

Algorithmen und Datenstrukturen (für ET/IT)

Sommersemester 2017

Dr. Stefanie Demirci

Computer Aided Medical Procedures
Technische Universität München



Programm heute

7 Fortgeschrittene Datenstrukturen

8 Such-Algorithmen

9 Graph-Algorithmen

10 Numerische Algorithmen

Matrizen

Lineare Gleichungen

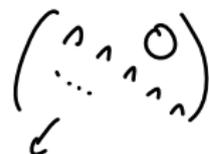
Die LUP-Zerlegung

Least Squares Probleme

Wiederholung: LUP Zerlegung

LUP Zerlegung:

- Sei $A \in \mathbb{R}^{n \times n}$ invertierbar.
- Dann gibt es eine untere Einheitsdreiecksmatrix $L \in \mathbb{R}^{n \times n}$, eine obere Dreiecksmatrix $U \in \mathbb{R}^{n \times n}$ sowie eine Permutationsmatrix $P \in \mathbb{R}^{n \times n}$, so dass



$$PA = LU.$$

- Aufwand: $\sim \frac{2n^3}{3}$ FLOPs
- Lösung von $Ax = b$: betrachte $\underline{PA}x = \underline{P}b \Leftrightarrow \underline{L}Ux = \underline{P}b$,
 - löse $Lz = \underline{P}b$ mit Vorwärts-Substitution nach z auf
 - löse $Ux = z$ mit Rückwärts-Substitution nach x auf

LUP-Zerlegung: Beispiel

- Lineares Gleichungssystem:

mit $\det(A) \neq 0 \Rightarrow$ invertierbar $Ax = b,$

$$A = \begin{pmatrix} 1 & 2 & 0 \\ 3 & 4 & 4 \\ 5 & 6 & 3 \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ 7 \\ 8 \end{pmatrix}$$

- LUP Zerlegung:

$$\underline{PA = LU}$$

mit

$$L = \begin{pmatrix} \textcircled{1} & 0 & 0 \\ 0,2 & \textcircled{1} & 0 \\ 0,6 & 0,5 & \textcircled{1} \end{pmatrix}, \quad U = \begin{pmatrix} \textcircled{5} & 6 & 3 \\ 0 & \textcircled{0,8} & -0,6 \\ 0 & 0 & \textcircled{2,5} \end{pmatrix}, \quad P = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

LUP-Zerlegung: Beispiel (Forts.)

- statt $Ax = b$ löse $LUx = Pb$
- **Schritt 1:** löse $Lz = Pb$ (Vorwärts-Substitution):

$$\begin{pmatrix} 1 & 0 & 0 \\ 0,2 & 1 & 0 \\ 0,6 & 0,5 & 1 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} 8 \\ 3 \\ 7 \end{pmatrix} \implies z = \begin{pmatrix} 8 \\ 1,4 \\ 1,5 \end{pmatrix}$$

- **Schritt 2:** löse $Ux = z$ (Rückwärts-Substitution):

$$\begin{pmatrix} 5 & 6 & 3 \\ 0 & 0,8 & -0,6 \\ 0 & 0 & 2,5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 8 \\ 1,4 \\ 1,5 \end{pmatrix} \implies x = \begin{pmatrix} -1,4 \\ 2,2 \\ 0,6 \end{pmatrix}$$

Die LU und LUP Zerlegung

LU Zerlegung:

- Sei $A \in \mathbb{R}^{n \times n}$ invertierbar und sei A diagonal dominant (d.h. $|a_{ii}| \geq \sum_{i \neq j} |a_{ij}|$ für alle $i = 1, \dots, n$).
- Dann gibt es eine untere Einheitsdreiecksmatrix $L \in \mathbb{R}^{n \times n}$ und eine obere Dreiecksmatrix $U \in \mathbb{R}^{n \times n}$ mit

$$A = LU.$$

LUP Zerlegung:

- Sei $A \in \mathbb{R}^{n \times n}$ invertierbar.
- Dann gibt es eine untere Einheitsdreiecksmatrix $L \in \mathbb{R}^{n \times n}$, eine obere Dreiecksmatrix $U \in \mathbb{R}^{n \times n}$ sowie eine Permutationsmatrix $P \in \mathbb{R}^{n \times n}$ mit

$$PA = LU.$$

Berechnung der LU Zerlegung 1

- Berechnung der LU Zerlegung mit Gauss Elimination
- Rekursive Strategie:

- Fall $n = 1$:

$$L = (1), U = A.$$

- Fall $n > 1$:

$$A = \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} = \begin{pmatrix} a_{11} & w^T \\ v & A' \end{pmatrix}$$

The diagram shows the matrix A partitioned into four blocks. The top-left element a_{11} is circled. The top-right row (a_{21}, \dots, a_{1n}) is circled and labeled w^T . The bottom-left column (a_{21}, \dots, a_{n1}) is circled and labeled v . The bottom-right submatrix $(a_{22}, \dots, a_{2n}; \dots; a_{n2}, \dots, a_{nn})$ is circled and labeled A' .

mit $v, w \in \mathbb{R}^{n-1}$ und $A' \in \mathbb{R}^{n-1 \times n-1}$.

Berechnung der LU Zerlegung 2

- Faktorisierung von A :

$$\begin{aligned} A &= \begin{pmatrix} a_{11} & w^T \\ v & A' \end{pmatrix} \leftarrow \begin{array}{l} \text{wird zu } L \\ \text{wird zu } U \end{array} \\ &= \begin{pmatrix} \underline{1} & \underline{0} \\ \underline{v/a_{11}} & \underline{I_{n-1}} \end{pmatrix} \begin{pmatrix} \underline{a_{11}} & \underline{w^T} \\ \underline{0} & \underline{A' - vw^T/a_{11}} \end{pmatrix}, \end{aligned}$$

wobei I_{n-1} die $n-1 \times n-1$ Einheitsmatrix ist.

- $A' - vw^T/a_{11}$ heißt **Schur-Komplement** von A bezüglich a_{11}
- Falls A eine LU Zerlegung hat, hat auch das Schur-Komplement eine LU Zerlegung
- **Annahme:**

$$A' - vw^T/a_{11} = L'U'$$

mit $L' \in \mathbb{R}^{n-1 \times n-1}$ untere Einheitsdreiecksmatrix,
 $U' \in \mathbb{R}^{n-1 \times n-1}$ obere Dreiecksmatrix.

Berechnung der LU Zerlegung 3

- Faktorisierung von A zusammen mit Annahme ergibt:

$$\begin{aligned} A &= \begin{pmatrix} 1 & 0 \\ v/a_{11} & I_{n-1} \end{pmatrix} \begin{pmatrix} a_{11} & w^T \\ 0 & \underbrace{A' - vw^T/a_{11}}_{L'U'} \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ v/a_{11} & I_{n-1} \end{pmatrix} \begin{pmatrix} a_{11} & w^T \\ 0 & \underline{L'U'} \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ \underline{v/a_{11}} & \underline{L'} \end{pmatrix} \begin{pmatrix} a_{11} & w^T \\ \underline{0} & \underline{U'} \end{pmatrix} \\ &= \underline{LU} \end{aligned}$$

- dann Rekursion für $A' - vw^T/a_{11} = L'U'$
- **Vorsicht:** $a_{11} \neq 0$ ist notwendig, ebenso für Rekursionsschritte
→ ok, da A diagonal dominant

Algorithmus LU Zerlegung

Input: Matrix $A \in \mathbb{R}^{n \times n}$

Output: Matrizen $L, U \in \mathbb{R}^{n \times n}$

LU-Zerlegung(A):

$U = 0; L = I_n$; // Initialisierung

1. for $k = 1$ to n {

$u_{kk} = a_{kk}$; // Diagonalelement

2. *additiv* for $i = k + 1$ to n {

$l_{ik} = \underline{a_{ik}} / u_{kk}$; // entspricht v_k v_k / a_{ii}

$u_{ki} = \underline{a_{ki}}$; // entspricht w_i^T

}

2. for $i = k + 1$ to n { ←

3. for $j = k + 1$ to n { ←

multiplicativ $a_{ij} = a_{ij} - l_{ik} u_{kj}$; // Schur-Komplement A'

}

}

}

- Rekursion wurde durch Iteration ersetzt

- Aufwand: $\Theta(n^3)$

Algorithmus LU Zerlegung

Input: Matrix $A \in \mathbb{R}^{n \times n}$

Output: Matrizen $L, U \in \mathbb{R}^{n \times n}$

LU-Zerlegung(A):

$U = 0; L = I_n;$

for $k = 1$ **to** n {

$u_{kk} = a_{kk};$

for $i = k + 1$ **to** n {

$l_{ik} = a_{ik} / u_{kk};$ // entspricht v_k

$u_{ki} = a_{ki};$ // entspricht w_i^T

}

for $i = k + 1$ **to** n {

for $j = k + 1$ **to** n {

$a_{ij} = a_{ij} - l_{ik} u_{kj};$

}

}

}

- Rekursion wurde durch Iteration ersetzt
- Aufwand: $\Theta(n^3)$
- Optimierung: falls A nicht weiter benötigt wird, können L, U direkt in A abgespeichert werden (in place)
 - L hat nur 1 auf der Diagonalen, wird nicht explizit gespeichert
 - Diagonale von U in Diagonale von A
 - Modifikation: ersetze l, u durch a

Beispiel-Ablauf: LU Zerlegung

- **LU-Zerlegung**(A) mit in place Optimierung:

w^T
 v y x m

| | | | |
|---|----|----|----|
| 2 | 3 | 1 | 5 |
| 6 | 13 | 5 | 19 |
| 2 | 19 | 10 | 23 |
| 4 | 10 | 11 | 31 |

A

| | | | |
|---|----|---|----|
| 2 | 3 | 1 | 5 |
| 3 | 4 | 2 | 4 |
| 1 | 16 | 9 | 18 |
| 2 | 4 | 9 | 21 |

k=1

| | | | |
|---|---|---|----|
| 2 | 3 | 1 | 5 |
| 3 | 4 | 2 | 4 |
| 1 | 4 | 1 | 2 |
| 2 | 1 | 7 | 17 |

k=2

u

| | | | |
|---|---|---|----|
| 2 | 3 | 1 | 5 |
| 3 | 4 | 2 | 4 |
| 1 | 4 | 1 | 2 |
| 2 | 1 | 7 | 17 |

k=3

- Ergebnis:

α

$$A = L \begin{pmatrix} 2 & 3 & 1 & 5 \\ 6 & 13 & 5 & 19 \\ 2 & 19 & 10 & 23 \\ 4 & 10 & 11 & 31 \end{pmatrix} = \underbrace{L}_{\text{Rest}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 2 & 1 & 7 & 1 \end{pmatrix} \underbrace{u} \begin{pmatrix} 2 & 3 & 1 & 5 \\ 0 & 4 & 2 & 4 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 3 \end{pmatrix}$$

LU vs. LUP Zerlegung

- **Problem** bei LU Zerlegung: Division durch a_{ii}
 - a_{ii} kann 0 sein (vermieden durch A diagonal dominant)
 - a_{ii} kann sehr klein sein, verursacht numerische Instabilitäten
- **Lösung:** vertauschen von Zeilen, so dass $a_{ii} \neq 0$ und groß
 - ist elementare Zeilenumformung, ändert also Lösung von Gleichungssystem nicht
 - dargestellt als Permutationsmatrix $P \in \mathbb{R}^{n \times n}$ so dass

$$PA = LU$$

also LUP Zerlegung oder LU Zerlegung mit partiellem Pivoting

Permutationsmatrizen

Permutationsmatrix

$P \in \mathbb{R}^{n \times n}$ heißt **Permutationsmatrix**, falls sie in jeder Zeile und jeder Spalte jeweils eine 1 enthält und sonst nur 0.

- Beispiel:

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

- Anwendung:

$$P_{\tilde{x}} = \begin{pmatrix} 0 & \tilde{1} & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_2 \\ x_3 \\ x_1 \end{pmatrix}$$

Permutationsmatrizen

- Ist $A \in \mathbb{R}^{n \times n}$ und $P \in \mathbb{R}^{n \times n}$ Permutationsmatrix, dann
 - PA ist A mit vertauschten Zeilen
 - AP ist A mit vertauschten Spalten

- Beispiel:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \quad P = \begin{pmatrix} 0 & \textcircled{1} & 0 \\ 0 & 0 & \textcircled{1} \\ \textcircled{1} & 0 & 0 \end{pmatrix}$$

Spalte/
Setze 2. Zeile in 1.
Spalte/
Setze 3. Zeile in 2.
Spalte/
Setze 1. Zeile in 3.

$$PA = \begin{pmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \\ 1 & 2 & 3 \end{pmatrix}, \quad AP = \begin{pmatrix} 3 & 1 & 2 \\ 6 & 4 & 5 \\ 9 & 7 & 8 \end{pmatrix}$$

- Implementation:** oft einfacher als Feld π statt als Matrix P
 - $\pi[i] = j$ bedeutet P hat 1 in Zeile i , Spalte j

Berechnung der LUP Zerlegung 1

- LUP Zerlegung ist LU Zerlegung mit **Pivoting-Strategie**
- **Pivoting-Strategie**: vertausche Zeilen, so daß Element mit **höchstem Absolutwert** an Stelle von a_{jj} steht
 - Bedingung A diagonal dominant ist nicht mehr nötig
 - A invertierbar sichert, dass keine 0 Spalten existieren
- **Rekursiver Ansatz** für $n > 1$:
 - Q Permutationsmatrix, so dass a_{k1} Pivot-Element

$$QA = \begin{pmatrix} \overset{\text{Stelle } a_{kn}}{\circlearrowleft} a_{k1} & w^T \\ v & A' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ v/a_{k1} & I_{n-1} \end{pmatrix} \begin{pmatrix} a_{k1} & w^T \\ 0 & A' - vw^T/a_{k1} \end{pmatrix}$$

mit $v, w \in \mathbb{R}^{n-1}$, $A' \in \mathbb{R}^{(n-1) \times (n-1)}$ und

$$\begin{aligned} v &= (a_{21}, a_{31}, \dots, a_{k-1,1}, a_{k+1,1}, \dots, a_{n1})^T \\ w^T &= (a_{k2}, a_{k3}, \dots, a_{kn}) \end{aligned}$$

Berechnung der LUP Zerlegung 2

- Annahme:

$$P'(A' - vw^T/a_{k1}) = L'U'$$

mit P' Permutationsmatrix

- Mit Permutationsmatrix

$$P = \begin{pmatrix} 1 & 0 \\ 0 & P' \end{pmatrix} Q$$

folgt

$$\begin{aligned} PA &= \begin{pmatrix} 1 & 0 \\ 0 & P' \end{pmatrix} QA \\ &= \begin{pmatrix} 1 & 0 \\ 0 & P' \end{pmatrix} \begin{pmatrix} 1 & 0 \\ v/a_{k1} & I_{n-1} \end{pmatrix} \begin{pmatrix} a_{k1} & w^T \\ 0 & A' - vw^T/a_{k1} \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ \textcircled{P'} v/a_{k1} & \textcircled{P'} \end{pmatrix} \begin{pmatrix} a_{k1} & w^T \\ 0 & A' - vw^T/a_{k1} \end{pmatrix} \end{aligned}$$

Berechnung der LUP Zerlegung 3

$$\begin{aligned} PA &= \begin{pmatrix} 1 & 0 \\ P'v/a_{k1} & P' \end{pmatrix} \begin{pmatrix} a_{k1} & w^T \\ 0 & A' - vw^T/a_{k1} \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ P'v/a_{k1} & I_{n-1} \end{pmatrix} \begin{pmatrix} a_{k1} & w^T \\ 0 & P'(A' - vw^T/a_{k1}) \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ P'v/a_{k1} & I_{n-1} \end{pmatrix} \begin{pmatrix} a_{k1} & w^T \\ 0 & L'U' \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ P'v/a_{k1} & L' \end{pmatrix} \begin{pmatrix} a_{k1} & w^T \\ 0 & U' \end{pmatrix} \\ &= LU \end{aligned}$$

Algorithmus LUP Zerlegung

Input: Matrix $A \in \mathbb{R}^{n \times n}$

Output: Matrizen L, U direkt in A , Feld π

LUP-Zerlegung(A):

π = Feld Länge n ;

for $i = 1$ **to** n { $\pi[i] = i$; }

for $k = 1$ **to** n {

$p = 0$; // Pivot-Element

for $i = k$ **to** n {

if ($|a_{ik}| > p$)

$p = |a_{ik}|$; $k' = i$;

}

if ($p == 0$) **error**("Matrix singular");

vertausche $\pi[k]$ mit $\pi[k']$;

for $i = 1$ **to** n

vertausche a_{ki} mit $a_{k'i}$;

for $i = k + 1$ **to** n {

$a_{ik} = a_{ik}/a_{kk}$;

for $j = k + 1$ **to** n

$a_{ij} = a_{ij} - a_{ik}a_{kj}$;

}

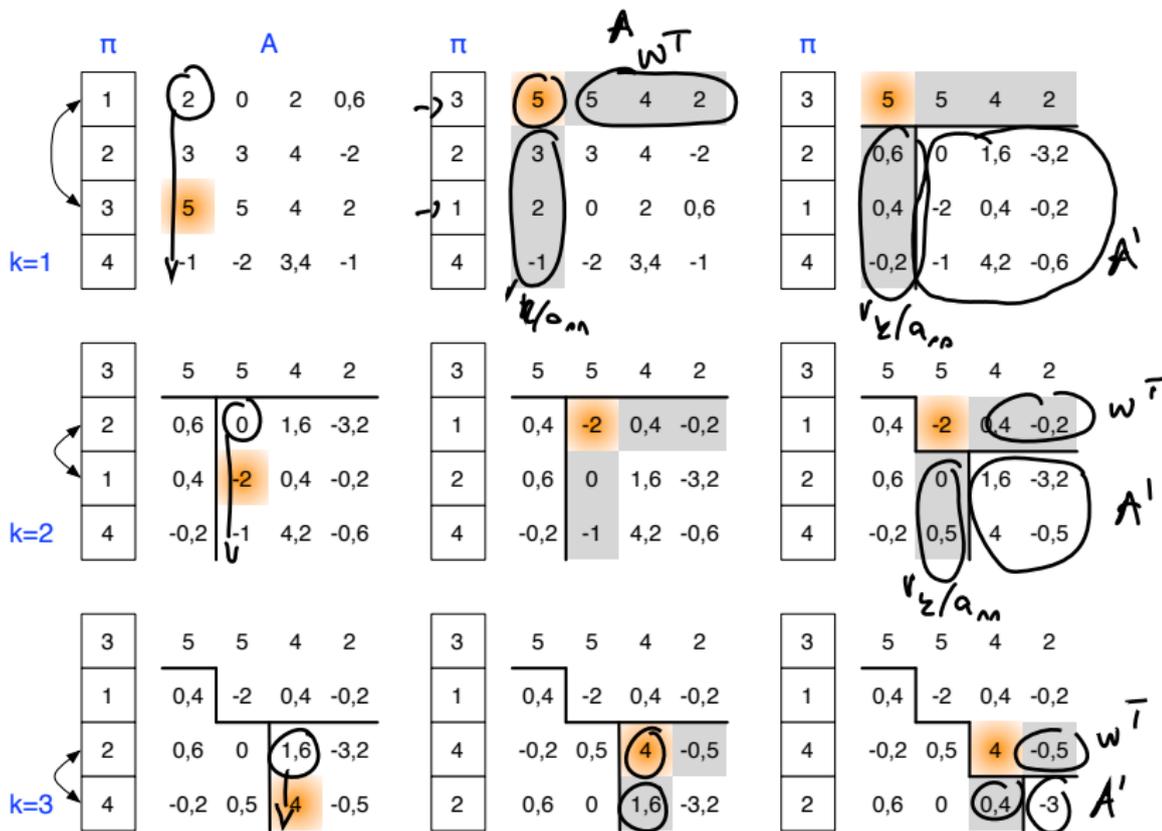
}

Pivotsierung

LU-Zerlegung

- Rekursion wurde durch Iteration ersetzt
- Variante mit **in place** Optimierung
- Aufwand: $\Theta(n^3)$
 - extra Aufwand Pivoting nur quadratisch

Beispiel-Ablauf: LUP Zerlegung



Beispiel-Ablauf: LUP Zerlegung 2

- Ergebnis von **LUP-Zerlegung**:

| | | | | |
|---|------|-----|-----|------|
| 3 | 5 | 5 | 4 | 2 |
| 1 | 0,4 | -2 | 0,4 | -0,2 |
| 4 | -0,2 | 0,5 | 4 | -0,5 |
| 2 | 0,6 | 0 | 0,4 | -3 |

A

- bzw. $PA = LU$ mit

$$\begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 & 2 & 0,6 \\ 3 & 3 & 4 & -2 \\ 5 & 5 & 4 & 2 \\ -1 & -2 & 3,4 & -1 \end{pmatrix} =$$

$$L \left\{ \begin{pmatrix} \textcircled{1} & 0 & 0 & 0 \\ 0,4 & \textcircled{1} & 0 & 0 \\ -0,2 & 0,5 & \textcircled{1} & 0 \\ 0,6 & 0 & 0,4 & \textcircled{1} \end{pmatrix} \begin{pmatrix} 5 & 5 & 4 & 2 \\ 0 & -2 & 0,4 & -0,2 \\ 0 & 0 & 4 & -0,5 \\ 0 & 0 & 0 & -3 \end{pmatrix} \right\} U$$

Anwendung der LU(P) Zerlegung

- Lösen von linearen System $Ax = b$ mittels:

- $PAx = Pb \Leftrightarrow LUx = Pb$
- löse $Lz = Pb$ mit Vorwärts-Substitution nach z auf
- löse $Ux = z$ mit Rückwärts-Substitution nach x auf

- Berechnen von Inverser von $A \in \mathbb{R}^{n \times n}$

- Matrix-Gleichung $AX = I_n$ mit $X = (X_1, \dots, X_n) \in \mathbb{R}^{n \times n}$
- berechne $PA = LU$
- löse $AX_i = e_i$ nach X_i mit LUP-Zerlegung für $i = 1, \dots, n$
(e_i i -ter Einheitsvektor)
- am Ende: $X = A^{-1}$
- Aufwand: $\Theta(n^3)$

Spaltenvektoren

unbekannte

Programm heute

7 Fortgeschrittene Datenstrukturen

8 Such-Algorithmen

9 Graph-Algorithmen

10 Numerische Algorithmen

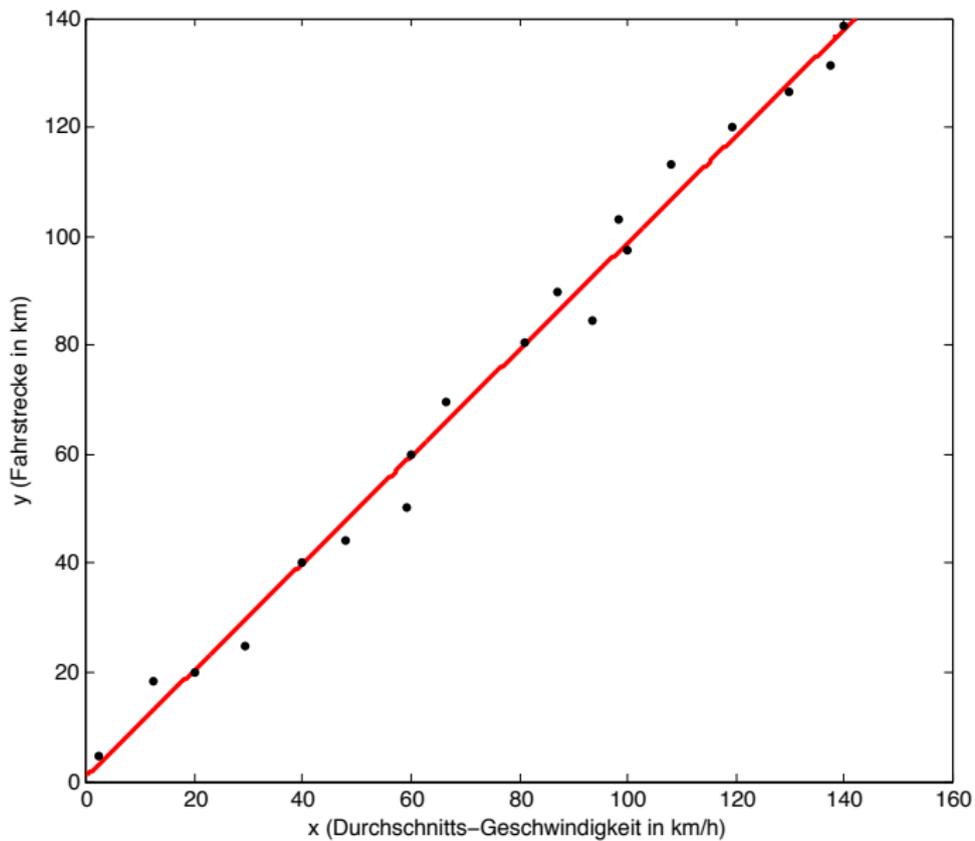
Matrizen

Lineare Gleichungen

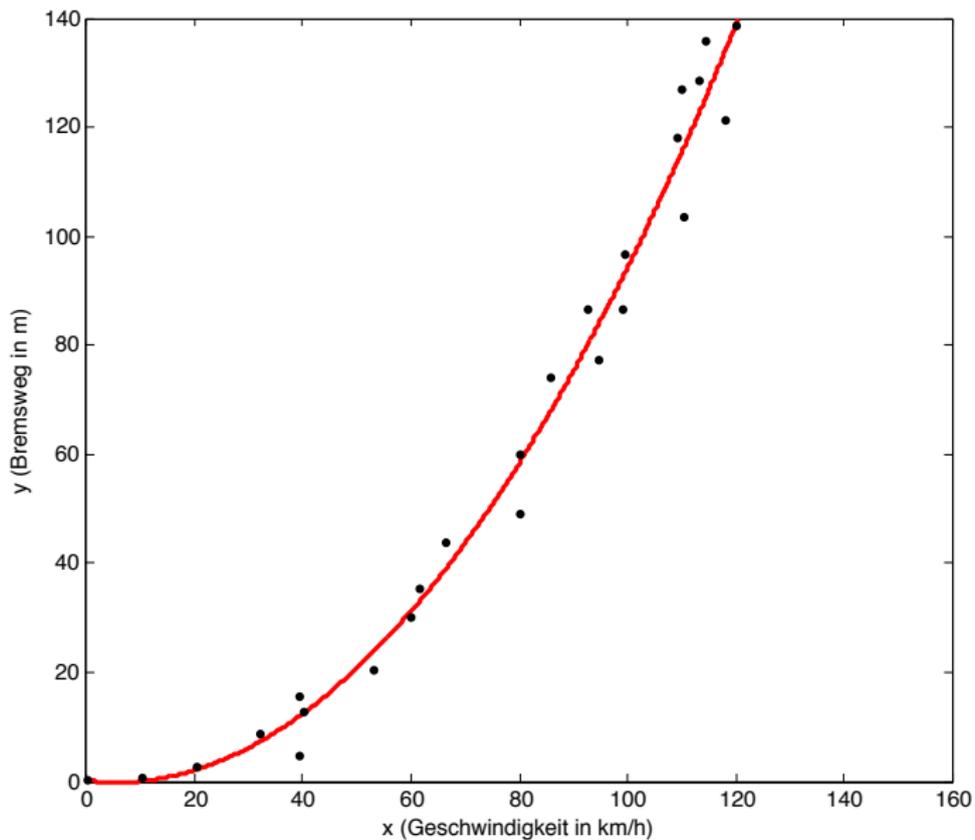
Die LUP-Zerlegung

Least Squares Probleme

Beispiel-Problem: Geschwindigkeit vs. Fahrstrecke



Beispiel-Problem: Geschwindigkeit vs. Bremsweg



Problemstellung Least Squares

- **Gegeben:** Datenreihe mit m Datenpunkten

$$(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$$

mit $x_j, y_j \in \mathbb{R}$ für $j = 1, \dots, m$

- **Erwartung:** y_j enthalten Messfehler ($j = 1, \dots, m$)
- **Gesucht:** Funktion $F : \mathbb{R} \rightarrow \mathbb{R}$ so dass **Approximationsfehler**

$$\eta_j = F(x_j) - y_j$$

für alle $j = 1, \dots, m$ möglichst gering

Problemstellung Least Squares

- **Gegeben:** Datenreihe mit m Datenpunkten

$$(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)$$

mit $x_j, y_j \in \mathbb{R}$ für $j = 1, \dots, m$

- **Erwartung:** y_j enthalten Messfehler ($j = 1, \dots, m$)
- **Gesucht:** Funktion $F : \mathbb{R} \rightarrow \mathbb{R}$ so dass **Approximationsfehler**

$$\eta_j = F(x_j) - y_j$$

für alle $j = 1, \dots, m$ möglichst gering

- **Annahme:** F lässt sich darstellen als Summe von Basisfunktionen f_i ,

$$F(x) = \sum_{i=1}^n c_i f_i(x)$$

Wahl der Basisfunktionen

- Wahl der Basisfunktionen f_i für F :

- typische Wahl: $f_i(x) = x^{i-1}$, dann

$$F(x) = c_1 + c_2x + c_3x^2 + \dots + c_nx^{n-1}$$

Koeffizienten

(Polynom in x vom Grad $n - 1$)

- auch oft mit $n = 2$, dann

$$F(x) = c_1 + c_2x$$

(Gerade) auch genannt **lineare Regression**

- im Fall von Tomographie: Pixel- oder Voxel-Basisfunktionen

Matrix-Notation

- für die verschiedenen Datenpunkte $x_j, j = 1, \dots, m$, kann F geschrieben werden als:

$$\begin{pmatrix} F(x_1) \\ \vdots \\ F(x_m) \end{pmatrix} = \underbrace{\begin{pmatrix} f_1(x_1) & \cdots & f_n(x_1) \\ \vdots & & \vdots \\ f_1(x_m) & \cdots & f_n(x_m) \end{pmatrix}}_{=:A} \underbrace{\begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}}_{=:c}$$

mit $A \in \mathbb{R}^{m \times n}$, $c \in \mathbb{R}^n$.

- untersucht wird dann der **Approximationsfehler** η

$$\eta = \underbrace{(Ac)}_{\tilde{f}(x_i)} - y$$

mit $\eta = (\eta_1, \dots, \eta_m) \in \mathbb{R}^m$, $y = (y_1, \dots, y_m) \in \mathbb{R}^m$.

Daten mit Meßfehler

- ist $m = n$ und A invertierbar, so kann direkt

$$Ac - y = 0$$

gelöst werden.

- **Problem:**
 - selbst wenn A invertierbar ist, y enthält Meßfehler
 - Lösung F ist dann meist **nicht** die **gewünschte** Lösung
 - passt sich zu sehr an "Ausreißer" an
- **besser:** mehr Datenpunkte, $m \gg n$

Minimierung des Approximationsfehlers

- zur Minimierung des Approximationsfehlers kann z.B. die Norm $\|\eta\|$ betrachtet werden

$$\|\eta\| = \left(\sum_{j=1}^m \eta_j^2 \right)^{\frac{1}{2}} = \sqrt{\sum_{j=1}^m \eta_j^2}$$

- zur Vereinfachung betrachtet man üblicherweise

$$\|\eta\|^2 = \|Ac - y\|^2 = \sum_{j=1}^m \left(\sum_{i=1}^n a_{ji} c_i - y_j \right)^2$$

- daher der Name: Methode der kleinsten Quadrate oder Least squares

Least-squares Lösung

Least-squares Lösung

Sei $A \in \mathbb{R}^{m \times n}$, $y \in \mathbb{R}^m$ mit $m \geq n$. Eine Lösung $c \in \mathbb{R}^n$ des Minimierungs-Problems

$$\min_c \|Ac - y\| \quad \text{or} \quad \min_c \|Ac - y\|^2$$

heißt **Least-squares Lösung**.

Anwendungs-Beispiele:

- Tracking von Objekten mit Kameras
- Kalibrierung von Kameras, Robotern, etc.
- Iterative Rekonstruktion für Tomographie

Normalengleichung

- Berechnung der **Least-squares Lösung** mit Standard-Technik
"Ableitung gleich null setzen"
- hier: partielle Ableitungen für $k = 1, \dots, n$

$$\frac{\partial \|\eta\|^2}{\partial c_k} = \sum_{j=1}^m \underbrace{\left(\sum_{i=1}^n a_{ji} c_i - y_j \right)}_{Ac - y} \underbrace{\left(a_{jk} \right)}_{A^T} = 0$$

nachdifferenzieren

- daraus folgt

$$\nabla \|\eta\|^2 = \underbrace{A^T (Ac - y)} = 0$$

- umgeformt ergibt sich

$$A^T Ac = A^T y$$

auch genannt die **Normalengleichung**.

Normalengleichung

Normalengleichung

Sei $A \in \mathbb{R}^{m \times n}$, $y \in \mathbb{R}^m$ mit $m \geq n$. $c \in \mathbb{R}^n$ ist eine Least-squares Lösung von $\min_c \|Ac - y\|$ genau dann, wenn die Normalengleichung gilt:

$$A^T A c = A^T y$$

Insbesondere ist die Normalengleichung lösbar, falls $\text{rank}(A) = n$.

Normalengleichung

Normalengleichung

Sei $A \in \mathbb{R}^{m \times n}$, $y \in \mathbb{R}^m$ mit $m \geq n$. $c \in \mathbb{R}^n$ ist eine Least-squares Lösung von $\min_c \|Ac - y\|$ genau dann, wenn die Normalengleichung gilt:

$$A^T A c = A^T y$$

Insbesondere ist die Normalengleichung lösbar, falls $\text{rank}(A) = n$.

- Die Matrix $A^T A$ ist immer **symmetrisch**.
- Falls $\text{rank}(A) = n$ ist $A^T A$ auch **positiv definit** und damit invertierbar
- Die Lösung der Normalengleichung ist dann

$$c = ((A^T A)^{-1} A^T) y.$$

Pseudoinverse

Pseudoinverse

Sei $A \in \mathbb{R}^{m \times n}$ mit $m \geq n$ und $\text{rank}(A) = n$. Die Matrix

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

heißt **Pseudoinverse** von A (auch **Moore-Penrose Inverse** genannt).

- Die Pseudoinverse verallgemeinert das Konzept der Inversen für nicht-quadratische Matrizen.
- Ist A invertierbar, dann gilt $A^+ = A^{-1}$.

Algorithmen zur Berechnung der Pseudoinversen 1

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

- Option 1: QR Methode (Verallgemeinerung f. $m \geq n$)

- berechne QR-Zerlegung $A = QR$, dann $Q^T Q = I$

$$A^T A = (QR)^T (QR) = R^T \underbrace{Q^T Q} R = R^T R$$

- löse die Matrix-Gleichung nach A^+

$$A^+ = (A^T A)^{-1} A^T \Leftrightarrow (A^T A) A^+ = A^T \Leftrightarrow \underline{(R^T R) A^+ = A^T}$$

mittels Vorwärts- und Rückwärts-Substitution.

Algorithmen zur Berechnung der Pseudoinversen 2

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

- Option 2: SVD Methode $f. m, n, k$

- berechne SVD-Zerlegung $A = U \Sigma V^T$, dann

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

- berechne $(A^T A)^{-1}$

$$(A^T A)^{-1} = V \Sigma_+^{-2} V^T,$$

wobei $\Sigma_+^{-2} = \text{diag}(1/\sigma_1^2, \dots, 1/\sigma_k^2, 0, \dots, 0)$
mit k größter Index so dass $\sigma_k > 0$

- berechne A^+

$$A^+ = (A^T A)^{-1} A^T = V \Sigma_+^{-2} V^T V \Sigma U^T = V \Sigma_+^{-1} U^T$$

Beispiel: Least squares Problem 1

- Gegeben: 5 Datenpunkte

$$(x_1, y_1) = (-1, 2), \quad (x_2, y_2) = (1, 1), \quad (x_3, y_3) = (2, 1)$$

$$(x_4, y_4) = (3, 0), \quad (x_5, y_5) = (5, 3)$$

- Gesucht: quadratisches Polynom

$$F(x) = c_1 + c_2x + c_3x^2$$

$$n = 3$$

Beispiel: Least squares Problem 2

- Aufstellen der Matrix A :

Vandermonde-Matrix

$$A = \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \\ 1 & x_4 & x_4^2 \\ 1 & x_5 & x_5^2 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 5 & 25 \end{pmatrix}$$

- Berechnen der Pseudoinversen A^+ :

$$A^+ = \begin{pmatrix} 0,500 & 0,300 & 0,200 & 0,100 & -0,100 \\ -0,388 & 0,093 & 0,190 & 0,193 & -0,088 \\ 0,060 & -0,036 & -0,048 & -0,036 & 0,060 \end{pmatrix}$$

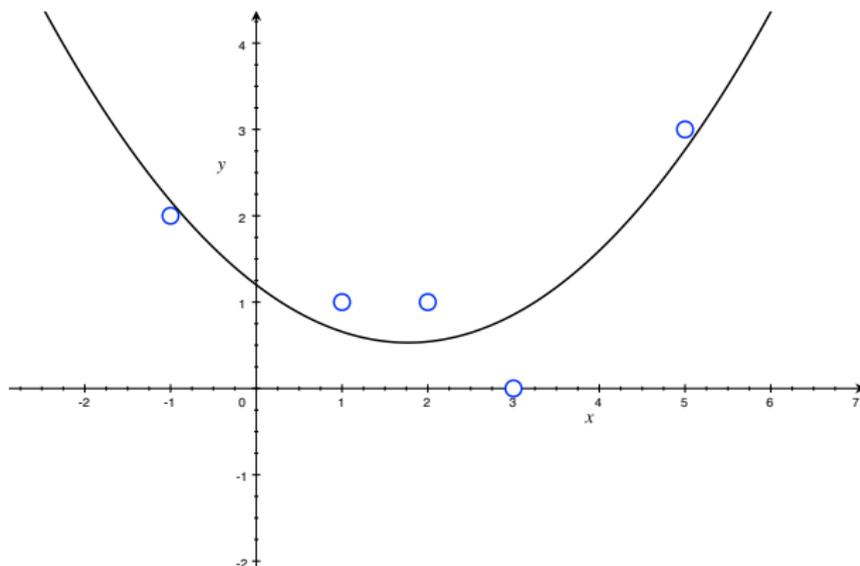
- Lösen nach c :

$$c = \begin{pmatrix} 1,200 \\ -0,757 \\ 0,214 \end{pmatrix}$$

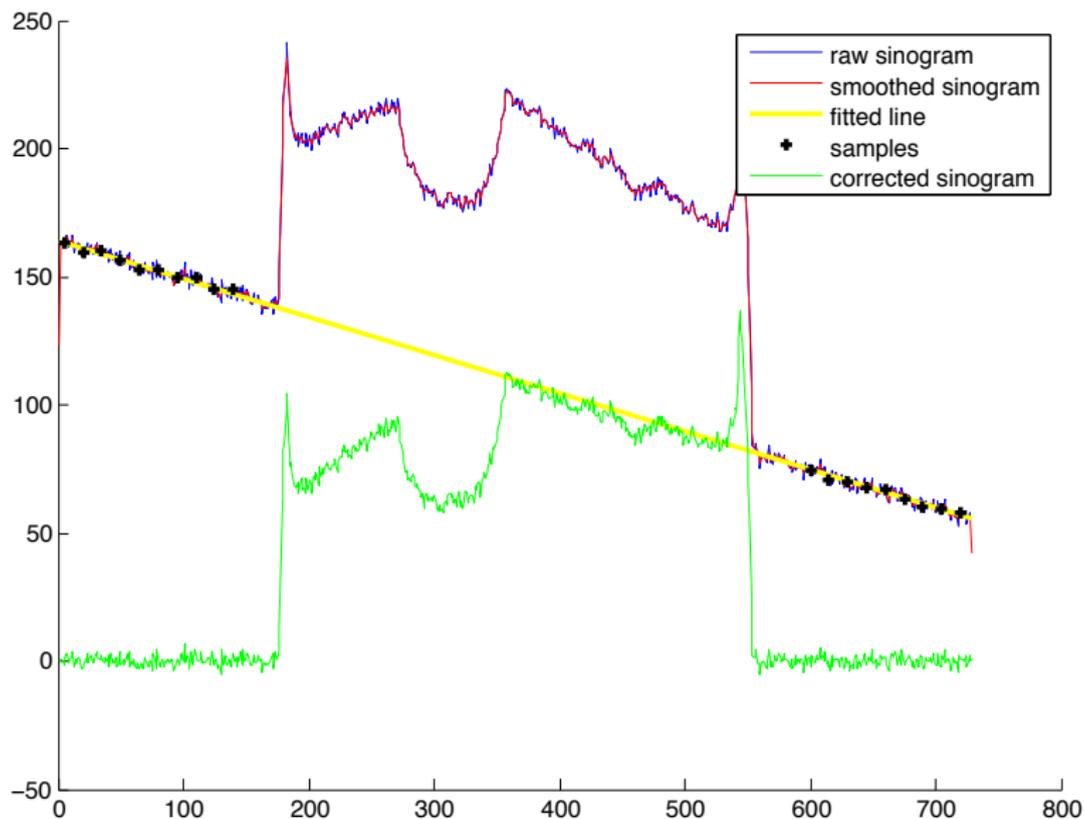
Beispiel: Least squares Problem 3

Lösung:

$$F(x) = 1,200 - 0,757x + 0,214x^2$$



Anwendungsbeispiel Tomographie



Zusammenfassung

7 Fortgeschrittene Datenstrukturen

8 Such-Algorithmen

9 Graph-Algorithmen

10 Numerische Algorithmen

Matrizen

Lineare Gleichungen

Die LUP-Zerlegung

Least Squares Probleme