

Algorithmen und Datenstrukturen (für ET/IT)

Wintersemester 2012/13

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

Lineare Gleichungen

Linear Gleichung

Eine **lineare Gleichung** hat die Form

$$a_1x_1 + a_2x_2 + \dots + a_nx_n = b,$$

wobei $x_1, \dots, x_n \in \mathbb{R}$ die **Unbekannten** sind, und die $a_1, \dots, a_n, b \in \mathbb{R}$ Konstanten.

- Schreibweise mit **Skalarprodukt**:

$$\langle a, x \rangle = b$$

oder auch

$$a^T x = b$$

mit $a = (a_i) \in \mathbb{R}^n$, $x = (x_i) \in \mathbb{R}^n$, $b \in \mathbb{R}$.

Lineares System

Lineares System

Ein **lineares System** bzw. **lineares Gleichungssystem** ist eine endliche Menge von linearen Gleichungen mit Unbekannten $x_1, \dots, x_n \in \mathbb{R}$:

$$\begin{array}{rcl} a_{11}x_1 + \dots + a_{1n}x_n & = & b_1 \\ \vdots & & \vdots \\ a_{m1}x_1 + \dots + a_{mn}x_n & = & b_m \end{array}$$

mit Konstanten $(a_{ji}) \in \mathbb{R}^{m \times n}$, $(b_i) \in \mathbb{R}^m$.

- wir betrachten hier meist den **quadratischen** Fall mit n Gleichungen und n Unbekannten

Lineares System

- Matrix-Schreibweise:

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$

- Kurz-Notation mit Matrix:

$$Ax = b$$

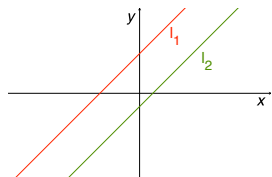
mit $A = (a_{ji}) \in \mathbb{R}^{m \times n}$, $b = (b_i) \in \mathbb{R}^m$ und Unbekannten $x = (x_i) \in \mathbb{R}^n$.

Beispiel: Lineares System

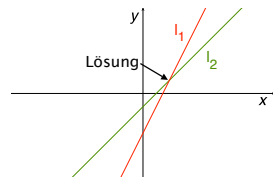
- Zwei Geraden in der Ebene (\mathbb{R}^2):

$$(l_1) \quad a_1x + b_1y = c_1 \quad (a_1, b_1) \neq (0, 0)$$

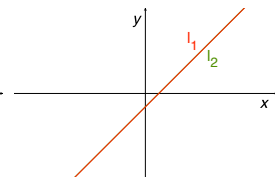
$$(l_2) \quad a_2x + b_2y = c_2 \quad (a_2, b_2) \neq (0, 0)$$



keine Lösung



eine Lösung



unendlich viele Lösungen

Lineares System: Eigenschaften

Sei $Ax = b$ lineares System mit $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ und $x \in \mathbb{R}^n$.

- Ein lineares System hat immer entweder **keine Lösung**, **eine Lösung** oder **unendlich viele Lösungen**.
- Ist $m < n$, so heißt das lineare System **unterbestimmt**.
- Ist $m > n$, so heißt das lineare System **überbestimmt**.
- Wir betrachten im weiteren den Fall $m = n$ (A quadratisch).

Lineare Systeme: Anwendungen

- Kalibrierung von Kameras
 - z.B. für Augmented Reality
- Registrierung von Bilddaten
- Segmentierung von Bilddaten
- Objekt-Detektierung und -Tracking
- Modelle für Finanzmärkte, Populationsbiologie
- Tomographische Rekonstruktion

Lösen von linearen Systemen

Inverse Matrix

Sei $A \in \mathbb{R}^{n \times n}$ quadratische Matrix. Falls ein $B \in \mathbb{R}^{n \times n}$ existiert mit

$$AB = BA = I_n$$

(wobei I_n die $n \times n$ Einheitsmatrix ist), dann heißt A **invertierbar** und wir nennen $B =: A^{-1}$ das **Inverse** von A .

Existiert A^{-1} , so heißt A **invertierbar**.

- Ist $Ax = b$ lineares System mit $A \in \mathbb{R}^{n \times n}$ invertierbar, dann ist

$$x = A^{-1}b$$

die **Lösung** des linearen Systems. Diese Lösung ist **eindeutig**.

Invertierbarkeit von Matrizen

Problem 1: wann ist eine Matrix $A \in \mathbb{R}^{n \times n}$ invertierbar?

- man findet explizit ein Inverses A^{-1}
 - für kleine Matrizen geeignet (z.B. 2×2)
- $\det(A) \neq 0$
- $\ker(A) = \{0\}$
- $\text{rank}(A) = n$
- $Ax = 0$ hat nur die triviale Lösung $x = 0$

Problem 2: falls A invertierbar, wie findet man das Inverse A^{-1} ?

- Berechnung mit Gauss-Jordan Elimination
- Umweg über Zerlegungen von A (z.B. LU-Zerlegung)
- **generell:** Berechnung der Inversen meist numerisch nicht stabil

Ansätze zum Lösen linearer Systeme

Gegeben sei lineares System $Ax = b$ mit $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$ und $x \in \mathbb{R}^n$.

- direktes Lösen über **Inverse** A^{-1}
 - numerisch oft nicht stabil
 - sehr effizient (besonders falls für mehrere b gelöst wird)
- Lösen über **Zerlegung** von A (z.B. $A = LU$)
 - numerisch stabil
 - sehr effizient (besonders falls für mehrere b gelöst wird)
- **iterative Algorithmen**, schrittweise Annäherung an Lösung
 - besonders geeignet für sehr grosse n
 - auch geeignet falls keine Lösung existiert, berechnet dann z.B. Approximation

$$\min \|Ax - b\|_2^2$$

Gauss Elimination

Sei $Ax = b$ lineares System mit $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$ und $x \in \mathbb{R}^n$.

- **Gauss-Elimination:** Überführung der Matrix A in **Dreiecksform** durch **elementare Zeilenumformungen**
- **Gauss-Jordan-Elimination:** Überführung der Matrix A in **Diagonalform** durch **elementare Zeilenumformungen**
- **Elementare Zeilenumformungen:**
 - Vielfaches einer Zeile zu einer anderen Zeile addieren
 - zwei Zeilen vertauschen
 - Vielfaches einer Zeile berechnen
- für lineare Systeme: Gauss-Elimination auf erweiterter Matrix

$$(A|b) = \left(\begin{array}{ccc|c} a_{11} & \cdots & a_{1n} & b_1 \\ \vdots & & \vdots & \vdots \\ a_{n1} & \cdots & a_{nn} & b_n \end{array} \right)$$

Gauss Elimination: Beispiel

$$2x + y - z = 8 \quad (G_1)$$

$$-3x - y + 2z = -11 \quad (G_2)$$

$$-2x + y + 2z = -3 \quad (G_3)$$

$$\left(\begin{array}{ccc|c} 2 & 1 & -1 & 8 \\ -3 & -1 & 2 & -11 \\ -2 & 1 & 2 & -3 \end{array} \right) \xrightarrow{G_2 + \frac{3}{2}G_1} \left(\begin{array}{ccc|c} 2 & 1 & -1 & 8 \\ 0 & \frac{1}{2} & \frac{1}{2} & 1 \\ -2 & 1 & 2 & -3 \end{array} \right)$$

$$\xrightarrow{G_3 + G_1} \left(\begin{array}{ccc|c} 2 & 1 & -1 & 8 \\ 0 & \frac{1}{2} & \frac{1}{2} & 1 \\ 0 & 2 & 1 & 5 \end{array} \right) \xrightarrow{G_3 + (-4)G_2} \left(\begin{array}{ccc|c} 2 & 1 & -1 & 8 \\ 0 & \frac{1}{2} & \frac{1}{2} & 1 \\ 0 & 0 & -1 & 1 \end{array} \right)$$

$$\xrightarrow{G_1/2, G_2 \cdot 2, G_3 \cdot (-1)} \left(\begin{array}{ccc|c} 1 & \frac{1}{2} & -\frac{1}{2} & 4 \\ 0 & 1 & 1 & 2 \\ 0 & 0 & 1 & -1 \end{array} \right)$$

Lösung nun durch **Rücksubstitution**: $z = -1$, $y = 3$ und $x = 2$

Berechnung der Inversen mit Gauss-Jordan Elimination

Sei $A \in \mathbb{R}^{n \times n}$.

- Bilde erweiterte Matrix

$$(A|I_n) = \left(\begin{array}{ccc|ccc} a_{11} & \cdots & a_{n1} & 1 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} & 0 & \cdots & 1 \end{array} \right)$$

wobei I_n die $n \times n$ Identitätsmatrix ist.

- Führe Gauss-Jordan Elimination für A aus (linker Teil) bis auf I_n reduziert, repliziere elementare Zeilenumformungen für I_n (rechter Teil).
- Am Ende entspricht rechter Teil A^{-1} .

Günstige Matrix-Formen

Sei $Ax = b$ lineares System mit $A \in \mathbb{R}^{n \times n}$, $x, b \in \mathbb{R}^n$.

- **A diagonal:** Lösung kann abgelesen werden

$$\begin{pmatrix} a_{11} & & 0 \\ & \ddots & \\ 0 & & a_{nn} \end{pmatrix} x = b$$

- **A obere Dreiecksmatrix:** Lösung durch Rückwärts-Substitution

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ & \ddots & \vdots \\ 0 & & a_{nn} \end{pmatrix} x = b$$

- **A untere Dreiecksmatrix:** Lösung durch Vorwärts-Substitution

$$\begin{pmatrix} a_{11} & & 0 \\ \vdots & \ddots & \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} x = b$$

Rückwärts-Substitution

Lineares System in oberer Dreiecksform:

$$\begin{aligned} a_{11}x_1 + a_{21}x_2 + \dots + a_{1,n-1}x_{n-1} + a_{1n}x_n &= b_1 \\ a_{22}x_2 + \dots + a_{2,n-1}x_{n-1} + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n &= b_{n-1} \\ a_{n,n}x_n &= b_n \end{aligned}$$

Rückwärts nacheinander nach x_n, x_{n-1}, \dots, x_1 auflösen:

$$x_i = \left(b_i - \sum_{j=i+1}^n a_{ij}x_j \right) / a_{ii}$$

Aufwand: n^2 arithmetische Operationen (FLOPs)

Vorwärts-Substitution

Lineares System in unterer Dreiecksform:

$$a_{11}x_1 = b_1$$

$$a_{21}x_1 + a_{22}x_2 = b_2$$

\vdots

$$a_{n-1,1}x_1 + a_{n-1,2}x_2 + \dots + a_{n-1,n-1}x_{n-1} = b_{n-1}$$

$$a_{n1}x_1 + a_{n2}x_2 + \dots + a_{n,n-1}x_{n-1} + a_{nn}x_n = b_n$$

Vorwärts nacheinander nach x_1, x_2, \dots, x_n auflösen:

$$x_i = \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j \right) / a_{ii}$$

Aufwand: n^2 FLOPs

Zerlegungen von Matrizen: Cholesky

Cholesky Zerlegung:

- Sei $A \in \mathbb{R}^{n \times n}$ symmetrisch ($A^T = A$) und positiv definit ($\langle x, Ax \rangle > 0$ für alle $x \in \mathbb{R}^n$, $x \neq 0$).
- Dann gibt es eine untere Dreiecksmatrix $L \in \mathbb{R}^{n \times n}$ mit strikt positiven Diagonalelementen, so daß

$$A = LL^T.$$

- Aufwand: $\sim \frac{n^3}{3}$ FLOPs
- Lösung von $Ax = b$: betrachte $LL^T x = b$ und
 - löse $Lz = b$ mit Vorwärts-Substitution nach z auf
 - löse $L^T x = z$ mit Rückwärts-Substitution nach x auf

Zerlegungen von Matrizen: LUP

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

Zerlegungen von Matrizen: QR

QR Zerlegung:

- Sei $A \in \mathbb{R}^{n \times n}$ invertierbar.
- Dann gibt es eine orthogonale Matrix $Q \in \mathbb{R}^{n \times n}$ ($Q^{-1} = Q^T$) und eine obere Dreiecksmatrix $R \in \mathbb{R}^{n \times n}$ mit positiven Diagonaleinträgen, so daß

$$A = QR.$$

- Aufwand: $\sim \frac{2n^3}{3}$ FLOPs
- Lösung von $Ax = b$: löse $Rx = Q^T b$ mit Rückwärts-Substitution nach x auf

Zerlegungen von Matrizen: SVD

Singular Value Decomposition (SVD):

- Sei $A \in \mathbb{R}^{m \times n}$.
- Dann existiert eine Matrix $U \in \mathbb{R}^{m \times m}$ mit $U^T U = I_m$, eine orthogonale Matrix $V \in \mathbb{R}^{n \times n}$ ($V^{-1} = V^T$) sowie eine diagonale Matrix $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_n) \in \mathbb{R}^{m \times n}$ mit nicht-negativen Diagonalelementen in monoton fallender Reihenfolge $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$, so daß

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

Die σ_i heißen **Singulärwerte** von A .

- **Aufwand:** $2mn^2 + 2n^3$ FLOPs (kann je nach Implementation höher sein)
- **Lösen von $Ax = b$:** (funktioniert nur falls alle $\sigma_i \neq 0$!)

$$x = A^{-1}b = (U \Sigma V^T)^{-1}b = V \Sigma^{-1} U^T b$$

Anwendungs-Beispiel



separate Präsentation

Zusammenfassung

7 Fortgeschrittene Datenstrukturen

8 Such-Algorithmen

9 Graph-Algorithmen

10 Numerische Algorithmen

Matrizen

Lineare Gleichungen