

# Algorithmen und Datenstrukturen (für ET/IT)

Sommersemester 2016

Dr. Tobias Lasser

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

# LUP-Zerlegung: Beispiel

- Lineares Gleichungssystem:

$$Ax = b,$$

mit

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

- LUP Zerlegung:

$$PA = LU$$

mit

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

## 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  $n = 1$ :

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

- Fall  $n > 1$ :

$$\begin{aligned} A &= \left( \begin{array}{c|ccc} a_{11} & a_{21} & \cdots & a_{1n} \\ \hline a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{array} \right) \\ &= \begin{pmatrix} a_{11} & w^T \\ v & A' \end{pmatrix} \end{aligned}$$

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  $A$ :

$$\begin{aligned} A &= \begin{pmatrix} a_{11} & w^T \\ v & A' \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ v/a_{11} & I_{n-1} \end{pmatrix} \begin{pmatrix} a_{11} & w^T \\ 0 & A' - vw^T/a_{11} \end{pmatrix}, \end{aligned}$$

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:

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

- 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):**

$U = 0; L = I_n;$

**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  $a$



# 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:

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

# 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:

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

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

- **Implementation:** oft einfacher als Feld  $\pi$  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
  - keine Bedingung  $A$  diagonal dominant mehr nötig
  - $A$  invertierbar sichert, daß keine 0 Spalten existieren
- **Rekursiver Ansatz** für  $n > 1$ :
  - $Q$  Permutationsmatrix, so daß  $a_{k1}$  Pivot-Element

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

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

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



## 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  $A \in \mathbb{R}^{n \times n}$

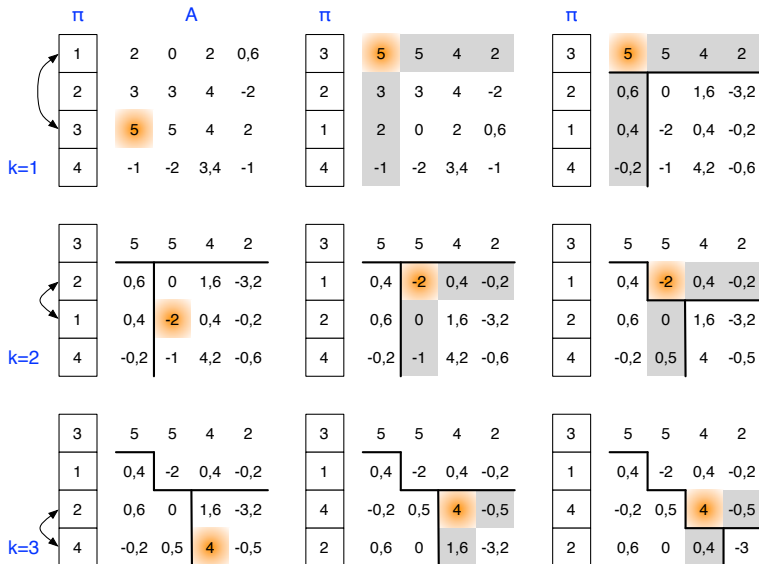
**Output:** Matrizen  $L, U$  direkt in  $A$ , Feld  $\pi$

**LUP-Zerlegung( $A$ ):**

```
 $\pi$  = 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  $n$   
       $a_{ij} = a_{ij} - a_{ik}a_{kj}$ ;  
    }  
}
```

- 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

- bzw.  $PA = LU$  mit

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

# 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:  $\Theta(n^3)$

# Programm heute

7 Fortgeschrittene Datenstrukturen

8 Such-Algorithmen

9 Graph-Algorithmen

10 Numerische Algorithmen

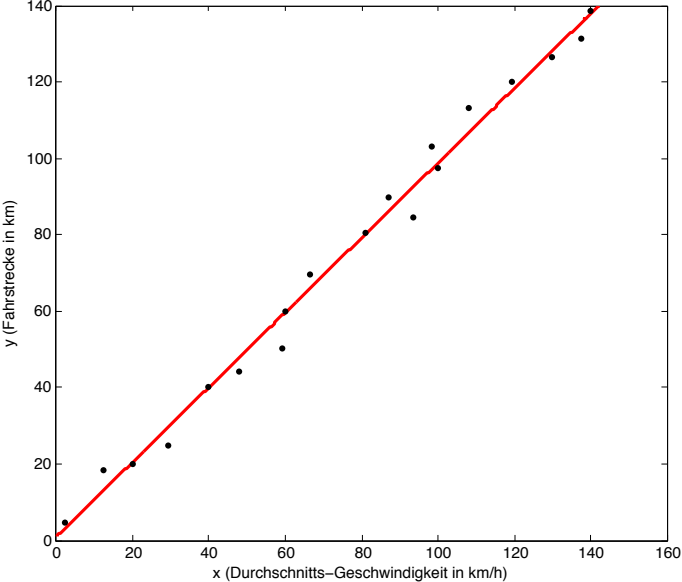
Matrizen

Lineare Gleichungen

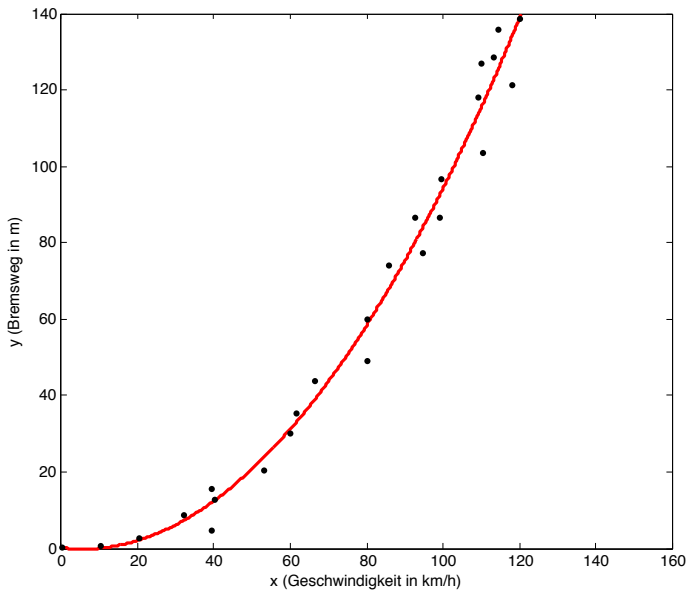
Die LUP-Zerlegung

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

- **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

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

(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**  $\eta$

$$\eta = Ac - y$$

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

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

- 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  $k = 1, \dots, n$

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

- daraus folgt

$$(Ac - y)^T A = 0 \quad \Leftrightarrow \quad A^T (Ac - y) = 0$$

- 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  $\text{rank}(A) = n$ .

- 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
  - berechne QR-Zerlegung  $A = QR$ , dann

$$A^T A = (QR)^T (QR) = R^T Q^T QR = R^T R$$

- löse die Matrix-Gleichung nach  $A^+$

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

mittels Vorwärts- und Rückwärts-Substitution.

## Algorithmen zur Berechnung der Pseudoinversen 2

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

- Option 2: SVD Methode

- 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

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

## Beispiel: Least squares Problem 2

- Aufstellen der Matrix  $A$ :

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

- Berechnen der Pseudoinversen  $A^+$ :

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

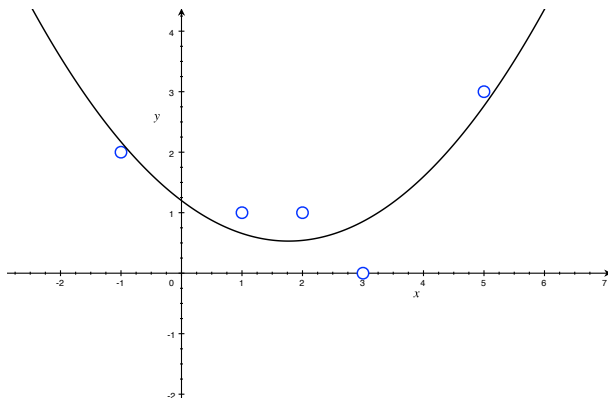
- Lösen nach  $c$ :

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

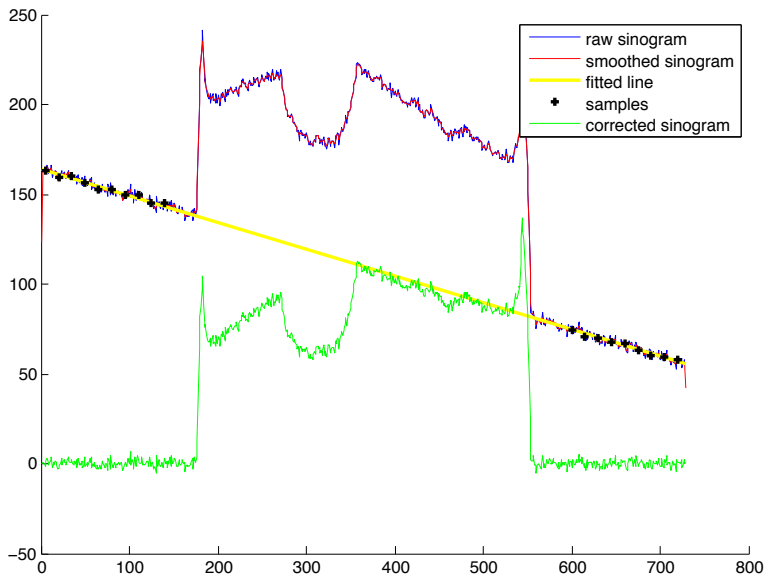
## Beispiel: Least squares Problem 3

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