Vorlesungsskript

Num. Lin. Algebra

Inhaltsverzeichnis

Definitionen

1. Einleitung

Wichtige Aufgabenklassen der linearen Algebra sind **lineare Gleichungssysteme**.

Gegeben: $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$

Gesucht: Ein/alle $x \in \mathbb{R}^m$ mit $Ax = b$

Herkunft:

- "direkt" aus der Anwendung, z.B. Beschreibung von Netzwerken, Tragwerk
- "indirekt" als Diskretisierung von stationären Prozessen, z.B. Belastung einer Membran
- "mittelbar" durch die Linearisierung nichtlinearer Modelle, z.B. Newton-Verfahren, Approximation von Lösungen gewöhnlicher DGL, notwendige Optimalitätsbedingungen

Klassifizierung:

- $m = n$: A quadratisch

  Generische Situation: A regulär

  $\Longrightarrow \exists!$ Lösung
- $m < n$: "Unterbestimmtes System"

  Generische Situation:

  $$\begin{aligned} \operatorname{rg}(A) &= m \ \text{(Vollrang)} \\ A &\mathrel{\widehat{=}} [A_1 A_2] \quad A_1 \in \mathbb{R}^{m \times m} \end{aligned}$$

  Lösungsmenge:

  $\mathcal{L} = \{x \in \mathbb{R}^n \mid Ax = b\} = \{x = x^+ + h, h \in \ker(A)\}$

  $=$ $(n - m)$-dimensionale lineare Mannigfaltigkeit

  Gesucht ist dann z.B. norm-minimale Lösung (Kap. 5)

- $m > n$: "Überbestimmtes System"

  $$\text{lösbar} \iff b \in \operatorname{im}(A) = \{y \in \mathbb{R}^m \mid \exists x : Ax = y\}$$

  Generisch nicht lösbar!

  Sinnvoll: Bestimme $\bar{x} \in \mathbb{R}^m$, so dass

  $$\|A\bar{x} - b\| = \min_{x \in \mathbb{R}^m} \|Ax - b\|$$

  $\| \, \| \mathrel{\widehat{=}}$ geeignete Norm, $\bar{x} \mathrel{\widehat{=}}$ Bestapproximierender für diese Norm.

  Mögliche Ansätze:

  - $\| \, \|_\infty$: $\|Ax - b\|_\infty = \max_{1 \leq i \leq m} \left|(Ax - b)_i\right|$
    Ein nichtglattes Optimierungsproblem auch als lineares Optimierungsproblem fomulierbar, schwierig zu lösen für m bzw. n groß.
  - $\| \, \|_1$: $\|Ax - b\|_1 = \sum_{i=1}^m |Ax - b|$
    Wie bei $\| \, \|_\infty$ stückweise lineares Optimierungsproblem.

    Aber stabil gegen Ausreißer.
  - $\| \, \|_2$: $\|Ax - b\|_2^2 = \sum_{i=1}^m \left((Ax - b)_i\right)^2$
    $\widehat{=}$ lineares Quadratmittelproblem, kleinste Quadrateproblem (Kap. 5)

Verfahren zur Lösung von LGS:

Direkte Verfahren:

- Transformation der Daten (A, b) in endlich viele in ein leichter zu lösendes LGS $\tilde{A}x = \tilde{b} \mathrel{\widehat{=}}$ CG-Verfahren
- Transformationen lassen sich oftmals als Faktorisierung von A interpretieren

  $$A = L \cdot R \quad \text{bzw.} \quad A = Q \cdot R$$
- Dafür i.d.R. Zugriff auf Elemente von $A \Longrightarrow$ limitiert die Größe der Matrix!

Kap. 2-5

Indirekte Verfahren:

- Ausgehend von einem Startvektor x^0 Iteration zur Berechnung von x^k mit $Ax^k \approx b$

  Hierbei wird oftmals nur das Matrix-Vektor-Produkt Av benötigt! (Kap. 6)
- Eigenwertprobleme

  Stabilitätsanalyse von Bauwerken. Verfahren dazu: numerische Optimierung

2. Das Gauß-Verfahren I

Jetzt: $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^n, \quad x : Ax = b?$

Satz 2.1: Existenz und Eindeutigkeit einer Lösung

Sei $A \in \mathbb{R}^{m \times n}$ eine Matrix mit $\det(A) \neq 0$ und $b \in \mathbb{R}^n$. Dann existiert genau ein $x \in \mathbb{R}^n$ mit

$$Ax = b$$

Beweis: lineare Algebra

□

$\Longrightarrow$ Anwendung von Algorithmen zur Berechnung von x sinnvoll! Wie?

2.1. Gaußsche Eliminationsverfahren und LR-Zerlegung

$\widehat{=}$ direktes Verfahren für quadratische System

Erste Idee: Systeme spezieller Struktur, z.B.

$$Rx = c, \quad R = \begin{pmatrix} r_{11} & \dots & r_{1n} \\ 0 & \ddots & \dots \\ 0 & 0 & r_{nn} \end{pmatrix} \in \mathbb{R}^{n \times n}, \quad c = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix} \in \mathbb{R}^n$$

$Rx = c$

$$\begin{aligned} & r_{nn} x_n = c_n \Longrightarrow x_n = \frac{c_n}{r_{nn}}, \quad r_{nn} \neq 0 \\ & r_{n-1n-1} x_{n-1} + r_{n-1n} x_n = c_{n-1} \\ & x_{n-1} = \frac{c_{n-1} - r_{n-1n} x_n}{r_{n-1n-1}}, \quad r_{n-1n-1} \neq 0 \end{aligned}$$

Algorithmus 2.2: Rückwärtssubsitution

$$x_n = \frac{c_n}{r_{nn}} \quad \text{falls } r_{nn} \neq 0$$
$$\vdots$$
$$x_i = \frac{c_i - \sum_{j=i+1}^{n} r_{ij} x_j}{r_{ii}} \quad \text{falls } r_{ii} \neq 0$$
$$\vdots$$
$$x_1 = \frac{c_1 - \sum_{j=2}^{n} r_{1j} x_j}{r_{11}} \quad \text{falls } r_{11} \neq 0$$

Algo. 2.2 anwendbar, wenn $\det(R) \neq 0$ (vgl. Theo. 2.1)

Wichtiger Aspekt dieser Vorlesung: Aufwandsabschätzung

Aufwand: i-te Zeile je $n - i$ Additionen und Multiplikationen und 1 Division

insgesamt:

$$\sum_{i=1}^{n} (i-1) = \frac{n(n-1)}{2} = \mathcal{O}(n^2)$$

Addition und Multiplikationen und n Divsionen.

Landau-Symbol: $\mathcal{O}(.)$

$$f(n) = \mathcal{O}(g(n)) \Longleftrightarrow \exists c > 0 : |f(n)| \leq C|g(n)|$$

Für ein lineares Gleichungssystem der Form

$$Lx = z, \quad L = \begin{pmatrix} l_{11} & & 0 \\ \vdots & \ddots & \\ l_{n1} & \dots & l_{nn} \end{pmatrix} \in \mathbb{R}^{n \times n} \quad z \in \begin{pmatrix} z_1 \\ \cdots \\ z_n \end{pmatrix} \in \mathbb{R}^n$$

gibt es einen analogen Algorithmus:

$$x_1 = \frac{z_1}{l_{11}} \quad l_{11} \neq 0$$
$$\vdots$$
$$x_n = \frac{z_n - \sum_{i=1}^{n-1} l_{ni} x_i}{l_{nn}} \quad l_{nn} \neq 0$$

$\Longrightarrow$ Vorwärtssubstitution mit gleichem Aufwand $\mathcal{O}(n^2)$

Lösungsidee für ein allgemeines Gleichungssystem:

Faktorisiere $A = L \cdot R$ und berechne die Lösung x von $Ax = b$ durch

$$Ax = L\underbrace{Rx}_{=:z} = b$$

$$\begin{aligned} Lz = b \Longrightarrow z = L^{-1}b \quad &\text{Vorwärtssubstitution} \\ Rx = z \Longrightarrow x = R^{-1}z \quad &\text{Rückwärtssubstitution} \end{aligned}$$

Mit Aufwand: $\mathcal{O}(n^2)$

Frage: Wie berechnet man Zerlegung $A = L \cdot R$

Man generiert eine Folge von Matrizen:

$$A = A^{(1)} \longrightarrow A^{(2)} \longrightarrow ... \longrightarrow A^n = R$$

von Matrizen der Gestalt

$$A^{(k)} = \begin{pmatrix} a_{11}^{(1)} & \dots & \dots & \dots & \dots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \dots & \dots & \dots & a_{2n}^{(2)} \\ & 0 & \ddots & & & \vdots \\ & & 0 & a_{kk}^{(k)} & \dots & a_{kn}^{(k)} \\ & & & \vdots & & \vdots \\ & & & a_{nk}^{(k)} & \dots & a_{nn}^{(k)} \end{pmatrix}$$

Wie?

Sei $x = (x_1, ..., x_n)^T \in \mathbb{R}^n, x_k \neq 0 \mathrel{\widehat{=}} k$-Spalte

Definiere: $l_{jk} = \frac{x_j}{x_k}$

$$l_k = \left(\underbrace{0, ..., 0}_{k \text{ mal}}, l_{k+1k}, ..., l_{nk}\right)^T$$

$e_k = k$-ter Einheitsvektor

$$L_k = I_n - l_k e_k^T \in \mathbb{R}^{n \times n}$$

Dann gilt

$$L_k x = \begin{pmatrix} 1 & & & 0 \\ & \ddots & & \\ & & 1 & \\ & & -l_{k+1k} & \ddots & \\ & & \vdots & \\ & & -l_{nk} & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} x_1 \\ \vdots \\ x_k \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Jeder Eliminationsschritt $A^{(k)} \longrightarrow A^{(k+1)}$ lässt sich damit als Multiplikation mit einer Matrix $L_k \in \mathbb{R}^{n \times n}$ von links

$$A^{k+1} = L_k A^{(k)} = \begin{pmatrix} I_k & 0 \\ 0 \;\; * & I_{n-k} \end{pmatrix} \begin{pmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ A_{21}^{(k)} & A_{22}^{(k)} \end{pmatrix} = \begin{pmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ 0 & A_{22}^{(k+1)} \end{pmatrix} \quad * \in \mathbb{R}^{n-k,1}$$

Eine Matrix der Gestalt L_k heißt Frobeniusmatrix $\rightarrow$ weitere Eigenschaften siehe Übung .

Der Eliminationsschritt ist genau dann durchführbar wenn $a_{kk}^{(k)} \neq 0$ gilt. Angenommen, dies gilt, dann erhält man

$$L_n \cdots L_2 L_1 A = R$$

$$A = \underbrace{L_1^{-1} \cdots L_{n-2}^{-1} L_{n-1}^{-1}}_{=:L} R$$

Induktiv beweißt man

$$L = L_1^{-1} \cdots L_{n-1}^{-1} = \begin{pmatrix} 1 & & & & 0 \\ l_{21} & \ddots & & & \\ \vdots & l_{32} & \ddots & & \\ \vdots & \vdots & & & \\ \vdots & \vdots & & \ddots & \\ l_{n1} & l_{n2} & \dots & l_{nn-1} & 1 \end{pmatrix}$$

Durch diese Struktur kann der Speicherplatz für A zum Speichern von L und R genutzt werden!

Algorithmus 2.3: LR-Zerlegung

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

```
for i = 1, ..., n
   for j = i, ..., n
      for k = 1, ..., i − 1
         a_ij = a_ij − a_ik * a_kj
      end
   end
   for j = i + 1, ..., n
      for k = 1, ..., i − 1
         a_ji = a_ji − a_jk * a_ki
      end
      a_ji = a_ji / a_ii
   end
end
```

Aufwand für die Dreieckszerlegung $A = L \cdot R$

#Operationen =

$$\sum_{i=1}^{n-1}\left((n-i)^2+(n-i)\right)=\frac{1}{3}n^3-\frac{1}{2}n^2+\frac{1}{6}n+\frac{1}{2}n^2-\frac{1}{2}n$$
$$=\frac{1}{3}n^3+\mathcal{O}(n^2)=\mathcal{O}(n^3)$$

$\Longrightarrow$ kubischer Aufwand! Nur akzeptabel für moderates n!

Algorithmus 2.4: Gaußsche Eliminationsverfahren

Gegeben: $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$

1) Berechne $A = L \cdot R$ $\quad\mathcal{O}(n^3)$

2) Berechne z aus $Lz = b$ $\quad\mathcal{O}(n^2)$

3) Berechne x aus $Rx = z$ $\quad\mathcal{O}(n^2)$

$\Longrightarrow$ Gesamtaufwand (Operationen): $\mathcal{O}(n^3)$, (Speicherplatz): n^2+n

Vorteil der Faktorisierung:

Zerlegung (teuer) kann für mehrere rechte Seiten nachgenutzt werden.

2.2. Pivot-Strategien

Beispiel 2.5: Algo 2.4 kann selbst für einfache Schritte scheitern:

$$Ax=b,\quad x=\begin{pmatrix}w\\z\end{pmatrix},\quad A=\begin{pmatrix}0&1\\1&0\end{pmatrix},\quad \det(A)=-1,\quad b=\begin{pmatrix}c\\e\end{pmatrix}$$

$\Longrightarrow a_{11}=0 \quad ↯$

Bei der völlig äquivalenten Formulierung

$$\tilde{A}\tilde{x}=\tilde{b},\quad \tilde{x}=\begin{pmatrix}w\\z\end{pmatrix},\quad \tilde{A}=\begin{pmatrix}1&0\\0&1\end{pmatrix},\quad \det\left(\tilde{A}\right)=1,\quad \tilde{b}=\begin{pmatrix}e\\c\end{pmatrix}$$

funktioniert Algo 2.4 mit

$$\tilde{A}=I_2=L\cdot R$$
$$L=I_2\quad R=I_2$$

$\Longrightarrow$ Zeilenvertauschung in A und der rechten Seite, **nicht** in x bzw. $\tilde{x}$!

Die LR-Zerlegung versagt nicht nur bei verschwindenen Diagonalelementen, sondern auch wenn diese betragsmäßig klein im Vergleich zu den restlichen Elementen sind.

$\rightsquigarrow$ Praktikum, Fehlertheorie (Kap. III)

Algorithmus 2.6: LR-Zerlegung mit Spaltenpivotisierung

1. $k=1, A^{(1)}=A$
2. Spaltenpivotisierung

Bestimme $p \in \{k, ..., n\}$ so, dass

$$\left|a_{pk}^{(k)}\right| \geq \left|a_{jk}^{(k)}\right| \text{ für } j = k, ..., n$$

3. Vertausche die Zeilen p und k durch

$$A^{(k)} \longrightarrow \tilde{A}^{(k)} \quad \text{mit} \quad \tilde{a}_{ij}^{(k)} = \begin{cases} a_{kj}^{(k)} & \text{falls } i = p \\ a_{pj}^{(k)} & \text{falls } i = k \\ a_{ij}^{(k)} & \text{sonst} \end{cases}$$

4. Führen der Eliminationsschritte

$$\tilde{A}^{(k)} \longrightarrow A^{(k+1)} \quad \text{setzte } k = k+1$$

5. Falls $k = n$ STOP

   Sonst gehe zu 2.

Alternative Pivotisierungsstrategien:

- Zeilenpivotisierung und Spaltentausch
- vollständige Pivotisierung, d.h. Suche des betragsmäßig größten Elements in der Restmatrix

Aufwand:

- Sowohl Spalten- als auch Zeilenpivotisierung: Im schlimmsten Fall $\mathcal{O}(n^2)$ zusätzliche Operationen
- vollständige Pivotisierung: Im schlimmsten Fall $\mathcal{O}(n^3)$ zusätzliche Operationen

Formale Beschreibung von Algo 2.6? Dazu: Permutationsmatrizen $P_\pi \in \mathbb{R}^{n \times n}$

Jede Permutation $\pi : \{1, ..., n\} \longrightarrow \{1, ..., n\}$ der Zahlen $1, ..., n$ bestimmt eine Matrix

$$P_\pi = \begin{pmatrix} e_{\pi(1)} & \cdots & e_{\pi(n)} \end{pmatrix}$$

Eine Zeilenvertauschung in A kann dann durch das Produkt $P_\pi A$ beschrieben werden, Spaltenvertauschung durch AP_π. Des Weiteren gilt $P_\pi^{-1} = P_\pi^T$, $\det(P_\pi) = \{-1, 1\}$.

Man kann beweisen, dass die LR-Zerlegung mit Spaltenpivotisierung <u>theoretisch</u> nur versagen kann, wenn $\det(A) = 0$

Satz 2.7: Durchführbarkeit der LR-Zerlegung

Für jede invertierbare Matrix A existiert eine Permutationsmatrix P derart, dass für PA die LR-Zerlegung mit Spaltenpivotisierung durchgeführt werden kann. D.h., man erhält $PA = LR$. Dabei kann man P so wählen, dass alle Elemente von L betragsmäßig kleiner gleich 1 sind, also $|L| \leq 1$

Beweis: Da A invertierbar ist, gilt $\det(A) \neq 0$. Damit existiert eine Permutationsmatrix P_{π_1}, so dass das erste Diagonalelement $\tilde{a}_{11}^{(1)}$ der Matrix

$$\tilde{A}^{(1)} = P_{\pi_1} A^{(1)}$$

von Null verschieden ist und das betragsmäßig größte Element in der ersten Spalte ist:

$$0 \neq \left|\tilde{a}_{11}^{(1)}\right| \geq \left|\tilde{a}_{i1}^{(1)}\right| \quad \text{für } i = 1, ..., n$$

Nach dem ersten Eliminationsschritt erhalten wir

$$A^{(2)} = L_1 \tilde{A}^{(1)} = L_1 P_{\pi_1} A = \begin{pmatrix} \tilde{a}_{11}^{(1)} & * \\ 0 & \breve{A}_2^{(2)} \end{pmatrix}$$

Wegen (*) gilt für L_1:

$$|l_{i1}| = \left|\frac{\tilde{a}_{i1}^{(1)}}{\tilde{a}_{11}^{(1)}}\right| \leq 1 \quad i = 2, ..., n$$

$\Longrightarrow |L_1| \leq 1, \quad \det(L_1) = 1$

$$\begin{aligned} \det\left(A^{(2)}\right) &= \underbrace{\det(L_1)}_{=1} \underbrace{\det\left(P_{\pi_1}\right)}_{\in\{-1,1\}} \underbrace{\det(A)}_{\neq 0} \\ &\neq 0 \end{aligned}$$

$$\det\left(\breve{A}^{(2)}\right) = \frac{\overbrace{\det\left(A^{(2)}\right)}^{\neq 0}}{\tilde{a}_{11}^{(1)}} \neq 0$$

Induktiv erhält man

$$R = A^{(n)} = L_{n-1} R_{\pi_{n-1}} L_{n-2} P_{\pi_{n-2}} \cdots L_1 P_{\pi_1} A$$

mit $|L_k| \leq 1$ und P_{π_k} entweder die Identität oder zwei Zeilen $j_1, j_2 \geq k$ vertauschen. Deswegen gilt für die Frobeniusmatrix

$$L_k = \begin{pmatrix} 1 & & & & 0 \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{k+1k} & & \\ & & \vdots & \ddots & \\ & & -l_{nk} & & 1 \end{pmatrix}, \text{dass}$$

$$\tilde{L}_k = P_{\pi_j} L_k P_{\pi_j^{-1}} = \begin{pmatrix} 1 & & & & 0 \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{\pi_j(k+1)k} & & \\ & & \vdots & \ddots & \\ & & -l_{\pi_j(n)k} & & 1 \end{pmatrix} \quad \text{für } j > k$$

Durch geschicktes Einfügen von $I = P_{\pi_{k+1}}^{-1} P_{\pi_{k+1}}$

$$\begin{aligned} R = A^{(n)} = L_{k-1} \left(P_{\pi_{n-1}} L_{n-2} P_{\pi_{n-1}}^{-1} \right) \left(P_{\pi_{n-1}} P_{\pi_{n-2}} L_{k-3} P_{\pi_{n-2}}^{-1} P_{\pi_{n-1}}^{-1} \right) \\ P_{\pi_{n-1}} P_{\pi_{n-2}} \cdot \ldots \cdot (\ldots L_1 P_{\pi_1} \ldots P_{\pi_{n-1}}^{-1} \left(P_{\pi_{n-1}} \ldots P_{\pi_1} A \right) \end{aligned}$$

$\Longrightarrow PA = \underbrace{\tilde{L}_1^{-1} \cdots \tilde{L}_{n-1}^{-1}}_{=:L} R$ mit

$$L = \begin{pmatrix} 1 & & & 0 \\ l_{\tilde{\pi}_1(l)1} & \ddots & & \\ & \ddots & \ddots & \\ \vdots & & & \\ l_{\tilde{\pi}_1(n)1} & \cdots & l_{\tilde{\pi}_{n-1}(n)(n-1)} & 1 \end{pmatrix}$$

und $|L| \leq 1$

Bemerkungen:

- Gilt $PA = LR$, dann berechnet man

$$\begin{aligned} Ax &= b \\ PAx &= Pb \\ LRx &= Pb \\ x &= R^{-1} L^{-1} Pb \end{aligned}$$

- Theoretisch sind die Formulierungen

$$Ax = b \qquad DAx = Db$$

für eine invertierbare Diagonalmatrix D äquivalent. Bei der praktischen Lösung auf dem Rechner haben solche Skalierungen aber u.U. einen **dramatischen** Einfluß, vgl. Kap. III.
- Auf dem Rechner: Verbesserung der unexakten Lösung durch sogenannte Nachiteration möglich, vgl. Kap. IV.

2.3. Cholesky-Verfahren für symm. pos. definite A

Gesucht: A spd eine $L \in \mathbb{R}^{n \times n}$ ($\det(L) > 0$) s.d. $A = LL^T$

siehe Übungen

3. Fehleranalyse

Situation

ideal: Eingabe $x \longrightarrow$ Algorithmus/Problemstellung $f \longrightarrow$ Ausgabe $y = f(x)$

real: $\tilde{x} = x + \varepsilon \longrightarrow \tilde{f} \longrightarrow \tilde{y} = \tilde{f}(\tilde{x})$

Frage: $y \longleftrightarrow \tilde{y}$?

Ursachen für den Gesamtfehler $\tilde{y} - y$

- **Modellfehler**
  - Idealisierungsfehler, z.B. in der Modellbildung
  - Datenfehler

  Modellfehler lassen in der Regel nicht vermeiden!

  Frage: Wie wirken sich solche Fehler **unabhängig** vom gewählten Algorithmus aus?

$$f(x) \longleftrightarrow f(\tilde{x})$$

  Kondition eines Problems
- **numerische Fehler**
  - Diskretisierungsfehler, kontinuierliches Problem versus diskretisierte Formulierung
  - Abbruchfehler, eigentlich unendliche Algorithmen werden nach endlichen Schritten abgebrochen
  - Approximationsfehler

$$\sin(x) = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!}$$
$$\Longrightarrow \widetilde{\sin}(x) = \sum_{n=0}^{k} (-1) \frac{x^{2n+1}}{(2n+1)!}$$

  - Rechengenauigkeit, reelle Zahlen versus Gleitkommazahlen

Rundungsfehler und Approximationsfehler $\Longrightarrow$ **Stabilität** eines Alogrithmus

$$f(x) \longleftrightarrow \tilde{f}(x)$$

Vernachlässigung von Fehlerbetrachtungen kann dramatische Auswrikungen haben:

- 1991: Untergang der Bohrinsel Sleipner, Fehler in den Kräften von 47%
- 1. Golfkrieg: Eine Patriotrakete verpasst angreifende Rakete. Im Steuerprogramm der Patriotrakete durch Multiplikation mit 0.1. Nach 100 Betriebsstunden: Differenz der berechneten Zeit und tatsächlich vergangener Zeit von 0.34 Sekunden
- Absturz der ersten Ariane 5 Rakete (1996), Umwandlung einer 64 bit Gleitkommazahl in 16 bit integer Zahl in Software der Arian 4
- London Millenium Bridge (2000), flasche Abschätzung der Fußgängerkräfte

3.1. Zahlendarstellung und Rundungsfehler

$\longrightarrow$ Einführung in das wissenschaftliche Rechnen

3.2. Kondition eines Problems

Erwartungshaltung: kleiner Fehler in der Aufgabenstellung ($x \to \tilde{x}$ verursacht einen kleinen Fehler in der Lösung $\tilde{y}$

Beispiel 3.1: Störung eines LGS

Gegeben ist das lineare Gleichungssystem

$$\underbrace{\begin{pmatrix} 1.2969 & 0.8648 \\ 0.2161 & 0.1441 \end{pmatrix}}_{=:A} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \underbrace{\begin{pmatrix} 0.8642 \\ 0.1440 \end{pmatrix}}_{=:b}$$

mit $\det(A) \neq 0$ und der eindeutig bestimmten Lösung $x = \begin{pmatrix} 2 & 2 \end{pmatrix}^T$.

Jetzt: Störung der rechten Seite

$$b \rightsquigarrow \tilde{b} = \begin{pmatrix} 0.86419999 \\ 0.14400001 \end{pmatrix}$$

liefert die Lösung $\tilde{x} = \begin{pmatrix} 0.9911 & -0.4870 \end{pmatrix}^T$. Ursache?

Dazu: Formalisierung Eigenschaften der Problemstellung

Wichtig: Notation: x - Eingabe, f - Problemstellung, y - Ausgabe

Definition 3.2: Numerisches Problem

Ein numerisches Problem ist ein Paar (f, x) wobei $f : D \subset \mathbb{R}^n \to \mathbb{R}^m$ eine Abbildung, $x \in D$ die Eingabe und $y = f(x)$ die Ausgabe ist.

Beispiel 3.3:

- Auswertung von $\sin(x)$: $x = 1.7, y = f(x) = \sin(x) = \sin(1.7)$
- Bestimmung von Nullstelle von $g(t) = at^2 + bt + c$

  Eingabe: $x = (a, b, c), y = f(x)$ definiert durch $g(f(a, b, c)) \overset{!}{=} 0$

Zur Lösung eines numerischen Problems können verschiedene Algorithmen genutzt werden

(Algorithmus: endliche Folge von Elementaroperationen, deterministisch bestimmt)

Hier: Die Kondition ist **unabhängig** vom gewählten Algorithmus!

Definition 3.4: wohl gestelltes Problem, schlecht gestelltes Problem

Das numerische Problem (f, x) heißt wohlgestellt, falls es eine konstante $L_{\text{abs}} \in \mathbb{R}^+$ gibt, so dass

$$\|f(\tilde{x}) - f(x)\| \leq L_{\text{abs}} \|x - \tilde{x}\|$$

für alle $\tilde{x} \to x$. Existiert keine solche Konstante L_{abs}, dann heißt (f, x) schlecht gestellt. Zur weiteren Analyse setzt man im wohldefinierten Fall

$$\kappa_{\text{abs}} := \inf\{L_{\text{abs}} \mid L_{\text{abs}} \geq 0 \text{ und } (*) \text{ gilt}\}$$

Gilt $x \neq 0 \neq f(x)$, definiert man analog κ_{rel} als die kleinste Konstante mit

$$\frac{\|f(\tilde{x}) - f(x)\|}{\|f(x)\|} \leq \kappa_{\text{rel}} \frac{\|\tilde{x} - x\|}{\|x\|}$$

für alle $\tilde{x}$ nahe x.

Bemerkungen:

- Die **absolute Kondition** κ_{abs} beschreibt die Verstärkung des absoluten Fehlers, die **relative Kondition** κ_{rel} die Verstärkung des relativen Fehlers
- Bei nichtlinearen Problemen hängen κ_{abs} und κ_{rel} meist stark von der Umgebung ab $\Longrightarrow$ linearisierte Fehlertheorie!
- κ_{abs} und κ_{rel} hängen stark von den verwendeten Normen ab! $\|\cdot\|_2, \|\cdot\|_\infty, \|\cdot\|_p, \|\cdot\|_1$

Definition 3.5: absolute und relative Kondition

Die Konstante κ_{abs} gibt die absolute Kondition eines numerischen Problems (f, x) und κ_{rel} die relative Kondition.

Das numerische Problem (f, x) ist **schlecht konditioniert**, wenn κ_{abs} bzw. κ_{rel} "groß" sind und gut konditioniert, wenn κ_{abs} bzw. κ_{rel} "klein" sind.

Wie berechnet man $\kappa_{\text{abs}}/\kappa_{\text{rel}}$? Dafür: Mittelwertsatz der Differentialrechnung

Es sei $f : [a, b] \to \mathbb{R}$ stetig auf $[a, b]$ und diffbar auf (a, b). Dann existiert $\bar{x} \in (a, b)$, so dass

$$f'(\bar{x}) = \frac{f(b) - f(a)}{b - a}$$

Anwendung in der Fehlertheorie: Für differenzierbare $f : \mathbb{R}^n \to \mathbb{R}$ existiert wegen der Taylorentwicklung für x und Δx ein $\bar{x} \in x + t\Delta x, t \in (0,1)$ mit

$$\Delta y := \tilde{y} - y = f(x + \Delta x) - f(x) = \nabla f(\bar{x})\Delta x$$

$\Longrightarrow \|\nabla f(\bar{x})\|$ ist ein Fehlermaß $\sim\to x$.

Deswegen verwendet man den Term $\|\nabla f(x)\|$ als Maß für die Fehlerverstärkung des absoluten Eingabefehlers $\|\Delta x\| = \|\tilde{x} - x\|$.

Der relative Fehler ist meist von größerer Bedeutung. Für $n = 1$ und $x \cdot y \neq 0$

$$\frac{\Delta y}{y} \approx \nabla f(x)\frac{\Delta x}{y} = \underbrace{\left(\nabla f(x)\frac{x}{f(x)}\right)}_{=\kappa_{\text{rel}}}\frac{\Delta x}{x}$$

Verallgemeinerung auf $n > 1$: $\kappa_{\text{rel}} = \left|\nabla f(x)^T x \cdot \frac{1}{f(x)}\right|$o **Beispiel 3.6:** Kondition der Addition

Problem: $f : \mathbb{R}^2 \to \mathbb{R}, f(x_1, x_2) = x_1 + x_2$ mit der l_1-Norm

$\nabla f(x) = (1,1)^T \Longrightarrow$

$$\kappa_{\text{abs}} = \|\nabla f(x)\|_1 = \left\|(1 \;\; 1)^T\right\|_1 = 2$$
$$\kappa_{\text{rel}} = \|\nabla f(x)^T x \frac{1}{f(x)}\|_1 = \frac{|x_1| + |x_2|}{|x_1 + x_2|}$$

Für die Addition zweier Zahlen mit gleichen Vorzeichen ergibt sich $\kappa_{\text{rel}} = 2 \Longrightarrow$ gut konditioniert!