . . . 19
4.2 Konstruktion der Singulärwertzerlegung . . . 20
4.3 Die Pseudoinverse . . . 21
4.4 Der Satz von Schmidt-Mirsky . . . 22

5 Interpolation **23**
5.1 Polynominterpolation . . . 23
5.2 Hermite-Interpolation . . . 25
5.3 Trigonometrische Interpolation . . . 26

6 Regression **28**
6.1 Lineare Regression . . . 28

7 Numerische Integration **30**
7.1 Einleitung . . . 30
7.2 Newton-Cotes . . . 31
7.3 Gauß-Quadratur . . . 32

8 Iterationsverfahren **34**
8.1 Einführung . . . 34
8.2 Konvergenz . . . 34
8.3 Lösen Linearer Gleichunssysteme: Splitting - Verfahren . . . 35
8.4 Lösen Linearer Gleichunssysteme: Gradienten - Verfahren . . . 37
8.5 Lösen Nicht-Linearer Gleichungssysteme: Fixpunktiteration . . . 38
8.6 Lösen Nicht-Linearer Gleichungssysteme: Newton-Verfahren . . . 39

Kapitel 1

Rechnen mit Maschinenzahlen

1.1 b-adische Darstellung

1.1.1 Definition

Es seien $b \in \mathbb{N}$, $b \geq 2$, $\sigma \in \{-1, +1\}$, $n \in \mathbb{Z}$ und für $k \geq -n, \alpha_k \in \{0, 1, ..., b-1\}$. Dann heißt für $r \in \mathbb{R}$ die Summe

$$r = \sum_{k=-n}^{\infty} \alpha_k b^{-k}$$

b-adische Darstellung von r.

1.1.2 Beispiele

- Binär: b=2
- Dezimal: b=10
- Hexadezimal: b=16

1.2 Gleitkommadarstellung

1.2.1 IEEE Standard

Beim IEEE-Standard unterscheidet man zwischen der Single- und Double-Precission. Diese unterscheiden sich in dem für eine Zahl benötigten Speicher. Dieser beträgt 32 (double-Precission: 64) Bit. Der IEEE-Standard besteht im Allgemeinen aus 3 Komponenten.

Vorzeichen: Für das Vorzeichen wird ein Bit benötigt. Das gespeicherte Element $s \in \{-1, +1\}$ definiert das Vorzeichen σ durch $\sigma = (-1)^s$

Exponent: Für den Exponenten werden 8 (11) Bit benötigt. Diese Ziffern stellen eine Zahl $\epsilon \in \{0, ..., 2^8 - 1\,(= 255)\}(\epsilon \in \{0, ..., 2^{11} - 1(= 2047)\})$. Der Exponent wird dann berechnet durch $e = \epsilon - 127$ $(\epsilon - 1023)$

Mantisse: Die Mantisse besteht aus 23 (52) Bit und definiert eine Folge α_k für $0 \leq k \leq 23$ $(0 \leq k \leq 52)$, wobei $\alpha_0 := 1$ gilt

Dann ist die dargestellte Zahl $r \in \mathbb{R}$ gegeben durch:

$$r = \sigma\, b^e \sum_{k=0}^{23} \alpha_k\, b^{-k} \quad \text{bzw.} \quad r = \sigma\, b^e \sum_{k=0}^{52} \alpha_k\, b^{-k}$$

1.2.2 Maschinenzahlbereich

Für eine Basis $b \in \mathbb{N}, b \geq 2$, eine Mantissenlänge $l \in \mathbb{N}$, sowie eine minimalen Exponenten $e_{min} \in \mathbb{N}$ und einen maximalen $e_{max} \in \mathbb{N}$ ist der Maschinenzahlbereich $\mathbb{M}(b, l, e_{min}, e_{max})$ gegeben durch

$$\mathbb{M} = \left\{ \sigma\, b^e \sum_{k=1}^{l} \alpha_k\, b^{-k} \right\}$$

Mit $\sigma \in \{-1, +1\}$, $\alpha_k \in \{0, 1, ..., b-1\}$ und $e \in [e_{min}, e_{max}] \cap \mathbb{N}$

1.2.3 Binäre Division

Hierbei handelt es sich um gewöhnliche schriftliche Division, die (ähnlich wie die Zerlegung von Zykeln in Transposotionen) jedes 5-Jährige Kind beherrscht. Als Beispiel berechnen wir $(\frac{1}{10})_D$. Offensichtlich gilt $(1)_D = (1)_B$ und $(10)_D = (1010)_B$. Wir erhalten also:

$$\begin{array}{rl}
(\frac{1}{10})_D = & 1 : 1010 = 0,0\overline{0011} \\
& \underline{10} \\
& -\ 0 \\
& 100 \\
& \underline{-\ \ 0} \\
& 1000 \\
& \underline{-\ \ \ \ 0} \\
& 10000 \\
& \underline{-\ 1010} \\
& \quad 1100 \\
& \underline{-\ \ 1010} \\
& \qquad 10\ 0
\end{array}$$

Es gilt also $(\frac{1}{10})_D = (0,0\overline{0011})_B$

1.3 Rechnen mit Maschinenzahlen

1.3.1 Definitionen

Für einen Maschinenzahlbereich $\mathbb{M} := \mathbb{M}(b, l, e_{min}, e_{max})$, sowie $X_{min} := min\left\{ |x| \mid x \in \mathbb{M} \right\}$ und $X_{max} := max\left\{ |x| \mid x \in \mathbb{M} \right\}$ heißt dass Auftreten einer Zahl $r \in \mathbb{R}$

Overflow: Falls $|r| > X_{max}$ gilt.

Underflow: Falls $|r| < X_{min}$ gilt.

Die Funktion

$$fl : \mathbb{R} \to \mathbb{M},\ r \mapsto fl(r) := \begin{cases} \sigma\, b^e \sum\limits_{k=1}^{l} \alpha_k\, b^{-k} & \alpha_{l+1} < \frac{b}{2} \\ \sigma\, b^e \left((\sum\limits_{k=1}^{l} \alpha_k\, b^{-k}) \ + b^{-l-1} \cdot (b - \alpha_{l+1}) \right) & \alpha_{l+1} \geq \frac{b}{2} \end{cases}$$

für $r = \sigma\, b^e \sum\limits_{k=1}^{\infty} \alpha_k\, b^{-k}$ heißt Rundung auf dem Maschinenzahlbereich $\mathbb{M}$.

Außerdem heißt

$$eps := inf\left\{\delta > 0 \mid fl(1+\delta) > 1\right\}$$

Maschinengenauigkeit von $\mathbb{M}$.

Für $r \in \mathbb{R}$, wie oben heißt

$$|r - fl(r)| \leq \left(\frac{b^{-m}}{2}\right) b^e$$

absoluter Rundungsfehler und

$$\frac{|r - fl(r)|}{|r|} \leq \frac{\left(\frac{b^{-m}}{2}\right) b^e}{b^{-1}\, b^e} = \frac{1}{2}\, b^{1-m} = eps$$

relativer Rundungsfehler.

1.3.2 Maschinenoperationen

Für eine arithmetische Operation $\times \in \{+, -, *, \div\}$ und $x, y \in \mathbb{M}(b, l)$ ist die Maschinenoperation $\otimes$ definiert durch

$$x \otimes y := fl(x \times y)$$

und es gilt

$$x \otimes y = (1 + \epsilon) * (x \times y)$$

für ein $\epsilon \in \mathbb{R}, |\epsilon| \leq eps$
Beim Rechnen mit Maschinenzahlen sind 2 Regeln zu beachten, die wir jeweils mit einem Beispiel motivieren:

Regel 1: Addiere möglichst in Reihenfolge aufsteigender Beträge der Summanden

Wir betrachten $\mathbb{M}(10, 3)$ und $x = 6590 = 0.659 \cdot 10^4, y = 1 = 0.1 \cdot 10^1$ und $z = 4 = 0.4 \cdot 10^1$. Dann gilt:

$$fl(x + y + z) = 0.660 \cdot 10^4 = 6600$$

und dabei

$$(x \oplus y) \oplus z = fl(6591) \oplus z = 0.659 \cdot 10^4 \oplus z = fl(6594) = 6590$$

aber

$$x \oplus (y \oplus z) = x \oplus fl(5) = x \oplus 0.5 \cdot 10^1 = fl(6955) = 6600$$

Regel 2: Vermeide Auslöschungseffekte bei Subtraktion annähernd gleich großer Zahlen

Wir betrachten $\mathbb{M}(10, 3)$ und $x = 0.73563$ und $y = 0.72441$. Dann gilt:

$$x - y = 0.00122 = 0.122 \cdot 10^{-2} \in \mathbb{M}$$

allerdings erhält man durch vorheriges Runden

$$fl(x) \ominus fl(y) = 0.736 \ominus 0.734 = 0.002 = 0.2 \cdot 10^{-2} \in \mathbb{M}$$

und damit einen relativen Fehler von

$$\frac{0.002 - 0.00122}{0.00122} \approx 0.64 \equiv 64\%$$

1.4 Kondition und Stabilität

1.4.1 Definitionen

Ein gegebenes Problem soll durch die Funktion

$$f : X \to \mathbb{R}^m, \qquad X \subseteq \mathbb{R}^n, n, m \in \mathbb{N}$$

beschrieben werden.
Dann heißt die kleinste Zahl $\kappa_{abs} \geq 0$ mit

$$\|f(x) - f(\tilde{x})\| \leq \kappa_{abs} \|x - \tilde{x}\|, \qquad \tilde{x} \to x$$

absolute Kondition des Problems $x \mapsto f(x)$ und die kleinste Zahl $\kappa_{rel} \geq 0$ mit

$$\frac{\|f(x) - f(\tilde{x})\|}{\|f(x)\|} \leq \kappa_{rel} \frac{\|x - \tilde{x}\|}{\|x\|}, \qquad \tilde{x} \to x$$

relative Kondition des Problems $x \mapsto f(x)$.
Sei

$$\tilde{f} : X \to \mathbb{R}^m, \qquad X \subseteq \mathbb{R}^n, n, m \in \mathbb{N}$$

der zu f gehörige Lösungsalgorithmus. Dann heißt $\tilde{f}$ stabil bezüglich des Problems $x \to f(x)$ falls es eine Stabilitätskonstante σ gibt, so dass für alle $\tilde{x} \in X$ mit $\|\tilde{x} - x\| \leq eps$ stets

$$\|f(\tilde{x}) - \tilde{f}(\tilde{x})\| \leq \kappa_{rel}\, \sigma\, eps$$

gilt.

1.4.2 Beispiele

Differenzierbare Problemfunktion f: Sei das f aus der Definition differenzierbar in x. Dann erhalten wir

$$\lim_{\tilde{x} \to x} \frac{\|f(x) - f(\tilde{x})\|}{\|x - \tilde{x}\|} = \|f'(x)\| = \kappa_{abs}$$

Lineares Gleichungssystem: Sei $A \in GL_n(\mathbb{R})$. Dann wird das Lineare Gleichungssystem beschrieben durch

$$f : b \mapsto A^{-1}b$$

Dann gilt $\kappa_{abs} = \|DA^{-1}(b)\| = \|A^{-1}\|$ und somit

$$\begin{aligned} \frac{\|f(b) - f(\tilde{b})\|}{\|b - \tilde{b}\|} \cdot \frac{\|b\|}{\|f(b)\|} &\leq \|A^{-1}\| \cdot \frac{\|b\|}{\|A^{-1}\|} \\ &\leq \|A^{-1}\| \cdot \frac{\|Ax\|}{\|x\|} \\ &\leq \|A^{-1}\| \cdot \|A\| \end{aligned}$$

Somit gilt $\kappa_{rel} \leq \|A^{-1}\| \cdot \|A\|$. Wir definieren deshalb

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

als Konditionszahl von A

1.5 Orthogonale Projektion

1.5.1 Definition

Sei V ein reeller Vektorraum mit Skalarprodukt $\langle \cdot, \cdot \rangle$ und $W \subseteq V$ ein Unterraum von V. Dann heißt $y^* \in V$ orthogonale Projektion von x auf W, falls einerseits

$$y^* \in W \tag{1.1}$$

und andererseits $\forall y \in W$

$$\langle y^* - x, y \rangle = 0 \tag{1.2}$$

gilt.

1.5.2 Eindeutigkeit

Die Abbildung $P : V \to W, x \mapsto y^*$ ist eindeutig und linear. Für eine Orthogonalbasis $(w_1, ..., w_m)$ von W und einer erweiterten Basis $(x_1, .., x_n) := (w_1, ..., w_m, v_{m+1}, ..., v_n)$ von V ist für

$$v := \sum_{i=1}^{n} \lambda_i \, x_i \; \in V$$

$P(v)$ gegeben durch

$$P(v) = \sum_{i=1}^{m} \lambda_i \, x_i = \sum_{i=1}^{m} \lambda_i \, w_i \; \in W$$

1.5.3 Element bester Approximation

Sei V ein reeller Vektorraum mit Skalarprodukt $\langle \cdot, \cdot \rangle$ und $W \subseteq V$ ein Unterraum von V. Dann heißt $y^* \in V$ Element bester Approximation von x auf W, falls sowohl

$$y^* \in W, \tag{1.1}$$

als auch $\forall y \in W$

$$\|x - y^*\| \leq \|x - y\| \tag{1.2}$$

gilt.
Insbesondere stimmt das Element bester Approximation von x auf W immer mit der orthogonalen Projektion von x auf W überein.

Kapitel 2

Lösung linearer Gleichungssysteme — Die LR-Zerlegung/Cholesky-Zerlegung

2.1 Einführung

Es sei ein lineares Gleichungssystem mit n Gleichungen und m Unbekannten der folgenden Form gegeben:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1m}x_m &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2m}x_m &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nm}x_m &= b_n \end{aligned}$$

Gesucht sind die Unbekannten $x_1, \ldots, x_m \in \mathbb{R}$ die dieses Gleichungssystem lösen.
Sei nun $A = [a_{ij}] \in \mathbb{R}^{n,m}$, $x = [x_1, \ldots, x_m]^T \in \mathbb{R}^{m,1}$ und $b = [b_1, \ldots, b_n]^T \in \mathbb{R}^{n,1}$ gegeben, dann lässt sich das obige Gleichungssystem auch darstellen als

$$Ax = b.$$

2.2 LR-Zerlegung

Es sei im folgenden $n = m$. Sind alle Diagonaleinträge von A nicht Null, d.h. $a_{ii} \neq 0 \; \forall i \in [n] := \{1, \ldots, n\}$, so wissen Sie aus der linearen Algebra, dass das Gaußsche Eliminierungsverfahren die Lösung des LGS durch Rückwärtseinsetzen liefert, man also eine Lösung der Form

$$\begin{aligned} \tilde{a}_{11}x_1 + \tilde{a}_{12}x_2 + \cdots + \tilde{a}_{1n}x_n &= \tilde{b}_1 \\ \tilde{a}_{22}x_2 + \cdots + \tilde{a}_{2n}x_n &= \tilde{b}_2 \\ &\vdots \\ \tilde{a}_{nn}x_n &= \tilde{b}_n \end{aligned}$$

erhält. Dieses Gleichungssystem lässt sich in der Form

$$Rx = \tilde{b}$$

darstellen, wobei $R \in \mathbb{R}^{n,n}$ eine obere Dreiecksmatrix ist. Gesucht ist also eine Zerlegung

$$A = LR,$$

wobei $R \in \mathbb{R}^{n,n}$ eine obere Dreiecksmatrix und $L \in \mathbb{R}^{n,n}$ eine untere Dreiecksmatrix mit Einsen auf der Diagonalen ist. Diese existiert genau dann, wenn A regulär, also invertierbar, ist. Sind alle Diagonaleinträge von A nicht Null, so liefert uns der Gauß-Algorithmus gerade diese Zerlegung. Schreiben wir für die Teilmatrix $\mathbb{R}^{n-r,n-r} \ni [a_{ij}]$, $i, j \in \{r, \dots, n\}$ für $r \in [n]$ die Kurzform $A^{(r)}$, so ist der Gauß-Algorithmus gegeben durch

1. Eliminiere die erste Spalte mit dem ersten Diagonaleintrag von A. Dies liefert eine Zerlegung der Form $A = L_r \tilde{A}$.

2. Wiederhole dies für die Matrix $\tilde{A}^{(r)}$, wobei $r - 1$ die Anzahl der Rekursionen ist.

In Pseudocode haben wir also den folgenden Algorithmus:

Algorithmus 1 Gauß-Algorithmus

```
for j = 1 to n do
    for i = j + 1 to n do
        l_ij = -a_ij / a_jj
        for k = j to n do
            a_ik ← a_ik + l_ij a_jk
        end for
    end for
end for
```

Daraus können wir ablesen, dass der Gauß-Algorithmus eine asymptotische Laufzeit von $O(n^3)$ hat.

Pivotisierung

Um für gewisse $i \in [n]$ den Fall $a_{ii} = 0$ oder Auslöschungseffekte durch das Dividieren durch betragsmäßig kleine Zahlen zu vermeiden, können wir Spaltenpivotisierung vornehmen um den ersteren Fall zu umgehen oder gegebenenfalls die Stabilität des Algorithmus zu verbessern.
Dazu nutzen wir Permutationsmatrizen, um in jedem Schritt Zeilenvertauschungen so vorzunehmen, dass das betragsmäßig größte Element in der Diagonalposition der Matrix steht. Den resultierenden Algorithmus nennen wir *Gaußelimination mit partieller Pivotisierung*.
Ist $P \in \mathbb{R}^{n,n}$ also eine Permutationsmatrix, so ist die LR-Zerlegung gegeben durch

$$PA = LR$$

und es gilt weiter

$$Ax = b \quad \Leftrightarrow \quad Rx = L^{-1}Pb.$$

Beispiel

Es sei

$$A = \begin{bmatrix} 2 & 1 & 7 \\ 4 & 3 & 6 \\ 1 & 5 & 8 \end{bmatrix} \quad \text{und} \quad b = \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix}.$$

Wir berechnen mit der Gaußelimination mit partieller Pivotisierung ein $x \in \mathbb{R}^3$ so dass

$$Ax = b.$$

$$A = \begin{bmatrix} 2 & 1 & 7 \\ \boxed{4} & 3 & 6 \\ 1 & 5 & 8 \end{bmatrix} \qquad \Rightarrow P_1 = P_{12}$$

$$\longrightarrow \quad P_1 A = \begin{bmatrix} 4 & 3 & 6 \\ 2 & 1 & 7 \\ 1 & 5 & 8 \end{bmatrix} \qquad \Rightarrow l_{21} = -\tfrac{1}{2},\ l_{31} = -\tfrac{1}{4}$$

$$\Rightarrow L_1 = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ -\frac{1}{4} & 0 & 1 \end{bmatrix}$$

$$\longrightarrow \quad L_1 P_1 A = \begin{bmatrix} 4 & 3 & 6 \\ 0 & -\frac{1}{2} & 4 \\ 0 & \boxed{4\frac{1}{4}} & 6\frac{1}{2} \end{bmatrix} \qquad \Rightarrow P_2 = P_{23}$$

$$\longrightarrow \quad P_2 L_1 P_1 A = \begin{bmatrix} 4 & 3 & 6 \\ 0 & 4\frac{1}{4} & 6\frac{1}{2} \\ 0 & -\frac{1}{2} & 4 \end{bmatrix} \qquad \Rightarrow l_{22} = \tfrac{2}{17}$$

$$L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{2}{17} & 1 \end{bmatrix}$$

$$\longrightarrow \quad L_2 P_2 L_1 P_1 A = \begin{bmatrix} 4 & 3 & 6 \\ 0 & 4\frac{1}{4} & 6\frac{1}{2} \\ 0 & 0 & 4\frac{13}{17} \end{bmatrix} =: R$$

Da weiterhin

$$P_2 L_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ -\frac{1}{4} & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{4} & 0 & 1 \\ -\frac{1}{2} & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{4} & 1 & 0 \\ -\frac{1}{2} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} =: \tilde{L}_1 P_2$$

gilt, erhalten wir so

$$\underbrace{P_2 P_1}_{=:P} A = \underbrace{\tilde{L}_1^{-1} L_2^{-1}}_{=:L} R$$

mit

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

und

$$L = \tilde{L}_1^{-1} L_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{4} & 1 & 0 \\ \frac{1}{2} & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\frac{2}{17} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{4} & 1 & 0 \\ \frac{1}{2} & -\frac{2}{17} & 1 \end{bmatrix}.$$

Damit haben wir dann

$$
\begin{aligned}
& Rx = L^{-1}Pb \\
\Leftrightarrow \quad & \begin{bmatrix} 4 & 3 & 6 \\ 0 & 4\frac{1}{4} & 6\frac{1}{2} \\ 0 & 0 & 4\frac{13}{17} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{4} & 1 & 0 \\ -\frac{1}{2} & \frac{2}{17} & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix} \\
\Leftrightarrow \quad & \begin{bmatrix} 4 & 3 & 6 \\ 0 & 4\frac{1}{4} & 6\frac{1}{2} \\ 0 & 0 & 4\frac{13}{17} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{4} & 1 & 0 \\ -\frac{1}{2} & \frac{2}{17} & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 0 \\ 1 \end{bmatrix} \\
\Leftrightarrow \quad & \begin{bmatrix} 4 & 3 & 6 \\ 0 & 4\frac{1}{4} & 6\frac{1}{2} \\ 0 & 0 & 4\frac{13}{17} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ -\frac{3}{4} \\ -\frac{1}{2} \end{bmatrix}.
\end{aligned}
$$

Per Rückwärtseinsetzen erhalten wir damit

$$x_3 = -\frac{13}{162}, \qquad x_2 = -\frac{74}{1377}, \qquad x_1 = \frac{1672}{459}.$$

2.3 Spezialfall: Symmetrische positiv definite Matrizen

Es sei $A \in \mathbb{R}^{n,n}$. Wir nennen A symmetrisch, falls $A = A^T$ gilt, und positiv definit, falls

$$x^T A x > 0 \quad \forall x \in \mathbb{R}^{n,1}.$$

Positive Definitheit ist äquivalent dazu, dass alle Eigenwerte von A reell und positiv sind. Ebenso besagt der *Satz von Sylvester* (auch *Kriterium von Hurwitz*), dass eine Matrix genau dann positiv definit ist, wenn alle ihre Hauptminoren $A(1:k, 1:k), k \in [n]$ positive Determinanten haben (analog sind die Vorzeichen der Determinanten der Hauptminoren, negativ beginnend, alternierend, so ist die Matrix negativ definit).
Eine weitere Äquivalenz zur positiven Definitheit einer symmetrischen Matrix ist die Existenz einer unteren Dreiecksmatrix L so dass

$$A = LL^T.$$

Diese Zerlegung, die wir als einen Spezialfall der LR-Zerlegung betrachten können, nennen wir die *Cholesky-Zerlegung*. Dies bedeutet, dass falls uns eine symmetrische, positiv definite Matrix vorliegt, wir die LR-Zerlegung durch die Cholesky-Zerlegung ersetzen und verbessern können!
Betrachten wir also für $A \in \mathbb{R}^{n,n}$ symmetrisch und positiv definit und $L = [l_1, \ldots, l_n] \in \mathbb{R}^{n,n}$, $l_i \in \mathbb{R}^{n,1}$ die Gleichung

$$A = LL^T = \begin{bmatrix} l_{11} & & 0 \\ \vdots & \ddots & \\ l_{n,1} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} l_{11} & \cdots & l_{n1} \\ & \ddots & \vdots \\ 0 & & l_{nn} \end{bmatrix},$$

so gilt erhalten wir

$$
\begin{aligned}
a_{jj} &= \sum_{i=1}^{j} l_{ij}^2 = \langle l_j, l_j \rangle \\
\Rightarrow a_{11} &= l_{11}^2 \;\Rightarrow\; l_{11} = \sqrt{a_{11}} \\
\Rightarrow l_{jj} &= \sqrt{a_{jj} - \sum_{i=1}^{j-1} l_{ij}^2}
\end{aligned}
$$

und für $i > j$ folgt

$$a_{ij} = \sum_{k=1}^{j} l_{ik} l_{kj}$$
$$\Rightarrow l_{ij} = \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{kj} \right).$$

In Pseudocode erhalten wir also den Algorithmus

Algorithmus 2 Cholesky-Zerlegung

```
for j = 1 to n do
    l_jj = sqrt(a_jj − Σ_{k=1}^{j−1} l_jk^2)
    for i = j + 1 to n do
        l_ij = (1/l_jj)(a_ij − Σ_{k=1}^{j−1} l_ik l_kj)
    end for
end for
```

an dem wir erkennen dass die asymptotische Laufzeit der Berechnung der Cholesky-Zerlegung im Vergleich zum Gauß-Algorithmus besser ist.

Beispiel

Es sei $A \in \mathbb{R}^{3,3}$ mit

$$A = \begin{bmatrix} 4 & 2 & 6 \\ 2 & 10 & 9 \\ 6 & 9 & 14 \end{bmatrix}.$$

Dann gilt

$$l_{11} = \sqrt{a_{11}} = 2,\ l_{21} = \tfrac{a_{21}}{l_{11}} = 1,\ l_{31} = \tfrac{a_{31}}{l_{11}} = 3,$$
$$\Rightarrow \quad l_{22} = \sqrt{a_{22} - l_{21}^2} = 3,\ l_{32} = \tfrac{1}{l_{22}}(a_{32} - l_{31} l_{12}) = 2$$
$$\Rightarrow \quad l_{33} = \sqrt{a_{33} - l_{31}^2 - l_{32}^2} = 1.$$

Die Cholesky-Zerlegung ist also gegeben durch

$$L = \begin{bmatrix} 2 & 0 & 0 \\ 1 & 3 & 0 \\ 3 & 2 & 1 \end{bmatrix} \quad \text{mit} \quad A = LL^T.$$

Kapitel 3

Ausgleichsprobleme – Die QR-Zerlegung

3.1 Einführung

Betrachten wir den allgemeinen Fall $A \in \mathbb{R}^{n,m}$, $x \in \mathbb{R}^{m,1}$ und $b \in \mathbb{R}^{n,1}$ und die Gleichung

$$Ax = b,$$

so kann es dazu kommen, dass diese Gleichung keine Lösung besitzt, selbst wenn A vollen Rang hat. In vielen Fällen ersetzen wir dann die Suche nach einer Lösung durch ein alternatives Problem: Finde ein $x^* \in \mathbb{R}^{m,1}$ so dass

$$\min_{x \in \mathbb{R}^{m,1}} \|Ax - b\| = \|Ax^* - b\|,$$

sprich wir suchen einen Vektor, der so nahe wie möglich an der gesuchten Lösung liegt.
Wir betrachten von nun an den Regelfall $n \geq m$. Hat A vollen Rang, so ist Ihnen aus der linearen Algebra bekannt, dass das Gram-Schmidt-Verfahren eine Zerlegung von A in eine orthogonale Matrix $Q \in GL_n(\mathbb{R})$ und eine verallgemeinerte obere Dreiecksmatrix R der Form

$$R = \begin{bmatrix} \tilde{R} \\ 0 \end{bmatrix} \in \mathbb{R}^{n,m}, \tilde{R} = \begin{bmatrix} * & \cdots & * \\ & \ddots & \vdots \\ & & * \end{bmatrix} \in \mathbb{R}^{m,m}$$

liefert, so dass

$$A = QR$$

gilt. Unter Verwendung dieser Zerlegung und den Eigenschaften von orthogonalen Matrizen, erhalten wir für das obige Ausgleichsproblem

$$\min_{x \in \mathbb{R}^{m,1}} \|Ax - b\| = \min_{x \in \mathbb{R}^{m,1}} \|QRx - b\| = \min_{x \in \mathbb{R}^{m,1}} \|Q(Rx - Q^T b)\| = \min_{x \in \mathbb{R}^{m,1}} \|Rx - Q^T b\|.$$

Deklarieren wir nun

$$R =: \begin{bmatrix} \tilde{R} \\ 0 \end{bmatrix} \quad \text{und} \quad Q^T b =: \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

mit $\tilde{R} \in \mathbb{R}^{m,m}$, $b_1 \in \mathbb{R}^{m,1}$, $b_2 \in \mathbb{R}^{n-m,1}$, so gilt weiter

$$\begin{aligned}\min_{x\in\mathbb{R}^{m,1}} \|Ax-b\|^2 &= \min_{x\in\mathbb{R}^{m,1}} \left\| \begin{bmatrix} \tilde{R} \\ 0 \end{bmatrix} x - \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \right\|^2 \\ &= \min_{x\in\mathbb{R}^{m,1}} \|\tilde{R}x - b_1\|^2 + \|0 - b_2\|^2 \\ &= \|b_2\|^2 + \min_{x\in\mathbb{R}^{m,1}} \|\tilde{R}x - b_1\|^2.\end{aligned}$$

Die Lösung unseres Ausgleichsproblems ist also gegeben durch

$$x^* = \tilde{R}^{-1} b_1$$

mit dem Residuum

$$\|Ax^* - b\| = \|b_2\|.$$

Jedoch ist die Berechnung der obigen Zerlegung, genannt *QR-Zerlegung*, mit dem Gram-Schmidt-Verfahren rechenlastig, so dass wir auf andere Methoden zur Berechnung dieser Zerlegung zurückgreifen.

3.2 Berechnung der QR-Zerlegung mit Householder-Matrizen

Eine solche Methode ist die Zerlegung mittels sogenannter *Householder-Matrizen*:

$$H_w := I_n - \frac{2}{w^T w} w w^T$$

Diese Transformation entspricht der Spiegelung eines Vektors an der Hyperebene, die orthogonal zu w ist, denn es gilt

$$H_w x = x - \frac{2}{w^T w}(w^T x) w = x - \frac{\langle x, w\rangle w}{\|w\|^2} - \frac{\langle x, w\rangle w}{\|w\|^2}.$$

Householder-Matrizen haben die folgenden Eigenschaften:

- $H = H^T$ (Symmetrie)
- $H^2 = I_n$ (Involution)
- $H^T H = I_n$ (Orthogonalität)

Um mit diesen Matrizen die Spalten einer Matrix zu orthogonalisieren, wählen wir zur Bestimmung der Householder-Matrizen bestimmte von den Spaltenvektoren abhängige Vektoren. Ist also $0 \neq x \in \mathbb{R}^{n,1}$ mit $x \notin \operatorname{span}\{e_1\}$ und $\sigma := \operatorname{sgn}(x_1)\|x\|$ so wählen wir

$$w = \frac{x + \sigma e_1}{\|x + \sigma e_1\|},$$

dann gilt

$$\|x + \sigma e_1\|^2 = \|x\|^2 + 2\sigma e_1^T x + \sigma^2 = \|x\|^2 + 2\sigma e_1^T x + \|x\|^2 = 2(x + \sigma e_1)^T x,$$

womit weiter

$$2w^T x = \frac{2(x + \sigma e_1)^T x}{\|x + \sigma e_1\|} = \|x + \sigma e_1\|$$

und somit insbesondere

$$x - 2ww^T x = x - (x + \sigma e_1) = -\sigma e_1$$

folgt. Anschaulich bedeutet dies also

$$H_w x = H_w \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = -\sigma e_1 = -\sigma \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

was gerade unser Ziel war. Damit können wir nun einen Algorithmus zur QR-Zerlegung mit Householder-Matrizen konstruieren:
Es sei $A = A^{(1)} \in \mathbb{R}^{n,m}$. Wir wollen Matrizen $A^{(k)}$ der Form

$$A^{(k)} = \begin{bmatrix} a_{11}^{(k)} & a_{12}^{(k)} & & \cdots & & a_{1m}^{(k)} \\ & \ddots & & & & \vdots \\ & & a_{k-1k-1}^{(k)} & & & a_{k-1m}^{(k)} \\ & & & a_{kk}^{(k)} & \cdots & a_{km}^{(k)} \\ & & & \vdots & & \vdots \\ & & & a_{nk}^{(k)} & \cdots & a_{nm}^{(k)} \end{bmatrix}, \quad k \in [m]$$

berechnen, so dass $A^{(m)} = R$ eine verallgemeinerte obere Dreiecksmatrix ist. Dies ist gegeben durch Transformationen der Form

$$A^{(k)} = \hat{H}_k A^{(k-1)}, \quad \hat{H}_k = \begin{bmatrix} I_{k-1} & 0 \\ 0 & H_{w_k} \end{bmatrix},$$

wobei

$$w_k = \frac{a_k^{(k)} + \operatorname{sgn}(a_{kk}^{(k)}) \left\| a_k^{(k)} \right\| e_1}{\left\| a_k^{(k)} + \operatorname{sgn}(a_{kk}^{(k)}) \left\| a_k^{(k)} \right\| e_1 \right\|}, \; e_1 \in \mathbb{R}^{n-(k-1),1}$$

mit

$$a_k^{(k)} = \begin{bmatrix} a_{kk}^{(k)} \\ \vdots \\ a_{nk}^{(k)} \end{bmatrix},$$

denn dann gilt

$$H_{w_k} a_k^{(k)} = \sigma e_1 \quad \text{mit } \sigma = \pm \left\| a_k^{(k)} \right\|.$$

In Pseudocode ausgedrückt erhalten wir den folgenden Algorithmus:

Algorithmus 3 QR-Zerlegung mit Householder-Matrizen

Let $A = A^{(1)} = [a_{ij}] \in \mathbb{R}^{n,m}$
for $k = 1$ **to** m **do**
  $a_k \leftarrow [0_1, \dots, 0_{k-1}, a_{kk}, \dots, a_{nk}]^T$
  $w_k \leftarrow a_k + \operatorname{sgn}(a_{kk})\|a_k\|e_k$
  $v_k \leftarrow \frac{w_k}{(w_k)_1}$
  $\beta_k \leftarrow \frac{2}{\|v_k\|^2}$
  $A_1^{(k)} \leftarrow v_k^T A$
  $A_2^{(k)} \leftarrow v_k A_1$
  $A^{(k+1)} \leftarrow A^{(k)} - \beta_k \tilde{A}_2^{(k)}$
end for

So erhalten wir dann die Zerlegung

$$A = QR \quad \text{mit } Q := \hat{H}_1 \hat{H}_2 \dots \hat{H}_{m-1},$$

wobei Q orthogonal ist da alle $\hat{H}_k$ orthogonal sind.

Beispiel

Es sei

$$A = A^{(1)} = \begin{bmatrix} 0 & -4 \\ 6 & -3 \\ 8 & 1 \end{bmatrix} \in \mathbb{R}^{3,2}.$$

Wir berechnen die QR-Zerlegung von A mittels Householder-Matrizen. Dann haben wir

$$w_1 = \begin{bmatrix} 0 \\ 6 \\ 8 \end{bmatrix} + \begin{bmatrix} \sqrt{0^2 + 6^2 + 8^2} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 10 \\ 6 \\ 8 \end{bmatrix}$$

$$\Rightarrow \quad v_1 = \begin{bmatrix} 1 \\ \frac{3}{5} \\ \frac{4}{5} \end{bmatrix}, \ \beta_1 = \frac{2}{\sqrt{2}^2} = 1$$

$$\Rightarrow \quad A_1^{(1)} = v_1^T A^{(1)} = \begin{bmatrix} 10 & -5 \end{bmatrix} \quad \Rightarrow \quad A_2^{(1)} = v_1 A_1^{(1)} = \begin{bmatrix} 10 & -5 \\ 6 & -3 \\ 8 & -4 \end{bmatrix}$$

$$\Rightarrow \quad A^{(2)} = A^{(1)} - \beta A_2^{(1)} = \begin{bmatrix} -10 & 1 \\ 0 & 0 \\ 0 & 5 \end{bmatrix}$$

$$w_2 = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} + \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 5 \\ 5 \end{bmatrix}$$

$$\Rightarrow \quad v_2 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, \ \beta_2 = \frac{2}{\sqrt{2}^2} = 1$$

$$\Rightarrow \quad A_1^{(2)} = v_2^T A^{(2)} = \begin{bmatrix} 0 & 5 \end{bmatrix} \quad \Rightarrow \quad A_2^{(2)} = v_2 A_1^{(2)} = \begin{bmatrix} 0 & 0 \\ 0 & 5 \\ 0 & 5 \end{bmatrix}$$

$$\Rightarrow A^{(3)} = A^{(2)} - \beta_2 A_2^{(2)} = \begin{bmatrix} -10 & 1 \\ 0 & -5 \\ 0 & 0 \end{bmatrix} =: R.$$

Insgesamt erhalten wir also

$$H_1 = I_3 - \beta_1 v_1 v_1^T = I_3 - \begin{bmatrix} 1 & \frac{3}{5} & \frac{4}{5} \\ \frac{3}{5} & \frac{9}{25} & \frac{12}{25} \\ \frac{4}{5} & \frac{12}{25} & \frac{16}{25} \end{bmatrix} = \begin{bmatrix} 0 & -\frac{3}{5} & -\frac{4}{5} \\ -\frac{3}{5} & \frac{16}{25} & -\frac{12}{25} \\ -\frac{4}{5} & -\frac{12}{25} & \frac{9}{25} \end{bmatrix}$$

und

$$H_2 = I_3 \beta_2 v_2 v_2^T = I - 3 - \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

und damit

$$H_2 H_1 A = R \quad \Leftrightarrow \quad A = \underbrace{H_1 H_2}_{=:Q} R$$

mit

$$Q = \begin{bmatrix} 0 & \frac{4}{5} & \frac{3}{5} \\ -\frac{3}{5} & \frac{12}{25} & -\frac{16}{25} \\ -\frac{4}{5} & -\frac{9}{25} & \frac{12}{25} \end{bmatrix}.$$

3.3 Berechnung der QR-Zerlegung mit Givens-Rotationen

Während bei vergleichsweise dicht besetzten Matrizen die Faktorisierung mit Householder-Matrizen wesentliche Vorteile im Vergleich zum Gram-Schmidt-Verfahren liefert, treten aber oft auch dünn besetzte Matrizen wie beispielsweise Tridiagonalmatrizen auf, bei denen wir diese spärliche Besetzung durch sogenannte *Givens-Rotationen* ausnutzen können. Es sei $A = [a_{ij}] \in \mathbb{R}^{n,m}$, dann ist eine *Givens-Rotation* des (i,j)-ten Eintrags von A $(i > j)$ gegeben durch

$$G_{ij} := \begin{bmatrix} I_{j-1} & & & & \\ & c & & s & \\ & & I_{i-j} & & \\ & -s & & c & \\ & & & & I_{n-i} \end{bmatrix} \in \mathbb{R}^{n,n}$$

mit

$$r := \sqrt{a_{jj}^2 + a_{ij}^2}, \; c = \frac{a_{jj}}{r}, \; s = \frac{a_{ij}}{r}.$$

Betrachten wir dann nämlich den vereinfachten Fall $n = 2, m = 1$ und $A = \left(\begin{smallmatrix} a \\ b \end{smallmatrix}\right) \in \mathbb{R}^{2,1}$, so gilt

$$G_{21} = \frac{1}{\sqrt{a^2 + b^2}} \begin{bmatrix} a & b \\ -b & a \end{bmatrix}$$

und somit

$$G_{21} A = \frac{1}{\sqrt{a^2 + b^2}} \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{\sqrt{a^2 + b^2}} \begin{bmatrix} a^2 + b^2 \\ -ab + ab \end{bmatrix} = \begin{bmatrix} \sqrt{a^2 + b^2} \\ 0 \end{bmatrix}.$$

Da diese Matrizen orthogonal sind, erhalten wir so eine Zerlegung

$$A = QR \quad \text{mit} \quad Q := G_{21}^T, \ldots, G_{nm}^T.$$

Der zugehörige Algorithmus in Pseudocode ist folgendermaßen gegeben:

Algorithmus 4 QR-Zerlegung mit Givens-Rotationen

```
Let A = [a_ij] ∈ ℝ^{n,m}
for j = 1 to m do
    for i = j + 1 to n do
        if a_ij ≠ 0 then
            r_ij ← √(a_jj^2 + a_ij^2)
            c_ij ← a_jj / r_ij
            s_ij ← a_ij / r_ij
            A ← G_ij A
        end if
    end for
end for
```

Beispiel

Sei

$$A = \begin{bmatrix} 1 & 5 \\ -2 & 1 \\ 2 & 0 \end{bmatrix} \in \mathbb{R}^{3,2}.$$

Wir berechnen die QR-Zerlegung von A unter Verwendung von Givens-Rotationen.

$$A = \begin{bmatrix} 1 & 5 \\ \boxed{-2} & 1 \\ 2 & 0 \end{bmatrix} \qquad \Rightarrow r_{21} = \sqrt{1^2 + (-2)^2} = \sqrt{5}$$

$$\Rightarrow c_{21} = \frac{1}{\sqrt{5}}, \; s_{21} = -\frac{2}{\sqrt{5}}$$

$$\Rightarrow G_{21} = \begin{bmatrix} \frac{1}{\sqrt{5}} & -\frac{2}{\sqrt{5}} & 0 \\ \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\longrightarrow \quad G_{21}A = \begin{bmatrix} \sqrt{5} & \frac{3}{\sqrt{5}} \\ 0 & \frac{11}{\sqrt{5}} \\ \boxed{2} & 0 \end{bmatrix} \qquad \Rightarrow r_{31} = \sqrt{\sqrt{5}^2 + 2^2} = 3$$

$$\Rightarrow c_{31} = \frac{\sqrt{5}}{3}, \; s_{31} = \frac{2}{3}$$

$$\Rightarrow G_{31} = \begin{bmatrix} \frac{\sqrt{5}}{3} & 0 & \frac{2}{3} \\ 0 & 1 & 0 \\ -\frac{2}{3} & 0 & \frac{\sqrt{5}}{3} \end{bmatrix}$$

$$\longrightarrow \quad G_{31}G_{21}A = \begin{bmatrix} 3 & 1 \\ 0 & \frac{11}{\sqrt{5}} \\ 0 & \boxed{-\frac{2}{\sqrt{5}}} \end{bmatrix}$$

$$\Rightarrow r_{32} = \sqrt{\frac{121}{5} + \frac{4}{5}} = 5$$

$$\Rightarrow c_{32} = \tfrac{11}{5\sqrt{5}}, \; s_{32} = -\tfrac{2}{5\sqrt{5}}$$

$$\Rightarrow G_{32} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{11}{5\sqrt{5}} & -\frac{2}{5\sqrt{5}} \\ 0 & \frac{2}{5\sqrt{5}} & \frac{11}{5\sqrt{5}} \end{bmatrix}$$

$$\longrightarrow \quad G_{32}G_{31}G_{21}A = \begin{bmatrix} 3 & 1 \\ 0 & 5 \\ 0 & 0 \end{bmatrix} =: R$$

Damit haben wir

$$A = \underbrace{G_{21}^T G_{31}^T G_{32}^T}_{=:Q} R$$

mit

$$Q = \frac{1}{75} \begin{bmatrix} 25 & 70 & 1 \\ -50 & 25 & 50 \\ 50 & -10 & 55 \end{bmatrix}.$$

Kapitel 4

Matrixapproximation und Pseudoinverse – Die Singulärwertzerlegung

4.1 Einführung

Aus der linearen Algebra ist bekannt, dass symmetrische Matrizen vollständig durch ihre Eigenwerte und Eigenvektoren beschrieben werden. Ist $A \in \mathbb{R}^{n,n}$ symmetrisch, so lässt sich diese reell diagonalisieren, sprich es gibt eine Orthogonalbasis aus Eigenvektoren Q und eine Diagonalmatrix D aus Eigenwerten von A mit $A = QDQ^T$.
Die *Singulärwertzerlegung* liefert uns eine Möglichkeit, diese Beschreibung über „Eigenwerte“ auf beliebige Matrizen zu verallgemeinern. Es sei also $A \in \mathbb{R}^{n,m}$ eine Matrix, dann gibt es orthogonale Matrizen $U \in \mathbb{R}^{n,n}$ und $V \in \mathbb{R}^{m,m}$ sowie eine Matrix $\Sigma \in \mathbb{R}^{n,m}$, so dass

$$A = U\Sigma V^T \quad \text{mit} \quad \Sigma = \begin{bmatrix} \Sigma_r & 0_{r,m-r} \\ 0_{n-r,r} & 0_{n-r,m-r} \end{bmatrix} \in \mathbb{R}^{n,m}, \quad \Sigma_r = \operatorname{diag}(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{r,r}$$

gilt, wobei $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ und $r = \operatorname{Rang}(A)$. Diese Zerlegung ist die *Singulärwertzerlegung* von A (auch kurz *SVD* vom englischen *singular value decomposition*). Weiterhin heißen die Werte $\sigma_1, \ldots, \sigma_r$ die *Singulärwerte* und die Spalten von U bzw. V *linke* bzw. *rechte Singulärvektoren* von A.
Schreiben wir $U = [u_1, \ldots, u_n]$ und $V = [v_1, \ldots, v_m]$, so erhalten wir die Eigenschaft dass $\operatorname{Bild}(A) = \operatorname{span}\{u_1, \ldots, u_r\}$ und $\operatorname{Kern}(A) = \operatorname{span}\{v_{r+1}, \ldots, v_m\}$.
Die Singulärwertzerlegung kann auch geschrieben werden als

$$A = \sum_{j=1}^{r} \sigma_j u_j v_j^T,$$

also aus Matrizen der Form $\sigma_j u_j v_j^T$, wobei $\operatorname{Rang}\left(\sigma_j u_j v_j^T\right) = 1$ gilt. Damit definieren wir die Matrizen

$$A_k := \sum_{j=1}^{k} \sigma_j u_j v_j^T, \quad 1 \leq k \leq r,$$

dann gilt $\operatorname{Rang}(A_k) = k$ und bei Verwendung der Spektralnorm folgt

$$\|A - A_k\|_2 = \left\|\operatorname{diag}(\sigma_{k+1}, \ldots, \sigma_r)\right\|_2 = \sigma_{k+1}.$$

Der Satz von Schmidt-Mirsky besagt sogar, dass

$$\|A - A_k\|_2 \leq \|A - B\|_2$$

für jede Matrix $B \in \mathbb{R}^{n,m}$ mit $\mathrm{Rang}(B) = k$ gilt, also A_k die Lösung der Approximationsaufgabe

$$\min_{B \in \mathbb{R}^{n,m}, \mathrm{Rang}(B) \leq k} \|A - B\|_2$$

ist. Sie ist sogar die Lösung der Approximationsaufgabe

$$\min_{B \in \mathbb{R}^{n,m}, \mathrm{Rang}(B) \leq k} \|A - B\|_F$$

bezüglich der Frobenius-Norm mit dem Residuum

$$\|A - A_k\|_F = \sqrt{\sum_{j=k+1}^{r} \sigma_j^2}.$$

4.2 Konstruktion der Singulärwertzerlegung

Eine mögliche Konstruktionsweise der SVD ist folgendermaßen gegeben:

1) Setze $B := A^T A$, dann ist $B \in \mathbb{R}^{m,m}$ eine symmetrische Matrix. Wir bestimmen die Eigenwerte λ_i und orthonormale Eigenvektoren v_i von B, dann gilt

$$v_i^T B v_i = \lambda_i v_i^T v_i$$

und weiter

$$v_i^T B v_i = v_i^T A^T A v_i = (A v_i)^T A v_i \geq 0,$$

also sind alle Eigenwerte nichtnegativ. O.B.d.A. sei $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m \geq 0$. Da der Rang von B gleich dem Rang von$A$ ist, sind genau die ersten r Eigenwerte $\lambda_1, \ldots, \lambda_r$ positiv. Seien $v_1, \ldots, v_m$ die zugehörigen Eigenvektoren.

2) Für $j = 1, \ldots, r$ setzen wir

$$u_j := \frac{1}{\sqrt{\lambda_j}} A v_j.$$

3) Wir ergänzen die Vektoren $u_1, \ldots, u_r$ mit $n - r$ orthonormalen Vektoren $u_{r+1}, \ldots, u_n$ zu einer Orthonormalbasis von $\mathbb{R}^m$.

4) Wir bilden die Matrizen

$$U := [u_1, \ldots, u_n] \qquad \text{und} \qquad V := [v_1, \ldots, v_m]$$

aus den orthonormalen Vektoren $u_1, \ldots, u_n$ und $v_1, \ldots, v_m$ und weiter

$$\Sigma = [s_{ij}] \not\in \mathbb{R}^{n,m}, \quad s_{ij} = \begin{cases} \sqrt{\lambda_j} & i = j < r, \\ 0 & \text{sonst} \end{cases}.$$

An der Konstruktion kann man erkennen, dass die Singulärwerte von A eindeutig bestimmt sind, aber die Singulärvektoren dies nicht sind.

Beispiel

Wir berechnen die Singulärwertzerlegung der Matrix

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{3,2}.$$

Wir setzen

$$B = A^T A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}.$$

Dann hat B die Eigenwerte $\lambda_1 = 3$ und $\lambda_2 = 1$ mit zugehörigen Eigenvektoren

$$\tilde{v}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \text{und} \quad \tilde{v}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Diese normalisieren wir zu

$$v_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \quad \text{und} \quad v_2 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}.$$

Damit setzen wir

$$u_1 = \tfrac{1}{\sqrt{3}} A v_1 = \begin{bmatrix} \frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} \end{bmatrix} \quad \text{und} \quad u_2 = \tfrac{1}{\sqrt{1}} A v_2 = \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}$$

und wählen dazu

$$u_3 = \begin{bmatrix} -\frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{bmatrix}.$$

Damit haben wir

$$U := \begin{bmatrix} \frac{2}{\sqrt{6}} & 0 & -\frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \end{bmatrix}, \qquad V := \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

und

$$\Sigma := \begin{bmatrix} \sqrt{3} & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

4.3 Die Pseudoinverse

Wie bei der Lösung von Ausgleichsproblemen mit QR-Zerlegungen kann man hier nach einem x^* minimaler euklidischer Länge fragen, so dass

$$\|Ax^* - b\| = \min_{x \in \mathbb{R}^{m,1}} \|Ax - b\|$$

für gegebene $A \in \mathbb{R}^{n,m}$ und $b \in \mathbb{R}^{n,1}$. Dies liefert uns gerade die Pseudoinverse:
Sei $A = U\Sigma V^T$ eine Singulärwertzerlegung von A, dann ist durch

$$A^+ := V\Sigma^+ U^T, \quad \Sigma^+ = \begin{bmatrix} \Sigma_r^{-1} & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{m,n}$$

die *Pseudoinverse* von A gegeben.
Für diese gilt dann

$$A^+A = V\Sigma^+U^TU\Sigma V^T = I_m$$

und

$$AA^+ = U\Sigma V^TV\Sigma^+U^T = I_n,$$

also folgt dass $x^* = A^+b$ gilt. Hat A weiterhin vollen Rang und ist $n \geq m$, dann gilt

$$A^+ = (A^TA)^{-1}A^T,$$

und ist $m > n$ so haben wir

$$A^+ = A^T(AA^T)^{-1}.$$

Ist bei vollem Rang $A = QR$ die QR-Zerlegung von A, so haben wir damit

$$A^+ = (R^TQ^TQR)^{-1}R^TQ^T = R^{-1}(R^T)^{-1}R^TQ^T = R^{-1}Q^T,$$

also ist die Lösung dieses Problems mit der QR-Zerlegung identisch mit der Lösung mit der Singulärwertzerlegung.

4.4 Der Satz von Schmidt-Mirsky

Um mit geringeren Datenmengen rechnen zu können ist es in der Numerik von großem Nutzen mit Matrizen geringen Ranges zu agieren. Mithilfe der Singulärwertzerlegung erhalten wir dafür den folgenden Satz:

Satz (von Schmidt-Mirsky):

Für $A \in \mathbb{R}^{n,m}$ mit der Singulärwertzerlegung

$$A = U\Sigma V^T \quad \text{und Sigulärwerten} \quad \sigma_1 \leq \sigma_2 \leq ... \leq \sigma_r$$

besitzt für $k \in \mathbb{N}$, $k \leq \operatorname{Rang} A$ und $\Sigma_k = \operatorname{diag}(\sigma_1, \ldots, \sigma_k) \in \mathbb{R}^{k,k}$ die Matrix

$$A_k := U \begin{bmatrix} \Sigma_k & 0_{k,m-k} \\ 0_{n-k,k} & 0_{n-k,m-k} \end{bmatrix} V^T \in \mathbb{R}^{n,m}$$

Rang k und ist die beste Approximation an A mit Rang kleinergleich k, d.h.:

$$A_k = \underset{\operatorname{Rang} B = k,\, B \in \mathbb{R}^{n,m}}{\operatorname{argmin}} \|B - A\|$$

Kapitel 5

Interpolation

5.1 Polynominterpolation

Gegeben: n+1 Paare $(x_k, f_k) \in \mathbb{R} \times \mathbb{R}$, k = 0 , . . . , n aus Stützstellen und Stützwerten

Gesucht: Eine Funktion $f : [a, b] \to \mathbb{R}$ mit

$$f(x_k) = y_k$$

Dies hat nun jedoch unendlich viele Lösungen, deshalb müssen wir unseren Lösungsraum einschränken mit $f \in V$("Hypothesenraum"). Sei

$$V = \text{span}\{\varphi_0, ..., \varphi_n \mid \varphi_0, ..., \varphi_n \text{ linear unabhängig}\}$$

wobei

$$B := \{\varphi_0, ..., \varphi_n\}$$

eine Basis von V ist.

Typisch: Wähle $V := \Pi_n$ (Polynome vom Grad $\leq$ n) mit $\varphi_i := x^i$ für i=0 , . . . , n

Ansatz: Suche ein Polynom vom Grad höchstens n

$$f(x) := \alpha_0\varphi_0(x) + \alpha_1\varphi_1(x) + \ldots + \alpha_n\varphi_n(x) \in V$$

Bedingung: Definiere $\eta_i(f) = f(x_i)$. Es muss gelten

$$\begin{aligned}
f(x_0) = \eta_0(f) &= \alpha_0\eta_0(\varphi_0) + \alpha_1\eta_0(\varphi_1) + \ldots + \alpha_n\eta_0(\varphi_n) = f_0 \\
f(x_1) = \eta_1(f) &= \alpha_0\eta_1(\varphi_0) + \alpha_1\eta_1(\varphi_1) + \ldots + \alpha_n\eta_1(\varphi_n) = f_1 \\
&\vdots \\
f(x_n) = \eta_n(f) &= \alpha_0\eta_n(\varphi_0) + \alpha_1\eta_n(\varphi_1) + \ldots + \alpha_n\eta_n(\varphi_n) = f_n
\end{aligned}$$

Wenn wir nun also für die Matrix $A \in \mathbb{R}^{n+1,n+1}$ und den Vektor $b \in \mathbb{R}^{n+1}$ wählen als

$$A = (a_{i,j}) = (\eta_j(\varphi_i))_{i,j=0,\ldots,n} \text{ und } b = (f_i)_{i=1,\ldots,n},$$

so sehen wir, dass gerade der Vektor $\alpha = (\alpha_0, \ldots, \alpha_n)^T \in \mathbb{R}^{n+1}$ der das Gleichungssystem

$$A * \alpha = b$$

löst, uns die Lösung $f \in V$ liefert.

Lemma: Es existiert genau eine Lösung $f \in V$ bzw. $\alpha \in \mathbb{R}^{n+1}$, falls
$$\det(\mathrm{A}) = \det(\eta_j(\varphi_i)) \neq 0.$$
Diese Lösung nennen wir Interpolationspolynom. (Dies gilt offensichtlich, wenn alle Stützstellen paarweise verschieden sind. Wir berechnen die Lösung jedoch nicht auf diese Weise, da dies zu teuer und instabil ist.)
Außerdem gilt $\det(\mathrm{A}) = \prod\limits_{0 \leq i < j \leq n} (x_j - x_i)$

Idealfall: Im Idealfall gilt
$$(\eta_j(\varphi_i))_{i,j} = I_n,$$
also
$$\eta_j(\varphi_i) = \delta_{ij}.$$
Wir suchen nun also Polynome, für die das gilt. Wenn wir diese gefunden haben, so nennen wir die Menge
$$\{\varphi_i \mid i = 0, \ldots, n\}$$
die Interpolationsbasis.

5.1.1 Lagrange-Darstellung

Satz: Zu paarweise verschiedenen $x_0, \ldots, x_n \in [a, b]$ und $f_0, \ldots, f_n \in \mathbb{R}$ existiert genau ein Interpolationspolynom $f \in \Pi_n$, $x \mapsto f(x) = p(f \mid x_0, \ldots, x_n)(x)$. Es gilt für alle $x \in [a, b]$:
$$f(x) = \sum_{j=0}^{n} f_j l_j(x) \tag{5.1}$$
mit den Lagrange-Polynomen
$$l_j(x) = \prod_{\substack{k=0 \\ k \neq j}}^{n} \tfrac{x - x_k}{x_j - x_k} \in \Pi_n.$$
Die Menge $\{l_j \mid j = 0, \ldots, n\}$ bildet eine Basis von Π_n und es gilt $l_j(x_i) = \delta_{ij}$.

Bemerkung:

1. Das Auswerten des Polynoms benötigt O((2n+1)(n+1)) Multiplikationen und Additionen.
2. Falls man eine neue Stützstelle $x_{n+1} \in [a, b]$ hinzufügt, so muss man alle $l_j, j = 0, \ldots, n+1$ neu berechnen.

Fehlerabschätzung: Sei $f \in C^{n+1}([a, b])$. Für $i = 0, \ldots, n$ seien $x_i \in [a, b]$, $f_i = f(x_i)$ und $x \mapsto p_n(f \mid x_0, \ldots, x_n)(x) \in \Pi_n$ das zugehörige Interpolationspolynom. Definiere
$$l(x) := \prod_{j=0}^{n} (x - x_j).$$
Sei $x \in [a, b]$ beliebig, dann existiert $\xi \in conv(x_0, \ldots, x_n) \cup x$ (Also das kleinste Intervall [c,d] mit $x_0, \ldots, x_n \in [c, d]$ vereinigt mit x), so dass
$$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} l(x),$$
außerdem können wir ξ selten exakt berechnen, deshalb gilt außerdem
$$\|f - p_n\|_\infty \leq \frac{1}{(n+1)!} \|f^{(n+1)}\|_\infty \|l\|_\infty$$

5.1.2 Newtondarstellung

Idee: Bei der Lagrangedarstellung benötigt das Hinzufügen neuer Stützstellen eine komplett neue Berechnung aller $l_j, j = 0, \ldots, n+1$. Wir suchen nun nach einem Verfahren, bei dem man das nicht benötigt. Die Idee ist nun eine rekursive Darstellung zu wählen. Doch zu erst etwas Vorarbeit:

Definition: Seien die n+1 Stützpunkte $(x_0, f_0), \ldots, (x_n, f_n)$ an paarweise verschiedenen Stützstellen gegeben und seien $0 \leq j < j + k \leq n$. Die dividierten Differenzen der Ordnung k werden definiert durch die rekursive Formel

$$f[x_i] := f_i, i = 0, \ldots, n$$

$$f[x_j, \ldots, x_{j+k}] := \frac{f[x_{j+1}, \ldots, x_{j+k}] - f[x_j, \ldots, x_{j+k-1}]}{x_{j+k} - x_j}.$$

Außerdem definieren wir

$$f[\underbrace{x, \ldots, x}_{\substack{m+1 \\ mal}}] := \frac{1}{m!} f^{(m)}(x).$$

Definition: Zu gegebenen Stützstellen $x_0, \ldots, x_n$ definieren wir

$$N_k(x) := \prod_{i=0}^{k-1} (x - x_i), k = 1, \ldots, n$$

und $N_0(x) := 1$ als die Newton-Basispolynome.

Newtondarstellung: Seien die n+1 Stützpunkte $(x_0, f_0), \ldots, (x_n, f_n)$ an paarweise verschiedenen Stützstellen. Das eindeutige interpolierende Polynom vom Grad $\leq n$ auf den Stützpunkten ist gegeben durch

$$p_n(x) := \sum_{k=0}^{n} f[x_0, \ldots, x_k] N_k(x) \tag{5.2}$$

Bemerkung: Das Polynom in (1) ist das Gleiche wie das Polynom in (2)(Das Interpolationspolynom ist eindeutig!), nur die Form ist anders, denn man kann $p_n(x)$ in O(n) rekursiv berechnen mit

$$p_n(x) = p_{n-1} + f[x_0, \ldots, x_n] N_n(x)$$

dadurch liegt das Hinzufügen eines neuen Stützpunktes nur noch in O(n).

5.2 Hermite-Interpolation

Idee: Bei bestimmten Interpolationsverfahren kann es dazu kommen, dass die Funktion auf einzelnen Intervallen interpoliert wird, jedoch entstehen dabei teilweise Interpolationen, die nicht mehr differenzierbar sind. Um dies zu vermeiden, macht es Sinn, sich auch die Ableitung in den Stützstellen anzuschauen.

Satz: Seien $(x_0, f_0), \ldots, (x_n, f_n)$ Stützpunkte mit $x_0 = x_1 = \ldots = x_l \neq x_{l+1}$ und an der Stelle x_0 seien die Stützstellen vorgegeben durch $f_0 := f(x_0), f_1 := f^{(1)}(x_0), \ldots, f_l := f^{(l)}(x_0)$. Dann existiert genau ein Polynom vom Grad n, das heißt eine Lösung der Hermite-Interpolationsaufgabe:

$$x \mapsto p_n[f \mid x_0, \ldots, x_k](x) = p_n(x) = \sum_{k=0}^{n} f[x_0, \ldots, x_k] \prod_{j=0}^{k-1} (x - x_j)$$

Falls $x_0 = x_1 = \ldots = x_n$, so gilt

$$p_n(x) = \sum_{k=0}^{n} \frac{(x-x_0)^k}{k!} f^{(k)}(x_0).$$

Problem: Bei schlecht gewählten Stützstellen kann die Berechnung extrem auswendig werden(Betrachte Beispiel von Runge). Was ist also eine gute Wahl für die Stützstellen, damit $\|f - p_n\|_\infty$ möglichst klein bleibt?

Lösung: Wir wählen als Stützstellen die Nullenstellen der Tschebyscheff-Polynome

$$T_n : x \mapsto T_n(x) := \cos(n \arccos(x)),$$

also ergeben sich die Stützstellen

$$x_j = \frac{a+b}{2} + \frac{b-a}{2} \cos(\frac{2j+1}{2n}\pi).$$

5.3 Trigonometrische Interpolation

Idee: Viele physikalische Größen sind periodisch, wenn man sie also mit einem Polynom wie oben interpoliert, so wird der Fehler bei Grenzwertbetrachtung extrem groß. Also wollen wir periodische Funktionen benutzen um sie zu interpolieren. Als Konvention benutzen wir im Folgenden, dass die zu interpolierende Funktion f 2π-periodisch ist(falls nicht: skalieren!).
Bei der trigonometrischen Interpolation benutzt man die äquidistanten Stützstellen

$$x_j := \frac{2\pi j}{n}, j = 0, \ldots, n-1$$

die interessanterweise nicht nur einfach, sondern sogar optimal sind.

Definition: Wir definieren

$$\omega_j := e^{ix_j} = e^{2\pi i \frac{j}{n}}, j = 0, \ldots, n-1$$

als die n-ten Einheitswurzeln.

Definition: Sei

$$G_n := span\{\varphi_k(x) = e^{ikx} \mid k = 0, \ldots, n-1\}$$

die Menge der komplexen trigonometrischen Polynome.

Ansatz: Zu den Stützwerten $f_0, \ldots, f_{n-1}$ und den äquidistanten Stützstellen $x_0, \ldots, x_{n-1}$ ist eine Funktion $t_n \in G_n$ gesucht mit

$$t_n(x_i) = f_i, i = 0, \ldots, n.$$

Es gilt für $\alpha_0, \ldots, \alpha_n \in \mathbb{C}$

$$t_n(x) = \sum_{j=0}^{n-1} \alpha_j e^{ijx}.$$

Offenbar muss dann gelten

$$\begin{aligned} t_n(x_0) &= \alpha_0\omega_0^0 + \alpha_1\omega_0^1 + \ldots + \alpha_n\omega_0^n = f_0 \\ t_n(x_1) &= \alpha_0\omega_1^0 + \alpha_1\omega_1^1 + \ldots + \alpha_n\omega_1^n = f_1 \\ &\vdots \\ t_n(x_{n-1}) &= \alpha_0\omega_{n-1}^0 + \alpha_1\omega_{n-1}^1 + \ldots + \alpha_n\omega_{n-1}^n = f_{n-1} \end{aligned}$$

Satz: Seien die Stützwerte $f_0, \ldots, f_{n-1}$ und die äquidistanten Stützstellen $x_0, \ldots, x_{n-1}$ gegeben. Dann ist die Lösung des Interpolationspolynoms

$$t_n(x_k) = \sum_{j=0}^{n-1} \alpha_j \omega_k^j = \sum_{j=0}^{n-1} \alpha_j e^{2\pi i j \frac{k}{n}}, k = 0, \ldots, n-1$$

gegeben durch

$$\alpha_k = \frac{1}{n} \sum_{j=0}^{n-1} f_j \overline{\omega_k}^j = \frac{1}{n} \sum_{j=0}^{n-1} f_j e^{-2\pi i j \frac{k}{n}}, k = 0, \ldots, n-1$$

Definition: Sei $n \in \mathbb{N}$. Die Abbildung $F : \mathbb{C}^n \to \mathbb{C}^n$, $f \mapsto d$, die durch

$$d_k := (DFT(f))_k := \frac{1}{n} \sum_{j=0}^{n-1} f_j \overline{\omega_k}^j = \frac{1}{n} \sum_{j=0}^{n-1} f_j e^{-2\pi i j \frac{k}{n}}, k = 0, \ldots, n-1$$

gegeben ist, heißt die Diskrete Fourier-Transformation(DFT).
Die Inverse DFT ist durch

$$(IDFT(d))_k := \sum_{j=0}^{n-1} d_j e^{-2\pi i j \frac{k}{n}} = f_k, k = 0, \ldots, n-1$$

definiert.

Bemerkung: Für die DFT benötigt man $\mathrm{O}(n^2)$ Multiplikationen. Es existiert jedoch eine schnellere Variante, die sogenannte schnelle Fourier-Transformation(fast Fourier transform)(FFT), die nur $\mathrm{O}(n \log n)$ Multiplikationen benötigt.

Lemma: Sei n=2m und $DFT(f_0, \ldots, f_{n-1}) = (d_0, \ldots, d_{n-1})$. Dann gilt

$$d_{2l} = \frac{1}{m} \sum_{k=0}^{m-1} \frac{1}{2}(f_k + f_{k+m}) e^{-2\pi i k \frac{l}{m}} = DFT(\frac{1}{2}(f_0 + f_m), \ldots, \frac{1}{2}(f_{m-1} + f_{2m-1}))$$

bzw.

$$\begin{aligned} d_{2l+1} &= \frac{1}{m} \sum_{k=0}^{m-1} \frac{1}{2}(f_k - f_{k+m}) e^{-2\pi i \frac{k}{m}} e^{-2\pi i k \frac{l}{m}} \\ &= DFT(\frac{1}{2}(f_0 + f_m) e^{-2\pi i \frac{0}{m}}, \ldots, \frac{1}{2}(f_{m-1} + f_{2m-1}) e^{-2\pi i \frac{m-1}{m}}) \end{aligned}$$

Beachte, dass die DFTs auf der rechten Seite nur DFTs aus Vektoren halber Länge sind. Wir können nun also eine DFT der Länge $2m$ berechnen, indem wir zwei DFTs der Länge m berechnen und dann geeignet zusammenfügen. Dies ist nun die sogenannte schnelle Fourier-Transformation mit der Komplexität $\mathrm{O}(n \log n)$.

Kapitel 6

Regression

6.1 Lineare Regression

Idee: Die lineare Regression ist ein Verfahren, das benutzt wird, wenn man untersuchen möchte, wie sich eine beobachtete abhängige Variable durch eine oder mehrere unabhängigen Variablen erklären lässt. In der Schule hat man die Idee dieses Konzepts benutzt, als man beispielsweise im Physikunterricht eine Gerade durch Messpunkte gezeichnet hat, die möglichst nah an allen diesen Messpunkten liegt. Wir wollen nun das mathematische Modell dahinter betrachten.

Gegeben: Es sind Messpunkte $x_j = (x_1^{(j)}, \ldots, x_n^{(j)}), j = 1, \ldots, m$ und Messwerte $y_j, j = 1, \ldots, m$ gegeben.

Gesucht: Wir suchen eine Funktion $f : \mathbb{R}^n \to \mathbb{R}$ mit $f(x_j) \approx y_j$, $j = 1, \ldots, m$. Bei der linearen Regression soll f die Form haben

$$f = \sum_{k=1}^{m} a_k x_k (+a_0)$$

für $a_k \in \mathbb{R}$. Außerdem wollen wir f so wählen, dass

$$\frac{1}{n} \sum_{j=1}^{m} \|y_j - f(x_j)\|^2$$

minimal wird.

Alternativ: Alternativ kann das lineare Regressionsproblem auch so beschrieben werden, dass $A \in \mathbb{R}^{m,n}$ und $b \in \mathbb{R}^m$ gegeben sind und $x \in \mathbb{R}^n$ gesucht ist, sodass $\|Ax - b\|_2^2$ minimal wird.

Satz: Für $A \in \mathbb{R}^{m,n}$ und $b \in \mathbb{R}^m$ mit Rang(A)=n und $n \geq m$ hat das lineare Regressionsproblem $\|Ax - b\|_2^2$ soll minimal werden genau eine Lösung $\tilde{x}$ =argmin$\{(\|Ax - b\|^2) |\ x \in \mathbb{R}^n\}$. Diese erfüllt die "Gaußsche Normalengleichung"$A^T A\tilde{x} = A^T b$.

Berechnung: Um diese Lösung zu finden benutzt man häufig die QR-Zerlegung einer Matrix. Das heißt, sind zu einer Matrix $A \in \mathbb{R}^{m,n}$ mit vollem Rang, wobei $n \leq m$, eine orthogonale Matrix $Q \in \mathbb{R}^{m,m}$ und eine verallgemeinerte obere Dreiecksmatrix R der Form

$$R = \begin{bmatrix} \tilde{R} \\ 0 \end{bmatrix} \in \mathbb{R}^{m,n}, \tilde{R} = \begin{bmatrix} * & \cdots & * \\ & \ddots & \vdots \\ & & * \end{bmatrix} \in \mathbb{R}^{n,n}$$

bekannt, sodass

$$A = QR$$

gilt. Aufgrund der Eigenschaften von orthogonalen Matrizen erhalten wir nun

$$\|Ax - b\|^2 = \|QRx - b\|^2 = \|Q(Rx - Q^T b)\|^2 = \|Rx - Q^T b\|.$$

Sei nun

$$R := \begin{bmatrix} \tilde{R} \\ 0 \end{bmatrix}, \text{ und } Q^T b := \begin{bmatrix} c \\ d \end{bmatrix},$$

dann folgt somit

$$\|Ax - b\|^2 = \left\| \begin{bmatrix} \tilde{R} \\ 0 \end{bmatrix} x - \begin{bmatrix} c \\ d \end{bmatrix} \right\|^2 = \|\tilde{R}x - c\|^2 + \|d\|^2,$$

also müssen wir nur noch $\|\tilde{R}x - c\|^2$ minimieren.

Kapitel 7

Numerische Integration

7.1 Einleitung

Ziel: Berechne für stetige Funktion $f : [a, b] \to \mathbb{R}$ das Integral

$$\mathcal{I}(f) := \int_a^b f(x)\,dx \quad \in \mathbb{R}.$$

Ansatz: Wähle Stützstellen $a \leq t_0 < t_1 < \cdots < t_n \leq b$ mit $f(t_k) = f_k \in \mathbb{R}$. Die allgemeine Quadraturformel ist definiert als

$$\mathcal{Q}_n(f) := \sum_{k=0}^{n} \omega_k \cdot f_k.$$

Finde also gute Quadraturgewichte $\omega_0, \ldots, \omega_n$. $\Rightarrow$ Wann ist meine Quadraturformel gut?

- Fehler $|\mathcal{I}(f) - \mathcal{Q}_n(f)|$ minimal
- hoher **Exaktheitsgrad**: $\mathcal{Q}_n$ exakt von Ordnung $q \in \mathbb{N}$, falls $\mathcal{I}(p) = \mathcal{Q}_n(p)$ für alle Polynome $p \in \mathbb{R}[t]_{\leq q}$. Es reicht aus, die Gleichheit $\mathcal{I}(p) = \mathcal{Q}_n(p)$ für Basispolynome $p_0, p_1, \ldots, p_q$ zu überprüfen. Dann lassen sich alle Polynome $p \in \mathbb{R}[t]_{\leq q}$ schreiben als

  $$p = \sum_{k=0}^{q} \alpha_k \cdot p_k.$$

  Dann gilt auch

  $$\mathcal{Q}_n(p) = \sum_{k=0}^{q} \omega_k \cdot p(t_k) = \sum_{k=0}^{q} \omega_k \cdot \left(\sum_{i=0}^{q} \alpha_i \cdot p_i(t_k) \right) = \sum_{i=0}^{q} \alpha_i \cdot \int_a^b p_i(s)\,ds = \mathcal{I}(p).$$

7.2 Newton-Cotes

Idee ist Integration des Interpolationspolynoms:

$$f(x) \approx P_n(x)\ \forall x \in [a,b] \quad \Rightarrow \quad \int_a^b f(x)\,dx \approx \int_a^b P_n(x)\,dx.$$

Vorgehen:

1. Finde äquidistante Stützstellen $a = t_0 < t_1 < \cdots < t_n = b$.
2. Berechne Lagrange-Polynome

$$P_n(t) = \sum_{k=0}^{n} f(t_k) \cdot l_k(t) \quad \text{mit} \quad l_k(t) := \prod_{\substack{j=0 \\ j\neq k}}^{n} \frac{t-t_j}{t_k-t_j}.$$

3. Quadraturgewichte $\omega_k := \int_a^b l_k(x)\,dx$
4. Nun ist die Quadraturformel gegeben durch

$$\mathcal{Q}_n(f) = \sum_{k=0}^{n} f(t_k) \cdot \omega_k = \sum_{k=0}^{n} f(t_k) \cdot \int_a^b l_k(x)\,dx = \int_a^b P_n(x)\,dx.$$

Fehlerabschätzung: Falls f $(q+1)$mal stetig differenzierbar ist, so gilt:

$$\begin{aligned} |\mathcal{I}(f) - \mathcal{Q}_n(f)| &\leq \frac{b-a}{(q+1)!} \cdot \max_{t\in[a,b]} \left| \prod_{j=0}^{q} (t-t_j) \right| \cdot ||f^{(q+1)}||_{\sup} \\ &\leq \frac{(b-a)^{q+2}}{(q+1)!} \cdot ||f^{(q+1)}||_{\sup} \end{aligned}$$

Exaktheitsgrad: Newton-Cotes exakt für alle Lagrange-Basispolynome $l_0, \ldots, l_n$, also auch für beliebige Polynome $p \in \mathbb{R}[t]_{\leq n}$.

Spezialfälle: Für $a = 0$ und $b = 1$ wurden Quadraturgewichte bereits berechnet:

n	Name	Stützstellen	Gewichte
1	Trapezregel	$0, 1$	$\frac{1}{2}, \frac{1}{2}$
2	Simpsonregel	$0, \frac{1}{2}, 1$	$\frac{1}{6}, \frac{4}{6}, \frac{1}{6}$
3	$\frac{3}{8}$-Regel	$0, \frac{1}{3}, \frac{2}{3}, 1$	$\frac{1}{8}, \frac{3}{8}, \frac{3}{8}, \frac{1}{8}$
4	Milne-Regel	$0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1$	$\frac{7}{90}, \frac{32}{90}, \frac{12}{90}, \frac{32}{90}, \frac{7}{90}$

Beobachtung: Gewichte sind symmetrisch, d.h. $\omega_k = \omega_{n-k}$ für $k = 0, \ldots, \lfloor \frac{n}{2} \rfloor$.

Weitere Verbesserung: Ist $n \geq 7$, so treten negative Gewichte ω_k auf. Wir wollen aber den Auslöschungseffekt vermeiden. Also: Zerlege $[a,b]$ in $a = z_0 < z_1 < \cdots < z_m = b$ (äquidistant). Berechne nun für jedes Intervall $[z_i, z_{i+1}]$ mit dem oben beschriebenen Algorithmus $\mathcal{Q}_n\left(f|_{[z_i,z_{i+1}]}\right)$. Dann ist

$$1 \int_a^b f(x)\,dx = \sum_{i=0}^{m-1} \int_{z_i}^{z_{i+1}} f(x)\,dx \approx \sum_{i=0}^{m-1} \mathcal{Q}_n\left(f|_{[z_i,z_{i+1}]}\right).$$

7.3 Gauß-Quadratur

Idee: Nutze orthogonale Polynome, um möglichst hohen Exaktheitsgrad zu erreichen. Beliebige stetige Funktionen lassen sich dann durch Polynome approximieren.

Orthogonale Polynome: Definiere für stetige Gewichtsfunktion $w : G \to [0, \infty[, G \subseteq \mathbb{R}$ (nicht Nullfunktion) ein Skalarprodukt

$$\langle \cdot, \cdot \rangle_w : C^0(G, \mathbb{R}) \times C^0(G, \mathbb{R}) \to \mathbb{R}, \quad (f, g) \mapsto \int_a^b f(x) \cdot g(x) \cdot w(x)\, dx.$$

Wir wollen für $\mathbb{R}[t]_{\leq n}$ eine ONB bezüglich $\langle \cdot, \cdot \rangle_w$ finden. Wende dazu Gram-Schmidt auf Basis $\{1, t, t^2, \ldots, t^n\}$ an:

$$q_j(t) := t^j - \sum_{k=0}^{j-1} \frac{\langle t^j, q_k \rangle_w}{\langle q_k, q_k \rangle_w} \cdot q_k(t).$$

Dann gelten:

- $q_0, q_1, \ldots, q_n$ sind orthogonal bzgl. $\langle \cdot, \cdot \rangle_w$, es gilt sogar: $\langle u, q_n \rangle_w = 0$ für alle $u \in \mathbb{R}[t]_{<n}$.
- Basis $\{q_0, \ldots, q_n\}$ eindeutig mit der Eigenschaft, dass der Leitkoeffizient jeweils 1 ist.
- Jedes q_j besitzt genau j Nullstellen in G (offenbar war G entsprechend gewählt).

Gauß-Quadratur für Polynome: Ziel ist Integration von Polynomen vom Grad $2n - 1$ mit Quadraturformeln vom Grad n. Gehe folgendermaßen vor:

1. Berechne ONB $q_0, q_1, \ldots, q_n$ wie oben (Leitkoeffizient jeweils 1).
2. Ermittle Nullstellen $t_1 < \cdots < t_n$ von q_n
3. Lagrange-Basispolynome $l_k(t) = \prod_{j=1, j \neq k}^{n} \frac{t - t_j}{t_k - t_j}$ bezüglich $t_1 \ldots t_n$.
4. Aus Gewichten $\omega_k := \int_a^b l_k(s) \cdot w(s)\, ds > 0$ ergibt sich $\mathcal{Q}_n(p) := \sum_{k=1}^{n} \omega_k \cdot p(t_k)$

Exaktheitsgrad ist genau $2n - 1$:
1. Für $p \in \mathbb{R}[t]_{\leq 2n-1}$ gibt es eindeutige Polynome $u, r \in \mathbb{R}[t]_{\leq n-1}$, sodass $p = q_n \cdot u + r \in \mathbb{R}[t]_{\leq 2n-1}$ ist (Polynomdivision). Dann gilt einerseits:

$$\begin{aligned}
\mathcal{Q}_n(p) &= \sum_{k=1}^{n} \omega_k \cdot p(t_k) = \sum_{k=1}^{n} \omega_k \cdot (\underbrace{q_n(t_k)}_{=0} \cdot u(t_k) + r(t_k)) \\
&= \sum_{k=1}^{n} \omega_k \cdot r(t_k) = \mathcal{Q}_n(r).
\end{aligned}$$

Andrerseits ist

$$\begin{aligned}
\mathcal{I}(p \cdot w) &= \int_a^b p(t) \cdot w(t)\, dt = \int_a^b q_n(t) \cdot u(t) \cdot w(t)\, dt + \int_a^b r(t) \cdot w(t)\, dt \\
&= \langle q_n, u \rangle_w + \mathcal{I}(r \cdot w) = 0 + \mathcal{I}(r \cdot w).
\end{aligned}$$

Wie oben lässt sich zeigen, dass wegen $\mathrm{Grad}(r) \leq n-1$ auch $\mathcal{Q}_n(r) = \mathcal{I}(r \cdot w)$ ist und es folgt $\mathcal{Q}_n(p) = \mathcal{I}(p \cdot w)$.

2. Für Polynome vom Grad $2n$ funktioniert das Verfahren nicht mehr:
Betrachte beliebige n Stützstellen $a \leq t_1 < \cdots < t_n \leq b$ sowie Gewichte $\omega_1, \ldots, \omega_n$. Dann gibt es ein Polynom

$$p(t) := \prod_{k=1}^{n} (t - t_k)^2 \quad \in \mathbb{R}[t]_{\leq 2n},$$

sodass gilt

$$\mathcal{Q}_n(p) = \sum_{k=1}^{n} \omega_k \cdot \underbrace{p(t_k)}_{=0} = 0 < \int_a^b \underbrace{p(x)}_{\geq 0} \cdot \underbrace{w(x)}_{>0} \, dx = \mathcal{I}(p).$$

Das heißt eine Quadraturformel vom Grad n ist höchstens exakt vom Grad $2n-1$.

Spezialfälle:

	Gewichtsfunktion	Intervall	Orthogonale Polynome
Legendre	$w(x) = 1$	$[-1, 1]$	$q_j(x) = \frac{1}{2^j \cdot j!} \cdot \frac{d^j}{dx^j}[(x^2-1)^j]$
Tschebyschow	$w(x) = \frac{1}{\sqrt{1-x^2}}$	$[-1, 1]$	$q_j(x) = \cos(j \cdot \arccos(x))$
Hermite	$w(x) = e^{-x^2}$	$]-\infty, \infty[$	$q_j(x) = (-1)^j \cdot e^{x^2} \cdot \frac{d^j}{dx^j} e^{-x^2}$
Laguerre	$w(x) = e^{-x}$	$[0, \infty[$	$q_j(x) = \frac{e^x}{j!} \cdot \frac{d^j}{dx^j}[x^j \cdot e^{-x}]$

Stetige Funktionen: Für Quadraturformeln $\mathcal{Q}_j$ vom Grad j betrachte Quadraturgewichte $\left(\omega_k^{(j)}\right)_{k=1\ldots j}$ sowie Stützstellen $\left(t_k^{(j)}\right)_{k=1\ldots j}$. Dann gilt für alle stetigen Funktionen f

$$\lim_{j\to\infty} \mathcal{Q}_j(f) = \lim_{j\to\infty} \sum_{k=1}^{j} f\left(t_k^{(j)}\right) \cdot \omega_k^{(j)} = \int_a^b f(x) \cdot w(x) \, dx.$$

Nutze Satz von Weierstraß: Für $\varepsilon > 0$ gibt es $n \in \mathbb{N}$ und $p \in \mathbb{R}[t]_{\leq n}$, sodass $||f - p||_{\sup} < \varepsilon$ ist auf $[a, b]$. Dann geht aber auch $|\mathcal{I}(f) - \mathcal{Q}_n(f)|$ gegen 0.

Kapitel 8

Iterationsverfahren

8.1 Einführung

Iterationsverfahren dienen der schrittweisen Annäherung an eine Lösung durch wiederholte Anwendung der gleichen Rechenschritte. Der große Vorteil besteht darin, dass es möglich ist, bereits mit wenig Rechenaufwand eine akzeptable Lösung zu erhalten und dann das Verfahren abzubrechen.

8.2 Konvergenz

8.2.1 Eigenschaften von Problemen und Algorithmen

Es gibt vier Kriterien der Problem- und Algorithmusbewertung.

Kondition: $\|f(\widetilde{x}) - f(x)\|$

• "Wie stark schwankt das Problem bei Störungen in der Eingabe?"

Stabilität: $\|\widetilde{f}(\widetilde{x}) - \widetilde{f}(x)\|$

• "Wie stark schwankt der Algorithmus bei Störungen in der Eingabe?"

Konsistenz: $\|\widetilde{f}(x) - f(x)\|$

• "Wie gut löst der Algorithmus das Problem bei exakter Eingabe?"

Konvergenz: $\|\widetilde{f}(\widetilde{x}) - f(x)\|$

• "Wie gut löst der Algorithmus das Problem bei Störungen in der Eingabe, also in realer Umgebung?"

8.2.2 Konvergenzordnung

Definition:

Sei $\|\cdot\|$ eine Norm auf dem $\mathbb{K}$-Vektorraum V, $x^* \in V$ und $(x_n) \subset V$ mit $(x_n) \to x^*$ eine gegen x^* konvergente Folge in V.

1. (x_n) konvergiert (mindestens) **linear**, wenn es ein q $<$ 1 und ein $n_0 \in \mathbb{N}$ gibt, sodass

$$\|x_{n+1} - x^*\| \leq q \cdot \|x_n - x^*\| \quad \forall n \geq n_0$$

2. (x_n) konvergiert (mindestens) **superlinear**, wenn $\exists n_0 \in \mathbb{N}, (\epsilon_n) \subset \mathbb{R}^+$ mit $\epsilon_n \to 0$, sodass

$$\|x_{n+1} - x^*\| \leq \epsilon_n \cdot \|x_n - x^*\| \quad \forall n \geq n_0$$

3. (x_n) besitzt (mindestens) Konvergenzordnung α, wenn $\exists c > 0$ und (x_n) mit

$$\|x_{n+1} - x^*\| \leq c \cdot \|x_n - x^*\|^\alpha \quad \forall n \geq n_0$$

   Im Fall $\alpha = 2$ spricht man von **quadratischer Konvergenz**, bei $\alpha = 3$ von **kubischer Konvergenz**.

4. Ein Iterationsverfahren

$$x_{n+1} := \phi(x_n) \quad n \geq 0 \; mit \; x_0 \in V, \phi : V \to V$$

   besitzt bezüglich des Fixpunktes von ϕ, also $x^* = \phi(x^*)$, lokal (mindestens) die Konvergenzordnung $\alpha \geq 1$, wenn $\exists U(x^*)$, sodass jede Iterationsfolge mit x_0 aus $U(x^*)$ gegen x^* konvergiert und (x_n) die Konvergenzordnung (mindestens) α besitzt.

8.3 Lösen Linearer Gleichunssysteme: Splitting - Verfahren

8.3.1 Konzept

Ziel: Löse $Ax = b$
Idee: Forme das Problem um in

$$\phi(x) := (I - Q^{-1}A)x + Q^{-1}b = x,$$

wobei Q eine beliebige reguläre Matrix ist, sodass das Verfahren konvergiert.

Definiere nachfolgend:

$$M := I - Q^{-1}A \quad und \quad N := Q^{-1}$$

Satz:

Iterationsverfahren $x_{k+1} = Mx_k + Nb$ konvergiert für alle Startwerte $x_0 \Leftrightarrow$ Spektralradius $\rho(M) := max_j|\lambda_j(M)| < 1$

Fehlerabschätzung:

$$\|x - x_k\| \leq \frac{\|M\|^k}{1 - \|M\|}\|x_1 - x_0\|$$

8.3.2 Verschiedene Verfahren

Betrachte das Splitting

$$A = L + D + R$$

mit L untere Dreiecksmatrix, D Diagonalmatrix, R obere Dreiecksmatrix.

Verfahren:

Richardson-Verfahren:	$Q = I$
Jacobi-Verfahren:	$Q = D$
(auch Gesamtschrittverfahren)	
Gauß-Seidel-Verfahren:	$Q = D + L$
(auch Einzelschrittverfahren)	
SOR-Verfahren:	$Q = \frac{1}{\omega} D + L$ mit $\omega \in (0, 2)$
(successive over relaxation)	

Beim SOR-Verfahren wählt man $\omega > 1$ um die Konvergnz zu beschleunigen und $\omega < 1$ um leicht divergierende Lösungen zu stabilisieren.

8.3.3 Konvergenzen

Das Jacobi-Verfahren und Gauß-Seidel-Verfahren konvergieren für jeden Startwert (linear), wenn A strikt diagonaldominant. ist
Das Gauß-Seidel-Verfahren konvergiert für jeden Startwert (linear), wenn A spd-Matrix ist.
Das SOR-Verfahren konvergiert mit $\omega \in (0, 2)$ für jede spd-Matrix.

8.3.4 Beispiele:

Man löse $\begin{pmatrix} 6 & 2 \\ 2 & 6 \end{pmatrix} x = \begin{pmatrix} 10 \\ -2 \end{pmatrix}$. (Lösungsvektor $\tilde{x} = \begin{pmatrix} 2 \\ -1 \end{pmatrix}$)
Wir erhalten

Richardson: $Q_R = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad M_R = I_2 - Q_R^{-1} A = \begin{pmatrix} -5 & -2 \\ -2 & -5 \end{pmatrix}$

Jacobi: $Q_J = \begin{pmatrix} 6 & 0 \\ 0 & 6 \end{pmatrix}, \quad M_J = I_2 - Q_J^{-1} A = \begin{pmatrix} 0 & -\frac{1}{3} \\ -\frac{1}{3} & 0 \end{pmatrix}$

Gauß-Seidel: $Q_G = \begin{pmatrix} 6 & 0 \\ 2 & 6 \end{pmatrix}, \quad M_G = I_2 - Q_G^{-1} A = \begin{pmatrix} 0 & -\frac{1}{3} \\ 0 & -\frac{1}{9} \end{pmatrix}$

SOR($\omega = \frac{2}{3}$): $Q_S = \begin{pmatrix} 4 & 0 \\ 2 & 4 \end{pmatrix}, \quad M_S = I_2 - Q_S^{-1} A = \begin{pmatrix} -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{4} & -\frac{1}{4} \end{pmatrix}$

Wir wissen bereits, dass sowohl das Jacobi- als auch das Gauß-Seidel-Verfahren konvergieren, da A strikt diagonaldominant ist. Berechnung des Spektralradius liefert:

Richardson: $p_R = det \begin{pmatrix} \lambda+5 & 2 \\ 2 & \lambda+5 \end{pmatrix} = (\lambda 5+)^2 - 4$

$\lambda \in \{-3, 7\} \Rightarrow$ Verfahren konvergiert nicht

Jacobi: $p_J = det \begin{pmatrix} \lambda & \frac{1}{3} \\ \frac{1}{3} & \lambda \end{pmatrix}$

$\lambda \in \left\{\pm\frac{1}{3}\right\} \Rightarrow$ Verfahren konvergiert

Gauß-Seidel: $p_G = \begin{pmatrix} \lambda & \frac{1}{3} \\ 0 & \lambda+\frac{1}{9} \end{pmatrix}$

$\lambda \in \left\{0, \frac{1}{9}\right\} \Rightarrow$ Verfahren konvergiert

SOR: $p_S = \begin{pmatrix} \lambda+\frac{1}{2} & \frac{1}{2} \\ -\frac{1}{4} & \lambda+\frac{1}{4} \end{pmatrix}$

$\lambda \in \{0,375 \pm 0,331\} \Rightarrow \lambda < 1 \Rightarrow$ Verfahren konvergiert

Für den Startvektor $x_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ erhalten wir

Jacobi: $x_1 = \begin{pmatrix} 0 & -\frac{1}{3} \\ -\frac{1}{3} & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} \frac{1}{6} & 0 \\ 0 & \frac{1}{6} \end{pmatrix} \begin{pmatrix} 10 \\ -2 \end{pmatrix} = \begin{pmatrix} \frac{5}{3} \\ -\frac{1}{3} \end{pmatrix}$

$x_2 = \begin{pmatrix} 0 & -\frac{1}{3} \\ -\frac{1}{3} & 0 \end{pmatrix} \begin{pmatrix} \frac{5}{3} \\ -\frac{1}{3} \end{pmatrix} + \begin{pmatrix} \frac{1}{6} & 0 \\ 0 & \frac{1}{6} \end{pmatrix} \begin{pmatrix} 10 \\ -2 \end{pmatrix} = \begin{pmatrix} \frac{1}{9} \\ -\frac{5}{9} \end{pmatrix} + \begin{pmatrix} \frac{5}{3} \\ -\frac{1}{3} \end{pmatrix} = \begin{pmatrix} \frac{16}{9} \\ -\frac{8}{9} \end{pmatrix} \approx \begin{pmatrix} 2 \\ -1 \end{pmatrix}$

Gauß-Seidel: $x_1 = \begin{pmatrix} 0 & -\frac{1}{3} \\ 0 & -\frac{1}{9} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} \frac{1}{6} & 0 \\ -\frac{1}{18} & \frac{1}{6} \end{pmatrix} \begin{pmatrix} 10 \\ -2 \end{pmatrix} = \begin{pmatrix} \frac{5}{3} \\ -\frac{8}{9} \end{pmatrix}$

$x_2 = \begin{pmatrix} 0 & -\frac{1}{3} \\ 0 & -\frac{1}{9} \end{pmatrix} \begin{pmatrix} \frac{5}{3} \\ -\frac{8}{9} \end{pmatrix} + \begin{pmatrix} \frac{1}{6} & 0 \\ -\frac{1}{18} & \frac{1}{6} \end{pmatrix} \begin{pmatrix} 10 \\ -2 \end{pmatrix} = \begin{pmatrix} \frac{8}{27} \\ -\frac{8}{81} \end{pmatrix} + \begin{pmatrix} \frac{5}{3} \\ -\frac{8}{9} \end{pmatrix} = \begin{pmatrix} \frac{53}{27} \\ -\frac{80}{81} \end{pmatrix} \approx \begin{pmatrix} 2 \\ -1 \end{pmatrix}$

SOR: $x_1 = \begin{pmatrix} -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{4} & -\frac{1}{4} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} \frac{1}{4} & 0 \\ -\frac{1}{8} & \frac{1}{4} \end{pmatrix} \begin{pmatrix} 10 \\ -2 \end{pmatrix} = \begin{pmatrix} \frac{5}{2} \\ -\frac{7}{4} \end{pmatrix}$

$x_2 = \begin{pmatrix} -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{4} & -\frac{1}{4} \end{pmatrix} \begin{pmatrix} \frac{5}{2} \\ -\frac{7}{4} \end{pmatrix} + \begin{pmatrix} \frac{1}{4} & 0 \\ -\frac{1}{8} & \frac{1}{4} \end{pmatrix} \begin{pmatrix} 10 \\ -2 \end{pmatrix} = \begin{pmatrix} -\frac{3}{8} \\ \frac{17}{16} \end{pmatrix} + \begin{pmatrix} \frac{5}{2} \\ -\frac{7}{4} \end{pmatrix} = \begin{pmatrix} \frac{17}{8} \\ -\frac{11}{16} \end{pmatrix} \approx \begin{pmatrix} 2 \\ -1 \end{pmatrix}$

8.4 Lösen Linearer Gleichunssysteme: Gradienten - Verfahren

8.4.1 "Klassisches"Gradientenverfahren

Für A $\in \mathbb{R}^{n\times n}$ spd ist das Lösen von $Ax = b$ äquivalent zur Minimierung der quadratischen Form $Q : \mathbb{R}^n \to \mathbb{R}$

$$Q(x) = \frac{1}{2}\langle x, Ax\rangle \ - \langle x, b\rangle$$

Im klassischen Gradientenverfahren wählen wir als Laufrichtung $-\bigtriangledown Q(x) = b - Ax$, das Negative des Gradienten von Q, da dieses in die Richtung des steilsten Abstiegs zeigt. Wir erhalten als Iterationsformel

$$x_{k+1} := x_k + t_k v_k$$

mit Schrittweite

$$t_k := \frac{\langle v_k, v_k\rangle}{\langle Av_k, v_k\rangle}$$

und Laufrichtung

$$v_k := b - Ax_k.$$

Für die Fehlerabschätzung gilt

$$\|x - x_k\|_A \leq (\frac{cond(A) - 1}{cond(A) + 1})^k \cdot \|x - x_0\|_A,$$

wobei $cond(A)$ wie auf Seite 5 die spektrale Kondition $cond(A) = \|A\|_2 \cdot \|A^{-1}\|_2 = \sqrt{\lambda_{max}/\lambda_{min}}$ bezeichnet.

8.4.2 Konjugiertes Gradientenverfahren (CG)

Problem: Die Richtung des steilsten Abstiegs ist i.A. nicht die Richtung zum Minimum. Idee: Wähle neue Suchrichtung orthogonal zum Span der vorherigen Suchrichtungen. Dadurch konvergiert das Verfahren im $dim(A)$-ten Schritt. Für eine Orthogonalisierung benutzen wir das **Gram-Schmidt-Verfahren**.

Für die Fehlerabschätzung gilt

$$\|x - x_k\|_A \leq 2(\frac{\sqrt{cond(A)} - 1}{\sqrt{cond(A)} + 1})^k \cdot \|x - x_0\|_A,$$

8.4.3 Vergleich

Das Konjugierte Gradientenverfahren ist das schnellste Iterationsverfahren für spd-Matrizen. Zudem sinkt der Fehler monoton. Allerdings müssen die vorherigen Laufrichtungen in irgendeiner Form gespeichert werden.

8.5 Lösen Nicht-Linearer Gleichungssysteme: Fixpunktiteration

8.5.1 Konzept

Wir formen das Problem in **Fixpunktgleichung**, also eine Gleichung der Form

$$\phi(x) = x$$

um und konstruieren mit Hilfe der Iteration

$$x_{k+1} = \phi(x_k),\ k = 0, 1, 2...$$

für einen gegebenen Startwert x_0 eine gegen den Fixpunkt x* (mit $\phi(x^*) = x^*$) konvergierende Folge (x_k). Dieser Fixpunkt x* soll dabei das ursprüngliche Problem lösen.

Man definiere analog zu Analysis die Begriffe *Kontrakion* und *Lipschitz-Konstante* und den *Banachschen Fixpunktsatz.*

8.5.2 Sätze

Satz:

Ist $\phi : I \to \mathbb{R}, \phi$ stetig diffbar, und Iein abgeschlosses Intervall so gilt

$$\sup_{x,y \in I} \frac{|\phi(x) - \phi(y)|}{|x - y|} = \sup_{z \in I} |\phi'(z)| =: L < \infty$$

Ist $L < 1$, handelt es sich um eine Kontraktion.
Die Aussage gilt analog für mehrdimensionale Räume: Falls $\phi : D \to D,\ D \subset \mathbb{R}^n$ abgeschlossen, zusammenhängend, konvex und ϕ stetig differenzierbar mit

$$\sup_{x \in D} \|\phi'(x)\|_{Op} =: L < 1,$$

so ist ϕ eine Kontraktion mit Lipschitz-Konstante L.

Satz:

Sind die Bedingungen des Banachschen Fixpunktsatzes erfüllt, also insbesondere eine Kontraktion ϕ auf einer abgeschlossenen, zusammenhängenden, konvexen Menge I mit Lipschitz-Konstant $L < 1$ vorhanden, dann konvergiert die Fixpunktiteration für jeden Startwert $x_0 \in I$ gegen den Fixpunkt x^* mit

$$|x_{k+1} - x_k| \leq L|x_k - x_{k-1}| \ \ und \ \ |x_k - x^*| \leq \frac{L^k}{1-L}|x_1 - x_0|$$

8.6 Lösen Nicht-Linearer Gleichungssysteme: Newton-Verfahren

8.6.1 Konzept

Das Newton-Verfahren ist eine spezielle Fixpunktiteration. Unter der Voraussetzung $F'(x)$ ist regulär hat

$$\phi(x) = x - (F'(x))^{-1}F(x)$$

den Fixpunkt x^* mit $F(x^*) = 0$.
Wir erhalten die Iterationsvorschrift

$$x_{k+1} = x_k - (F'(x_k))^{-1}F(x_k).$$

8.6.2 Vereinfachtes Newton-Verfahren

Für $n >> 1$ ist die Berechnung von $(F'(x_n))^{-1}$ sehr mühsam. Man verwendet daher im vereinfachten Newton-Verfahren die Rundung $F'(x_k) \approx F'(x_0)$ und erhält damit

$$x_{k+1} = x_k - (F'(x_0))^{-1}F(x_k).$$

8.6.3 Gedämpftes Newton-Verfahren

Um den Konvergenzbereich des Newton-Verfahrens zu vergrößern, ist es möglich eine Folge von Dämpfungsfaktoren $\omega_k \in (0,1)$ einzufügen. Man erhält dann als Iterationsvorschrift

$$x_{k+1} = x_k - \omega_k(F'(x_k))^{-1}F(x_k).$$

8.6.4 Beispiele:

Wir bemühen nun einen "Klassiker" und berechnen die Quadratwurzel aus 2 als Nullstelle der Gleichung $f(x) = x^2 - 2$ mit $f'(x) = 2x$. Als Startwert wählen wir $x_0 = 5$. Für das gedämpfte Newtonverfahren wählen wir $(\omega_k)_{k\in\mathbb{N}} = ((\frac{1}{2})^k)_{k\in\mathbb{N}}$

Allgemeines Newton-Verfahren:

Wir erhalten

$$
\begin{aligned}
x_1 &= 5 - \frac{5^2 - 2}{2 \cdot 5} = 5 - \frac{23}{10} = \frac{27}{10} \\
x_2 &= \frac{27}{10} - \frac{(\frac{27}{10})^2 - 2}{2 \cdot \frac{27}{10}} = 1,35 \\
x_3 &= 1,35 - \frac{1,35^2 - 2}{2 \cdot 1,35} = 1,42 \approx \sqrt[2]{2}
\end{aligned}
$$

Vereinfachtes Newton-Verfahren:

Wir erhalten

$$
\begin{aligned}
x_1 &= 5 - \frac{5^2 - 2}{2 \cdot 5} = 5 - \frac{23}{10} = \frac{27}{10} \\
x_2 &= \frac{27}{10} - \frac{(\frac{27}{10})^2 - 2}{10} = 2,171 \\
x_3 &= 2,17 - \frac{2,17^2 - 2}{10} = 1,9 \\
x_4 &= 1,9 - \frac{1,9^2 - 2}{10} = 1,74 \\
x_5 &= 1,74 - \frac{1,174^2 - 2}{10} \approx 1,637
\end{aligned}
$$

Gedeämpftes Newton-Verfahren:

Wir erhalten

$$
\begin{aligned}
x_1 &= 5 - 1 \cdot \frac{5^2 - 2}{2 \cdot 5} = 5 - \frac{23}{10} = \frac{27}{10} \\
x_2 &= \frac{27}{10} - \frac{1}{2} \cdot \frac{(\frac{27}{10})^2 - 2}{2 \cdot \frac{27}{10}} = 2,21 \\
x_3 &= 2,21 - \frac{1}{4} \cdot \frac{2,21^2 - 2}{2 \cdot 2,21} = 2.05 \\
x_4 &= 2,05 - \frac{1}{8} \cdot \frac{2,05^2 - 2}{2 \cdot 2,05} = 1,98
\end{aligned}
$$