

# Numerik I

WiSe22/23

Prof. Dr. Luise Blank

12. Dezember 2022



# Inhaltsverzeichnis

<b>1</b>	<b>Einführung</b>	<b>3</b>
<b>2</b>	<b>Lineare Gleichungssysteme: Direkte Methoden</b>	<b>9</b>
2.1	Gaußsches Eliminationsverfahren	9
2.2	Gaußsches Eliminationsverfahren mit Pivotisierung	15
<b>3</b>	<b>Fehleranalyse</b>	<b>21</b>
3.1	Zahlendarstellung und Rundungsfehler	21
3.2	Kondition eines Problems	25
3.3	Stabilität von Algorithmen	33
3.4	Beurteilung von Näherungslösungen linearer Gleichungssysteme	38
<b>4</b>	<b>Lineare Gleichungssysteme: Direkte Methoden (Fortsetzung)</b>	<b>41</b>
4.1	Gaußsches Eliminationsverfahren mit Äquilibration und Nachiteration	41
4.2	Cholesky-Verfahren	42
4.3	Lineare Ausgleichsprobleme	45
4.4	Orthogonalisierungsverfahren	49
<b>5</b>	<b>Numerische Lösung nichtlinearer Gleichungssysteme</b>	<b>57</b>
5.1	Einführung	57
5.2	Fixpunktiteration	58
5.3	Konvergenzordnung und Fehlerabschätzungen	61
5.4	Newton-Verfahren für skalare Gleichungen	63
5.5	Das Newton-Verfahren im Mehrdimensionalen	66
5.6	Abbruchkriterien beim Newton-Verfahren	68
5.7	Varianten des Newton-Verfahrens	69



Dieses Skript ist eine Überarbeitung des freundlicherweise von Gesine Schwalbe erstellten Skriptes.

Es wird keinerlei Anspruch auf Richtigkeit und Vollständigkeit dieses Dokuments erhoben.  
Fehler und Anmerkungen bitte an:

**luise.blank@ur.de**



# Kapitel 1

## Einführung

17.10.22

### Wozu?

- viele Probleme mit gleicher Struktur sind zu lösen, z.B.
  - $ax^2 + bx + c = 0 \Rightarrow x_{\pm} = -\frac{b}{2a} \pm \frac{1}{2a} \sqrt{b^2 - 4ac}$
  - bestimme den größten gemeinsamen Teiler zweier Zahlen  
     $\leadsto$  euklidischer Algorithmus $\Rightarrow$  Algorithmus erwünscht
- ein Problem kann analytisch gelöst werden, aber es dauert zu lange bis das Ergebnis bestimmt ist  
z.B. bei der numerischen Simulation von Strömungen tauchen bis zu 1 Million Unbekannte auf. Es sind Systeme mit  $\approx 10^6$  Gleichungen zu lösen  
 $\Rightarrow$  effiziente Algorithmen notwendig,  
Näherungslösung häufig ausreichend
- ein Problem kann nicht analytisch gelöst werden. Z.B. ist bei Differentialgleichungen vielleicht Existenz- und Eindeutigkeit gewährleistet, aber eine konstruktive Methode zur Berechnung ist nicht bekannt.  
 $\Rightarrow$  Näherungslösung erwünscht

### Seit wann?

Algorithmen gibt es schon lange bevor es Rechner gab, z.B. den euklidischen Algorithmus seit um 300 v.Chr. Die Fragestellungen und der Blickwinkel verschieben sich jedoch in Abhängigkeit von den zu lösenden Problemen und der existierenden Computer-Hardware. Beispiele sind: Strömungsprobleme modelliert mit partiellen Differentialgleichungen, Big Data, Parallelrechner, Vektorrechner, größere Speicher, etc..

### Anwendungen

- Herd, Waschmaschine, Heizungsanlage
- Handy (über welchen Satelliten wird übertragen), Digitalkamera, MP3-Player
- Navigationssysteme
- Erstellung des Zugfahrplans
- Robotersteuerung (bis hin zu Roboterfußball)

- Fahrzeugindustrie
  - Fahrzeugbau (Crash-Simulation, Strömungsmodellierung)
  - Fahrzeugsteuerung
- Finanzmarkt z.B. Risikoanalyse im Wertpapierhandel
- Klimaanalyse z.B. Vorhersage von Erdbeben, Hurrikans, Überflutungen
- Medizinische Versorgung z.B. Bildverarbeitung, Prognose für Epidemieentwicklungen, Beschreibung des Blutkreislaufs
- Raffinerie-Industrie
- Kontrolle und Optimierung chemischer und biologischer Prozesse, z.B. Regelung von Wärmezufuhr
- u.s.w.

## Fragestellungen

1. Rechengeschwindigkeit, Rechenaufwand (Anzahl der Rechenoperationen, Rechenzeit auf welchem Rechner...), Komplexität des Algorithmus  
 Beispiel: Berechnung der Lösung eines Gleichungssystems  

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$
 mit einer  $n \times n$ -Matrix  $\mathbf{A}$ 
  - (a) Cramersche Regel:  
 $x_j = \frac{\det(A)_j}{\det A}$  (ersetze die  $j$ -te Spalte von  $A$  durch  $b$ )  
 und  $\det A = \sum_{\pi} \text{sign}(\pi) a_{1m_1} \cdot \dots \cdot a_{nm_n}$   
 Benötigt etwa  $n!$  Multiplikationen und Additionen. Bei einer  $20 \times 20$ -Matrix  $\mathbf{A}$  (was heutzutage klein ist) wären dies  
 $\approx 2,5 \cdot 10^{18}$  Operationen. Falls jede arithmetische Operation  $10^{-6}$  Sekunden (also eine Mikrosekunde) benötigt, ist eine Rechenzeit von mehr als *eine Millionen Jahre* nötig!
  - (b) Gaußsches Eliminationsverfahren:  
 Benötigt etwa  $n^3$  Operationen, also ungefähr 8000 Operationen und weniger als *0,005 Sekunden* (Golub, Ortega: Scientific Computing [6]).
2. Verhalten bei Störungen, Stabilität des Verfahrens (Eingabefehler, Rundungsfehler, Diskretisierungsfehler)  
 Beispiele:
  - (a)  $\frac{1}{10^{-8}} = 10^8$  mit Störung des Nenners  $\frac{1}{2 \cdot 10^{-8}} = 5 \cdot 10^7$   
 $\leadsto$  kleine Störung im Nenner kann zu großen Störungen im Ergebnis führen.
  - (b)  $x^2 + 314x - 2 = 0$ . Falls diese Gleichung mit der  $p, q$ -Formel (Mitternachtsformel) gelöst wird und immer auf 5 signifikante Stellen gerundet wird, ergibt sich ein

$$\text{relativer Fehler} := \frac{|\text{Fehler}|}{|\text{Lösung}|}$$

von  $\approx 57\%$ .

Eine geschickte Formatierung liefert ein Ergebnis mit einem relativen Fehler von  $\approx 1,5 \cdot 10^{-5}$ , d.h. die ersten 4 Stellen sind exakt. Dabei werden für  $ax^2 + bx + c = 0$  folgende Ausdrücke verwendet:

$$\begin{aligned} x_1 &= \frac{1}{2a} (-b - \text{sign}(b) \sqrt{b^2 - 4ac}) \\ x_2 &= \frac{2c}{-b - \text{sign}(b) \sqrt{b^2 - 4ac}} \end{aligned}$$

Die Lösung ist 0.0063693, das Resultat mit der p,q-Formel lautet 0.01 und die letzte Formel liefert 0.0063692.



### 3. Genauigkeit des Verfahrens, Fehleranalyse, Konvergenzgeschwindigkeit, Konvergenzordnung.

Beispiel: Numerische Approximation von Ableitungen.

Für  $f \in C^3(I)$  gilt die Taylor-Entwicklung:

$$f(x \pm h) = f(x) \pm h f'(x) + \frac{h^2}{2} f''(x) + R(x) \text{ mit } |R(x)| \leq ch^3.$$

(a) *Vorwärtsgenommener Differenzenquotient:*

$$(D_h^+ f)(x) := \frac{f(x+h) - f(x)}{h} \approx f'(x),$$

$$\left| f'(x) - \frac{f(x+h) - f(x)}{h} \right| \leq ch,$$

konvergiert also mit linearer Abhängigkeit von der Schrittweite  $h$ .

(b) *Zentraler Differenzenquotient:*

$$(D_h^0 f)(x) := \frac{f(x+h) - f(x-h)}{2 \cdot h} \approx f'(x),$$

$$\left| f'(x) - \frac{f(x+h) - f(x-h)}{2 \cdot h} \right| \leq ch^2,$$

konvergiert mit quadratischer Ordnung bei gleichem Aufwand!

## Some disasters attributable to bad numerical computing

(Last modified August 26, 1998 by Douglas N. Arnold, arnold@ima.umn.edu)

Have you been paying attention in your numerical analysis or scientific computation courses? If not, it could be a costly mistake.

Here are some real life examples of what can happen when numerical algorithms are not correctly applied.

- The Patriot Missile failure, in Dhahran, Saudi Arabia, on February 25, 1991 which resulted in 28 deaths, is ultimately attributable to poor handling of *rounding errors*.
- The explosion of the Ariane 5 rocket just after lift-off on its maiden voyage off French Guiana, on June 4, 1996, was ultimately the consequence of a *simple overflow*.
- The sinking of the Sleipner A offshore platform in Gandsfjorden near Stavanger, Norway, on August 23, 1991, resulted in a loss of nearly one billion dollars. It was found to be the result of *inaccurate finite element analysis*.

## Weitere praxisrelevante Fragestellungen:

- soll ein *black box solver* genutzt werden oder eine Lösungsmethode entwickelt werden, welche auf das spezielle Problem angepaßt ist,
- wie teuer ist die Implementierung (wieviel Arbeitszeit),
- ist der implementierte Algorithmus vielseitig einsetzbar, (welche Problemklassen deckt er ab, welche Rechnerstruktur ist vorausgesetzt)?

## 'Gute' Programme sind

- zuverlässig (fehlerfrei)
- robust (z.B. behandeln Ausnahmesituationen und filtern ungeeignete Daten heraus)
- portierbar auf andere Rechenanlagen

Abbildung 1.1: Bandmatrix

$$\begin{pmatrix} * & * & * & & & 0 \\ * & * & * & * & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & * & * & * & * \\ & & & * & * & * \\ 0 & & & & * & * \end{pmatrix}$$

- wartungsfreundlich (leicht zu "ändern oder zu erweitern)
- gut dokumentiert
- ausgiebig getestet

Dies soll durch die Programmieraufgaben trainiert werden.

Eine Faustregel der numerischen Mathematik:

Zu jedem noch so eleganten numerischen Verfahren gibt es ein Gegenbeispiel, für welches die Methode völlig versagt (Teubner Taschenbuch).

## Welche Probleme werden hier behandelt?

### 1. Lineare Gleichungssysteme $Ax = b$

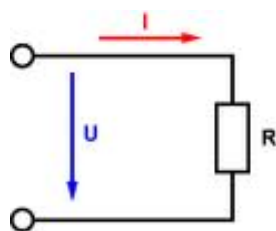
- 'kleine' bis 'mittelgroße' Matrizen  
→ **direkte Methoden:** nach endlich vielen Schritten ist die *exakte Lösung* bis auf Rundungsfehler berechnet (z.B. Gauß-Elimination)
- strukturierte Matrizen  
Symmetrie, Bandstruktur,
- große Matrizen (mit zusätzlichen Eigenschaften)  
→ **iterative Methoden:** kenne Startwert  $x_0$ , berechne neue Approximation  $x_i$  unter Ausnutzung der vorherigen bis die *Näherungslösung*  $x_i$  'gut genug' ist'.

### 2. Lineare Ausgleichsprobleme

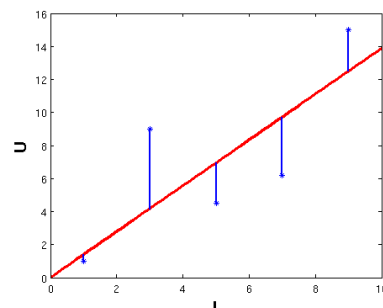
Beispiel: für die Spannung  $U$  und die Stromstärke  $I$  gilt das Ohm'sches Gesetz:  $U = R \cdot I$ .  $(I_i, U_i)$  seien die

Abbildung 1.2: Lineares Ausgleichsproblem

(a) Einfacher Stromkreis



(b) Messwerte linear approximiert



Meßdaten mit möglichen Meßfehlern in  $U$ . Gesucht ist der Widerstand  $R$ . Finde  $R$ , sodass gilt  $f(R) = \min_r \sum_i (U_i - r I_i)^2$ .

3. *Lösung nichtlinearer Gleichungen*, z.B.

- Berechnung von Nullstellen  $g(x) = 0$ ,
- Berechnung von Fixpunkten  $f(x) = x$ .

4. *Eigenwertwertberechnung*

$$A\mathbf{x} = \lambda\mathbf{x}, \quad \lambda \in \mathbb{C}.$$

5. *Interpolation*

Setze Meßdaten zu einer kontinuierlichen Funktion fort, z.B. polynomial, stückweise konstant, stückweise linear, oder, falls sie eine Schwingung repräsentieren, berechne die zugehörige Fourierreihe

6. *Berechnung von Integralen* (Quadraturformeln)

Approximation von

$$\int_a^b f(x) dx .$$

Bei allem spielt die *Fehleranalyse* eine große Rolle und die Grundbegriffe werden in einem extra Abschnitt behandelt.



## Kapitel 2

# Lineare Gleichungssysteme: Direkte Methoden

Sei  $A \in \mathbb{R}^{n \times n}$  und  $b \in \mathbb{R}^n$ . Gesucht ist  $x \in \mathbb{R}^n$  mit

$$Ax = b$$

Die Existenz und Eindeutigkeit einer Lösung seien vorausgesetzt.

Anmerkung: Ein verlässlicher Lösungsalgorithmus überprüft dies und behandelt alle Fälle.

Bekannte Ansätze aus der linearen Algebra:

- Die Cramersche Regel ist ineffizient (siehe Einleitung).
- Die Inverse  $A^{-1}$  für  $x = A^{-1}b$  aufzustellen ist ebenso ineffizient. Dieses würde dem Lösen für alle  $b \in \mathbb{R}^n$  entsprechen.

!Invertieren von Matrizen vermeiden!  
Lineare Gleichungssysteme werden gelöst!

## 2.1 Gaußsches Eliminationsverfahren

Das Verfahren wurde 1809 von Friedrich Gauß, 1759 von Joseph Louis Lagrange beschrieben und war seit dem 1. Jhd. v. Chr. in China bekannt.

Das Gaußverfahren löst ein lineares Gleichungssystem  $Ax = b$  mit  $A = (a_{ij})_{i,j \leq n} \in K^{n \times n}$  und  $b = (b_i)_{i \leq n} \in K^n$ .

Der Algorithmus sieht folgendermaßen aus:

Bearbeite die erste Spalte ( $A \rightarrow A^{(1)}, b \rightarrow b^{(1)}$ ): für alle Zeilen  $i = 2, \dots, n$

$$\begin{array}{cccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ \vdots & & \vdots & & & & \vdots & & \vdots \\ a_{i1}x_1 & + & a_{i2}x_2 & + & \cdots & + & a_{in}x_n & = & b_i \quad | \quad \text{ite Zeile} - 1\text{te Zeile} \cdot \frac{a_{i1}}{a_{11}} \\ \vdots & & \vdots & & & & \vdots & & \vdots \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array}$$

Es ergibt sich  $A^{(1)}$  und  $b^{(1)}$  mit

$$\begin{array}{lll} a_{1j}^{(1)} & = & a_{1j} \quad \text{für } j = 1, \dots, n \\ a_{i1}^{(1)} & = & 0 \quad \text{für } i = 2, \dots, n \\ a_{ij}^{(1)} & = & a_{ij} - a_{1j} \frac{a_{i1}}{a_{11}} \quad \text{für } i, j = 2, \dots, n \\ b_i^{(1)} & = & b_i - b_1 \frac{a_{i1}}{a_{11}} \quad \text{für } i = 2, \dots, n \end{array}$$

Dann bearbeite die zweite Spalte ( $A^{(1)} - A^{(2)}, b^{(1)} \rightarrow b^{(2)}$ ): für jede Zeile  $i = 3, \dots, n$

$$\begin{array}{cccccc} a_{11}x_1 & + & a_{12}x_2 & + & \dots & + & a_{1n}x_n & = & b_1 \\ & & a_{22}^{(1)}x_2 & + & \dots & + & a_{2n}^{(1)}x_n & = & b_2^{(1)} \\ & & \vdots & & & & \vdots & & \vdots \\ & & a_{n2}^{(1)}x_2 & & & & a_{nn}^{(1)}x_n & = & b_n^{(1)} \end{array} \quad \left| \begin{array}{l} \text{ite Zeile} - 2\text{te Zeile} \cdot \frac{a_{i2}}{a_{22}^{(1)}} \end{array} \right.$$

und so weiter.

In Schritt  $A^{(k-1)}$  zu  $A^{(k)}$ ,  $b^{(k-1)}$  zu  $b^{(k)}$  werden die Einträge der  $k$ -ten Spalte unterhalb der Diagonalen (also  $k = 1, \dots, n-1$ ) eliminiert durch:

$$(i\text{-te Zeile}) - (k\text{-te Zeile}) \cdot \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} \quad \text{für } i = k+1, \dots, n$$

Dadurch bleiben die ersten  $k$ -Zeilen und die ersten  $(k-1)$  Spalten von  $A^{(k-1)}$  erhalten und es gilt  $a_{ik}^{(k)} = 0$  für  $i = k+1, \dots, n$ . Diese Werte müssen nicht berechnet werden. Die restlichen Einträge werden somit berechnet durch: setze für die Zeilen  $i = k+1, \dots, n$

$$l_{ik} := \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}}, \quad (2.1.1)$$

$$a_{ij}^{(k)} := a_{ij}^{(k-1)} - a_{kj}^{(k-1)} l_{ik} \quad \text{für } j = k+1, \dots, n, \quad (2.1.2)$$

$$b_i^{(k)} := b_i^{(k-1)} - b_k^{(k-1)} l_{ik}. \quad (2.1.3)$$

So werden sukzessive die Spalten  $k = 1, \dots, n-1$  bearbeitet.

$$A \rightarrow A^{(1)} \rightarrow A^{(2)} \rightarrow \dots \rightarrow A^{(n-1)},$$

um mit  $A^{(n-1)}$  eine Matrix mit oberer Dreiecksgestalt zu erhalten.

$$\underbrace{\begin{pmatrix} a_{11} & \dots & \dots & a_{1n} \\ & a_{22}^{(1)} & \dots & a_{2n}^{(1)} \\ & & \ddots & \vdots \\ 0 & & & a_{nn}^{(n-1)} \end{pmatrix}}_{A^{(n-1)}=:R} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}}_{=:z} = \underbrace{\begin{pmatrix} b_1 \\ b_2^{(1)} \\ \vdots \\ b_n^{(n-1)} \end{pmatrix}}_{=:z} \quad (2.1.4)$$

Es gilt

$$Rx = z \text{ mit } R = \begin{pmatrix} \square \\ \circ \end{pmatrix}. \quad (2.1.5)$$

Dieser Prozess wird *Vorwärtselimination* genannt. Der zugehörige Algorithmus lautet:

---

**Algorithmus 2.1.1 :** Vorwärtselimination

---

```

for  $k = 1, \dots, n-1$  do
  for  $i = k+1, \dots, n$  do
     $l_{ik} = a_{ik}/a_{kk}$ 
    for  $j = k+1, \dots, n$  do
       $a_{ij} = a_{ij} - l_{ik}a_{kj}$ 
    end for
     $b_i = b_i - l_{ik}b_k$ 
  end for
end for

```

---



- Wird eine neue rechte Seite  $b$  betrachtet, muss 2.1.1 nicht komplett neu ausgeführt werden, da sich  $\tilde{L}$  nicht ändert. Es reicht (2.1.3) zu wiederholen.

**Notation 2.1.5.** Die *Dreieckszerlegung* einer Matrix  $A$  in  $\tilde{L}$  und  $R$  ist durch das Verfahren der Vorwärtselemination 2.1.1 ohne die Zeile für  $b_i$  gegeben.

**Notation 2.1.6.** Die *Vorwärtssubstitution* bestimmt aus  $b$  den Vektor  $z$  mit Hilfe der bereits bestimmten Matrix  $\tilde{L}$ . Dies entspricht der in (2.1.4) bzw. den Schritten aus 2.1.1 ohne die Bestimmung von  $l_{ik}$  und  $R$ , also nur den Schleifen für  $b_i$ .

---

**Algorithmus 2.1.7 :** Gauß-Elimination zur Lösung von  $Ax = b$

---

1. Dreieckszerlegung  $A \rightsquigarrow \tilde{L}, R$
  2. Vorwärtssubstitution  $b \rightsquigarrow z$
  3. Rückwärtssubstitution  $R, z \rightsquigarrow x$
- 

### 2.1.8. Rechenaufwand (gezählt in „flops“),

„flops“ = floating point operations

hierzu zählen Additionen, Subtraktionen, Multiplikationen und Divisionen

#### 1. Rechenaufwand der **Dreieckszerlegung**:

Für  $j = k + 1, \dots, n$ , je 1 Addition, 1 Multiplikation für  $a_{ij}$

also je  $(n - k)$  Additionen und Multiplikationen.

Diese werden je für  $i = k + 1, \dots, n$ , je 1 Division zusätzlich für  $l_{ik}$ .

Dieser Aufwand ist je für  $k = 1, \dots, n - 1$  benötigt und zusätzlich nötig. Also werden

$$\begin{aligned} \sum_{k=1}^{n-1} (n-k)^2 &= \sum_{k=1}^{n-1} k^2 = \frac{(n-1)n(2n-1)}{6} = \frac{2n^3 - 3n^2 + n}{6} \\ &\approx \frac{1}{3}n^3 \text{ Additionen und Multiplikationen für große } n \text{ und} \end{aligned}$$

$$\sum_{k=1}^{n-1} (n-k) = \frac{n^2 - n}{2} \approx \frac{1}{2}n^2 \text{ Divisionen für große } n \text{ ausgeführt.}$$

Damit ergibt sich ein Gesamtaufwand von

$$2 \frac{2n^3 - 3n^2 + n}{6} + \frac{n^2 - n}{2} = \frac{2}{3}n^3 - \frac{1}{2}n^2 - \frac{1}{6}n \approx \frac{2}{3}n^3 \text{ flops für große } n.$$

#### 2. Rechenaufwand der **Vorwärts- bzw. Rückwärtssubstitution**

Hier ergeben sich

$$\text{je } \sum_{k=1}^{n-1} (n-k) = \frac{n^2 - n}{2} \approx \frac{1}{2}n^2 \text{ Multiplikationen und Additionen}$$

sowie zusätzlich  $n$  Divisionen für die Rückwärtssubstitution und damit insgesamt

$$n^2 + n \text{ flops.}$$

Die Dreieckszerlegung (LR-Zerlegung) benötigt  $\mathcal{O}(n^3)$  flops  
und die Vorwärts- bzw. Rückwärtssubstitution  $\mathcal{O}(n^2)$  flops.

Für z.B.  $n = 1000$  ist  $n^2 = 10^6$  und  $n^3 = 10^9$ , bzw.

falls  $n \hat{=} 1000$  mikrosek = 1 sek ergibt sich  $n \hat{=} 16$  min und  $n^3 = 11,6$  Tage.



**Definition 2.1.9** (Landau-Symbole).

Seien  $f, g: D \rightarrow \mathbb{R}$ ,  $D \subseteq \mathbb{R}$ ,  $-\infty \leq a \leq \infty$  und  $(a_n)_{n \in \mathbb{N}}, (b_n)_{n \in \mathbb{N}}$  Folgen in  $\mathbb{R}$ .

- a)  $f(x) = \mathcal{O}(g(x))$   
für  $x \rightarrow a \neq \pm\infty$ , falls

$$\exists U(a), c \in \mathbb{R} \forall x \in U(a): |f(x)| \leq c|g(x)|,$$

für  $x \rightarrow \pm\infty$  falls

$$\exists N \in \mathbb{N} \forall x > N \text{ bzw. } x < -N: |f(x)| < c|g(x)|.$$

Falls  $\lim_{x \rightarrow a} \frac{|f(x)|}{|g(x)|} < \infty \Rightarrow f(x) = \mathcal{O}(g(x))$ . (“ $f$  verhält sich wie  $g$ ”)

- b)  $f(x) = o(g(x))$  für  $x \rightarrow a$ , falls

$$\lim_{x \rightarrow a} \frac{|f(x)|}{|g(x)|} = 0.$$

- c)  $a_n = \mathcal{O}(b_n)$  für  $n \rightarrow \infty$ , falls

$$\exists N \in \mathbb{N}, c \in \mathbb{R} \forall n \in \mathbb{N}: |a_n| \leq c|b_n|$$

- d)  $a_n = o(b_n)$  für  $n \rightarrow \infty$ , falls

$$\forall \varepsilon > 0 \exists N \in \mathbb{N}: \forall n \geq N: |a_n| \leq \varepsilon |b_n|.$$

**2.1.10. Allgemeines zur Aufwandsbetrachtung**

Die Anzahl der Rechenoperationen ist nicht immer ausschlaggebend für den Aufwand, z.B.

- *Parallelrechner*: In manchen Algorithmen sind Rechenschritte parallel ausführbar. Damit entspricht die Zeit nicht der Anzahl an Operationen. Zusätzlich wird „Kommunikationszeit“ benötigt.
- *Sortieralgorithmen und die Indexverwaltung*: benötigen Zeit aber keine Rechenoperationen
- *If-When-Abfragen und Ähnliches*: benötigen ebenfalls Zeit aber keine Rechenoperationen

Rechenoperationen liefern jedoch oft eine gute Schätzung.

Für *Funktionsauswertungen* ist die Anzahl der Rechenoperationen in der Regel unbekannt. Hier werden -falls möglich- *skalare Funktionsauswertungen* gezählt.

24.10.22

**2.1.11. Formalisieren des Gauß-Algorithmus:**

- a) Die **Rückwärtssubstitution** entspricht

$$Rx = z.$$

- b) Die **Vorwärtssubstitution** ist gegeben durch

$$\begin{aligned} b_i^{(k)} &= b_i^{(k-1)} - l_{ik} b_k^{(k-1)} && \text{für } i = k+1, \dots, n \\ \Rightarrow b^{(k)} &= b^{(k-1)} - l_k b_k^{(k-1)} && \text{mit } l_k := \begin{pmatrix} 0 & \cdots & 0 & l_{k+1,k} & \cdots & l_{n,k} \end{pmatrix}^T \end{aligned}$$

Sei  $e_k \in \mathbb{R}^n$  der  $k$ -te Einheitsvektor und

$$L_k := I - l_k e_k^T = \begin{pmatrix} 1 & & & & \\ 0 & \ddots & & & \\ & & 1 & & \\ \vdots & -l_{k+1,k} & 1 & & \\ & \vdots & & \ddots & \\ & -l_{n,k} & & & 1 \end{pmatrix} \quad (2.1.11)$$

dann gilt

$$\begin{aligned} b^{(k)} &= L_k b^{(k-1)} \\ z &= L_{n-1} \cdot L_{n-2} \cdot \dots \cdot L_1 b. \end{aligned}$$

Sei

$$L := L_1^{-1} \cdot \dots \cdot L_{n-1}^{-1}. \quad (2.1.12)$$

Hiermit entspricht der Vorwärtssubstitution

$$Lz = b. \quad (2.1.13)$$

c) Für die **Dreieckszerlegung**: wie für die Vorwärtssubstitution ergibt sich

$$A^{(k)} = L_k A^{(k-1)}$$

und somit

$$\begin{aligned} R &= L_{n-1} \cdot \dots \cdot L_1 A \\ \Rightarrow A &= L \cdot R. \end{aligned} \quad (2.1.14)$$

Frage: Wie hängt  $\tilde{L}$  bzw. die Zerlegung  mit  $L$  und  $R$  zusammen?

**Lemma 2.1.12.**

1.  $L_k$  ist eine Frobeniusmatrix, d.h. sie unterscheidet sich höchstens in einer Spalte von der Einheitsmatrix  $I$ .
2.  $L_k^{-1} = I + l_k e_k^T$ .
3. Es gilt:

$$L = L_1^{-1} \cdot \dots \cdot L_{n-1}^{-1} = I + \sum_{i=1}^{n-1} l_i e_i^T = \begin{pmatrix} 1 & & 0 \\ L_{ij} & \ddots & \\ & & 1 \end{pmatrix} = I + \tilde{L}. \quad (2.1.15)$$

*Beweis.* Siehe Übungsaufgabe. □

Hiermit ergibt sich der folgende Satz:

**Satz 2.1.13** (LR- oder LU-Zerlegung). *Das obige Verfahren (2.1.1) -(2.1.2) erzeugt unter der Voraussetzung von nicht-nullwertigen Pivotelementen eine Faktorisierung*

$$A = L \cdot R = \begin{pmatrix} \text{Lower Triangle } L & \cdot & \text{Upper Triangle } R \end{pmatrix}$$

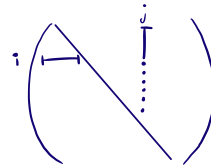
wobei  $R$  eine obere Dreiecksmatrix und  $L$  eine untere, normierte Dreiecksmatrix ist, d.h. für  $i = 1, \dots, n$  gilt  $l_{ii} = 1$ . Weiterhin existiert zu jeder regulären Matrix höchstens eine solche Zerlegung.

*Beweis.* Zur Eindeutigkeit:

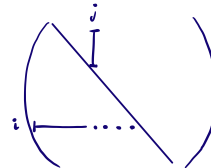
$$A = LR \Leftrightarrow a_{ij} = \sum_{k=1}^{\min j, j} l_{ik} r_{kj}$$

Da  $0 \neq \det(A) = \det(L) \cdot \det(R) = \prod_{i=1}^n r_{ii}$ , ist  $R$  regulär und  $r_{ii} \neq 0$  für  $i = 1, \dots, n$ . Mit  $l_{ii} = 1$  gilt, folgt

$$\text{für } i \leq j \quad r_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} r_{kj}$$



$$\text{für } i > j \quad l_{ij} = \frac{1}{r_{ij}} \left( a_{ij} - \sum_{k=1}^{j-1} l_{ik} r_{kj} \right),$$



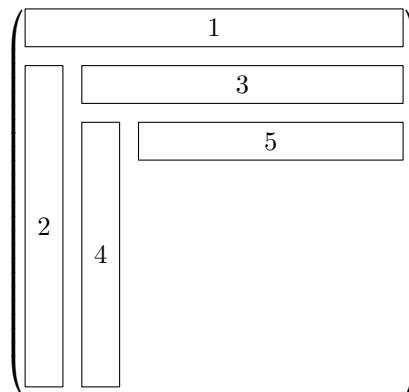
Diese können rekursiv, also eindeutig, berechnet werden, wenn auch die Reihenfolge der Berechnung nicht eindeutig ist. Mögliche Verfahren sind z.B.

das *Verfahren von Crout*

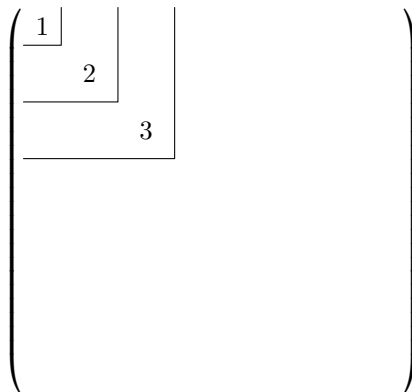
```

for  $i = 1, \dots, n$  do
  for  $j = i, \dots, n$  do
     $r_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} r_{kj}$ 
  end for
  for  $j = i + 1, \dots, n$  do
     $l_{ji} = \left( a_{ij} - \sum_{k=1}^{i-1} l_{ik} a_{kj} \right) / r_{jj}$ 
  end for
end for

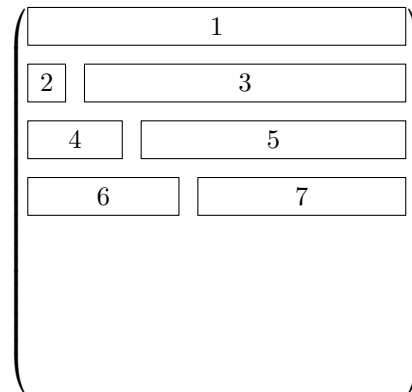
```



oder



oder zeilenweise



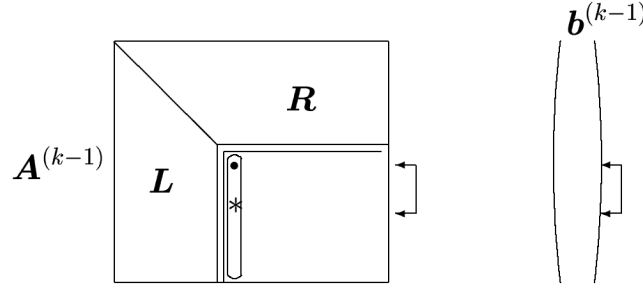
oder wie in Algorithmus 2.1.1 (ohne die Zeile mit  $b_j$ ).

□

## 2.2 Gaußsches Eliminationsverfahren mit Pivotisierung

**Beispiel:** Die Matrix  $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$  ist invertierbar, aber die Gauß-Elimination 2.1.1 versagt. Permutiere die erste mit der zweiten Zeile und der Algorithmus wird anwendbar.

Abbildung 2.1: Ein Schritt der Spaltenpivotisierung

**Allgemein:**

Vermeide die Division durch betragsmäßig kleine Zahlen!

**2.2.1. Spaltenpivotisierung:** *Spaltenpivotisierung* wird auch *partielle* oder *halbmaximale Pivotisierung* genannt.

Im  $k$ -ten Eliminationsschritt  $A^{(k-1)} \rightarrow A^{(k)}$  wird die Vorwärtselimination 2.1.1 durch  $a_{kk}^{(k-1)}$  dividiert. Die Spaltenpivotisierung zur Vermeidung von  $a_{kk}^{(k-1)} = 0$  geht folgendermaßen vor (siehe auch Abbildung 2.1).

1. Bestimme das Pivotelement  $a_{pk}^{(k-1)}$  als betragsmäßig größtes der „Rest-Spalte“, d.h.

$$|a_{pk}^{(k-1)}| \geq |a_{jk}^{(k-1)}| \quad \text{für } j = k, \dots, n$$

2. Vertausche in  $A^{(k-1)}$  die  $k$ -te mit der  $p$ -ten Zeile
3. Führe einen Gauß-Eliminationsschritt aus.

*Bemerkung 2.2.2.*

- a) Hiermit gilt  $|l_{jk}| \leq 1$ .
- b) Anstelle von Spaltenpivotisierung kann eine *Zeilenpivotisierung* durchgeführt werden. Welche günstiger (in cpu-time) ist, hängt von der Rechnerarchitektur und der damit zusammenhängenden Umsetzung des Gauß-Algorithmus ab.  
(Beispielsweise greifen Vektorrechner entweder auf die gesamte Spalte oder auf die gesamte Zeile einer Matrix zu und bevorzugen dementsprechend Operationen spalten- bzw. zeilenweise.)
- c) Der Aufwand enthält (bis auf  $|\cdot|$ ) keine Rechenoperationen (flops), aber  $\mathcal{O}(n^2)$  Vergleiche und Vertauschungen.
- d) Eine *vollständige Pivotsuche* sucht das betragsmäßig größte Element der gesamten Restmatrix und benötigt  $\mathcal{O}(n^3)$  Vergleiche. Sie wird so gut wie nie angewendet.

Damit die LR-Zerlegung mit Pivotisierung unabhängig von der rechten Seite erstellt werden kann, müssen die Permutationen gespeichert werden. Hierfür wird ein sogenannter *Permutationsvektor*  $\pi$  verwendet. Wenn

$$\pi^{(k-1)}(r) = s$$

ist, soll dies bedeuten, dass im  $(k-1)$ -ten Eliminationsschritt in der  $r$ -ten Zeile von  $A^{(k-1)}$  die  $s$ -te bearbeitete Zeile von  $A$  steht.

(Vorsicht: Hier wird  $x_i^{(neu)} = x_{\pi(i)}^{(alt)}$  verwendet, während in der Literatur für Permutationen  $\Pi$  oft die Notation  $x_{\Pi(i)}^{(neu)} = x_i^{(alt)}$  benutzt wird.)

Somit setze

$$\begin{aligned} \pi^{(k)}(k) &= \pi^{(k-1)}(p), \\ \pi^{(k)}(p) &= \pi^{(k-1)}(k), \quad \text{und} \\ \pi^{(k)}(i) &= \pi^{(k-1)}(i) \quad \text{für } i \neq k, p. \end{aligned}$$

**Algorithmus 2.2.3** : Gauß-Elimination (LR-Zerlegung) mit Spaltenpivotisierung

---

```

 $\pi(1:n) = [1:n]$ 
for  $k = 1, \dots, n-1$  do
  bestimme Pivotzeile  $p$ , so dass
   $|a_{pk}| = \max\{|a_{jk}|, j = k, \dots, n\}$ 
  vertausche  $\pi(k)$  mit  $\pi(p)$ 
  vertausche  $A(k, 1:n)$  mit  $A(p, 1:n)$ 
  if  $a_{kk} \neq 0$  then
     $zeile = [k+1:n]$ 
     $A(zeile, k) = A(zeile, k)/a_{kk}$ 
     $A(zeile, zeile) = A(zeile, zeile) - A(zeile, k)A(k, zeile)$ 
  else
    "A ist singulär"
  end if
end for

```

---

**Beachte:** Im Pseudoalgorithmus steht  $a_{kk} \neq 0$ .

Im Programm teste reelle Zahlen (float, double, ...) NIE auf  $= 0$ !

sondern immer, ob diese betragsmäßig kleiner einer geeigneten Toleranz ist, d.h. hier auf  $|a_{kk}| \leq tol$ .

Die **Permutationsmatrix**

$$P_\pi := (e_{\pi(1)}^T, \dots, e_{\pi(n)}^T) = (e_{\pi(1)}, \dots, e_{\pi(n)})^T$$

mit den  $j$ -ten Einheitsvektor  $e_j$  realisiert durch

$$P_\pi A$$

die beschriebene Zeilenpermutation. Allgemein gilt für Permutationen

$$P^{-1} = P^T,$$

$$\det P_\pi = \text{sign}(\pi) = \begin{cases} +1 & \text{falls } \pi \text{ durch eine gerade} \\ -1 & \text{Anzahl von Transpositionen erzeugt wird.} \\ & \text{ungerade} \end{cases}$$

26.10.2022

**Satz 2.2.4.** Für jede invertierbare Matrix  $A$  existiert eine Permutationsmatrix  $P$ , so dass eine Dreieckszerlegung

$$PA = LR$$

existiert.  $P$  kann so gewählt werden, dass alle Elemente von  $L$  betragsmäßig kleiner oder gleich 1 sind, d.h. es gilt  $|l_{ij}| \leq 1 \quad \forall i, j$ .

*Beweis.* Da  $\det A \neq 0$  ist, existiert eine Transposition  $\tau_1$ , sodass

$$a_{11}^{(1)} = a_{\tau_1(1),1} \neq 0$$

und

$$|a_{\tau_1(1),1}| \geq |a_{i1}| \quad \forall i = 1, \dots, n.$$

Wir erhalten damit

$$A^{(1)} = L_1 P_{\tau_1} A = \begin{pmatrix} a_{11}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & & \\ \vdots & B^{(1)} & \\ 0 & & \end{pmatrix}$$

und alle Elemente von  $L_1$  sind betragsmäßig kleiner oder gleich 1 sowie  $\det L_1 = 1$ .  
Daraus folgt

$$\det B^{(1)} = \frac{1}{a_{11}^{(1)}} \det A^{(1)} = \frac{1}{a_{\tau_1(1),1}} \det(L_1) \det(P_{\tau_1}) \det(A) \neq 0.$$

Also ist  $B^{(1)}$  invertierbar.

Induktiv erhalten wir dann die Existenz von  $\tau_k$  mit  $a_{kk}^{(k)} := a_{\tau_k(k),k}^{(k-1)} \neq 0$  und

$$\begin{aligned} R &= A^{(n-1)} = L_{n-1} P_{\tau_{n-1}} \cdot \dots \cdot L_1 P_{\tau_1} \cdot A \\ &= L_{n-1} (P_{\tau_{n-1}} L_{n-2} P_{\tau_{n-1}}^{-1}) (P_{\tau_{n-1}} P_{\tau_{n-2}} L_{n-2} P_{\tau_{n-2}}^{-1} P_{\tau_{n-1}}^{-1}) \dots (P_{\tau_{n-1}} \cdot \dots \cdot P_{\tau_1}) A \\ &= L_{n-1} \cdot \hat{L}_{n-2} \cdot \dots \cdot \hat{L}_1 P_{\pi} A. \end{aligned}$$

Da  $\tau_i$  nur zwei Zahlen  $\geq i$  vertauscht, ist

$$\pi_i := \tau_{n-1} \circ \dots \circ \tau_i \quad \text{für } i = 1, \dots, (n-1)$$

eine Permutation der Zahlen  $\{i, \dots, n\}$ , d.h. insbesondere gilt:

$$\begin{aligned} \pi_i(j) &= j & \text{für } j = 1, \dots, (i-1) \\ \pi_i(j) &\in \{i, \dots, n\} & \text{für } j = i, \dots, n. \end{aligned}$$

$$\Rightarrow P_{\pi_{i+1}} = (e_1, \dots, e_i, e_{\pi_{i+1}(i+1)}, \dots, e_{\pi_{i+1}(n)})^T =: \begin{pmatrix} I_i & 0 \\ 0 & P_{\sigma} \end{pmatrix}$$

Damit folgt:

$$\begin{aligned} P_{\pi_{i+1}} L_i P_{\pi_{i+1}}^{-1} &= P_{\pi_{i+1}} \left( \begin{array}{c|c} I_i & 0 \\ \hline 0 & I_{n-i} \end{array} \right) \begin{pmatrix} I_i & 0 \\ 0 & P_{\sigma}^{-1} \end{pmatrix} \\ &= \begin{pmatrix} I_i & 0 \\ 0 & P_{\sigma} \end{pmatrix} \left( \begin{array}{c|c} I_i & 0 \\ \hline 0 & P_{\sigma}^{-1} \end{array} \right) \\ &= \left( \begin{array}{c|c} I_i & 0 \\ \hline 0 & I_{n-i} \end{array} \begin{array}{c} -l_{\pi_{i+1}(i+1),i} \\ \vdots \\ -l_{\pi_{i+1}(n),i} \end{array} \right) = I - (P_{\pi_{i+1}} l_i) e_i^T =: \hat{L}_i. \end{aligned}$$

Mit  $R = L_{n-1} \hat{L}_{n-2} \dots \hat{L}_1 P_{\pi_1} A$  und Lemma 2.1.12 gilt: es existiert eine Permutation  $\pi_1$  mit

$$P_{\pi_1} A = LR,$$

wobei  $R$  obere Dreiecksgestalt hat und

$$L = \begin{pmatrix} 1 & & & 0 \\ l_{\pi_2(2),1} & \ddots & & \\ \vdots & \ddots & 1 & \\ l_{\pi_2(n),1} & \dots & l_{\pi_n(n),n-1} & 1 \end{pmatrix} \quad \text{mit } |l_{ij}| \leq 1$$

gilt. □

**2.2.5. Lösen eines Gleichungssystems  $Ax = b$** 

Das Lösen eines linearen Gleichungssystems der Form  $Ax = b$  mit invertierbarer Matrix  $A$  kann mit der LR-Zerlegung durch folgende drei Schritte durchgeführt werden:

1. Zerlege  $A$  durch  $PA = LR$ ,
2. Löse durch Vorwärtssubstitution  $Lz = Pb$ ,
3. Löse durch Rückwärtssubstitution  $Rx = z$ .

*Bemerkung 2.2.6.*

a)  $P_\pi A = LR$  kann zur Berechnung von  $\det(A)$  genutzt werden:

$$\det(A) = \text{sign}(\pi) \cdot r_{11} \cdot \dots \cdot r_{nn}.$$

b) Algorithmus 2.2.3 testet, ob die Matrix singulär ist, bis auf den Fall  $r_{nn} = a_{nn}^{(n-1)} = 0$ .

c)  $\det(A) = 0$  sollte nicht als (numerischer) Nachweis für die Singularität von  $A$  genutzt werden.  
Z.B. ist  $10^{-8}I$  regulär, aber  $\det(10^{-8}I) = 10^{-8n} \approx 0$  für große  $n$ , also ist  $A$  „numerisch singulär“.

*Beispiel 2.2.7.* (Gauß-Elimination unter Berücksichtigung von Rundungsfehlern).

Lösung von  $Ax = b$  mit

$$A = \begin{pmatrix} 10^{-4} & 1 \\ 1 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

wobei in jedem Schritt auf 3 Dezimalstellen gerundet wird.

a) Gauß-Elimination ohne Pivotisierung: (kleines Pivotelement bedeutet großer Multiplikator),

$$\begin{aligned} l_{21} &= 10^4 \\ r_{22} &= a_{22} - l_{21}a_{12} = 1 - 10^4 \cdot 1 = -9999 \approx -10^4 =: \tilde{r}_{22} \\ b_2^{(1)} &= b_2 - l_{21}b_1 = 2 - 10^4 = -9998 \approx -10^4 =: \tilde{b}_2^{(1)} \end{aligned}$$

Die Rückwärtssubstitution ergibt

$$\begin{aligned} x_2 &= \frac{b_2^{(1)}}{r_{22}} = \frac{9998}{9999} \approx 1 \\ \tilde{x}_2 &= \frac{\tilde{b}_2}{\tilde{r}_{22}} = \frac{-10^4}{-10^4} = 1 \\ x_1 &= \frac{1}{r_{11}}(b_1 - r_{12}x_2) = 10^4(1 - 1 \frac{9998}{9999}) = \frac{10^4}{9999} \approx 1 \\ \tilde{x}_1 &= \frac{1}{\tilde{r}_{11}}(1 - 1\tilde{x}_2) = 10^4(1 - 1 \cdot 1) = 0 \end{aligned}$$

b) Gauß-Elimination mit Spaltenpivotisierung:

$$l_{21} = 10^{-4} < 1, \tilde{r}_{22} = 1, \tilde{b}_2^{(2)} = 1, \tilde{x}_2 = 1, \tilde{x}_1 = 1.$$

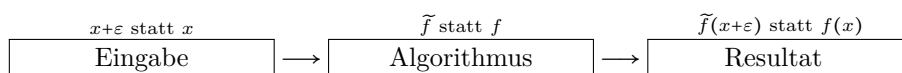
Spaltenpivotisierung führt im Allgemeinen auch bei Rundungsfehlern zu besseren Ergebnissen.





# Kapitel 3

## Fehleranalyse



### Eingabefehler:

z.B. Rundungsfehler, Fehler in Messdaten, Fehler im Modell (falsche Parameter)

### Fehler im Algorithmus:

z.B. Rundungsfehler durch Rechenoperationen, Approximationen (z.B. Ableitung durch Differenzenquotient, die Berechnung von Sinus durch abgebrochene Reihenentwicklung)

Es stellen sich insbesondere zwei Fragen:

1. Wie wirken sich Eingabefehler auf das Resultat unabhängig vom gewählten Algorithmus aus?

$$f(x) \leftrightarrow f(x + \varepsilon)$$

↪ Kondition des Problems

2. Wie wirken sich (Rundungs-)Fehler des Algorithmus aus und wie verstärkt der Algorithmus Eingabefehler?

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

↪ Stabilität des Algorithmus

## 3.1 Zahlendarstellung und Rundungsfehler

Auf Rechnern können nur endlich viele Zahlen realisiert werden. Die wichtigsten Typen sind:

- **ganze Zahlen** (integer):

$$z = \pm \sum_{i=0}^m z_i \beta^i$$

mit  $\beta = \text{Basis des Zahlensystems (oft } \beta = 2\text{)}$   
 $z_i \in \{0, \dots, \beta - 1\}$

Beispiele:

$$\begin{array}{cccccc}
 10^3 & 10^2 & 10^1 & 10^0 & & \\
 \hline
 5 & 3 & 4 & 2 & \text{Dezimalsystem} & 
 \end{array}
 \qquad
 \begin{array}{cccccc}
 16 & 8 & 4 & 2 & 1 & \\
 \hline
 1 & 0 & 1 & 1 & 0 & \text{Dualsystem}
 \end{array}$$

$$\begin{array}{cccc}
 \beta^3 & \beta^2 & \beta^1 & \beta^0 \\
 \hline
 z_3 & z_2 & z_1 & z_0
 \end{array}
 \text{ System zur Basis } \beta$$

- **Gleitpunktzahlen** (floating point)

**Definition 3.1.1.** Eine Zahl  $x \in \mathbb{Q}$  mit einer Darstellung

$$\begin{aligned} x &= \sigma \cdot (a_1.a_2 \dots a_t)_\beta \beta^e \\ &= \sigma \beta^e \sum_{\nu=1}^t a_\nu \beta^{-\nu+1} \end{aligned}$$

wobei

$\beta \in \mathbb{N}$	die Basis des Zahlensystems,
$\sigma \in \{\pm 1\}$	das Vorzeichen,
$m = (a_1.a_2 \dots a_t)_\beta$ $= \sum_{\nu=1}^t a_\nu \beta^{-\nu+1}$	die Mantisse,
$a_i \in \{0, \dots, \beta - 1\}$	die Ziffern der Mantisse,
$t \in \mathbb{N}$	die Mantissenlänge,
$e \in \mathbb{Z}$	der Exponent mit $e_{\min} \leq e \leq e_{\max}$ sind,

heißt *Gleitkommazahl* mit  $t$  Stellen und Exponent  $e$  zur Basis  $b$ .

Ist  $a_1 \neq 0$ , so heißt  $x$  *normalisierte Gleitkommazahl*.

Die Menge  $\mathcal{F}(\beta, t, e_{\min}, e_{\max})$  bezeichne die Menge aller normalisierten Gleitkommazahlen zur Basis  $\beta$  mit  $t$  Stellen und Exponenten  $e$  mit  $e_{\min} \leq e \leq e_{\max}$ .

31.10.22

*Bemerkung 3.1.2.*

- $0 \notin \mathcal{F}(\beta, t, e_{\min}, e_{\max})$ , d.h. 0 ist keine normalisierte Gleitkommazahl, da  $a_1 = 0$  ist.
- $a_1 \neq 0$  stellt sicher, dass die Gleitkommadarstellung eindeutig ist.
- In der Praxis werden auch nicht-normalisierte Darstellungen verwendet.
- Heutige Rechner verwenden meist  $\beta = 2$ , aber auch  $\beta = 8, \beta = 16$ .

*Beispiel 3.1.3.* (Bit-Darstellung nach IEEE-Standard 754 von floating point numbers)

Die Basis ist  $\beta = 2$ .

	Speicherplatz	$t$	$e_{\min}$	$e_{\max}$
einfache Genauigkeit (float)	32bits = 4Bytes	24	-126	127
doppelte Genauigkeit (double)	64bits = 8Bytes	52	-1022	1023

Die Darstellung im Rechner (Bitmuster) für float ist:

$s$	$b_0 \dots b_7$	$a_2 \dots a_{24}$
-----	-----------------	--------------------

Da  $a_1 \neq 0$ , also  $a_1 = 1$  gilt, wird  $a_1$  nicht gespeichert. Es gilt  $s, b, a_i \in \{0, 1\} \forall i$  Interpretation des Bitmusters:

- $s$  Vorzeichenbit:  $\sigma = (-1)^s$
- $b = \sum_{i=0}^7 b_i \cdot 2^i \in \{1, \dots, 254\}$  speichert den Exponenten mit  
 $e = b - \underbrace{127}_{\text{Basiswert}}$  (kein Vorzeichen nötig).  
 Beachte:  $b_0 = \dots = b_7 = 1$  sowie  $b_0 = \dots = b_7 = 0$  sind bis auf Ausnahmen keine gültigen Exponenten
- $m = (a_1.a_2 \dots a_{24}) = 1 + \sum_{\nu=2}^{24} a_\nu 2^{1-\nu}$  stellt die Mantisse dar.  
 $a_1 = 1$  wird nicht abgespeichert.

- Besondere Zahlen per Konvention:

$x = 0$ :	$s$ bel., $b = 0$ , $m = 1$	<table border="1"><tr><td><math>s</math></td><td><math>0 \dots 0</math></td><td><math>0 \dots 0</math></td></tr></table>	$s$	$0 \dots 0$	$0 \dots 0$
$s$	$0 \dots 0$	$0 \dots 0$			
$x = \pm\infty$ :	$s$ bel., $b = 255$ , $m = 1$	<table border="1"><tr><td><math>s</math></td><td><math>1 \dots 1</math></td><td><math>0 \dots 0</math></td></tr></table>	$s$	$1 \dots 1$	$0 \dots 0$
$s$	$1 \dots 1$	$0 \dots 0$			
$x = NaN$	$s$ bel., $b = 255$ , $m \neq 1$				
$x = (-1)^s (0 + \sum_{\nu=2}^{24} a_\nu \cdot 2^{1-\nu}) 2^{126}$	$s$ bel., $b = 0$ , $m \neq 1$	denormalized number			

20.10.2014

Die betragsmäßig **größte Zahl** ist:

$$\begin{array}{|c|c|c|} \hline 0 & 01 \dots 1 & 1 \dots 1 \\ \hline \end{array} \quad x_{max} = (2 - 2^{-23}) \cdot 2^{127} \approx 3,4 \cdot 10^{38}$$

Die betragsmäßig **kleinste Zahl** ist eine denormalized number:

$$\begin{array}{|c|c|c|} \hline 0 & 0 \dots 0 & 0 \dots 01 \\ \hline \end{array} \quad x_{min} = 2^{-23} \cdot 2^{-126} = 2^{-149} \approx 1,4 \cdot 10^{-45}$$

### 3.1.4. Verteilung der Maschinenzahlen

Die Maschinenzahlen sind im Dezimalsystem ungleichmäßig verteilt, z. B.

$$x = \pm a_1.a_2a_3 \cdot 2^e \quad \text{mit } -2 \leq e \leq 1 \text{ und } a_i \in \{0, 1\}$$

ist im Dualsystem gleichmäßig, jedoch im Dezimalsystem ungleichmäßig verteilt, siehe Abbildung 3.1.

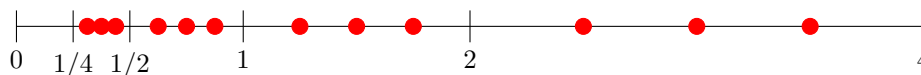


Abbildung 3.1

#### Notation 3.1.5.

**overflow:** Es ergibt sich eine Zahl, die betragsmäßig größer ist als die größte maschinendarstellbare Zahl.

**underflow:** Es ergibt sich eine Zahl, die betragsmäßig kleiner ist als die kleinste positive Zahl.

Beispiel 3.1.6.

- a) overflow bei ganzen Zahlen, falls diese wie der Exponent  $e$  dargestellt werden, also  $b = e + 127$ :

$$\begin{array}{rcl} b & = & 254 \\ + & 3 & \\ \hline b + 3 = 257 \bmod 2^8 & = & 1 \end{array} \quad \begin{array}{l} 11111110 \\ 00000011 \\ \hline 00000001 \end{array}$$

- b) Vermeidung von overflow:  
statt

$$\|x\|_2 := \sqrt{\sum_{i=1}^n x_i^2}$$

wie dargestellt zu berechnen, berechne erst

$$\bar{x}_{max} := \max\{x_1, \dots, x_n\}$$

und dann

$$\|x\|_2 = \bar{x}_{max} \sqrt{\sum_{i=1}^n \left(\frac{x_i}{\bar{x}_{max}}\right)^2}.$$

**3.1.7. Rundungsfehler.** Habe  $x \in \mathbb{R}$  die normalisierte Darstellung

$$\begin{aligned} x &= \sigma \cdot \beta^e \left( \sum_{\nu=1}^t a_{\nu} \beta^{1-\nu} + \sum_{\nu=t+1}^{\infty} a_{\nu} \beta^{1-\nu} \right) \\ &= \sigma \cdot \beta^e \left( \sum_{\nu=1}^t a_{\nu} \beta^{1-\nu} + \beta^{1-t} \sum_{l=1}^{\infty} a_{t+l} \beta^{-l} \right) \end{aligned}$$

mit  $e_{\min} \leq e \leq e_{\max}$ , dann wird mit  $fl(x)$  die *gerundete Zahl* bezeichnet, wobei  $fl(x)$  gegeben ist durch die Schranke an den *absoluten Rundungsfehler*

$$|fl(x) - x| \leq \begin{cases} \frac{1}{2} \beta^{e+1-t} & \text{bei symmetrischem Runden} \\ \beta^{e+1-t} & \text{bei Abschneiden} \end{cases}.$$

Für die *relative Rechengenauigkeit* folgt somit

$$\frac{|fl(x) - x|}{|x|} \leq \begin{cases} \frac{1}{2} \beta^{1-t} & \text{bei symmetrischem Runden} \\ \beta^{1-t} & \text{bei Abschneiden} \end{cases}.$$

Die *Maschinengenauigkeit* des Rechners ist daher durch

$$\text{eps} := \beta^{1-t} \quad (\text{für float} \approx 10^{-7}, \text{ für double} \approx 10^{-16})$$

gegeben.

Die Mantissenlänge bestimmt also die Maschinengenauigkeit. Bei einfacher Genauigkeit ist  $fl(x)$  bis auf ungefähr 7 signifikante Stellen genau.

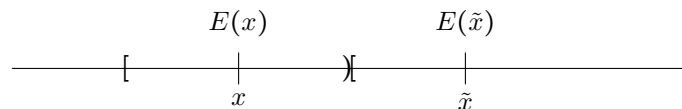
Im Folgenden betrachten wir symmetrisches Runden und definieren daher  $\tau := \frac{1}{2} \text{eps}$ . Weiterhin gilt:

a) Die *kleinste Zahl am Rechner*, welche größer als 1 ist, ist

$$1 + \text{eps}$$

b) Eine Maschinenzahl  $x$  repräsentiert eine Eingabemenge

$$E(x) = \{\tilde{x} \in \mathbb{R} : |\tilde{x} - x| \leq \tau|x|\}$$



*Bemerkung 3.1.8.* Gesetze der arithmetischen Operationen gelten i.A. nicht, z.B.:

- $x$  Maschinenzahl  $\Rightarrow fl(x + \nu) = x$  für  $|\nu| < \tau|x|$
- Assoziativ- und Distributivgesetze gelten nicht, z.B. für  $\beta = 10$ ,  $t = 3$ ,  $a = 0$ ,  $1$ ,  $b = 105$ ,  $c = -104$  gilt:

$$\begin{aligned} fl(a + fl(b + c)) &= 1, 1 \\ fl(fl(a + b) + c) &= fl(fl(105, 1) + (-104)) = fl(105 - 104) = 1 \end{aligned}$$

$\Rightarrow$  Für einen Algorithmus ist die Reihenfolge der Operationen wesentlich! Mathematisch äquivalente Formulierungen können zu verschiedenen Ergebnissen führen.

**3.1.9. Auslöschung von signifikanten Stellen.** Sei  $x = 9,995 \cdot 10^{-1}, y = 9,984 \cdot 10^{-1}$ . Runde auf drei signifikante Stellen und berechne  $x - y$ :

$$\begin{aligned} f(x, y) &:= x - y = 0,0011 = 1,1 \cdot 10^{-3} \\ \tilde{f}(x, y) &:= fl(fl(x) - fl(y)) = fl(1,00 \cdot 10^0 - 9,98 \cdot 10^{-1}) \\ &= fl(0,02 \cdot 10^{-1}) = fl(2,00 \cdot 10^{-3}) = 2 \cdot 10^{-3} \end{aligned}$$

Daraus ergibt sich der relative Fehler

$$\frac{|\tilde{f}(x, y) - f(x, y)|}{|f(x, y)|} := \frac{|2 \cdot 10^{-3} - 1,1 \cdot 10^{-3}|}{|1,1 \cdot 10^{-3}|} \approx 82\%$$

Der Grund hierfür ist, dass das Problem der Subtraktion zweier annähernd gleich großer Zahlen schlecht konditioniert ist. Dies wird im nächsten Abschnitt erläutert.

### 3.1.10. Zwei Regeln:

1. Umgehbare Subtraktion annähernd gleich großer Zahlen vermeiden!
2. Unumgängliche Subtraktion möglichst an den Anfang des Algorithmus stellen! (siehe später).

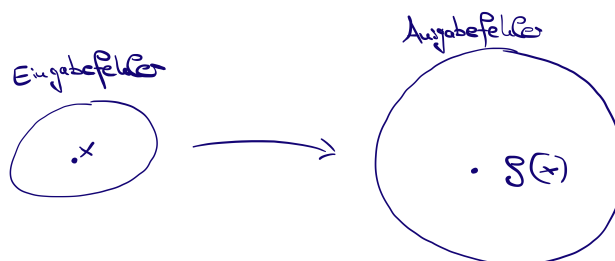
2.11.22

## 3.2 Kondition eines Problems

Es wird das Verhältnis

$$\frac{\text{Ausgabefehler}}{\text{Eingabefehler}}$$

untersucht.



**Definition 3.2.1.** Sei  $f: U \rightarrow \mathbb{R}^m$  mit  $U \subseteq \mathbb{R}^n$  offen und  $x \in U$ . Dann bezeichne  $(f, x)$  das Problem, zu einem gegebenen  $x$  die Lösung  $f(x)$  zu finden.

**Definition 3.2.2.** Sei  $x \in \mathbb{R}^n$  und  $\tilde{x} \in \mathbb{R}^n$  eine Näherung an  $x$ . Weiterhin sei  $\|\cdot\|$  eine Norm auf  $\mathbb{R}^n$ .

- a)  $\|\tilde{x} - x\|$  heißt *absoluter Fehler*,
- b)  $\frac{\|\tilde{x} - x\|}{\|x\|}$  heißt *relativer Fehler* für  $x \neq 0$ .

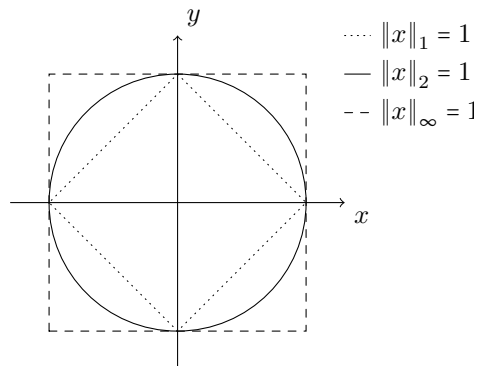
Da der relative Fehler *skalierungsinvariant* ist, d.h. unabhängig von der Größe von  $x$  ist, ist dieser i.d.R. von größerem Interesse. Beide Fehler hängen von der Wahl der Norm ab!

Häufig werden Fehler auch komponentenweise gemessen:

$$\begin{array}{lll} \text{Für } i = 1, \dots, n : & |\tilde{x}_i - x_i| \leq \delta & \text{(absolut)} \\ & |\tilde{x}_i - x_i| \leq \delta |x_i| & \text{(relativ) für } x_i \neq 0. \end{array}$$

Wiederholung 3.2.3 (Normen im  $\mathbb{R}^n$ ).

$$\begin{aligned} \text{Summennorm } (l_1\text{-Norm}): \quad & \|x\|_1 := \sum_{i=1}^n |x_i| \\ \text{Euklidische Norm } (l_2\text{-Norm}): \quad & \|x\|_2 := \sqrt{\sum_{i=1}^n |x_i|^2} \\ \text{Maximumsnorm } (l_\infty\text{-Norm}): \quad & \|x\|_\infty := \max\{|x_i| \mid i = 1, \dots, n\} \\ \text{Hölder-Norm } (l_p\text{-Norm}): \quad & \|x\|_p := \left( \sum_{i=1}^n |x_i|^p \right)^{\frac{1}{p}} \end{aligned}$$



**Definition 3.2.4.** Auf dem  $\mathbb{R}^n$  sei die Norm  $\|\cdot\|_a$  und auf dem  $\mathbb{R}^m$  die Norm  $\|\cdot\|_b$  gegeben. Dann ist die zugehörige *Matrixnorm* von  $A \in \mathbb{R}^{m \times n}$  gegeben durch

$$\|A\|_{a,b} := \sup_{x \neq 0} \frac{\|Ax\|_b}{\|x\|_a} = \sup_{\|x\|_a=1} \|Ax\|_b. \quad (3.2.1)$$

Also ist  $\|A\|_{a,b}$  die kleinste Zahl  $c > 0$  mit

$$\|Ax\|_b \leq c \|x\|_a \quad \forall x \in \mathbb{R}^n.$$

**Definition 3.2.5.** Sei  $A \in \mathbb{R}^{m \times n}$ . Dann sind folgende Normen definiert:

- a) *Frobeniusnorm (Schurnorm)*:  $\|A\|_F := \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2}$ ,
- b) *p-Norm*:  $\|A\|_p := \|A\|_{p,p}$ ,
- c) Eine Matrixnorm heißt *verträglich mit den Vektornormen*  $\|\cdot\|_a, \|\cdot\|_b$ , falls gilt

$$\|Ax\|_b \leq \|A\| \|x\|_a \quad \forall x \in \mathbb{R}^n.$$

*Bemerkung 3.2.6.* Sei  $A \in \mathbb{R}^{m \times n}$ .

- a) Die Normen  $\|\cdot\|_F$  und  $\|\cdot\|_p$  sind *submultiplikativ*, d.h.

$$\|AB\| \leq \|A\| \|B\|.$$

- b) Die Norm  $\|\cdot\|_{1,1}$  heißt *Spaltensummennorm* und es gilt:

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^m |a_{ij}|.$$

- c) Die Norm  $\|\cdot\|_{\infty, \infty}$  heißt *Zeilensummennorm* und es gilt:

$$\|A\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^n |a_{ij}|.$$

- d) Die Wurzeln aus den Eigenwerten von  $A^T A$  heißen *Singulärwerte*  $\sigma_i$  von A. Es gilt

$$\begin{aligned} \|A\|_2 &= \max\{\sqrt{\mu} \mid A^T A x = \mu x \text{ für ein } x \neq 0\} \\ &= \sigma_{\max}. \end{aligned}$$

- e) Die Frobeniusnorm  $\|\cdot\|_F$  ist verträglich mit der euklidischen Norm  $\|\cdot\|_2$ .

*Beweis.* Siehe Übungsaufgaben. □

### 3.2 a) Normweise Konditionsanalyse

**Definition 3.2.7.** Sei  $(f, x)$  ein Problem mit  $f: U \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$  und Normen  $\|\cdot\|_a$  auf  $\mathbb{R}^n$  und  $\|\cdot\|_b$  auf  $\mathbb{R}^m$ .

a) Die *absolute normweise Kondition* eines Problems  $(f, x)$  ist die kleinste Zahl  $\kappa_{abs} > 0$  mit

$$\|f(\tilde{x}) - f(x)\|_b \leq \kappa_{abs}(f, x) \|\tilde{x} - x\|_a + o(\|\tilde{x} - x\|_a). \quad (3.2.2)$$

b) Die *relative normweise Kondition* eines Problems  $(f, x)$  mit  $x \neq 0, f(x) \neq 0$  ist die kleinste Zahl  $\kappa_{rel} > 0$  mit

$$\frac{\|f(\tilde{x}) - f(x)\|_b}{\|f(x)\|_b} \leq \kappa_{rel}(f, x) \frac{\|\tilde{x} - x\|_a}{\|x\|_a} + o\left(\frac{\|\tilde{x} - x\|_a}{\|x\|_a}\right) \quad \text{für } \tilde{x} \rightarrow x. \quad (3.2.3)$$

Sprechweise:

- falls  $\kappa$  „klein“ ist, ist das Problem „gut konditioniert“,
- falls  $\kappa$  „groß“ ist, ist das Problem „schlecht konditioniert“.

**Lemma 3.2.8.** Falls  $f$  differenzierbar ist, gilt

$$\kappa_{abs}(f, x) = \|Df(x)\|_{a,b}, \quad (3.2.4)$$

und für  $x \neq 0, f(x) \neq 0$  gilt

$$\kappa_{rel}(f, x) = \frac{\|x\|_a}{\|f(x)\|_b} \|Df(x)\|_{a,b}, \quad (3.2.5)$$

wobei  $Df(x)$  die *Jakobi-Matrix* bezeichnet.

*Beispiel 3.2.9.* (Normweise Kondition der Addition).

Betrachte  $f(x_1, x_2) := x_1 + x_2, f: \mathbb{R}^2 \rightarrow \mathbb{R}$  und  $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ .

$$\Rightarrow Df(x_1, x_2) = \nabla f(x)^T = \left( \frac{\partial}{\partial x_1} f(x), \frac{\partial}{\partial x_2} f(x) \right) = (1, 1).$$

Wähle z.B. die  $l_1$ -Norm auf  $\mathbb{R}^2$  und  $\mathbb{R}$ , dann folgt

$$\begin{aligned} \kappa_{abs}(f, x) &= \|Df(x)\|_{1,1} = \|Df(x)\|_1 && \text{(Matrix-Norm!)} \\ &= 1, \\ \kappa_{rel}(f, x) &= \frac{\|x\|_1}{\|f(x)\|_1} \|Df(x)\|_1 \\ &= \frac{|x_1| + |x_2|}{|x_1 + x_2|}. \end{aligned}$$

Daraus folgt:

Die Addition zweier Zahlen mit gleichem Vorzeichen ergibt

$$\kappa_{rel} = 1.$$

Die Subtraktion zweier annähernd gleich großer Zahlen ergibt eine sehr schlechte relative Kondition:

$$\kappa_{rel} \gg 1.$$

*Beispiel 3.2.10.* (Lösen eines linearen Gleichungssystems).

Sei  $A \in \mathbb{R}^{n \times n}$  invertierbar und  $b \in \mathbb{R}^n$ . Es soll

$$Ax = b$$

gelöst werden. Mögliche Störungen können in  $A$  und in  $b$  auftreten. Die Fehler werden bzgl.  $\|\cdot\|_p$  und  $\|\cdot\|_q$  gemessen.

a) Betrachte Störungen in  $b$ :

Für  $f: b \mapsto x = A^{-1}b$  berechne  $\kappa(f, b)$ : Betrachte

$$\begin{aligned} A(x + \Delta x) &= b + \Delta b \\ \Rightarrow f(b + \Delta b) - f(b) &= \Delta x = A^{-1}\Delta b \quad \text{da } x = A^{-1}b \\ \Rightarrow \|\Delta x\|_p &= \|A^{-1}\Delta b\|_p \leq \|A^{-1}\|_{q,p} \|\Delta b\|_q \quad \forall b, \Delta b. \end{aligned}$$

Die Abschätzung ist nach Definition 3.2.4 *scharf*, d.h. es gibt ein  $\Delta b \in \mathbb{R}^n$ , so dass „ $=$ “ gilt. Also gilt

$$\kappa_{abs}(f, b) = \|A^{-1}\|_{q,p} \quad \forall b \in \mathbb{R}^n. \quad (3.2.6)$$

Dies folgt auch aus Lemma 3.2.8, d.h. aus  $\kappa_{abs}(f, b) = \|Df(b)\|$ .

Ebenso folgt die scharfe Abschätzung

$$\begin{aligned} \frac{\|f(b + \Delta b) - f(b)\|}{\|f(b)\|} &= \frac{\|\Delta x\|}{\|x\|} = \frac{\|A^{-1}\Delta b\|}{\|x\|} \leq \frac{\|A^{-1}\| \|b\| \|\Delta b\|}{\|x\| \|b\|} \\ \Rightarrow \kappa_{rel}(f, b) &= \|A^{-1}\| \frac{\|b\|}{\|A^{-1}b\|}. \end{aligned} \quad (3.2.7)$$

Da  $\|b\| \leq \|A\| \|x\| = \|A\| \|A^{-1}b\|$  ergibt sich für alle möglichen (rechten Seiten)  $b$  die Abschätzung

$$\kappa_{rel}(f, b) \leq \|A\| \|A^{-1}\|. \quad (3.2.8)$$

Die Abschätzung (3.2.8) ist scharf in dem Sinne, dass es ein  $\widehat{b} \in \mathbb{R}^n$  gibt mit  $\|\widehat{b}\| = \|A\| \|\widehat{x}\|$  und somit ein  $\widehat{b} \in \mathbb{R}^n$  existiert mit

$$\kappa_{rel}(f, \widehat{b}) = \|A\| \|A^{-1}\|.$$

b) Betrachte Störungen in  $A$ :

Sei also

$$(A + \Delta A)(x + \Delta x) = b.$$

Die zugehörige Funktion ist

$$f: \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^n, A \mapsto x = A^{-1}b.$$

Berechne  $\kappa(f, A)$  mittels der Ableitung  $Df(A): \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^n$ . Für eine Matrix  $C$  liefert  $Df(A)C$  die Richtungsableitung:

$$\begin{aligned} C \mapsto Df(A)C &= \left. \frac{d}{dt} ((A + tC)^{-1}b) \right|_{t=0} \\ &= \left. \frac{d}{dt} ((A + tC)^{-1}) \right|_{t=0} b \end{aligned}$$

Weiterhin gilt

$$\left. \frac{d}{dt} ((A + tC)^{-1}) \right|_{t=0} = -A^{-1}CA^{-1}, \quad (3.2.9)$$

denn es ist, falls  $(A + tC)$  invertierbar,

$$\begin{aligned} 0 &= \frac{d}{dt} I = \frac{d}{dt} ((A + tC)(A + tC)^{-1}) \\ &= C(A + tC)^{-1} + (A + tC) \frac{d}{dt} (A + tC)^{-1} \\ \Leftrightarrow \frac{d}{dt} (A + tC)^{-1} &= -(A + tC)^{-1}C(A + tC)^{-1}. \end{aligned}$$



Für ein genügend kleines  $t$  ist die Invertierbarkeit gewährleistet, da  $A$  invertierbar ist (siehe auch Lemma 3.2.12).

7.11.22

Also gilt

$$Df(A)C = -A^{-1}CA^{-1}b.$$

$$\begin{aligned} \Rightarrow \kappa_{abs}(f, A) &= \|Df(A)\| = \sup_{\substack{C \in \mathbb{R}^{n \times n} \\ C \neq 0}} \frac{\|A^{-1}CA^{-1}b\|}{\|C\|} \\ &\leq \sup_{\substack{C \in \mathbb{R}^{n \times n} \\ C \neq 0}} \frac{\|A^{-1}\| \|C\| \|A^{-1}b\|}{\|C\|} = \|A^{-1}\| \|x\| \leq \|A^{-1}\|^2 \|b\| \end{aligned} \quad (3.2.10)$$

$$\begin{aligned} \Rightarrow \kappa_{rel}(f, A) &= \frac{\|A\|}{\|f(A)\|} \|Df(A)\| \\ &\leq \|A\| \|A^{-1}\|. \end{aligned} \quad (3.2.11)$$

c) Betrachte Störungen in  $A$  und  $b$  :

$$(A + \Delta A)(x + \Delta x) = (b + \Delta b).$$

Für  $\kappa$  müsste  $\|(A, b)\|$  festgelegt werden. Dies wird jedoch nicht betrachtet. Es gilt folgende Abschätzung für invertierbare Matrizen  $A \in \mathbb{R}^{n \times n}$  und Störungen  $\Delta A \in \mathbb{R}^{n \times n}$  mit  $\|A^{-1}\| \|\Delta A\| < 1$ :

$$\frac{\|\Delta x\|}{\|x\|} \leq \|A\| \|A^{-1}\| (1 - \|A^{-1}\| \|\Delta A\|)^{-1} \underbrace{\left( \frac{\|\Delta b\|}{\|b\|} + \frac{\|\Delta A\|}{\|A\|} \right)}_{\neq \frac{\|(\Delta A, \Delta b)\|}{\|(A, b)\|}} \quad (3.2.12)$$

*Beweis.* Siehe Übungsaufgabe

□

**Definition 3.2.11.** (Kondition einer Matrix).

Sei  $\|\cdot\|$  eine Norm auf  $\mathbb{R}^{n \times n}$  und  $A \in \mathbb{R}^{n \times n}$  eine reguläre Matrix. Die Größe

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

heißt *Kondition der Matrix* bzgl. der Norm  $\|\cdot\|$ .

Ist  $\|\cdot\|$  von einer Vektor-Norm  $\|\cdot\|_p$  induziert, bezeichnet

$$\text{cond}_p(A) \text{ die } \text{cond}_{\|\cdot\|_p}(A).$$

Oft wird  $\text{cond}(A)$  für  $\text{cond}_2(A)$  verwendet.

$\text{cond}_{\|\cdot\|}(A)$  schätzt die relative Kondition eines linearen GLS  $Ax = b$  für alle möglichen Störungen in  $b$  oder in  $A$  ab und diese Abschätzung ist scharf.

Es stellt sich nun die Frage:

Wann existiert die Inverse der gestörten invertierbaren Matrix  $A$ ? Da  $A + \Delta A = A(I + A^{-1}\Delta A)$  wird dies zurückgeführt auf die Existenz der Inversen zu  $(I + C)$ .

**Lemma 3.2.12.** (Neumannsche Reihe).

Sei  $C \in \mathbb{R}^{n \times n}$  mit  $\|C\| < 1$  und mit einer submultiplikativen Norm  $\|\cdot\|$ , so ist  $(I - C)$  invertierbar und es gilt

$$(I - C)^{-1} = \sum_{k=0}^{\infty} C^k,$$

$$\text{sowie} \quad \|(I - C)^{-1}\| \leq \frac{1}{1 - \|C\|}. \quad (3.2.14)$$

*Beweis.* Es gilt zu zeigen, dass  $\sum_{k=0}^{\infty} C^k$  existiert. Da  $\|C\| < 1$  gilt

$$\begin{aligned} \left\| \sum_{k=0}^m C^k \right\| &\leq \sum_{k=0}^m \|C^k\| \leq \sum_{k=0}^m \|C\|^k && (\text{da } \|\cdot\| \text{ submultiplikativ}) \\ &= \frac{1 - \|C\|^{m+1}}{1 - \|C\|} && (\text{geometrische Reihe}) \\ &\leq \frac{1}{1 - \|C\|} && \forall m \in \mathbb{N}. \end{aligned}$$

Mit dem Majorantenkriterium folgt damit die Existenz von  $\sum_{k=0}^{\infty} C^k$ .

$$\begin{aligned} (I - C) \sum_{k=0}^{\infty} C^k &= \lim_{m \rightarrow \infty} (I - C) \sum_{k=0}^m C^k = \lim_{m \rightarrow \infty} (C^0 - C^{m+1}) \\ &= I - \lim_{m \rightarrow \infty} C^{m+1} = I. \end{aligned}$$

□

*Bemerkung 3.2.13.*

a) Für eine symmetrische, positiv definite Matrix  $A \in \mathbb{R}^{n \times n}$  gilt

$$\kappa_2(A) = \frac{\lambda_{\max}}{\lambda_{\min}}, \quad (3.2.15)$$

wobei  $\lambda_{\max}$  und  $\lambda_{\min}$  der maximale bzw. minimale Eigenwert von  $A$  ist.

b) Eine andere Darstellung von  $\kappa(A)$  ist

$$\kappa(A) := \frac{\max_{\|x\|=1} \|Ax\|}{\min_{\|x\|=1} \|Ax\|} \in [0, \infty]. \quad (3.2.16)$$

Diese ist auch für nicht invertierbare Matrizen und Matrizen  $A \in \mathbb{R}^{n \times m}$  mit  $n \neq m$  wohldefiniert.

c)  $\kappa(A) \geq 1$ .

d)  $\kappa(\alpha A) = \kappa(A)$  für  $0 \neq \alpha \in \mathbb{R}$  (skalierungsinvariant).

e) Eine Matrix  $A \neq 0$  und  $A \in \mathbb{R}^{n \times n}$  ist genau dann singular, wenn  $\kappa(A) = \infty$ .

Wegen der Skalierungsinvarianz ist die Kondition zur numerischen Überprüfung der Regularität von  $A$  besser geeignet als die Determinante.

*Beweis.* a) siehe Übungsaufgabe, c)-e) folgt aus b). □

*Beispiel 3.2.14.* (Kondition eines nichtlinearen Gleichungssystems.)

Sei  $f: \mathbb{R}^n \rightarrow \mathbb{R}^n$  stetig differenzierbar und  $y \in \mathbb{R}^n$  gegeben. Zur Lösung von

$$f(x) = y$$

ist die Kondition gesucht, also  $\kappa(f^{-1}, y)$  mit  $f^{-1}: y \mapsto x$ .

Sei  $Df(x)$  invertierbar, dann existiert aufgrund des Satzes für implizite Funktionen die inverse Funktion  $f^{-1}$  lokal in einer Umgebung von  $y$  mit  $f^{-1}(y) = x$ , sowie

$$D(f^{-1})(y) = (Df(x))^{-1}.$$

Hiermit folgt

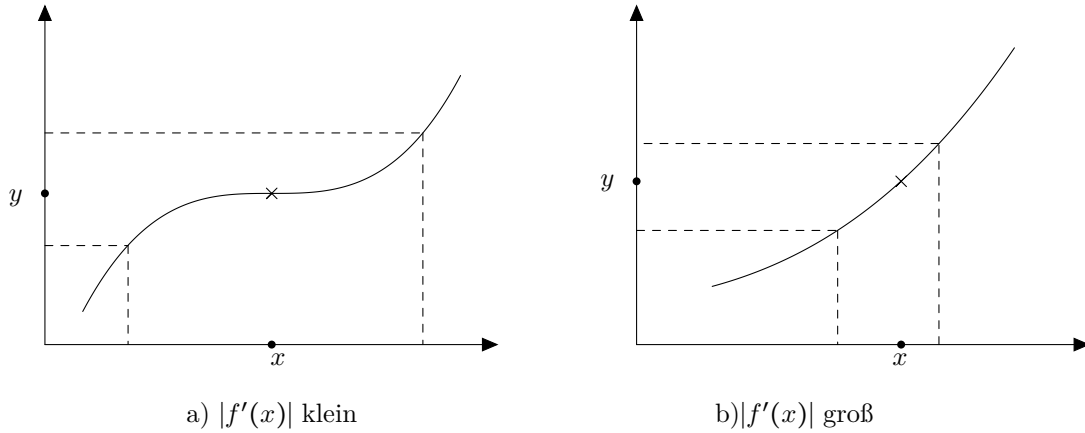
$$\begin{aligned}\kappa_{abs}(f^{-1}, y) &= \|(Df(x))^{-1}\|, \\ \kappa_{rel}(f^{-1}, y) &= \frac{\|f(x)\|}{\|x\|} \|(Df(x))^{-1}\|. \end{aligned} \quad (3.2.17)$$

Für skalare Funktionen  $f: \mathbb{R} \rightarrow \mathbb{R}$  folgt somit

$$\kappa_{rel}(f^{-1}, y) = \frac{|f(x)|}{|x|} \frac{1}{|f'(x)|}.$$

Falls  $|f'(x)| \approx 0$ , ergibt sich eine schlechte absolute Kondition, dagegen ist  $|f'(x)| \gg 0$  die absolute Kondition der Umkehrabbildung  $f^{-1}$  in  $y$  sehr gut. Dies ist auch in Abbildung 3.2 zu sehen. In Abbildung 3.2 a) bewirkt eine kleine Störung in  $y$  eine große in  $x = f^{-1}(y)$ , während in Abbildung 3.2 b) die Störung in  $y$  durch  $f^{-1}$  verringert wird.

Abbildung 3.2: Kondition  $\kappa_{abs}(f^{-1}, y)$



### 3.2 b) Komponentenweise Konditionsanalyse

*Beispiel 3.2.15.* Falls  $A$  Diagonalgestalt hat, sind die Gleichungen unabhängig voneinander (*entkoppelt*). Die erwartete relative Kondition wäre dann – wie bei skalaren Gleichungen – gleich 1. Ebenso sind Störungen nur in der Diagonale zu erwarten. Jedoch gilt für

$$\begin{aligned} A = \begin{pmatrix} 1 & 0 \\ 0 & \varepsilon \end{pmatrix} &\Rightarrow A^{-1} = \begin{pmatrix} 1 & 0 \\ 0 & \varepsilon^{-1} \end{pmatrix} \\ \Rightarrow \kappa_\infty = \kappa_2 &= \frac{1}{\varepsilon} \quad \text{für } 0 < \varepsilon \leq 1, \end{aligned}$$

welche für  $\varepsilon \rightarrow 0$  sehr groß sind.

**Definition 3.2.16.** Sei  $(f, x)$  ein Problem mit  $f(x) \neq 0$  und  $x = (x_i)_{i=1, \dots, n}$  mit  $x_i \neq 0$  für alle  $i = 1, \dots, n$ . Die *komponentenweise Kondition* von  $(f, x)$  ist die kleinste Zahl  $\kappa_{rel} \geq 0$ , so dass

$$\frac{\|f(\tilde{x}) - f(x)\|_\infty}{\|f(x)\|_\infty} \leq \kappa_{rel} \cdot \max_i \frac{|\tilde{x}_i - x_i|}{|x_i|} + o\left(\max_i \frac{|\tilde{x}_i - x_i|}{|x_i|}\right) \quad \text{für } \tilde{x} \rightarrow x.$$

Vorsicht:  $\frac{\|\tilde{x} - x\|_\infty}{\|x\|_\infty} \neq \max_i \frac{|\tilde{x}_i - x_i|}{|x_i|}$ .

**Lemma 3.2.17.** Sei  $f$  differenzierbar und fasse  $|\cdot|$  komponentenweise auf, d.h.  $|x| := \begin{pmatrix} |x_1| \\ \vdots \\ |x_n| \end{pmatrix}$ . Dann gilt

$$\kappa_{rel} = \frac{\| |Df(x)| |x| \|_\infty}{\|f(x)\|_\infty}. \quad (3.2.18)$$

*Beweis.* Vergleiche seien ebenfalls komponentenweise zu verstehen. Nach dem Satz von Taylor gilt

$$\begin{aligned} f_i(\tilde{x}) - f_i(x) &= \left( \frac{\partial f_i}{\partial x_1}(x), \dots, \frac{\partial f_i}{\partial x_n}(x) \right) \begin{pmatrix} \tilde{x}_1 - x_1 \\ \vdots \\ \tilde{x}_n - x_n \end{pmatrix} + o(\|\tilde{x} - x\|) \\ \Rightarrow |f(\tilde{x}) - f(x)| &\leq |Df(x)| \begin{pmatrix} |x_1| \frac{|\tilde{x}_1 - x_1|}{|x_1|} \\ \vdots \\ |x_n| \frac{|\tilde{x}_n - x_n|}{|x_n|} \end{pmatrix} + o\left(\max_i \frac{|\tilde{x}_i - x_i|}{|x_i|}\right) \quad \text{da } x_i \text{ fest und } \tilde{x}_i \rightarrow x_i \\ &\leq |Df(x)| |x| \max_i \frac{|\tilde{x}_i - x_i|}{|x_i|} + o\left(\max_i \frac{|\tilde{x}_i - x_i|}{|x_i|}\right) \\ \Rightarrow \frac{\|f(\tilde{x}) - f(x)\|_\infty}{\|f(x)\|_\infty} &\leq \frac{\| |Df(x)| |x| \|_\infty}{\|f(x)\|_\infty} \max_i \frac{|\tilde{x}_i - x_i|}{|x_i|} + o\left(\max_i \frac{|\tilde{x}_i - x_i|}{|x_i|}\right). \end{aligned}$$

9.11.22

Bei diesen Abschätzungen wird genutzt, dass  $x$  und  $f(x)$  bekannt sind und somit keine Rolle bei  $o$  spielen. Wähle  $\tilde{x}_i = x_j + h \operatorname{sign} \frac{\partial f_i}{\partial x_j}(x)$  mit  $h > 0$ , dann gilt

$$|Df_i(x)(\tilde{x} - x)| = Df_i(x)(\tilde{x} - x)$$

und in obiger Rechnung gilt Gleichheit. Also folgt

$$\kappa_{rel} = \frac{\| |Df(x)| |x| \|_\infty}{\|f(x)\|_\infty}.$$

□

*Beispiel 3.2.18.*

a) Komponentenweise Kondition der Multiplikation:

$$\begin{aligned} f: \mathbb{R}^2 &\rightarrow \mathbb{R}, f(x, y) := xy \\ \Rightarrow Df(x, y) &= (y, x) \\ \Rightarrow \kappa_{rel|\cdot|} f(x, y) &= \frac{\left\| \begin{pmatrix} |y|, |x| \end{pmatrix} \begin{pmatrix} |x| \\ |y| \end{pmatrix} \right\|_\infty}{|xy|} = \frac{2|x||y|}{|xy|} = 2. \end{aligned}$$

b) Komponentenweise Kondition eines linearen Gleichungssystems:

Löse  $Ax = b$  mit möglichen Störungen in  $b$ , also betrachte

$$\begin{aligned} f: b &\mapsto A^{-1}b \\ \kappa_{rel|\cdot|}(f, b) &= \frac{\| |A^{-1}| |b| \|_\infty}{\|A^{-1}b\|_\infty}. \end{aligned}$$

Falls  $A$  eine Diagonalmatrix ist, folgt

$$\kappa_{rel|\cdot|}(f, b) = 1.$$

c) Komponentenweise Kondition des Skalarproduktes  $\langle xy \rangle := \sum_{i=1}^n x_i y_i = x^T y$ :

Für  $f: \mathbb{R}^2 \rightarrow \mathbb{R}$ ,  $f(x, y) = \langle xy \rangle$  folgt

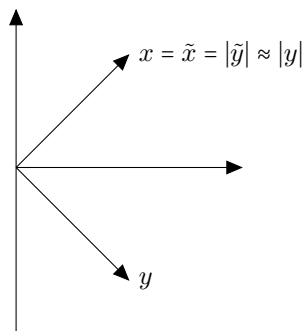
D  $f(x, y) = (y^T, x^T)$ . Also gilt mit  $\cos(x, y) = \frac{\langle xy \rangle}{\|x\|_2 \|y\|_2}$

$$\kappa_{rel} = \frac{\left\| \begin{pmatrix} y^T \\ x^T \end{pmatrix} \right\|_\infty \left\| \begin{pmatrix} x \\ y \end{pmatrix} \right\|_\infty}{\|\langle x, y \rangle\|_\infty} = \frac{2\|y^T\|_\infty \|x\|_\infty}{|\langle x, y \rangle|} = 2 \frac{\langle |x|, |y| \rangle}{|\langle x, y \rangle|} = 2 \frac{\cos(|x|, |y|)}{\cos(x, y)}.$$

Falls  $x$  und  $y$  nahezu senkrecht aufeinander stehen, kann das Skalarprodukt sehr schlecht konditioniert sein.

Betrachte zum Beispiel für  $x = \tilde{x} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$  und  $y = \begin{pmatrix} 1 + 10^{-10} \\ -1 \end{pmatrix}$ ,  $\tilde{y} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$ .

Abbildung 3.3: Vektoren mit nahezu gleichem komponentenweisen Betrag

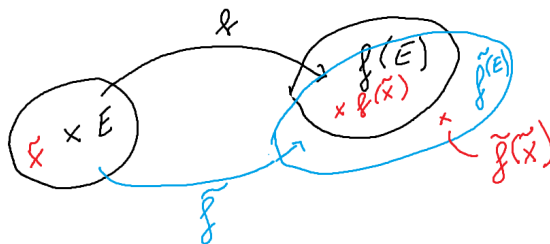


### 3.3 Stabilität von Algorithmen

Bislang haben wir die Kondition eines gegebenen Problems  $(f, x)$  betrachtet. Nun stellt sich die Frage:

Was passiert durch das Implementieren am Rechner?

Ein „stabiler“ Algorithmus sollte ein gut konditioniertes Problem nicht „kaputt machen“.



#### 3.3 a) Vorwärtsanalyse

Die Fehlerfortpflanzung durch die einzelnen Rechenschritte, aus denen die Implementierung aufgebaut ist, wird abgeschätzt.

Bemerkung 3.3.1. Für die Rechenoperationen  $+$ ,  $-$ ,  $\cdot$ ,  $/$ , kurz  $\nabla$ , gilt nach 3.1.7:

$$fl(a \nabla b) = (a \nabla b)(1 + \varepsilon) = (a \nabla b) \frac{1}{1 + \mu} \quad (3.3.1)$$

mit  $|\varepsilon|, |\mu| \leq \text{eps}$ .

Beispiel 3.3.2. Sei

$$f(x_1, x_2, x_3) := \frac{x_1 x_2}{x_3}$$

mit Maschinenzahlen  $x_1, x_2$  und  $x_3 \neq 0$  und bezeichne  $x := (x_1, x_2, x_3)$ . Weiterhin sei der Algorithmus durch

$$f(x_1, x_2, x_3) = (f^{(2)} \circ f^{(1)})(x_1, x_2, x_3)$$

gegeben mit

$$f^{(1)}(x_1, x_2, x_3) = (x_1 x_2, x_3) \quad \text{und} \quad f^{(2)}(y, z) = \frac{y}{z}.$$

Die Implementierung  $\tilde{f}$  von  $f$  beinhaltet Rundungsfehler.

$$\begin{aligned} \Rightarrow \tilde{f}^{(1)}(x) &= (fl(x_1 x_2), x_3) = (x_1 x_2(1 + \varepsilon_1), x_3) \text{ mit } |\varepsilon_1| \leq \text{eps:} \\ \tilde{f}(x) &= \tilde{f}^{(2)}(\tilde{f}^{(1)}(x)) \\ &= fl(f^{(2)}(x_1 x_2(1 + \varepsilon_1), x_3)) \\ &= \frac{x_1 x_2(1 + \varepsilon_1)}{x_3} (1 + \varepsilon_2) \\ &= f(x)(1 + \varepsilon_1)(1 + \varepsilon_2) \text{ mit } |\varepsilon_2| \leq \text{eps,} \\ \Rightarrow \frac{|\tilde{f}(x) - f(x)|}{|f(x)|} &= |\varepsilon_1 + \varepsilon_2 + \varepsilon_1 \varepsilon_2| \leq 2 \text{eps} + \text{eps}^2. \end{aligned}$$

Dies ist eine „worst case“ Analyse, da immer der maximale Fehler eps angenommen wird, und gibt i.d.R. eine starke Überschätzung des Fehlers an. Für qualitative Aussagen sind sie jedoch nützlich.

In Computersystemen stehen mehr Operationen wie  $\nabla$  zur Verfügung, die mit einer relativen Genauigkeit eps realisiert werden können.

**Definition 3.3.3.** Eine Abbildung  $\phi: U \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^m$  heißt *elementar ausführbar*, falls es eine *elementare Operation*  $\tilde{\phi}: \mathbb{F}^n \rightarrow \mathbb{F}^m$  gibt, wobei  $\mathbb{F}$  die Menge der Maschinenzahlen bezeichne, mit

$$|\tilde{\phi}_i(x) - \phi_i(x)| \leq \text{eps} |\phi_i(x)| \quad \forall x \in \mathbb{F}^n \text{ und } i = 1, \dots, m. \quad (3.3.2)$$

$\tilde{\phi}$  heißt dann *Realisierung* von  $\phi$ .

**Bemerkung:** Aus (3.3.2) folgt für  $1 \leq p \leq \infty$

$$\|\tilde{\phi}(x) - \phi(x)\|_p \leq \text{eps} \|\phi(x)\|_p \quad \forall x \in \mathbb{F}^n. \quad (3.3.3)$$

**Definition 3.3.4.** Sei  $f: E \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^m$  gegeben. Ein Tupel  $(f^{(1)}, \dots, f^{(l)})$  mit  $l \in \mathbb{N}$  von elementar ausführbaren Abbildungen

$$f^{(i)}: U_i \subseteq \mathbb{R}^{k_i} \rightarrow U_{i+1} \subseteq \mathbb{R}^{k_{i+1}}$$

mit  $k_1 = n$  und  $k_{l+1} = m$  heißt *Algorithmus von  $f$* , falls

$$f = f^{(l)} \circ \dots \circ f^{(1)}.$$

Das Tupel  $(\tilde{f}^{(1)}, \dots, \tilde{f}^{(l)})$  mit Abbildungen  $\tilde{f}^{(i)}$ , welche Realisierungen der  $f^{(i)}$  sind, heißt *Implementation* von  $(f^{(1)}, \dots, f^{(l)})$ . Die Komposition

$$\tilde{f} = \tilde{f}^{(l)} \circ \dots \circ \tilde{f}^{(1)}$$

heißt *Implementation von  $f$* .

Im Allgemeinen gibt es verschiedene Implementierungen einer Abbildung  $f$ .

**Lemma 3.3.5** (Fehlerfortpflanzung). *Sei  $x \in \mathbb{R}^n$  und  $\tilde{x} \in \mathbb{F}^n$  mit  $|\tilde{x}_i - x_i| \leq \text{eps} |x_i|$  für alle  $i = 1, \dots, n$ . Sei  $(f^{(1)}, \dots, f^{(l)})$  ein Algorithmus für  $f$  und  $(\tilde{f}^{(1)}, \dots, \tilde{f}^{(l)})$  eine zugehörige Implementation. Mit den Abkürzungen*

$$x^{(j+1)} := f^{(j)} \circ \dots \circ f^{(1)}(x), \quad x^{(1)} := x$$

*und mit entsprechenden  $\tilde{x}^{(j+1)}$  gilt, falls  $x^{(j+1)} \neq 0$  für alle  $j = 0, \dots, (l-1)$  und  $\|\cdot\|$  eine beliebige  $p$ -Norm ist:*

$$\frac{\|\tilde{x}^{(j+1)} - x^{(j+1)}\|}{\|x^{(j+1)}\|} \leq \text{eps} \mathcal{K}^{(j)} + o(\text{eps}) \quad (3.3.4)$$

*mit  $\mathcal{K}^{(j)} = (1 + \kappa^{(j)} + \kappa^{(j)} \kappa^{(j-1)} + \dots + \kappa^{(j)} \dots \kappa^{(1)})$ , wobei  $\kappa^{(j)} := \kappa_{\text{rel}}(f^{(j)}, x^{(j)})$  die Kondition der elementar ausführbaren Operation  $f^{(j)}$  ist.*

14.11.22

*Beweis.* Wir zeigen 3.3.5 durch Induktion. Betrachte hierfür vorerst folgende Abschätzung

$$\begin{aligned} \frac{\|\tilde{x}^{(j+1)} - x^{(j+1)}\|}{\|x^{(j+1)}\|} &= \frac{\|\tilde{f}^{(j)}(\tilde{x}^{(j)}) - f^{(j)}(x^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} \\ &\leq \frac{\|\tilde{f}^{(j)}(\tilde{x}^{(j)}) - f^{(j)}(\tilde{x}^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} + \frac{\|f^{(j)}(\tilde{x}^{(j)}) - f^{(j)}(x^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} \\ &= \frac{\|\tilde{f}^{(j)}(\tilde{x}^{(j)}) - f^{(j)}(\tilde{x}^{(j)})\|}{\|f^{(j)}(\tilde{x}^{(j)})\|} \frac{\|f^{(j)}(\tilde{x}^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} + \frac{\|f^{(j)}(\tilde{x}^{(j)}) - f^{(j)}(x^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} \\ &\stackrel{(3.3.3)}{\leq} \text{eps} \frac{\|f^{(j)}(\tilde{x}^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} + \frac{\|f^{(j)}(\tilde{x}^{(j)}) - f^{(j)}(x^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} \\ &\leq \text{eps} \left( 1 + \frac{\|f^{(j)}(\tilde{x}^{(j)}) - f^{(j)}(x^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} \right) + \frac{\|f^{(j)}(\tilde{x}^{(j)}) - f^{(j)}(x^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} \\ &= \text{eps} + (\text{eps} + 1) \frac{\|f^{(j)}(\tilde{x}^{(j)}) - f^{(j)}(x^{(j)})\|}{\|f^{(j)}(x^{(j)})\|} \\ &\stackrel{\text{Def. 3.2.7}}{\leq} \text{eps} + (\text{eps} + 1) \left( \kappa^{(j)} \frac{\|\tilde{x}^{(j)} - x^{(j)}\|}{\|x^{(j)}\|} + o\left(\frac{\|\tilde{x}^{(j)} - x^{(j)}\|}{\|x^{(j)}\|}\right) \right) \\ &= \text{eps} + (\text{eps} + 1) \left( \kappa^{(j)} \frac{\|\tilde{x}^{(j)} - x^{(j)}\|}{\|x^{(j)}\|} \right) + o\left(\frac{\|\tilde{x}^{(j)} - x^{(j)}\|}{\|x^{(j)}\|}\right). \end{aligned}$$

Nach Voraussetzung gilt (3.3.4) mit  $\mathcal{K}^{(0)} = 1$  für  $j = 0$ .

Für  $j = 1$  folgt mithilfe obiger Abschätzung und der Voraussetzung  $\frac{\|\tilde{x} - x\|}{\|x\|} = \frac{\|\tilde{x}^{(1)} - x^{(1)}\|}{\|x^{(1)}\|} \leq \text{eps}$

$$\begin{aligned} \frac{\|\tilde{x}^{(2)} - x^{(2)}\|}{\|x^{(2)}\|} &\leq \text{eps} + (\text{eps} + 1) (\kappa^{(1)} \text{eps} + o(\text{eps})) \\ &= \text{eps}(1 + \kappa^{(1)}) + o(\text{eps}) \\ &= \text{eps} \mathcal{K}^{(1)} + o(\text{eps}). \end{aligned}$$

Hiermit ist der Induktionsanfang gezeigt.

Induktionsschritt von  $j - 1$  zu  $j$ :

$$\begin{aligned}
 \frac{\|\tilde{x}^{(j+1)} - x^{(j+1)}\|}{\|x^{(j+1)}\|} &\leq \text{eps} + (\text{eps} + 1) \left( \kappa^{(j)} \frac{\|\tilde{x}^{(j)} - x^{(j)}\|}{\|x^{(j)}\|} + o\left(\frac{\|\tilde{x}^{(j)} - x^{(j)}\|}{\|x^{(j)}\|}\right) \right) \\
 &\stackrel{\text{Ind.-Vor.}}{\leq} \text{eps} + (1 + \text{eps}) \kappa^{(j)} [\text{eps} \mathcal{K}^{(j-1)} + o(\text{eps})] \\
 &\quad + (1 + \text{eps}) o(\text{eps} \mathcal{K}^{(j-1)} + o(\text{eps})) \\
 &= \text{eps} (1 + \kappa^{(j)} \mathcal{K}^{(j-1)}) + o(\text{eps}).
 \end{aligned}$$

Mit  $\mathcal{K}^{(j)} = 1 + \kappa^{(j)} \mathcal{K}^{(j-1)}$  folgt die Behauptung.  $\square$

Hiermit folgt:

**Korollar 3.3.6.** *Unter der Voraussetzung von Lemma 3.3.5 gilt*

$$\frac{\|\tilde{f}(\tilde{x}) - f(x)\|}{\|f(x)\|} \leq \text{eps} (1 + \kappa^{(l)} + \kappa^{(l)} \kappa^{(l-1)} + \dots + \kappa^{(l)} \dots \kappa^{(1)}) + o(\text{eps}) \quad (3.3.5)$$

*Bemerkung 3.3.7.* Mit Korollar 3.3.6 ist offensichtlich, dass schlecht konditionierte Probleme zu elementar ausführbaren Abbildungen so früh wie möglich ausgeführt werden sollten. Salopp formuliert ergibt sich dies auch aus der unpräzisen Begründung: Falls  $(f^{(j)}, x^{(j)})$  eine große Kondition hat, werden mit  $f^{(j)}$  alle vorher aufgelaufenen Fehler verstärkt. Daher sollte  $f^{(j)}$  so früh wie möglich angewendet werden, wo der Eingabefehler noch 'klein' ist. Allerdings hängt  $\kappa^{(j)}$  nicht nur von  $f^{(j)}$ , sondern auch vom Zwischenergebnis  $x^{(j)}$  ab, welches a priori unbekannt ist.

Nach Beispiel 3.2.9 ist die Subtraktion zweier annähernd gleicher Zahlen schlecht konditioniert. Deshalb sollten unvermeidbare Subtraktionen möglichst früh durchgeführt werden.

*Bemerkung 3.3.8.* (Sprechweise).

Der Quotient

$$\frac{\overbrace{\|\tilde{f}(\tilde{x}) - f(x)\|}^{\text{Gesamtfehler}}}{\|f(x)\|} \bigg/ \underbrace{\frac{\|f(\tilde{x}) - f(x)\|}{\|f(x)\|}}_{\substack{\text{Fehler} \\ \text{durch} \\ \text{das Problem}}} \underbrace{\frac{\|\tilde{x} - x\|}{\|x\|}}_{\substack{\text{Eingabe-} \\ \text{fehler}}} \quad (3.3.6)$$

gibt die *Güte des Algorithmus* an.

Als *Stabilitätsindikator* kann also

$$\sigma(f, \tilde{f}, x) := \frac{\mathcal{K}}{\kappa_{rel}(f, x)} \quad (3.3.7)$$

verwendet werden und es gilt

$$\frac{\|\tilde{f}(\tilde{x}) - f(x)\|}{\|f(x)\|} < \underbrace{\sigma(f, \tilde{f}, x)}_{\substack{\text{Beitrag} \\ \text{des} \\ \text{Algorithmus}}} \underbrace{\kappa_{rel}(f, x)}_{\substack{\text{Beitrag} \\ \text{des} \\ \text{Problems}}} \underbrace{\text{eps} + o(\text{eps})}_{\substack{\text{Rundungs-} \\ \text{fehler}}}.$$

Falls  $\sigma(f, \tilde{f}, x) < 1$ , dämpft der Algorithmus die Fehlerfortpflanzung der Eingabe- und Rundungsfehler und heißt *stabil*. Für  $\sigma(f, \tilde{f}, x) \gg 1$  heißt der Algorithmus *instabil*.

*Beispiel 3.3.9.* Nach Gleichung (3.3.3) gilt für die Elementaroperationen  $\mathcal{K} \leq 1$ . Da für die Subtraktion zweier annähernd gleich großer Zahlen  $\kappa_{rel} \gg 1$  gilt, ist der Stabilitätsfaktor der Subtraktion zweier annähernd gleich großer Zahlen sehr klein und der Algorithmus stabil, Falls es sich jedoch bei einer zusammengesetzten Abbildung  $f = h \circ g$  bei der zweiten Abbildung  $h$  um eine Subtraktion handelt, gilt

$$\mathcal{K} = (1 + \kappa(sub) + \kappa(sub) \kappa(g))$$

und die Stabilität ist gefährdet.



Genauere Abschätzungen und damit genauere Indikatoren können durch komponentenweise Betrachtungen erhalten werden. Ein Beispiel ist  $f(b, c) = -b + \sqrt{b^2 - c}$ , welches für  $b \gg a > 0$  gut konditioniert ist, aber der zugehörige Algorithmus instabil ist.

### 3.3 b) Rückwärtsanalyse

Die Fragestellung ist nun:

Kann  $\tilde{f}(\tilde{x})$  als exaktes Ergebnis von einer gestörten Eingabe  $\hat{x}$  unter der exakten Abbildung  $f$  aufgefasst werden?  
D.h.

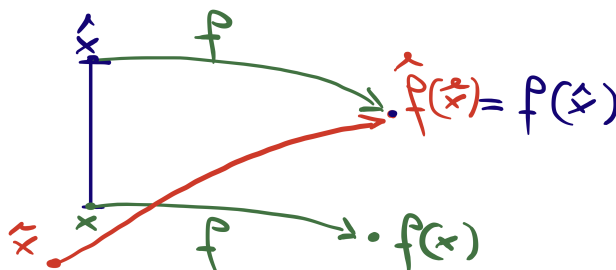
$$\exists \hat{x} \in \mathbb{R}^n: f(\hat{x}) = \tilde{f}(\tilde{x}),$$

dann schätze den Fehler

$$\|\hat{x} - x\|$$

ab bzw. falls  $f$  nicht injektiv ist, betrachte

$$\min_{\hat{x} \in \mathbb{R}^n} \{ \|\hat{x} - x\| \mid f(\hat{x}) = \tilde{f}(\tilde{x}) \}.$$



Interessant ist dies z.B. bei: Die Eingangsdaten seien Messdaten  $\tilde{x}$  mit 1% relativer Genauigkeit. Liefert die Rückwärtsanalyse, dass  $\tilde{f}(\tilde{x})$  als exaktes Ergebnis  $f(\hat{x})$  mit Eingangsdaten  $\hat{x}$ , die höchstens um 0,5 % schwanken, aufgefasst werden kann, so ist das Verfahren „geeignet“.

Die Rückwärtsanalyse ist

- in der Regel leichter durchführbar als die Vorwärtsanalyse und
- ebenfalls nur eine qualitative Schätzung der Genauigkeit der numerisch berechneten Werte.

Bemerkung 3.3.10.

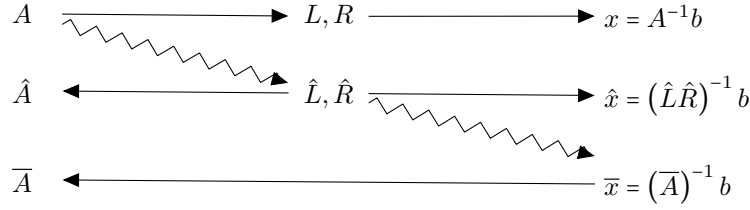
$$\begin{array}{rclcl} \text{Vorwärtsfehler} & \leq & \text{Kondition des Problems} & \cdot & \text{Rückwärtsfehler} \\ \|\tilde{f}(\tilde{x}) - f(x)\| & \leq & \kappa(f, x) & \cdot & \|\hat{x} - x\| \end{array}$$

Ein Beispiel ist die Rückwärtsanalyse der Gauß-Elimination. Sie geht auf Wilkinson zurück. Wir geben hier einige Ergebnisse dazu an.

**Satz 3.3.11.**  $A \in \mathbb{R}^{n \times n}$  besitze eine LR-Zerlegung. Dann berechnet die Gauß-Elimination Matrizen  $\hat{L}$  und  $\hat{R}$ , so dass  $\hat{L}\hat{R} = \hat{A}$  und

$$\begin{aligned} \|\hat{A} - A\| &\leq \frac{\text{eps}}{1 - n \text{eps}} \left( \|\hat{L}\| \left\| \begin{pmatrix} 1 & & 0 \\ & 2 & \\ 0 & & \ddots \\ & & & n \end{pmatrix} \right\| \|\hat{R}\| - \|\hat{R}\| \right) \\ &\leq \frac{n \text{eps}}{1 - n \text{eps}} \|\hat{L}\| \|\hat{R}\| \\ &= n \text{eps} \|\hat{L}\| \|\hat{R}\| + \mathcal{O}(n^2 \text{eps}^2) \end{aligned}$$

falls  $n \text{eps} \leq \frac{1}{2}$ .



*Beweis.* [siehe 12] Die letzte Abschätzung folgt aus der geometrischen Reihe.  $\square$

**Satz 3.3.12** (Sautter 1971).  $A \in \mathbb{R}^{n \times n}$  besitze eine LR-Zerlegung. Dann berechnet das Gaußsche Eliminationsverfahren für das Gleichungssystem  $Ax = b$  eine Lösung  $\bar{x}$  mit

$$\bar{A}\bar{x} = b \quad \text{und} \quad \|\bar{A} - A\| \leq 2n \text{eps} \|\hat{L}\| \cdot \|\hat{R}\| + \mathcal{O}(n^2 \text{eps}^2).$$

*Beweis.* [siehe 4].  $\square$

Weitere Abschätzungen existieren für Gauß-Elimination mit Pivotisierung und für spezielle Klassen von Matrizen.

### 3.3.13. Allgemeine Faustregeln für die LR-Zerlegung

- Falls für die Matrix  $n \|\hat{L}\| \|\hat{R}\|$  dieselbe Größenordnung wie  $\|A\|$  besitzt, ist der Algorithmus „gutartig“.
- Für tridiagonale Matrizen ist der Algorithmus mit Spaltenpivotisierung stabil.
- Falls  $A$  oder  $A^T$  strikt diagonal dominant ist, d.h.

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}| \quad \forall i = 1, \dots, n,$$

ist Spaltenpivotisierung überflüssig. Der Algorithmus ist stabil.

- Für symmetrisch positiv definite Matrizen sollte keine Pivotisierung durchgeführt werden, um die Symmetrie zu erhalten. Der Algorithmus ist stabil.

**Vorsicht:** Selbst wenn die LR-Zerlegung in dem Sinne stabil ist, dass  $\|\bar{A} - A\|$  für  $\bar{A}\bar{x} = b$  klein ist, kann die numerische Lösung  $\bar{x}$  sehr ungenau sein, da der Vorwärtsfehler  $\|\bar{x} - x\|$  von der Kondition  $\text{cond}(A)$  abhängt.

Ein Beispiel hierzu ist die Hilbertmatrix  $H_n = \left( \frac{1}{i+j-1} \right)_{i,j=1,\dots,n}$ , für die  $\text{cond}(H_n)$  exponentiell mit der Dimension  $n$  wächst ( $\text{cond}_2(H_3) \approx 524.06$ ,  $\text{cond}_2(H_4) \approx 15513.74$ ).

## 3.4 Beurteilung von Näherungslösungen linearer Gleichungssysteme

Zu  $Ax = b$  liege eine Näherungslösung  $\tilde{x}$  vor.

### 3.4 a) Im Sinne der Vorwärtsanalyse

Im Sinne der Vorwärtsanalyse und der Fehlerentwicklung durch das Problem gilt nach Beispiel 3.2.10

$$\frac{\|\tilde{x} - x\|}{\|x\|} \leq \text{cond}(A) \frac{\|\Delta b\|}{\|b\|} \leq \text{cond}(A) \frac{\|r(\tilde{x})\|}{\|b\|} \quad (3.4.1)$$

mit dem *Residuum*

$$r(\tilde{x}) := A\tilde{x} - b = \tilde{b} - b = \Delta b \quad (3.4.2)$$

Wie der absolute Fehler ist das Residuum skalierungsabhängig. Daher ist  $\|r(\tilde{x})\|$  ungeeignet, um Genauigkeitsaussagen zu treffen. Um den relativen Fehler in  $x$  abzuschätzen, ist die Betrachtung von

$$\frac{\|r(\tilde{x})\|}{\|b\|} \quad (3.4.3)$$

geeigneter. Für große  $\text{cond}(A)$  ist dieser Quotient jedoch weiterhin ungeeignet.

### 3.4 b) Im Sinne der Rückwärtsanalyse

**Satz 3.4.1** (Prager und Oettli, 1964). *Sei  $\tilde{x}$  eine Näherungslösung für  $Ax = b$ . Falls*

$$\|r(\tilde{x})\| \leq \varepsilon(\|A\| \|\tilde{x}\| + \|b\|). \quad (3.4.4)$$

*dann existiert eine Matrix  $\tilde{A}$  und ein Vektor  $\tilde{b}$  mit  $\tilde{A}\tilde{x} = \tilde{b}$  und*

$$\|\tilde{A} - A\| \leq \varepsilon \|A\| \quad \text{und} \quad \|\tilde{b} - b\| \leq \varepsilon \|b\|. \quad (3.4.5)$$

Aufgrund von (3.4.4) wird der komponentenweise relative Rückwärtsfehler durch

$$\max_i \frac{|A\tilde{x} - b|_i}{(|A|\|\tilde{x}\| + |b|)_i}$$

abgeschätzt.

Für den normweisen relativen Rückwärtsfehler gilt entsprechend (Rigal und Gaches 1967)

$$\frac{\|A\tilde{x} - b\|}{\|A\| \|\tilde{x}\| + \|b\|}.$$



# Kapitel 4

## Lineare Gleichungssysteme: Direkte Methoden (Fortsetzung)

### 4.1 Gaußsches Eliminationsverfahren mit Äquilibration und Nachiteration

Mit Skalierung

$D_z A$  (Zeilenskalierung) oder  
 $D_s A$  (Spaltenskalierung)

mittels Diagonalmatrizen  $D_z, D_s$  lässt sich eine Pivotstrategie beliebig abändern.

Was ist eine „gute“ Skalierung?

Skalierung ändert die Länge der Basisvektoren des Bild- bzw. des Urbildraumes. Durch Normierung der Länge auf 1 wird die Pivotstrategie unabhängig von der gewählten Einheit.

Sei  $A \in \mathbb{R}^{n \times m}$  und  $\|\cdot\|$  eine Vektornorm.

#### 4.1.1. Äquilibration der Zeilen

Alle Zeilen von  $D_z A$  haben die gleiche Norm, z.B.  $\|\cdot\| = 1$ , wofür

$$D_z = \begin{pmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \end{pmatrix} \quad \text{mit } \sigma_i := \frac{1}{\|(a_{i1}, \dots, a_{im})\|} \quad (4.1.1)$$

gesetzt wird.

#### 4.1.2. Äquilibration der Spalten

Alle Spalten von  $AD_s$  haben die gleiche Norm, z.B.  $\|\cdot\| = 1$ , wofür

$$D_s = \begin{pmatrix} \tau_1 & & 0 \\ & \ddots & \\ 0 & & \tau_m \end{pmatrix} \quad \text{mit } \tau_j := 1 / \left\| \begin{pmatrix} a_{1j} \\ \vdots \\ a_{nj} \end{pmatrix} \right\| \quad (4.1.2)$$

gesetzt wird.

Äquilibration von Zeilen **und** Spalten ergibt für die Bestimmung von  $\kappa_i, \tau_i$  ein nichtlineares Gleichungssystem und ist i.d.R. zu aufwendig.

**Lemma 4.1.3.** Sei  $A$  zeilenäquilibriert bzgl. der  $l_1$ -Norm, dann gilt:

$$\text{cond}_\infty(A) \leq \text{cond}_\infty(DA) \quad (4.1.3)$$

für alle regulären Diagonalmatrizen  $D$ .

*Beweis.* Siehe Übungsaufgabe. □

Wie in Kapitel 3 gesehen, kann die Näherungslösung  $\tilde{x}$  trotz Pivotisierung und Äquilibration noch sehr ungenau sein.

#### 4.1.4. Nachiteration

Die Näherung  $\tilde{x}$  kann durch Nachiteration verbessert werden. Falls  $\tilde{x}$  exakt ist, gilt:

$$r(\tilde{x}) := b - A\tilde{x} = 0. \quad (4.1.4)$$

Ansonsten ist  $A(x - \tilde{x}) = r(\tilde{x})$ .

Also löse die *Korrekturgleichung*

$$A\Delta x = r(\tilde{x}) \quad (4.1.5)$$

und setze

$$x^{(1)} := \tilde{x} + \Delta x.$$

Wiederhole dies sooft, bis  $x^{(i)}$  „genau genug“ ist.

Die Lösung  $\tilde{x}$  wird durch Nachiteration meist mit sehr gutem Erfolg verbessert. Genauer ist z.B. zu finden in [3].

Gleichung (4.1.5) wird mit der bereits vorhandenen LR-Zerlegung nur mit der neuen rechten Seite  $r(\tilde{x})$  gelöst, d.h. eine Vorwärts- und eine Rückwärtssubstitution mit  $\mathcal{O}(n^2)$  flops je Iteration sind anzuwenden.

*Bemerkung 4.1.5* (nach Skeel 1980). Die Gauß-Elimination mit Spaltenpivotsuche und einer Nachiteration ist komponentenweise stabil.

## 4.2 Cholesky-Verfahren

Im Folgenden sei  $A$  eine symmetrische, positiv definite Matrix in  $\mathbb{R}^{n \times n}$ , d.h.  $A = A^T$  und

$$\langle x, Ax \rangle = x^T Ax > 0 \text{ für alle } x \neq 0. \quad (4.2.1)$$

(Kurz: **spd Matrix**.)

**Satz 4.2.1.** Für jede spd Matrix  $A \in \mathbb{R}^{n \times n}$  gilt:

- i)  $A$  ist invertierbar.
- ii)  $a_{ii} > 0$  für  $i = 1, \dots, n$ .
- iii)  $\max_{ij} |a_{ij}| = \max_i a_{ii}$ .
- iv) Bei der Gauß-Elimination ohne Pivotsuche ist jede Restmatrix wieder eine spd Matrix.

*Beweis.*

- i) Folgt aus (4.2.1).
- ii) Sei  $e_i$  der  $i$ -te Einheitsvektor, so folgt  $a_{ii} = e_i^T A e_i > 0$ .
- iii) Siehe Übungsaufgabe.

iv) Es gilt:

$$\begin{aligned}
 A^{(1)} &:= A = \begin{pmatrix} a_{11} & z^T \\ z & B^{(1)} \end{pmatrix} \\
 A^{(2)} &:= L_1 A^{(1)} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ -\frac{1}{a_{11}}z & I & & \end{pmatrix} A^{(1)} = \begin{pmatrix} a_{11} & z^T \\ 0 & B^{(2)} \\ \vdots & \\ 0 & \end{pmatrix} \\
 \Rightarrow L_1 A^{(1)} L_1^T &= \begin{pmatrix} a_{11} & z^T \\ 0 & B^{(2)} \\ \vdots & \\ 0 & \end{pmatrix} \begin{pmatrix} 1 & -\frac{1}{a_{11}}z^T \\ 0 & I \\ \vdots & \\ 0 & \end{pmatrix} = \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & B^{(2)} & \\ 0 & & & \end{pmatrix}.
 \end{aligned}$$

Weiterhin gilt  $x \neq 0 \Leftrightarrow L_1 x \neq 0$  da  $L_1$  invertierbar.

$$\begin{aligned}
 \Rightarrow \tilde{x}^T B^{(2)} \tilde{x} &= x^T L_1 A^{(1)} L_1^T x && \text{für } x := \begin{pmatrix} 0 \\ \tilde{x} \end{pmatrix} \\
 &= (L_1^T x)^T A (L_1^T x) > 0 && \forall \tilde{x} \neq 0
 \end{aligned}$$

und damit ist auch  $B^{(2)}$  spd.

Induktiv folgt hiermit iv).

□

Insbesondere ergibt sich

$$(L_{n-1} \cdots L_1) A^{(1)} (L_1^T \cdots L_{n-1}^T) = \begin{pmatrix} d_1 & & 0 \\ & \ddots & \\ 0 & & d_n \end{pmatrix},$$

wobei  $d_i$  das  $i$ -te Diagonalelement von  $A^{(i)}$  ist und somit  $d_i > 0$  für  $i = 1, \dots, n$  gilt.

Sei  $L := (L_1^{-1} \cdots L_{n-1}^{-1})$  wie in (2.1.9), so ergibt sich:

**Folgerung 4.2.2** (Cholesky-Zerlegung). *Für jede spd Matrix  $A$  existiert eine eindeutige Zerlegung der Form*

$$A = L D L^T,$$

wobei  $L$  eine reelle unipotente (d.h.  $l_{ii} = 1$ ) (normierte) untere Dreiecksmatrix und  $D$  eine positive Diagonalmatrix ist. Diese Zerlegung heißt rationale Cholesky-Zerlegung. Die Zerlegung

$$A = \bar{L} \bar{L}^T \tag{4.2.2}$$

mit der reellen unteren Dreiecksmatrix

$$\bar{L} = L \begin{pmatrix} \sqrt{d_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{d_n} \end{pmatrix} = L D^{\frac{1}{2}}$$

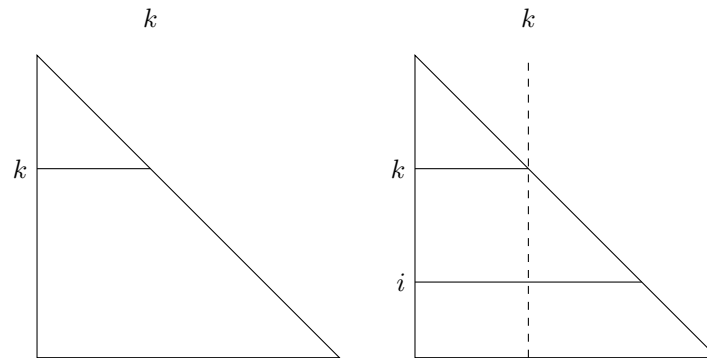
heißt Cholesky-Zerlegung.

Wegen (4.2.2) gilt:

$$a_{kk} = \bar{l}_{k1}^2 + \cdots + \bar{l}_{kk}^2 \tag{4.2.3}$$

$$a_{ik} = \bar{l}_{i1} \bar{l}_{k1} + \cdots + \bar{l}_{ik} \bar{l}_{kk} \quad \text{für } i > k \tag{4.2.4}$$

Abbildung 4.1: Spaltenweise Berechnung der Cholesky-Zerlegung



Demnach funktioniert die spaltenweise (und auch die zeilenweise) Berechnung der Cholesky-Zerlegung.

Es ergibt sich folgender Algorithmus.

---

**Algorithmus 4.2.3 : Cholesky-Zerlegung**

---

**for**  $k = 1, \dots, n$  **do**

$$l_{kk} = (a_{kk} - \sum_{j=1}^{k-1} l_{kj}l_{kj})^{\frac{1}{2}}$$

**for**  $i = k + 1, \dots, n$  **do**

$$l_{ik} = (a_{ik} - \sum_{j=1}^{k-1} l_{ij}l_{kj})/l_{kk}$$

**end for**

**end for**

---

#### 4.2.4. Rechenaufwand für die Cholesky-Zerlegung in flops.

Es sind je

$$\frac{1}{6}(n^3 - n) \text{ Additionen sowie Multiplikationen,}$$

$$\frac{1}{6}(3n^2 - 3n) \text{ Divisionen und}$$

$$n \text{ Quadratwurzeln}$$

also ca.  $\frac{2}{3}n^3$  flops für große  $n$  notwendig.

Im Vergleich zur LR-Zerlegung halbiert sich in etwa der Aufwand (siehe Übungsaufgabe).

*Bemerkung 4.2.5.*

- Wegen (4.2.3) gilt  $|\bar{l}_{kj}| \leq \sqrt{a_{kk}}$ , d.h. die Matrizeneinträge können nicht zu groß werden.
- Für spd Matrizen ist der Cholesky-Algorithmus stabil.
- Da  $A$  symmetrisch ist, muss nur die untere Dreiecksmatrix gespeichert werden. In Algorithmen kann  $\bar{L}$  in eine Kopie dieser Dreiecksmatrix geschrieben werden.
- „Fast singuläre“ Matrizen können durch die Diagonale erkannt werden.

Die rationale Cholesky-Zerlegung kann für eine größere Menge von symmetrischen Matrizen verwendet werden, da nicht  $d_{ii} > 0$  vorausgesetzt wird. Dieser Algorithmus ist jedoch im Allgemeinen nicht stabil, da die Faktoren

beliebig groß werden können, z.B. für  $\begin{pmatrix} \varepsilon & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1/\varepsilon & 1 \end{pmatrix} \begin{pmatrix} \varepsilon & 0 \\ 0 & -1/\varepsilon \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1/\varepsilon & 1 \end{pmatrix}^T$ .



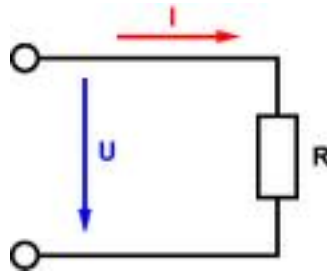
## 4.3 Lineare Ausgleichsprobleme

Beispiel 4.3.1. (siehe Einführung)

Seien  $m$  Messungen  $(I_i, U_i)$  für die Stromstärke  $I$  und die Spannung  $U$  gegeben. Gesucht ist der zugehörige Widerstand  $R$ . Das Ohmsche Gesetz liefert den Zusammenhang

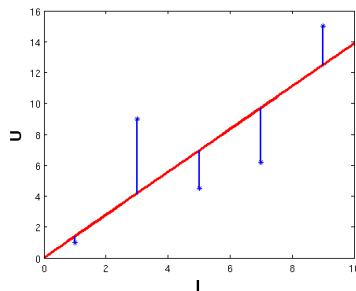
$$U = R \cdot I.$$

Abbildung 4.2: Schaltplan einer einfachen U-I-Messung



Wird davon ausgegangen, dass die  $I_i$  exakt sind, wird das  $R$  gesucht, für das  $RI_i$  im Mittel den minimalen Abstand bzgl. der  $\|\cdot\|_2$ -Norm zu  $U_i$  hat, siehe Abbildung 4.3.

Abbildung 4.3: Lineares Ausgleichsproblem für exakte Stromstärken  $I$  und gemessenen Spannungen  $U_i$

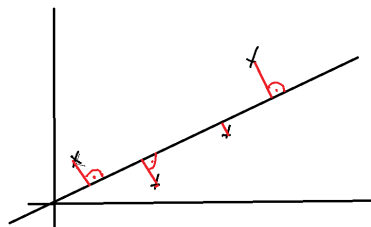


Genauer gesagt berechne

$$\min_{R \in \mathbb{R}} \sum_{i=1}^m (U_i - RI_i)^2.$$

**Vorsicht:** Es wird **nicht** die Gerade mit minimalem euklidischem Abstand zu  $(I_i, U_i)$  gesucht (siehe Abbildung 4.4)! Dieses Problem ist nichtlinear und aufwendig zu lösen.

Abbildung 4.4: Nichtlineares Ausgleichsproblem



21.11.22

**4.3.2. Lineares Ausgleichsproblem.**

Gegeben seien Messdaten  $(t_i, b_i)$  mit  $t_i, b_i \in \mathbb{R}$  für  $i = 1, \dots, m$  und die Abhängigkeit  $b(t)$  werde beschrieben durch eine Modellfunktion, welche linear von den unbekannten Parametern  $x_1, \dots, x_n$  des Modells abhängt, d.h.

$$b(t) = a_1(t)x_1 + \dots + a_n(t)x_n.$$

Für exakte Messdaten  $b_i$  würde

$$b(t_i) = b_i \quad \forall i \in \{1, \dots, m\}$$

gelten. Im Allgemeinen werden jedoch  $m \geq n$  Messwerte  $b_i$  bestimmt und hiermit die  $n$  Parameter  $x_i$  so gewählt, dass die kleinsten *Fehlerquadrate auftreten*:

$$\min_{x_1, \dots, x_n} \sum_{i=1}^m (b_i - b(t_i))^2. \quad (4.3.1)$$

(Nach Gauß kann (4.3.1) auch aus der Maximum-Likelihood-Methode für einen stochastischen Ansatz hergeleitet werden.)

Definiere:

$$\begin{aligned} b &= (b_i)_{i=1, \dots, m} \in \mathbb{R}^m \\ x &= (x_j)_{j=1, \dots, n} \in \mathbb{R}^n \\ A &= (a_j(t_i))_{\substack{i=1, \dots, m \\ j=1, \dots, n}} \in \mathbb{R}^{m \times n} \end{aligned}$$

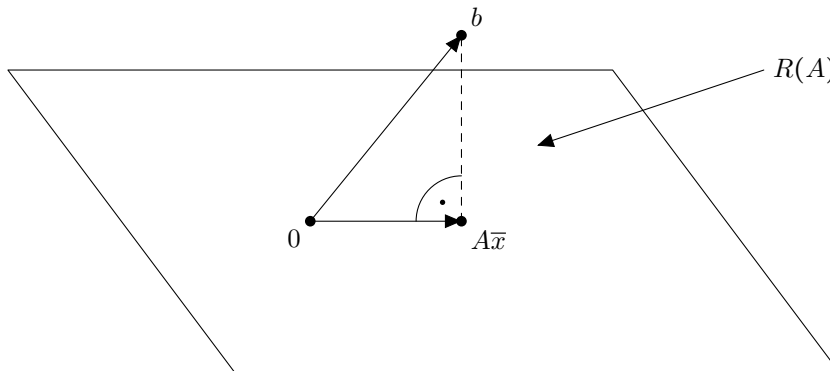
Damit ist (4.3.1) äquivalent zum *linearen Ausgleichsproblem*: Zu gegebenem  $b \in \mathbb{R}^m$  und  $A \in \mathbb{R}^{m \times n}$  mit  $m \geq n$  ist das  $\bar{x} \in \mathbb{R}^n$  gesucht mit

$$\|b - A\bar{x}\|_2 = \min_{x \in \mathbb{R}^n} \|b - Ax\|_2. \quad (4.3.2)$$

Das entspricht der „Lösung“ eines überbestimmten, i.A. nicht erfüllbaren GLS  $Ax = b$ .

Aufgrund der  $l_2$ -Norm ist  $\bar{x}$  gegeben durch die orthogonale Projektion von  $b$  auf den Bildraum  $R(A)$  (siehe Abb. 4.5), wie gleich gezeigt wird.

Abbildung 4.5: Projektion von  $b$  auf  $R(A)$ .



**Satz 4.3.3** (Projektionssatz). Sei  $V$  ein reeller Vektorraum mit einem Skalarprodukt  $\langle \cdot, \cdot \rangle$  und der induzierten Norm  $\|v\| := \sqrt{\langle v, v \rangle}$ . Sei  $U \subseteq V$  ein endlich dimensionaler Untervektorraum und sei

$$U^\perp := \{v \in V \mid \langle v, u \rangle = 0 \quad \forall u \in U\}$$

Dann gilt

- 1) Zu jedem  $v \in V$  existiert genau ein  $\bar{u} \in U$ , so dass  $v - \bar{u} \in U^\perp$ , d.h.  $\langle v - \bar{u}, u \rangle = 0 \quad \forall u \in U$ . Dies definiert die orthogonale Projektion  $P: V \rightarrow U, v \mapsto \bar{u} = Pv$ .
- 2) Zu jedem  $v \in V$  bestimmt  $Pv$  die eindeutige Lösung

$$\|v - Pv\| = \min_{u \in U} \|v - u\|.$$

Also gilt mit einem eindeutigen  $\bar{u} = Pv$ , dass

$$\|v - \bar{u}\| = \min_{u \in U} \|v - u\| \Leftrightarrow \langle v - \bar{u}, u \rangle = 0 \quad \forall u \in U. \quad (4.3.3)$$

*Beweis.*

- 1) Sei  $\{u_1, \dots, u_n\}$  eine Orthonormalbasis von  $U$  und  $\bar{u} \in U$ .

$$\Rightarrow \exists! (\alpha_i)_{i=1, \dots, n} \subset \mathbb{R}: \bar{u} = \sum_{i=1}^n \alpha_i u_i.$$

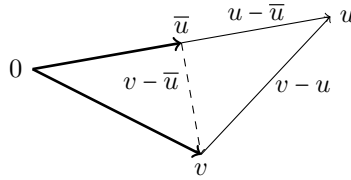
Damit gilt

$$\begin{aligned} 0 &= \langle v - \bar{u}, u \rangle & \forall u \in U \\ \Leftrightarrow 0 &= \langle v - \sum_{i=1}^n \alpha_i u_i, u_j \rangle & \forall j = 1, \dots, n \\ \Leftrightarrow \alpha_j &= \sum_{i=1}^n \alpha_i \langle u_i, u_j \rangle = \langle v, u_j \rangle \end{aligned}$$

$$\Rightarrow Pv = \bar{u} = \sum_{i=1}^n \langle v, u_i \rangle u_i \in U \quad (4.3.4)$$

und  $\bar{u}$  ist die eindeutig bestimmte Lösung für  $v - \bar{u} \in U^\perp$ .

Abbildung 4.6: Geometrische Konstruktion von  $\bar{u}$



- 2) Sei  $u \in U$  beliebig. Dann gilt für  $\bar{u} = Pv$  (siehe Abbildung 4.6)

$$\begin{aligned} \|v - u\|^2 &= \|v - \bar{u} + \bar{u} - u\|^2 \\ &= \|v - \bar{u}\|^2 + 2 \underbrace{\langle v - \bar{u}, \bar{u} - u \rangle}_{=0} + \|u - \bar{u}\|^2 \\ &= \|v - \bar{u}\|^2 + \|u - \bar{u}\|^2 \end{aligned} \quad (4.3.5)$$

(Dies ist anschaulich der Satz des Pythagoras.)

$$\Rightarrow \|v - \bar{u}\| \leq \|v - u\| \quad \forall u \in U.$$

□

**Satz 4.3.4.** Der Vektor  $\bar{x} \in \mathbb{R}^n$  ist genau dann Lösung des linearen Ausgleichsproblems

$$\min_{x \in \mathbb{R}^n} \|b - Ax\|_2,$$

falls er die Normalgleichung

$$A^T A \bar{x} = A^T b \quad (4.3.6)$$

erfüllt. Insbesondere ist  $\bar{x}$  eindeutig, falls  $A \in \mathbb{R}^{m \times n}$  maximalen Rang  $n \leq m$  hat.

*Beweis.* Sei  $V = \mathbb{R}^m$ ,  $U = R(A) = \{Ax \mid x \in \mathbb{R}^n\}$ ,  $b \in \mathbb{R}^m$ . Nach (4.3.3) gilt

$$\begin{aligned} \|b - A\bar{x}\|_2 &= \min_{x \in \mathbb{R}^n} \|b - Ax\|_2 \\ \Leftrightarrow \langle b - A\bar{x}, Ax \rangle &= 0 \quad \forall x \in \mathbb{R}^n \\ \Leftrightarrow \langle A^T(b - A\bar{x}), x \rangle &= 0 \quad \forall x \in \mathbb{R}^n \\ \Leftrightarrow A^T(b - A\bar{x}) &= 0 \\ \Leftrightarrow A^T A \bar{x} &= A^T b \end{aligned}$$

Nach dem Projektionssatz 4.3.3 existiert ein eindeutiges  $\bar{y} = Pb$ . Für dieses  $\bar{y}$  ist  $\bar{x} \in \mathbb{R}^n$  mit  $\bar{y} = A\bar{x}$  eindeutig bestimmt, falls  $A$  injektiv ist, d.h. falls  $\text{rang}(A) = n$ .  $\square$

Ähnlich zum Skalarprodukt ist die relative Kondition von  $(P, b)$  schlecht, falls  $b$  fast senkrecht zu  $U$  steht. Die relative Kondition des linearen Ausgleichsproblems hängt zusätzlich von  $\text{cond}(A)$  ab.

#### 4.3.5. Lösung der Normalgleichung

Falls  $\text{rang}(A) = n$ , ist  $A^T A$  spd und das Cholesky-Verfahren ist anwendbar. Dafür ist

1.  $A^T A$  zu berechnen:

**der Aufwand** beträgt ca.  $\frac{1}{2}n^2m$  Multiplikationen,

**die Kondition** der Berechnung ist häufig schlecht, da  $\frac{1}{2}n^2$  Skalarprodukte berechnet werden.

2. die Cholesky-Zerlegung von  $A^T A$  durchzuführen:

**der Aufwand** beträgt ca.  $\frac{1}{6}n^3$  Multiplikationen,

**die Kondition** zur Lösung von  $A^T Ax = z$  ist i.A. sehr groß.

Für  $A \in \mathbb{R}^{m \times n}$  mit  $m \geq n$  und  $\text{rang}(A) = n$  gilt (siehe Übungsaufgabe)

$$\text{cond}_2(A^T A) = \text{cond}_2(A)^2 \quad (4.3.7)$$

Also überwiegt für  $m \gg n$  der Aufwand  $A^T A$  zu berechnen. Die auftretenden Konditionen entsprechen i.d.R. nicht dem des Ausgangsproblems. Damit ist die **Cholesky-Zerlegung angewendet auf die Normalgleichungen ungeeignet**.

**Satz 4.3.6.** Sei  $A \in \mathbb{R}^{m \times n}$  mit  $m \geq n$  und  $\text{rang}(A) = n$ , sei  $b \in \mathbb{R}^m$  und besitze  $A$  eine Zerlegung

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$$

mit einer orthogonalen Matrix  $Q \in \mathbb{R}^{m \times m}$  und einer oberen Dreiecksmatrix  $R \in \mathbb{R}^{n \times n}$ . Dann ist  $R$  invertierbar. Bezeichne

$$\begin{pmatrix} \bar{b}_1 \\ \bar{b}_2 \end{pmatrix} := Q^T b \quad (4.3.8)$$

dann ist

$$\bar{x} = R^{-1} \bar{b}_1 \quad (4.3.9)$$

die Lösung des linearen Ausgleichsproblems und

$$\|\bar{b}_2\| = \|b - A\bar{x}\| = \min_{x \in \mathbb{R}^n} \|b - Ax\|.$$

Zur Erinnerung:

$Q$  orthogonal  $\Leftrightarrow QQ^T = I \Leftrightarrow Q^{-1} = Q^T$

Weiterhin ist  $Q$  längenerhaltend, d.h.  $\|Qv\|_2 = \|v\|_2$ ,

somit folgt

$$\|Q\|_2 = \|Q^{-1}\|_2 = 1, \text{cond}_2(Q) = 1. \quad (4.3.10)$$

*Beweis.*  $R$  ist invertierbar, da  $\text{rang}(R) = \text{rang}(Q^{-1}A) = \text{rang}(A) = n$ . Außerdem gilt

$$\begin{aligned} \|b - Ax\|_2^2 &= \left\| Q \left( Q^T b - \begin{pmatrix} R \\ 0 \end{pmatrix} x \right) \right\|_2^2 \\ Q \text{ längenerhaltend} &= \left\| Q^T b - \begin{pmatrix} Rx \\ 0 \end{pmatrix} \right\|_2^2 = \|\bar{b}_1 - Rx\|_2^2 + \|\bar{b}_2\|^2, \end{aligned}$$

was minimal für  $R\bar{x} = \bar{b}_1$  ist. □

Da  $Q$  längenerhaltend ist, folgt mit 3.2.13b) ( $\text{cond}(A) = \frac{\max\|Ax\|}{\min\|Ax\|}$ ) sofort

$$\text{cond}_2(R) = \text{cond}_2(A).$$

Die auftretende Kondition entspricht also der des Ausgleichsproblems.

*Bemerkung 4.3.7.* Sei  $A \in \mathbb{R}^{n \times n}$  invertierbar und habe eine QR-Zerlegung, d.h. es existiert eine orthogonale Matrix  $Q$  und eine obere Dreiecksmatrix  $R$ , so dass:

$$A = QR$$

Dann kann das Gleichungssystem  $Ax = b$  wie folgt gelöst werden:

1. Setze  $z = Q^T b$ , was Kondition 1 hat.
2. Löse durch Rückwärtssubstitution  $Rx = z$ .

23.11.22

## 4.4 Orthogonalisierungsverfahren

Konstruiere eine QR-Zerlegung

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix} \quad (4.4.1)$$

durch einen Eliminationsprozess:

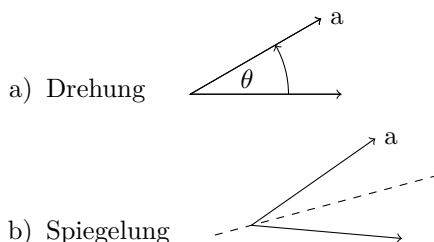
$$A \longrightarrow Q^{(1)}A \longrightarrow Q^{(2)}Q^{(1)}A \longrightarrow Q^{(p)}\dots Q^{(1)}A = \begin{pmatrix} R \\ 0 \end{pmatrix} \quad (4.4.2)$$

mit orthogonalen Matrizen  $Q^{(i)}$ . Dann gilt

$$Q = Q^{(1)T} \dots Q^{(p)T} \quad (4.4.3)$$

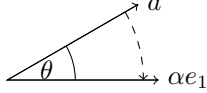
Dies ist im Gegensatz zur LR-Zerlegung aufgrund von  $\text{cond}(Q^{(i)}) = 1$  immer stabil.

Für  $Q \in \mathbb{R}^{2 \times 2}$  gibt es zwei mögliche Typen von orthogonalen Abbildungen, nämlich:



#### 4.4 a) Givens-Rotation

Es wird eine Drehung auf den ersten Einheitsvektor durchgeführt:

$$a = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \longrightarrow \begin{pmatrix} \alpha \\ 0 \end{pmatrix} = \alpha e_1$$


d.h.  $a_2$  wird eliminiert und es gelte  $\|\alpha e_1\|_2 = \|a\|_2$ .

Drehungen werden beschrieben durch

$$Q = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} =: \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \quad \theta \in [0, 2\pi).$$

Weiterhin muss gelten  $Qa = \begin{pmatrix} \alpha \\ 0 \end{pmatrix}$  mit  $\alpha = \pm\|a\|_2$ . Hiermit folgt

- für  $\|a\| = 0$  ist  $c = 1, s = 0$ ,
- für  $\|a\| \neq 0$  ist

$$c = \frac{a_1}{\alpha}, s = \frac{a_2}{\alpha} \quad \text{mit} \quad \alpha = \pm\sqrt{a_1^2 + a_2^2}. \quad (4.4.4)$$

Im Folgenden wird diese Zuweisung kurz mit

$$[c, s] = \text{givens}(a_1, a_2) \quad (4.4.5)$$

bezeichnet.

Als *Givens-Rotation* wird eine Matrix der Form

$$\Omega_{k,l} = \begin{pmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & & \mathbf{c} & & \mathbf{s} \\ & & & & 1 & \\ & & & & & \ddots \\ & & & -\mathbf{s} & & \mathbf{c} & \\ & & & & & & \ddots \\ & & & & & & & 1 \end{pmatrix} \begin{matrix} \leftarrow k\text{-te Zeile} \\ \\ \\ \leftarrow l\text{-te Zeile} \end{matrix} \quad (4.4.6)$$

mit  $c^2 + s^2 = 1$  und  $k < l$  bezeichnet.

Es folgt

$$\begin{aligned} \tilde{A} &= \Omega_{kl} A \quad \text{mit} \\ \tilde{a}_{ij} &= a_{ij} \quad \text{für } i \neq k, l, \\ \tilde{a}_{kj} &= ca_{kj} + sa_{lj}, \\ \tilde{a}_{lj} &= -sa_{kj} + ca_{lj}. \end{aligned} \quad (4.4.7)$$

Demnach werden nur die  $k$ -te und  $l$ -te Zeile verändert.

Falls nun  $[c, s] = \text{givens}(x_k, x_l)$  gilt

$$\Omega_{k,l}x = \begin{pmatrix} x_1 \\ \vdots \\ x_{k-1} \\ \alpha \\ x_{k+1} \\ \vdots \\ x_{l-1} \\ 0 \\ x_{l+1} \\ \vdots \\ x_n \end{pmatrix} \quad \text{mit } \alpha = \pm \left\| \begin{pmatrix} x_k \\ x_l \end{pmatrix} \right\|_2 \quad (4.4.8)$$

d.h. eine Givens-Rotation erzeugt eine Null.

Da nun

$$\begin{pmatrix} R \\ 0 \end{pmatrix} = \begin{pmatrix} * & * & * & \dots \\ 0 & * & & \\ 0 & 0 & \ddots & \\ \vdots & & & \\ 0 & 0 & 0 & \dots & * \\ 0 & \dots & \dots & \dots & 0 \\ \vdots & & & & \vdots \\ 0 & \dots & \dots & \dots & 0 \end{pmatrix} \in \mathbb{R}^{m \times n}$$

erzeugt werden soll, sind

$$p = \sum_{j=1}^n (m - j) \text{ Givens-Rotationen nötig,}$$

um eine QR-Zerlegung nach (4.4.1) zu erzeugen. Und eine Rotation, welche  $a_{ij}$  auf 0 setzt, ist durch zugehörige  $(c_{ij}, s_{ij})$  gegeben.

Eine Matrix  $A$  kann spaltenweise von unten nach oben, wie im folgenden Beispiel für eine 4x3-Matrix vorgeführt, in  $\begin{pmatrix} R \\ 0 \end{pmatrix}$  überführt werden:

$$\begin{aligned} A = \begin{pmatrix} * & & \\ * & * & \\ * \curvearrowleft & & \\ * \curvearrowleft & & \end{pmatrix} & \xrightarrow{\Omega_{3,4}^{(1)}} \begin{pmatrix} * & & \\ * \curvearrowleft & * & \\ * \curvearrowleft & & \\ 0 & & \end{pmatrix} \xrightarrow{\Omega_{2,3}^{(1)}} \begin{pmatrix} * \curvearrowleft & & \\ * \curvearrowleft & * & \\ 0 & & \\ 0 & & \end{pmatrix} \xrightarrow{\Omega_{1,2}^{(1)}} \begin{pmatrix} * & & \\ 0 & * & \\ 0 & & \\ 0 & & \end{pmatrix} \\ & \xrightarrow{\Omega_{3,4}^{(2)}} \begin{pmatrix} * & * & \\ 0 & * \curvearrowleft & * \\ 0 & * \curvearrowleft & \\ 0 & 0 & \end{pmatrix} \xrightarrow{\Omega_{2,3}^{(2)}} \begin{pmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \curvearrowleft \\ 0 & 0 & * \curvearrowleft \end{pmatrix} \\ & \xrightarrow{\Omega_{3,4}^{(3)}} \begin{pmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} R \\ 0 \end{pmatrix} \end{aligned}$$

Es ergibt sich:

---

**Algorithmus 4.4.1 : Givens-QR-Algorithmus**


---

```

for  $j = 1, \dots, n$  do
  for  $i = m, m-1, \dots, j+1$  do
    % setze  $a_{ij}$  auf 0
     $[c, s] = \text{givens}(a_{i-1,j}, a_{ij})$ 
    speichere  $c$  und  $s$  für  $a_{ij}$ 
     $A(i-1:i, j:n) = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} A(i-1:i, j:n)$ 
  end for
end for

```

---

*Bemerkung 4.4.2.*

- a)  $A(i-1:i, 1:j-1) = 0$  und ist daher nicht zu berechnen oder zu speichern. Der Speicherplatz kann für die Speicherung der Givensrotationen benutzt werden.
- b)  $R$  steht anschließend in  $A$ .
- c) Die Bestimmung der Länge  $|\alpha|$  wird so ausgeführt, dass over- oder underflow vermieden wird. Weiterhin wird das Vorzeichen von  $c$  oder  $s$  festgelegt, so dass aufgrund von  $c^2 + s^2 = 1$  nur ein Wert  $\rho$  gespeichert werden muss. Hiermit wird auch das Vorzeichen von  $\alpha$  festgelegt:

Falls  $a_2 = 0$ : Setze

$$c = 1, s = 0, \alpha = a_1, \\ \text{merke } \rho = 1.$$

Falls  $|a_2| > |a_1|$ : Setze

$$\tau = \frac{a_1}{a_2}, s = \frac{1}{\sqrt{1+\tau^2}}, c = s\tau, \alpha = a_2\sqrt{1+\tau^2}, \\ \text{merke } \rho = \frac{c}{2}.$$

Falls  $|a_2| \leq |a_1|$ : Setze

$$\tau = \frac{a_2}{a_1}, c = \frac{1}{\sqrt{1+\tau^2}}, s = c\tau, \alpha = a_1\sqrt{1+\tau^2}, \\ \text{merke } \rho = \frac{2}{s}.$$

- d) Aufgrund von c) muss nur  $\rho$  gespeichert werden:

Für  $\rho = 1$ : Setze  $c = 1, s = 0$ .

Für  $|\rho| < 1$ : Setze  $c = 2\rho, s = \sqrt{1-c^2}$ .

Für  $|\rho| > 1$ : Setze  $s = \frac{2}{\rho}, c = \sqrt{1-s^2}$ .

Hiermit können alle notwendigen Givens-Rotationen als untere Dreiecksmatrix zusammen mit  $R$  in  $A$  gespeichert werden.

#### 4.4.3. Aufwand des Givens-QR-Algorithmus

- a) Für  $m \approx n$  sind ungefähr

$$\frac{4}{3}n^3 \text{ Multiplikationen und } \frac{1}{2}n^2 \text{ Quadratwurzeln nötig.}$$

Die Givens-QR-Zerlegung ist somit ungefähr viermal so aufwändig wie die Gauß-Elimination, dafür jedoch stabil.



b) Für  $m \gg n$  sind ungefähr

$2n^2m$  Multiplikationen und  $mn$  Quadratwurzeln nötig.

Das Verfahren ist daher zwei- bis viermal so aufwändig wie das Cholesky-Verfahren für die Normalgleichungen, aber stabil.

c) Bei Hessenberg-Matrizen, d.h. Matrizen mit der Gestalt

$$A = \begin{pmatrix} * & & & \\ * & * & & \\ 0 & \ddots & \ddots & \\ & & * & * \end{pmatrix}, \quad (4.4.9)$$

also  $a_{ik} = 0$  für alle  $i > k + 1$ , sind nur

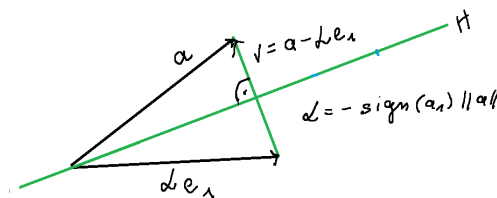
$(n - 1)$  Givens-Rotationen auszuführen.

Diese Matrizen tauchen z.B. bei Eigenwertberechnungen auf und sind dort ein wichtiger Bestandteil der Verfahren.

#### 4.4 b) Householder-Reflexion

Zu einem Vektor  $a \in \mathbb{R}^k$  ist die Reflexion  $Q$  gesucht, die  $a$  auf  $\alpha e_1$  spiegelt, d.h.

$$Qa = \alpha e_1 \quad \text{mit } \alpha = \pm \|a\|.$$



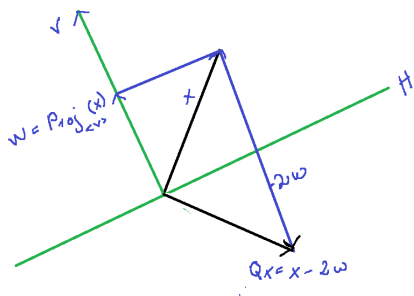
Die Hyperebene  $H$ , an der  $a$  gespiegelt wird, ist durch die Senkrechte

$$v = a - \alpha e_1 \quad (4.4.10)$$

gegeben. Damit Stellenauslöschungen in  $v$ , d.h. in  $v_1$ , vermieden werden, wird das Vorzeichen von  $\alpha$  geeignet gewählt, also

$$\alpha = -\text{sign}(a_1) \|a\|_2. \quad (4.4.11)$$

Die zugehörige Reflexion  $Q$  ist dann gegeben durch:



$$w := \text{Proj}_v(x) = \left\langle \frac{v}{\|v\|}, x \right\rangle \frac{v}{\|v\|} \quad \text{und}$$

$$Qx = x - 2w = x - 2 \frac{v^T x}{v^T v} v \quad (4.4.12)$$

$$= \left( I - 2 \frac{vv^T}{v^T v} \right) x.$$

$$\Rightarrow Q = I - 2 \frac{vv^T}{v^T v} \quad (4.4.13)$$

(beachte  $vv^T \in \mathbb{R}^{k \times k}$ ,  $v^T v \in \mathbb{R}$ ).

$Q$  heißt *Householder Reflexion* und wurde 1958 von Householder eingeführt. Für die spezielle Wahl (4.4.10) mit (4.4.11) vom Vektor  $v$  folgt

$$\begin{aligned} v^T v = \|v\|^2 &= \|a\|^2 - 2\alpha\langle a, e_1 \rangle + \alpha^2 \\ &= -2\alpha(a_1 - \alpha) = -2\alpha v_1, \end{aligned} \quad (4.4.14)$$

womit statt  $k$  Multiplikationen für  $v \in \mathbb{R}^k$  nur 2 nötig sind.

28.11.22

*Bemerkung 4.4.4.*

- a)  $Q$  ist symmetrisch,
- b)  $Q$  ist orthogonal,
- c)  $Q$  ist involutorisch, d.h.  $Q^2 = I$ ,  
bzw. es gilt  $Q^{-1} = Q^T = Q$ .

Die Householder Reflexion setzt nicht nur eine Null, sondern im Vektor gleich alle gewünschten Nullen. Um die erste Spalte in  $A$  auf die gewünschte Gestalt zu bringen, bestimme  $Q^{(1)}$  wie oben, indem die erste Spalte von  $A$  als  $a$  gewählt wird:

$$A \rightarrow A^{(1)} = Q^{(1)} A = \begin{pmatrix} \alpha^{(1)} & & \\ 0 & * & \\ \vdots & & \\ 0 & & \end{pmatrix}$$

In der  $k$ -ten Spalte sollen die  $(k-1)$ -ten Zeilen und die  $(k-1)$ -ten Spalten erhalten bleiben und die Restmatrix verändert werden.  $A^{(k-1)}$  hat die Gestalt

$$A^{(k-1)} = \begin{pmatrix} * & & & & \\ & \ddots & & & \\ & & * & & \\ & & 0 & - & - \\ & 0 & \vdots & T^{(k-1)} & - \\ & & \vdots & & \\ & & 0 & & \end{pmatrix} \begin{array}{l} \leftarrow (k-1)\text{-te Zeile} \\ \\ \\ \uparrow (k-1)\text{-te Spalte} \end{array}.$$

Setze also

$$Q^{(k)} = \begin{pmatrix} I_{k-1} & 0 \\ 0 & \bar{Q}^{(k)} \end{pmatrix}, \quad (4.4.15)$$

wobei  $\bar{Q}^{(k)}$  durch die erste Spalte von  $T^{(k)}$ , d.h. mittels

$$a = (a_{i,k}^{(k-1)})_{i=k,\dots,m} \in \mathbb{R}^{m+1-k} \quad (4.4.16)$$

bestimmt wird. Dann gilt

$$Q^{(k)} A^{(k-1)} = \begin{pmatrix} * & & & & \\ & \ddots & & & \\ & & * & & \\ & & 0 & - & - \\ & 0 & \vdots & 0 & \dots \\ & & \vdots & \vdots & * \\ & & 0 & 0 & * \end{pmatrix}.$$

Nach insgesamt

$$p = \min(m-1, n) \quad (4.4.17)$$

Schritten erhalten wir für  $A \in \mathbb{R}^{m \times n}$

$$Q^T A = Q^{(p)} \dots Q^{(1)} A = \begin{pmatrix} R \\ 0 \end{pmatrix}. \quad (4.4.18)$$

Bemerkung 4.4.4 liefert somit

$$Q = Q^{(1)} \dots Q^{(p)}. \quad (4.4.19)$$

#### 4.4.5. Speicherung

Gespeichert werden müssen die obere Dreiecksmatrix  $R$  und die *Householdervektoren*  $v^{(i)} \in \mathbb{R}^{m+1-i}$ . Die Diagonalelemente von  $R$  sind  $r_{ii} = \alpha^{(i)}$ .

Folgende Speicheraufteilung ist möglich:

$R$  ohne die Diagonale wird in  $A$  gespeichert und die Diagonale in einem extra Vektor. Die Vektoren  $v^{(i)}$  werden als untere Dreiecksmatrix in  $A$  gespeichert.

$$\left( \begin{array}{c|c|c|c|c} & & & & R \\ v^{(1)} & & & & \\ \hline & v^{(2)} & & & \\ & & \ddots & & \\ & & & v^{(p)} & \end{array} \right) \text{ und } \begin{pmatrix} \alpha^{(1)} \\ \vdots \\ \alpha^{(p)} \end{pmatrix}.$$

Falls zusätzlich  $v^{(i)}$  so normiert ist, dass  $v_1^{(i)} = 1$  ist, braucht diese Komponente nicht gespeichert werden und sowohl  $R$  als auch alle  $v^{(i)}$  ohne  $v_1^{(i)}$  können komplett in  $A$  gespeichert werden.

Für Matrizen  $A \in \mathbb{R}^{m \times n}$  mit vollem Rang ergibt sich folgender Algorithmus.

---

#### Algorithmus 4.4.6 : Householder QR-Algorithmus

---

```

for  $j = 1, \dots, \min(m-1, n)$  do
  % berechne  $\alpha$  für  $a = A(j : m, j)$ 
   $\alpha(j) = -\text{sign}(A(j, j)) \sqrt{A(j : m, j)^T A(j : m, j)}$  siehe (4.4.11)
  % berechne  $A(j : m, j) = v = a - \alpha e_1$ 
   $A(j, j) = A(j, j) - \alpha(j)$  siehe (4.4.10)
  % berechne  $-\langle v, v \rangle = \alpha(a_1 - \alpha) = \alpha v_1$ 
   $\beta = \alpha(j) A(j, j)$  siehe (4.4.14)
  % berechne  $\overline{Q}^{(j)} T^{(j)}$  aber nicht mehr die erste Spalte, welche  $\alpha(j) e_1$  ist
  for  $l = j+1 : n$  do
    % setze  $v_x = -2(v^T x) / (v^T v)$ 
     $v_x = A(j : m, j)^T A(j : m, l) / \beta$  siehe (4.4.12)
    % berechne  $\overline{Q}^{(j)} x = x + v_x v$ 
     $A(j : m, l) = A(j : m, l) + v_x A(j : m, j)$  siehe (4.4.12)
  end for
end for

```

---

Zur Lösung eines Gleichungssystems oder eines linearen Ausgleichsproblems muss noch  $Q^T b = Q^{(p)} \dots Q^{(1)} b$  berechnet werden.

---

**Algorithmus 4.4.7 :** Berechnung von  $Q^T b = Q^{(p)} \dots Q^{(1)} b$ 


---

```

for  $j = 1, \dots, \min\{m-1, n\}$  do
  % beachte (4.4.15), also  $b(1:j-1)$  bleibt gleich
  % setze  $v_x = -2v^T b / (v^T v)$ 
   $v_x = (A(j:m, j)^T b(j:m)) / (\alpha(j) A(j, j))$            siehe (4.4.14)
  % berechne  $\overline{Q}^{(j)} b = b + v_x v$ 
   $b(j:m) = b(j:m) + v_x A(j:m, j)$            siehe (4.4.12)
end for

```

---

Anschließend ist durch die Rückwärtssubstitution noch  $Rx = \bar{b}_1$  zu lösen.

#### 4.4.8. Aufwand für den Householder-QR-Algorithmus

a) Falls  $m \approx n$  ist, sind

ungefähr  $\frac{2}{3}n^3$  Multiplikationen notwendig

und somit ist die Householder-Reflexion ungefähr doppelt so teuer wie die LR-Zerlegung, ist aber stabil.

b) Falls  $m \gg n$  ist, sind

ungefähr  $2mn^2$  Multiplikationen notwendig.

Der Aufwand ist daher ungefähr so hoch wie bei der Anwendung des Cholesky-Verfahren für die Normalgleichung, aber die Householder-Reflexion ist stabil.

Wohlgermerkt können mit Householderreflexionen, Givens-Rotationen und auch mit der LR-Zerlegung ebenso Matrizen  $A \in \mathbb{R}^{m \times n}$  mit  $m < n$  bearbeitet werden. Dann wird  $A$  zu

$$\begin{pmatrix} \ddots & & & \\ & \ddots & R & \\ 0 & & \ddots & \end{pmatrix}.$$

# Kapitel 5

## Numerische Lösung nichtlinearer Gleichungssysteme

### 5.1 Einführung

*Beispiel 5.1.1.*

- 1) Berechnung der Wurzel:  $f(x) = x^2 - c = 0 \Leftrightarrow x = \pm\sqrt{c}$ .
- 2) Nullstellenbestimmung eines Polynoms  $p$ .
- 3) Lösung des nichtlinearen Randwertproblems

$$-\Delta u = f(u) \text{ in } \Omega = (0, 1) \times (0, 1) \text{ mit } u = 0 \text{ auf } \partial\Omega.$$

Mit dem Differenzenverfahren (siehe Übungsaufgabe) ergibt sich ein endlich dimensionales System nichtlinearer Gleichungen:

$$A\vec{u} = h^2 \vec{f}(\vec{u}).$$

*Nullstellenbestimmung:*

**Gegeben:**  $D \subseteq \mathbb{R}^n, f: D \rightarrow \mathbb{R}^m$  stetig,

**gesucht:**  $x^* \in D$  mit  $f(x^*) = 0$ .

*Fixpunktgleichung:*

**Gegeben:**  $D \subseteq \mathbb{R}^n, g: D \rightarrow \mathbb{R}^n$  stetig,

**gesucht:**  $x^* \in D$  mit  $g(x^*) = x^*$ .

Falls  $m=n$  sind diese zwei Aufgaben äquivalent.

#### 5.1.2. Das Bisektionsverfahren.

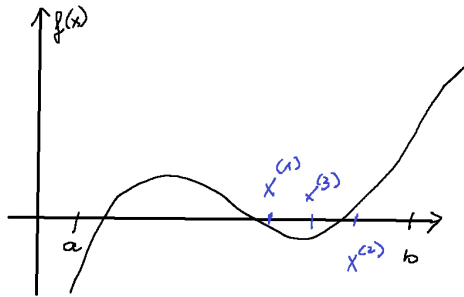
Sei  $f: [a, b] \rightarrow \mathbb{R}$  stetig und  $f(a)f(b) < 0$ . Dann folgt aus dem Zwischenwertsatz die Existenz mindestens einer Nullstelle  $x^* \in (a, b)$ .

Generiere eine Folge von Intervallen  $[a^{(i)}, b^{(i)}] \subseteq [a^{(i-1)}, b^{(i-1)}]$  mit  $b^{(i)} - a^{(i)} \rightarrow 0$ , die eine Nullstelle enthalten. Hierfür setze  $a^{(0)} = a, b^{(0)} = b$  und für  $i \geq 0$ :

$$x^{(i+1)} = \frac{1}{2}(b^{(i)} + a^{(i)}), \tag{5.1.1}$$

$$[a^{(i+1)}, b^{(i+1)}] := \begin{cases} [x^{(i+1)}, b^{(i)}] & \text{für } f(a^{(i)})f(x^{(i+1)}) > 0 \\ [a^{(i)}, x^{(i+1)}] & \text{für } f(a^{(i)})f(x^{(i+1)}) < 0 \end{cases} \quad (5.1.2)$$

und  $x^* = x^{(i+1)}$  für  $f(x^{(i+1)}) = 0$ .



Für jedes  $i \geq 1$  gilt somit

$$b^{(i)} - a^{(i)} = \frac{1}{2^i}(b - a)$$

und es existiert eine Nullstelle  $x^*$  in  $[a^{(i)}, b^{(i)}]$ . Damit folgt

$$|x^{(i+1)} - x^*| \leq \frac{1}{2}(b^{(i)} - a^{(i)}) = 2^{-(i+1)}(b - a) \rightarrow 0,$$

also  $\lim_{i \rightarrow \infty} x^{(i)} = x^*$ .

30.11.22

**Korollar 5.1.3.** Das oben angegebene Bisektionsverfahren konvergiert, falls  $f: [a, b] \rightarrow \mathbb{R}$  stetig ist und  $f(a)f(b) < 0$  gilt.

*Bemerkung 5.1.4.*

- a)  $x^{(i)}$  wird als Intervallmitte, also unabhängig von  $f(x^{(i)})$  gewählt. Die Konvergenzgeschwindigkeit hängt nur von der Länge des Intervalls  $[a, b]$  und der Lage von  $x^*$  bezüglich der Intervallhalbierung ab. Diese kann demnach sehr langsam sein.
- b) Ein Vorteil ist, dass keine Differenzierbarkeitsvoraussetzungen nötig sind.
- c) Das Verfahren ist nicht für  $f: D \rightarrow \mathbb{R}^n$  mit  $n \geq 2$  anwendbar.

## 5.2 Fixpunktiteration

Gesucht sei ein Fixpunkt  $x^* \in D \subseteq \mathbb{R}^n$  der stetigen Funktion  $g: D \rightarrow \mathbb{R}^n$ , d.h.

$$x^* = g(x^*) \quad (5.2.1)$$

*Idee:* Nutze (5.2.1) zur Iteration, d.h. wähle ein  $x^{(0)} \in D$  und setze

$$x^{(k+1)} = g(x^{(k)}) \quad \text{für } k = 0, 1, \dots \quad (5.2.2)$$

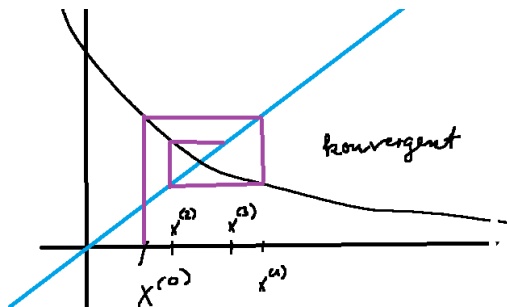
Dies bedarf der Voraussetzung  $x^{(k)} \in D \forall k$ .

Falls  $x^{(k)}$  konvergiert, ist der Grenzwert  $x^*$  ein Fixpunkt, denn für stetiges  $g$  gilt:

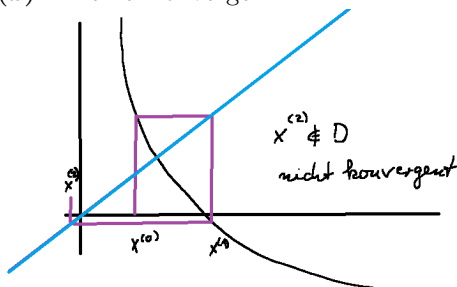
$$x^* = \lim_{k \rightarrow \infty} x^{(k+1)} = \lim_{k \rightarrow \infty} g(x^{(k)}) \stackrel{g \text{ stetig}}{=} g\left(\lim_{k \rightarrow \infty} x^{(k)}\right) = g(x^*). \quad (5.2.3)$$

Beispiel 5.2.1. Löse  $x - e^{-x} - 1 = 0$ .

a)  $x = 1 + e^{-x} =: g_1(x) \leadsto$  Konvergenz



b)  $e^{-x} = x - 1 \Leftrightarrow x = -\ln(x - 1) =: g_2(x) \leadsto$  keine Konvergenz

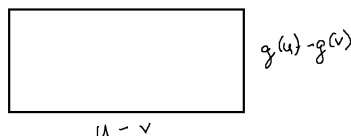


**Definition 5.2.2.** Sei  $D \subseteq \mathbb{R}^n$  abgeschlossen und  $\|\cdot\|$  eine Norm auf dem  $\mathbb{R}^n$ . Eine Abbildung  $g: D \rightarrow \mathbb{R}^n$  heißt *Kontraktion* bezüglich  $\|\cdot\|$ , falls es ein  $\kappa \in [0, 1)$  gibt mit

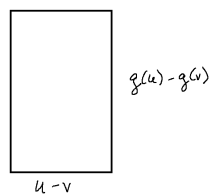
$$\|g(u) - g(v)\| \leq \kappa \|u - v\| \quad \forall u, v \in D.$$

Die kleinste solche Zahl  $\kappa$  heißt *Kontraktionszahl* von  $g$ .

Offensichtlich ist jede auf  $D$  kontrahierende Abbildung stetig.



falls die Box so liegt, sieht es nach einer Kontraktion aus.



falls die Box so liegt, sieht es nicht nach einer Kontraktion aus.

**Lemma 5.2.3.** Sei  $D = \overline{\Omega}$  mit  $\Omega \subseteq \mathbb{R}^n$  offen und konvex und  $\|\cdot\|$  eine Norm auf dem  $\mathbb{R}^n$ . Falls  $g: D \rightarrow D$  eine stetig differenzierbare Funktion ist und gelte bezüglich der zugeordneten Matrixnorm

$$\sup_{x \in \Omega} \|Dg(x)\| < 1,$$

so ist  $g$  kontrahierend bezüglich  $\|\cdot\|$ .

*Beweis.* Mit  $u, v \in D$  gilt  $u + t(v - u) \in D$ , da  $D$  konvex ist. Somit ist

$$h: [0, 1] \rightarrow \mathbb{R}^n \quad \text{mit} \quad h(t) := g(u + t(v - u))$$

wohldefiniert und stetig differenzierbar. Mit dem Hauptsatz der Differenzial- und Integralrechnung folgt:

$$\begin{aligned} \|g(u) - g(v)\| &= \|h(1) - h(0)\| = \left\| \int_0^1 h'(t) dt \right\| = \left\| \int_0^1 Dg(u + t(v - u))(v - u) dt \right\| \\ &\leq \int_0^1 \|Dg(u + t(v - u))\| dt \|v - u\| \leq \underbrace{\sup_{x \in \Omega} \|Dg(x)\|}_{=: \kappa < 1} \|v - u\| \end{aligned} \quad (5.2.4)$$

□

**Satz 5.2.4** (Banachscher Fixpunktsatz).

Sei  $D \subseteq \mathbb{R}^n$  abgeschlossen und die Abbildung  $g: D \rightarrow \mathbb{R}^n$  eine Kontraktion. Weiterhin gelte  $g(D) \subseteq D$ . Dann gilt:

1) Es existiert genau ein Fixpunkt  $x^*$  von  $g$ .

2) Für jeden Startwert  $x^{(0)} \in D$  konvergiert die Folge der Fixpunktiterierten

$$x^{(k+1)} := g(x^{(k)}) \quad (5.2.5)$$

gegen  $x^*$ .

3) Es gelte die a posteriori Fehlerabschätzung

$$\|x^{(k)} - x^*\| \leq \frac{\kappa}{1 - \kappa} \|x^{(k)} - x^{(k-1)}\|. \quad (5.2.6)$$

und die a priori Fehlerabschätzung

$$\|x^{(k)} - x^*\| \leq \frac{\kappa^k}{1 - \kappa} \|x^{(1)} - x^{(0)}\|. \quad (5.2.7)$$

*Beweis.*

Zu 2): Sei  $x^{(0)} \in D$  beliebig. (5.2.5) ist wohldefiniert, da  $g(D) \subseteq D$ .

$(x^{(k)})_{k \in \mathbb{N}}$  bildet eine Cauchyfolge, da

$$\begin{aligned} \|x^{(k+1)} - x^{(k)}\| &= \|g(x^{(k)}) - g(x^{(k-1)})\| \leq \kappa \|x^{(k)} - x^{(k-1)}\| \\ &\leq \kappa^k \|x^{(1)} - x^{(0)}\| \\ \Rightarrow \|x^{(k+l)} - x^{(k)}\| &\leq \sum_{i=0}^{l-1} \|x^{(k+i+1)} - x^{(k+i)}\| \leq \sum_{i=0}^{l-1} \kappa^{k+i} \|x^{(1)} - x^{(0)}\| \\ &= \|x^{(1)} - x^{(0)}\| \kappa^k \sum_{i=0}^{l-1} \kappa^i \leq \frac{\kappa^k}{1 - \kappa} \|x^{(1)} - x^{(0)}\| \quad \forall k, l \in \mathbb{N} \end{aligned} \quad (5.2.8)$$

Daraus folgt, dass  $\lim_{k \rightarrow \infty} x^{(k)} = x^*$  existiert. Da  $D$  abgeschlossen ist, ist  $x^* \in D$ .

Zu 1): Da  $g$  stetig ist, ist  $g(x^*) = x^*$  (siehe hierzu (5.2.3)).  $x^*$  ist eindeutiger Fixpunkt, da für einen weiteren Fixpunkt  $y^*$  gilt

$$0 \leq \|x^* - y^*\| = \|g(x^*) - g(y^*)\| \leq \kappa \|x^* - y^*\|.$$

Da  $\kappa < 1$ , muss  $\|x^* - y^*\| = 0$  sein und damit  $x^* = y^*$ .

Zu 3): Es gilt

$$\begin{aligned} \|x^{(k)} - x^*\| &= \lim_{l \rightarrow \infty} \|x^{(k+l)} - x^{(k)}\| \leq \frac{\kappa^k}{1 - \kappa} \|x^{(1)} - x^{(0)}\|, \\ \text{bzw. } \|x^{(k)} - x^*\| &= \lim_{l \rightarrow \infty} \|x^{(k+l)} - x^{(k)}\| \leq \lim_{l \rightarrow \infty} \sum_{i=0}^{l-1} \|x^{(k+i+1)} - x^{(k+i)}\| \\ &\leq \lim_{l \rightarrow \infty} \sum_{i=0}^{l-1} \kappa^{k+i+1} \|x^{(k)} - x^{(k-1)}\| \leq \frac{\kappa}{1 - \kappa} \|x^{(k)} - x^{(k-1)}\|. \end{aligned}$$

□



Bemerkung 5.2.5.

- 1) Als Voraussetzung in Satz 5.2.4 wäre bereits ausreichend, dass  $D$  ein vollständiger metrischer Raum mit einer Metrik  $d$  ist. Dann wird in Satz 5.2.4 die Norm durch die Metrik  $d$  ersetzt.
- 2) Im Allgemeinen ist der Nachweis  $g(D) \subseteq D$  schwierig!

**Folgerung 5.2.6.** Sei  $x^* \in \mathbb{R}^n$ , so dass  $g(x^*) = x^*$  gilt, und sei  $g$  in einer abgeschlossenen Umgebung von  $\overline{B_\varepsilon(x^*)} := \{x \in \mathbb{R}^n \mid \|x - x^*\| \leq \varepsilon\}$  stetig differenzierbar mit

$$\|g'(x)\| < 1 \quad \forall x \in \overline{B_\varepsilon(x^*)},$$

so ist Satz 5.2.4 mit  $D = \overline{B_\varepsilon(x^*)}$  anwendbar.

Beweis. Nutze (5.2.4) und Lemma 5.2.3. □

## 5.3 Konvergenzordnung und Fehlerabschätzungen

**Definition 5.3.1.** Eine Folge  $(x^{(k)})_{k \in \mathbb{N}}$  mit  $x^{(k)} \in \mathbb{R}^n$  konvergiert mit (mindestens) der Ordnung  $p \geq 1$  gegen  $x^*$ , falls  $\lim_{k \rightarrow \infty} x^{(k)} = x^*$  und falls es ein  $C > 0$  sowie  $N \in \mathbb{N}$  gibt, so dass

$$\|x^{(k+1)} - x^*\| \leq C \|x^{(k)} - x^*\|^p \quad \forall k \geq N.$$

Im Fall  $p = 1$  ist zusätzlich  $C < 1$  gefordert. Dann wird von *linearer Konvergenz* gesprochen.

Für  $p = 2$  heißt es *quadratische Konvergenz*.

Gilt

$$\lim_{k \rightarrow \infty} \frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|} = 0,$$

so konvergiert die Folge *superlinear*.

**Folgerung 5.3.2.**

Die Fixpunktiteration konvergiert unter der Voraussetzung in Satz 5.2.4 mindestens linear.

Bemerkung 5.3.3.

- a) Lineare Konvergenz hängt von der gewählten Norm ab.
- b) Hat die Folge bzgl. einer Vektornorm auf dem  $\mathbb{R}^n$  die Konvergenzordnung  $p > 1$ , hat sie diese bzgl. jeder Norm.

5.12.22

**Definition 5.3.4.**

- a) Ein *iteratives Verfahren* zur Bestimmung eines Wertes  $x^*$  hat die *Konvergenzordnung*  $p$ , falls es eine Umgebung  $U$  um  $x^*$  gibt, so dass für alle Startwerte aus  $U \setminus \{x^*\}$  die erzeugte Folge mit Ordnung  $p$  konvergiert.
- b) Das Verfahren heißt *lokal konvergent*, falls es für alle Startwerte in einer Umgebung von  $x^*$  konvergiert.
- c) Das Verfahren heißt *global konvergent*, falls es im gesamten Definitionsbereich des zugehörigen Problems konvergiert.

**Lemma 5.3.5.** Sei  $(x^{(k)})_{k \in \mathbb{N}}$  eine konvergente Folge in  $\mathbb{R}$  mit Grenzwert  $x^*$ .

a) Falls

$$\lim_{k \rightarrow \infty} \frac{x^{(k+1)} - x^*}{x^{(k)} - x^*} = A \in (-1, 1), \quad A \neq 0 \tag{5.3.1}$$

hat die Folge genau die Konvergenzordnung 1. Weiter gilt mit

$$A_k := \frac{x^{(k)} - x^{(k-1)}}{x^{(k-1)} - x^{(k-2)}} \\ \lim_{k \rightarrow \infty} \frac{\frac{A_k}{1-A_k}(x^{(k)} - x^{(k-1)})}{x^* - x^{(k)}} = 1 \quad \text{sowie} \quad \lim_{k \rightarrow \infty} A_k = A. \quad (5.3.2)$$

b) Falls die Folge Konvergenzordnung  $p > 1$  hat, gilt

$$\lim_{k \rightarrow \infty} \frac{x^{(k+1)} - x^{(k)}}{x^* - x^{(k)}} = 1. \quad (5.3.3)$$

*Beweis.* (skizzenhaft, siehe Übungsaufgaben) Sei  $e^{(k)} := x^* - x^{(k)}$ . Nutze  $x^{(k+1)} - x^{(k)} = e^{(k)} - e^{(k+1)}$ :

a) Zeige  $\lim_{k \rightarrow \infty} \frac{x^{(k)} - x^{(k-1)}}{e^{(k)}} = \frac{1-A}{A}$ , sowie  $\lim_{k \rightarrow \infty} A_k = A$ . Dann folgt die Behauptung.

b) Folgt aus  $\lim_{k \rightarrow \infty} \frac{e^{(k+1)}}{e^{(k)}} = 0$ .

□

**Folgerung 5.3.6** (a posteriori Fehlerabschätzung).

a) Für  $p = 1$ , große  $k$  und  $A_k$  in etwa konstant sowie  $x^{(k)} \in \mathbb{R}$  gilt

$$x^* - x^{(k)} \approx \frac{A_k}{1 - A_k} (x^{(k)} - x^{(k-1)}). \quad (5.3.4)$$

$|x^{(k)} - x^{(k-1)}|$  ist im Allgemeinen **keine** sinnvolle Schätzung des Fehlers  $|x^* - x^{(k)}|$ !

b) Für  $p > 1$ ,  $x^{(k)} \in \mathbb{R}$  und große  $k$  gilt

$$x^* - x^{(k)} \approx x^{(k+1)} - x^{(k)}. \quad (5.3.5)$$

*Bemerkung 5.3.7.*

Für Folgen im  $\mathbb{R}^n$  mit  $n > 1$  gibt es für  $p = 1$  kein Analogon zu (5.3.4).

Falls  $p > 1$ , lässt sich (5.3.3) für die Normen der Differenzen zeigen, woraus für große  $k$

$$\|x^* - x^{(k)}\| \approx \|x^{(k+1)} - x^{(k)}\| \quad (5.3.6)$$

folgt.

*Beweis.* Für  $p > 1$  nutze  $\lim_{k \rightarrow \infty} \frac{\|e^{(k+1)}\|}{\|e^{(k)}\|} = 0$  und

$$\|e^{(k)}\| - \|e^{(k+1)}\| \leq \|x^{(k+1)} - x^{(k)}\| \leq \|e^{(k)}\| + \|e^{(k+1)}\|.$$

□

**Folgerung 5.3.8.** Falls  $p > 1$  ist, kann  $p$  folgendermaßen approximiert werden:

$$p \approx \frac{\log(\|x^{(k+2)} - x^{(k+1)}\|)}{\log(\|x^{(k+1)} - x^{(k)}\|)} \quad \text{für große } k.$$

*Beweis.* Übungsaufgabe.

□

## 5.4 Newton-Verfahren für skalare Gleichungen

Sei  $f: [a, b] \rightarrow \mathbb{R}$  differenzierbar. Dann gilt

$$f(x^*) = f(x) + f'(x)(x^* - x) + o(\|x - x^*\|),$$

d.h.  $f$  kann lokal gut durch die Tangente approximiert werden.

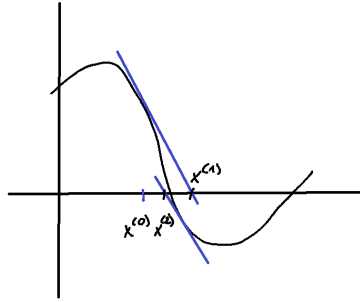
Anstatt nun die nichtlineare Nullstellengleichung  $f(x^*) = 0$  zu lösen, bestimme für gegebenes  $x$  die Nullstelle  $\bar{x}$  der Tangentengleichung

$$0 = f(x) + f'(x)(\bar{x} - x) \Leftrightarrow \bar{x} = x - \frac{f(x)}{f'(x)}$$

und iteriere dies. Notwendig ist hier die Bedingung  $f'(x) \neq 0$ .

### 5.4.1. Iterationsschritt des Newton(-Kantorowitsch)-Verfahrens.

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}. \quad (5.4.1)$$



Das Newton-Verfahren wird auch *Tangentenverfahren* genannt und stammt von J. Raphson (1630). Newton hat eine ähnliche Technik früher angewendet.

Die entscheidende Idee beim Newton-Verfahren ist der *iterative Linearisierungsprozess*, d.h. die Lösung einer nichtlinearen Gleichung wird durch eine Folge von Lösungen linearer Gleichungen ersetzt.

**Satz 5.4.2.** Sei  $f \in C^1(a, b)$  und  $x^* \in (a, b)$  eine einfache Nullstelle von  $f$ , d.h.  $f'(x^*) \neq 0$ . Dann gibt es ein  $\varepsilon > 0$ , so dass für jedes  $x^{(0)} \in \overline{B_\varepsilon(x^*)}$  das Newton-Verfahren (5.4.1) superlinear gegen  $x^*$  konvergiert.

Falls  $f \in C^2(a, b)$  tritt mindestens quadratische Konvergenz ein, d.h. das Verfahren konvergiert lokal quadratisch.

*Beweis.* Für  $f \in C^2(a, b)$ : Gleichung (5.4.1) definiert eine Fixpunktiteration mit

$$g(x) = x - \frac{f(x)}{f'(x)} \quad (5.4.2)$$

Für  $f \in C^2(a, b)$  gilt

$$g'(x) = 1 - \frac{f'(x)f'(x) - f(x)f''(x)}{(f'(x))^2} = \frac{f(x)f''(x)}{(f'(x))^2}.$$

Da  $f(x^*) = 0$  und  $f'(x^*) \neq 0$  gilt  $g'(x^*) = 0$ . Weiterhin gibt es aufgrund der Stetigkeit von  $f'$  eine Umgebung  $U$  von  $x^*$ , in der  $f'(x) \neq 0 \forall x \in U$ . In  $U$  ist somit  $g'$  stetig. Da  $g'(x^*) = 0$  ist, existiert also ein  $\varepsilon > 0$  mit

$$g'(x) \leq \kappa < 1 \quad \forall x \in \overline{B_\varepsilon(x^*)}.$$

Da  $g(x^*) = x^*$  ist, ist die Folgerung 5.2.6 anwendbar, also ist  $g$  eine Kontraktion und  $g(\overline{B_\varepsilon(x^*)}) \subseteq \overline{B_\varepsilon(x^*)}$ . Der Banachsche Fixpunktsatz liefert die Wohldefiniertheit der Newton-Iteration und die Konvergenz für alle  $x^{(0)} \in \overline{B_\varepsilon(x^*)}$ .

Die quadratische Konvergenz folgt aus

$$\begin{aligned} |x^{(k+1)} - x^*| &= \left| x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})} - x^* + \frac{f(x^*)}{f'(x^{(k)})} \right| \\ &= \frac{|f(x^{(k)}) - f(x^*) + f'(x^{(k)})(x^{(k)} - x^*)|}{|f'(x^{(k)})|} \\ &\leq \sup_{x \in B_\varepsilon(x^*)} \frac{1}{|f'(x)|} \sup_{x \in B_\varepsilon(x^*)} |f''(x)| \frac{1}{2} |x^{(k)} - x^*|^2 \end{aligned}$$

aufgrund der Taylorentwicklung.

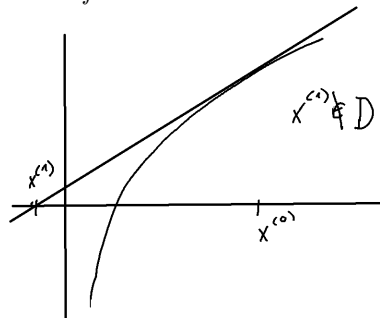
Für  $f \in C^1$  ist der Beweis eine Übungsaufgabe. □

*Bemerkung 5.4.3.*

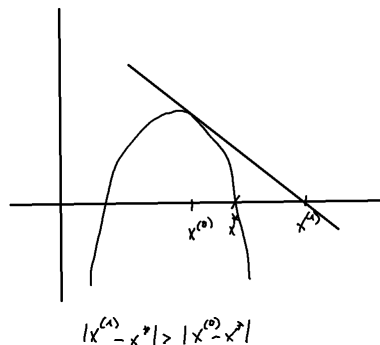
- a) Mehrfache Nullstellen können im Allgemeinen nicht mit (5.4.1) bestimmt werden.
- b) Die Ableitung  $f'$  muss analytisch (als Funktion) gegeben sein.
- c) Die Lage und Größe des Konvergenzintervalls ist a priori unbekannt.  
(Um in die Nähe der Nullstelle zu kommen, könnte z.B. das Bisektionsverfahren Anwendung finden.)

*Beispiel 5.4.4* (Newton-Verfahren ohne Konvergenz).

- $x^{(1)}$  liegt nicht mehr im Definitionsbereich von  $f$



- $|x^* - x^{(1)}| \not\leq |x^* - x^{(0)}|$  und somit ist  $g$  aus (5.4.2) keine Kontraktion für den Startwert  $x^{(0)}$ . Die Konvergenz des Newton-Verfahrens ist daher nicht gesichert.



*Beispiel 5.4.5.* Es ist die Lösung von  $x - e^{-\frac{1}{2}x} = 0$  gesucht. Als Startwert sei  $x^{(0)} = 0,8$  gewählt.

- a) Mit der Banachschen Fixpunktiteration angewendet auf die Fixpunktgleichung  $x = e^{-\frac{1}{2}x}$  ergibt sich

$$x^{(10)} = \mathbf{0,70347017}, \quad \text{was auf 4 Stellen exakt ist.}$$

b) Mit dem Newton-Verfahren ergibt sich

$$\begin{array}{ll} x^{(3)} = 0,70346742, & \text{was bis auf 17 Stellen exakt ist, und} \\ x^{(4)} & \text{ist bis auf Maschinengenauigkeit exakt.} \end{array}$$

7.12.22

Die Ableitung  $f'(x)$  ist nicht immer explizit bekannt. Eine Idee ist, sie mithilfe des Differenzenquotienten zu approximieren:

$$f'(x^{(k)}) \approx \frac{f(x^{(k)}) - f(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}.$$

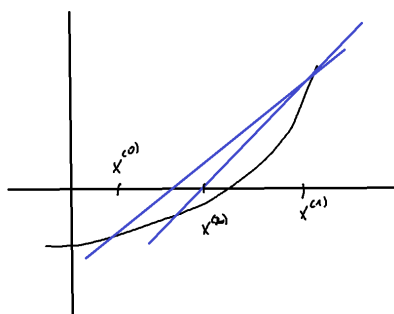
Damit ergibt sich

$$x^{(k+1)} = x^{(k)} - f(x^{(k)}) \frac{x^{(k)} - x^{(k-1)}}{f(x^{(k)}) - f(x^{(k-1)})},$$

d.h.  $x^{(k+1)}$  ist die Nullstelle der Sekante durch  $f(x^{(k)})$  und  $f(x^{(k-1)})$ .

#### 5.4.6. Iterationsschritt des Sekantenverfahrens.

$$x^{(k+1)} = \frac{x^{(k-1)} f(x^{(k)}) - x^{(k)} f(x^{(k-1)})}{f(x^{(k)}) - f(x^{(k-1)})}. \quad (5.4.3)$$



**Satz 5.4.7** (Konvergenz des Sekantenverfahrens). Sei  $f \in C^2([a, b])$  und  $x^* \in (a, b)$  eine einfache Nullstelle. Dann konvergiert das Sekantenverfahren in einer Umgebung von  $x^*$  superlinear mit Ordnung

$$p = \frac{1}{2}(1 + \sqrt{5}) \approx 1,618.$$

*Beweis.* Siehe z.B. [7, 12] (mittels Zwischenwertsatz und Fibonacci-Folge). □

#### Zu Beispiel 5.4.5:

Das Sekantenverfahren benötigt einen zweiten Startwert. Mit  $x^{(0)} = 0,8$  und  $x^{(1)} = 0,7$  ergibt sich

$$\begin{array}{ll} x^{(3)} = 0,7034674, & \text{was bis auf 7 Stellen exakt ist, und} \\ x^{(6)} & \text{ist bis auf Maschinengenauigkeit exakt.} \end{array}$$

*Bemerkung 5.4.8.*

- Das Sekanten-Verfahren ist keine Fixpunktiteration. Es benötigt  $x^{(k)}$  und  $x^{(k-1)}$  für  $x^{(k+1)}$  (**Mehrschrittverfahren**).
- Die Berechnung von  $f(x)$  und  $f'(x)$  ist im Allgemeinen sehr teuer. Das Sekanten-Verfahren benötigt pro Iteration nur eine (skalare) Funktionsauswertung. Das Newton-Verfahren benötigt hingegen zwei (skalare) Funktionsauswertungen. Also sind zwei Iterationen des Sekanten-Verfahrens so teuer wie eine des Newton-Verfahrens. Bei gleichem Aufwand konvergiert das Sekanten-Verfahren daher lokal schneller als das Newton-Verfahren mit der Konvergenzordnung  $p^2 = 2,618 \dots$  für  $x^{(k)} \rightarrow x^{(k+2)}$  (siehe auch Beispiel 5.4.5).
- Die Sekantenmethode ist i.A. nicht stabil, denn für  $f(x^{(k)}) \approx f(x^{(k+1)})$  können Stellenauslöschungen im Nenner auftreten. Stabilere Varianten, wie z.B. **regula falsi**, haben eine geringere Konvergenzordnung.

## 5.5 Das Newton-Verfahren im Mehrdimensionalen

Wie im 1-dimensionalen wird  $f: \Omega \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$  linearisiert

$$f(\bar{x}) \approx f(x) + Df(x)(\bar{x} - x) \quad (5.5.1)$$

mit der Jacobi-Matrix

$$Df(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x) & \dots & \frac{\partial f_1}{\partial x_n}(x) \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1}(x) & \dots & \frac{\partial f_n}{\partial x_n}(x) \end{pmatrix}$$

Falls nun für gegebenes  $x$  die Jacobi-Matrix  $Df(x)$  invertierbar ist und  $x^*$  mit  $f(x^*) = 0$  zu finden ist, approximiere  $x^*$  durch die Lösung  $\bar{x}$  der linearen Gleichung  $0 = f(x) + Df(x)(\bar{x} - x)$ :

$$\bar{x} = x - [Df(x)]^{-1}f(x).$$

### 5.5.1. Iterationsschritt des Newton-Verfahrens.

$$x^{(k+1)} = x^{(k)} - [Df(x^{(k)})]^{-1}f(x^{(k)}) \quad (5.5.2)$$

*Bemerkung 5.5.2.* Ein Newton-Iterationsschritt (5.5.2) wird aufgeteilt in die Berechnung der sogenannten *Newton-Korrektur*  $\Delta x^{(k)}$  mit

$$Df(x^{(k)})\Delta x^{(k)} = -f(x^{(k)}) \quad (5.5.3)$$

und dem *Korrekturschritt*

$$x^{(k+1)} = x^{(k)} + \Delta x^{(k)}. \quad (5.5.4)$$

---

#### Algorithmus 5.5.3 : Das mehrdimensionale Newton-Verfahren

---

```

setze Startwert  $x$ 
 $i = 0$ 
 $fx = f(x)$ 
while „Abbruchkriterium“ do
   $Dfx = Df(x)$ 
  Löse  $Dfx \cdot d = -fx$ 
   $x = x + d$ 
   $fx = f(x)$ 
   $i = i + 1$ 
end while

```

---

### 5.5.4. Aufwand pro Iteration.

$n$  eindimensionale Funktionsauswertungen für  $f(x)$ ,

$n^2$  eindimensionale Funktionsauswertungen für  $Df(x)$ ,

Lösen eines linearen Gleichungssystems mit i.d.R.  $\mathcal{O}(n^3)$  flops.

*Bemerkung 5.5.5.* Das Newton-Verfahren ist *affin-invariant*, d.h. die Folge  $(x^{(k)})$  ist zu gegebenem  $x^{(0)}$  unabhängig davon, ob  $f(x) = 0$  oder  $\tilde{f}(x) := Af(x) = 0$  mit regulärem  $A \in \mathbb{R}^{n \times n}$  gelöst wird. Dies gilt, da

$$[D\tilde{f}(x)]^{-1}\tilde{f}(x) = [ADf(x)]^{-1}(Af(x)) = [Df(x)]^{-1}f(x)$$

und damit die Newton-Korrektur  $\Delta x^{(k)}$  für  $f$  und  $\tilde{f}$  identisch also affin-invariant ist.

**Satz 5.5.6.** Sei  $\Omega \in \mathbb{R}^n$  offen und  $f: \Omega \rightarrow \mathbb{R}^n$  in  $C^2(\Omega)$ . Sei  $x^* \in \Omega$  eine Nullstelle von  $f$  mit einer invertierbaren Jacobi-Matrix  $Df(x^*)$ . Dann existiert eine Umgebung von  $x^*$ , so dass das Newton-Verfahren für jeden Startwert  $x^{(0)}$  in dieser Umgebung quadratisch gegen  $x^*$  konvergiert.

*Beweis.* Der Beweis kann wie im eindimensionalen durchgeführt werden. Aber Vorsicht:  $D^2f(x)$  ist eine **bilineare Abbildung**, d.h. in  $\mathcal{L}(\mathbb{R}^n, \mathcal{L}(\mathbb{R}^n, \mathbb{R}^n))$ .

Wir zeigen die Behauptung induktiv über die quadratische Konvergenz. Da  $Df(x^*)$  invertierbar ist und  $f \in C^2(\Omega)$ , existiert nach dem Satz über implizite Funktionen eine Umgebung  $\overline{B_\varepsilon(x^*)} \subset \Omega$ , auf der  $Df(x)$  invertierbar und stetig ist. Sei

$$c := \sup_{x \in B_\varepsilon(x^*)} \| [Df(x)]^{-1} \| (= \max_{x \in B_\varepsilon(x^*)} \| [Df(x)]^{-1} \|)$$

und

$$w := \sup_{x \in B_\varepsilon(x^*)} \| D^2f(x) \|.$$

Für  $x^{(k)} \in B_\varepsilon(x^*)$  ist  $x^{(k)} + t(x^* - x^{(k)}) \in B_\varepsilon(x^*)$  für  $t \in [0, 1]$  und

$$h^{(k)}(t) := f(x^{(k)} + t(x^* - x^{(k)})) \quad \forall t \in [0, 1]$$

ist wohldefiniert und in  $C^2([0, 1], \mathbb{R}^n)$ . Wie in 5.4.2 folgt

$$\begin{aligned} x^{(k+1)} - x^* &= [Df(x^{(k)})]^{-1} (f(x^*) - f(x^{(k)}) - Df(x^{(k)})(x^* - x^{(k)})) \\ &= [Df(x^{(k)})]^{-1} (h^{(k)}(1) - h^{(k)}(0) - Dh^{(k)}(0)) \\ &= [Df(x^{(k)})]^{-1} \int_0^1 D^2h^{(k)}(t)(1-t)dt \end{aligned} \quad (\text{Restglieddarst. der Taylorentw.})$$

$$\begin{aligned} \Rightarrow \|x^{(k+1)} - x^*\| &\leq c \frac{1}{2} \sup_{t \in [0, 2]} \|D^2h^{(k)}(t)\| \quad \forall x^{(k)} \in B_\varepsilon(x^*) \\ &\leq c \frac{1}{2} w \|x^{(k)} - x^*\|^2 \quad \forall x^{(k)} \in B_\varepsilon(x^*), \end{aligned} \quad (5.5.5)$$

wobei die Abschätzung (5.5.5) noch ausführlicher gezeigt wird.

Sei nun  $\delta \leq \varepsilon$ , so dass  $\frac{1}{2}cw\delta < 1$  gilt, so folgt induktiv für  $x^{(0)} \in B_\delta(x^*)$  mit (5.5.5)

$$\begin{aligned} \|x^{(k+1)} - x^*\| &\leq \frac{1}{2}cw\delta^2 < \delta \\ \Rightarrow x^{(k+1)} &\in B_\delta(x^*) \subseteq B_\varepsilon(x^*). \end{aligned}$$

Auf  $x^{(k+1)}$  ist der nächste Iterationsschritt anwendbar, das Verfahren also wohldefiniert für  $x^{(0)} \in B_\delta(x^*)$  und mit (5.5.5) folgt quadratische Konvergenz.

Es bleibt zu zeigen:

$$\|D^2h(t)\| \leq w \|x^{(k)} - x^*\|^2 \quad \forall t \in [0, 1].$$

Mit

$$h: [0, 1] \rightarrow \mathbb{R}^n, \quad h = \begin{pmatrix} h_1 \\ \vdots \\ h_n \end{pmatrix}$$

gilt für  $h_i: [0, 1] \rightarrow \mathbb{R}$

$$\begin{aligned} Dh_i^{(k)}(t) &= \underbrace{Df_i(x^{(k)} + t(x^* - x^{(k)}))}_{\in \mathbb{R}^{1 \times n}} (x^* - x^{(k)}) \in \mathbb{R}, \\ D^2h_i^{(k)}(t) &= (x^* - x^{(k)})^T \underbrace{D^2f_i(x^{(k)} + t(x^* - x^{(k)}))}_{\in \mathbb{R}^{n \times n}} (x^* - x^{(k)}) \in \mathbb{R}, \end{aligned}$$

woraus die Abschätzung folgt.  $\square$

12.12.2022

Unter genaueren Voraussetzungen kann die Existenz einer Nullstelle  $x^*$  gezeigt und eine Umgebung  $B_r(x^*)$  explizit angegeben werden. Dies liefert folgender Satz:

**Satz 5.5.7** (Satz von Kantorowitsch). *Sei  $\Omega \subseteq \mathbb{R}^n$  konvex und  $f: \Omega \rightarrow \mathbb{R}^n$  stetig differenzierbar auf  $\Omega$  und erfülle für ein  $x^{(0)} \in \Omega$  folgendes:*

- a)  $\|Df(x) - Df(y)\| \leq \gamma \|x - y\|$  für alle  $x, y \in \Omega$ ,
- b)  $\|[Df(x^{(0)})]^{-1}\| \leq \beta$ ,
- c)  $\|[Df(x^{(0)})]^{-1}f(x^{(0)})\| \leq \alpha$ .

Mit den Konstanten

$$h = \alpha\beta\gamma, \quad r_{\pm} = \frac{1 \pm \sqrt{1 - 2h}}{h}\alpha$$

folgt dann, falls  $h \leq \frac{1}{2}$  und  $\overline{B_{r_-}(x^{(0)})} \subset \Omega$ , dass  $f$  genau eine Nullstelle  $x^*$  in  $\Omega \cap B_{r_+}(x^{(0)})$  hat. Weiterhin bleibt die Folge der Newton-Iterierten in  $B_{r_-}(x^{(0)})$  und konvergiert gegen  $x^*$ .

Beweis. z.B. in Ortega/Rheinhold (2000)  $\square$

## 5.6 Abbruchkriterien beim Newton-Verfahren

- 1) Limitiere die Anzahl der Iterationen, u.a. um Endlosschleifen durch fehlerhafte Programme auszuschließen.
- 2) Breche ab, wenn das Verfahren nicht konvergiert, d.h. wenn  $x^{(k)}$  nicht im Konvergenzbereich liegt.
- 3) Breche ab, wenn das Ergebnis genau genug ist, d.h. der Fehler

$$e^{(k)} := \|x^* - x^{(k)}\|$$

klein genug ist.

### 5.6.1. Der Monotonietest.

Beim Newton-Verfahren sollte die Funktion  $g$  der zugehörigen Fixpunktiteration eine Kontraktion sein, d.h. es muss ein  $\kappa \in [0, 1)$ , so dass für alle  $k$  gilt

$$\begin{aligned} \|\Delta x^{(k)}\| &= \|x^{(k+1)} - x^{(k)}\| = \|g(x^{(k)}) - g(x^{(k-1)})\| \\ &\leq \kappa \|x^{(k)} - x^{(k-1)}\| = \kappa \|\Delta x^{(k-1)}\|. \end{aligned} \quad (5.6.1)$$

Als Abbruchkriterium für eine mögliche Divergenz des Verfahrens wähle ein  $\kappa \in [0, 1)$ , z.B.  $\kappa = \frac{1}{2}$ , und breche ab, falls

$$\|\Delta x^{(k)}\| > \kappa \|\Delta x^{(k-1)}\|. \quad (5.6.2)$$

Um im Mehrdimensionalen eine vielleicht unnötig (teure) Berechnung von  $Df(x^{(k)})$  für  $\Delta x^{(k)}$  zu vermeiden, kann  $\Delta x^{(k)}$  durch

$$\overline{\Delta x}^{(k)} = -[Df(x^{(k-1)})]^{-1}f(x^{(k)}) \quad (5.6.3)$$

approximiert werden.  $Df(x^{(k-1)})$  mit einer Zerlegung liegt bereits aus der Berechnung von  $\Delta x^{(k-1)}$  vor. Ebenso ist  $f(x^{(k)})$  bekannt. Die Lösung von (5.6.3) benötigt daher nur  $\mathcal{O}(n^2)$  flops. Statt (5.6.2) wird dann auf

$$\|\overline{\Delta x}^{(k)}\| > \kappa \|\Delta x^{(k-1)}\| \quad (5.6.4)$$

getestet.



**5.6.2. Kriterium für erreichte Genauigkeit.**

Es ist eine Nullstelle gesucht, also teste, ob  $x^{(k)}$  eine ist. Dieses *residuumbasierte Kriterium*

$$\|f(x^{(k)})\| \leq \text{tol} \quad (5.6.5)$$

ist nur bedingt anwendbar. Betrachte z.B.  $\tilde{f}(x) = \alpha f(x)$ , dann haben  $f$  und  $\tilde{f}$  dieselben Nullstellen. Das Abbruchkriterium (5.6.5) unterscheidet sich jedoch um den Faktor  $|\alpha|$ . Das Newton-Verfahren ist affin-invariant. Demnach bleibt  $(x^{(k)})_{k \in \mathbb{N}}$  gleich, ob nun  $f(x)$  oder  $\tilde{f}(x)$  betrachtet wird. Ein affin-invariantes Abbruchkriterium ist

$$\|[Df(x^{(k)})]^{-1} f(x^{(k)})\| \leq \text{tol} . \quad (5.6.6)$$

Nutze daher

$$\|\Delta x^{(k)}\| \leq \text{tol} . \quad (5.6.7)$$

(5.6.7) könnte aufgrund der quadratischen Konvergenz auch mit (5.3.5) der Approximation des Fehlers  $\|x^* - x^{(k)}\|$  motiviert werden. Diese Begründung gilt jedoch nur für große  $k$ .

**5.7 Varianten des Newton-Verfahrens**

$Df(x^{(k)})$  steht nicht immer analytisch zur Verfügung. Die exakte Jacobi-Matrix wird häufig durch eine andere Matrix  $B$  approximiert, z.B. durch den Differenzenquotienten im Eindimensionalen oder durch sogenanntes *automatisches Differenzieren*. Der Iterationsschritt lautet dann

$$\begin{aligned} \text{löse } B^{(k)} d^{(k)} &= -f(x^{(k)}) \\ x^{(k+1)} &= x^{(k)} + d^{(k)} \end{aligned} \quad (5.7.1)$$

Um den Aufwand zu verringern kann  $Df(x^{(k)})$  z.B. auch durch  $Df(x^{(0)})$  approximiert werden.

**5.7.1. Iterationsschritt des vereinfachten Newton-Verfahrens.**

$$x^{(k+1)} = x^{(k)} - [Df(x^{(0)})]^{-1} f(x^{(k)}) . \quad (5.7.2)$$

Das Verfahren konvergiert nur noch lokal linear. Der Aufwand je Iteration ist jedoch erheblich geringer (den genauen Aufwand zu berechnen ist eine Übung).

**5.7.2. Das Broyden-Verfahren.**

Das Broyden-Verfahren ist eine Verallgemeinerung des Sekantenverfahrens auf den  $\mathbb{R}^n$  mit  $n > 1$ .  $Df(x^{(k)})$  wird mittels der abgebrochenen Taylorentwicklung approximiert, d.h.

$$B^{(k)} \underbrace{(x^{(k)} - x^{(k-1)})}_{:= p^{(k-1)}} = \underbrace{f(x^{(k)}) - f(x^{(k-1)})}_{:= q^{(k-1)}} . \quad (5.7.3)$$

$B^{(k)}$  ist nicht eindeutig durch (5.7.3) festgelegt.

Die *Broyden-Aufdatierung* bestimmt  $B^{(k)}$  rekursiv durch eine Aufdatierung mit einer Rang-1-Matrix, auch „rang-1-update“ genannt ( $C_{\text{neu}} = C_{\text{alt}} + M$  mit  $\text{rang}(M) = 1$ ). Die Broyden-Aufdatierung lautet dann für gegebenes  $B^{(k)}$ ,  $x^{(k)}$  und  $x^{(k+1)}$ :

$$\begin{aligned} p^{(k)} &:= x^{(k+1)} - x^{(k)} , \\ q^{(k)} &:= f(x^{(k+1)}) - f(x^{(k)}) , \\ B^{(k+1)} &:= B^{(k)} + \frac{1}{p^{(k)T} p^{(k)}} (q^{(k)} - B^{(k)} p^{(k)}) p^{(k)T} . \end{aligned} \quad (5.7.4)$$

Aus (5.7.4) folgt die Bedingung (5.7.3), was selbst nachgewiesen werden soll.

Ein Iterationsschritt des *Broyden-Verfahrens* lautet somit:

---

**Algorithmus 5.7.3 :** Ein Iterationsschritt des Broyden-Verfahrens

---

$$\begin{aligned} &\text{löse } B^{(k)} d^{(k)} = -f(x^{(k)}) \\ &x^{(k+1)} := x^{(k)} + d^{(k)} \\ &B^{(k+1)} := B^{(k)} + \frac{1}{d^{(k)T} d^{(k)}} f(x^{(k+1)}) d^{(k)T} \end{aligned}$$


---

Für das Verfahren muss  $x^{(0)}$  und  $B^{(0)}$  gegeben sein. Unter bestimmten Voraussetzungen konvergiert das Verfahren lokal superlinear (siehe [12] und dortige Referenzen).

#### 5.7.4. Das gedämpfte Newton-Verfahren.

Falls  $f$  eine Nullstelle  $x^*$  hat, gilt

$$f(x^*) = 0 \quad \Leftrightarrow \quad f(x^*) = \arg \min_{x \in \mathbb{R}^n} \frac{1}{2} \|f(x)\|_2^2 \quad (5.7.5)$$

Betrachte nun die Funktion  $\Phi: \mathbb{R}^n \rightarrow \mathbb{R}$  mit

$$\Phi(x) := \frac{1}{2} \|f(x)\|_2^2 = \frac{1}{2} f(x)^T f(x).$$

Für  $\Phi$  ist die Newton-Korrektur  $d^{(k)} := \Delta x^{(k)} = -[Df(x^{(k)})]^{-1} f(x^{(k)})$  in  $x^{(k)} \neq x^*$  eine *Abstiegsrichtung*, d.h.

$$\exists \bar{\mu} > 0 \quad \forall 0 < \mu < \bar{\mu}: \quad \Phi(x^{(k)} + \mu d^{(k)}) < \Phi(x^{(k)}), \quad (5.7.6)$$

denn für  $f(x) \neq 0$  gilt

$$\begin{aligned} \left. \frac{d}{d\mu} \Phi(x + \mu d) \right|_{\mu=0} &= [f(x + \mu d)^T Df(x + \mu d) d]_{\mu=0} \\ &= -f(x)^T f(x) < 0. \end{aligned}$$

Die Idee ist nun, statt wie im Newton-Verfahren den ganzen Schritt  $d^{(k)}$  zu gehen, wähle eine „geeignete“ Schrittweite  $\alpha^{(k)} \in (0, 1]$  und setze

$$x^{(k+1)} := x^{(k)} + \alpha^{(k)} d^{(k)}, \quad (5.7.7)$$

d.h. dämpfe  $d^{(k)}$  mit einer Schrittweite  $\alpha^{(k)}$ . Ein möglicher „Eignungstest“ für die Schrittweite ist

$$\|f(x^{(k)} + \alpha d^{(k)})\| \leq (1 - \frac{1}{2}\alpha) \|f(x^{(k)})\|. \quad (5.7.8)$$

Eine mögliche Strategie um solch ein  $\alpha$  zu bestimmen ist das sogenannte *back tracking*:

1. Setze  $\alpha = 1$ .
2. Halbiere  $\alpha$  rekursiv solange, bis (5.7.8) gilt.

Es sind allerdings effektivere Dämpfungsstrategien bekannt.

Beim gedämpften Newton-Verfahren gibt es also eine äußere Iteration ( $k$ ) um  $x^*$  zu approximieren und eine innere Iteration, um für jedes ( $k$ ) ein geeignetes  $\alpha^{(k)}$  zu berechnen. Eine innere Iteration ist mit  $n$  eindimensionalen Funktionsauswertungen wesentlich weniger aufwendig als eine äußere Iteration.

Unter bestimmten Voraussetzungen ist **globale** Konvergenz gewährleistet.

# Literatur

- [1] M. Abramowitz und I. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. 1964.
- [2] Carl de Boor. *A Practical Guide to Splines*. Applied Mathematical Sciences. Springer New York, 2001. ISBN: 9780387953663.
- [3] Wolfgang Dahmen und Arnold Reusken. *Numerik für Ingenieure und Naturwissenschaftler*. 2. Aufl. Springer-Lehrbuch. Springer-Verlag, Berlin, 2008. ISBN: 978-3-540-76493-9.
- [4] Peter Deuffhard und Andreas Hohmann. *Numerische Mathematik*. 1. 4. Aufl. de Gruyter Lehrbuch. Eine algorithmisch orientierte Einführung. Walter de Gruyter & Co., Berlin, 2008. ISBN: 978-3-11-020354-7.
- [5] R.W. Freund und R.H.W. Hoppe. *Stoer/Bulirsch: Numerische Mathematik 1*. Springer-Lehrbuch Bd. 1. Springer-Verlag Berlin Heidelberg, 2007. ISBN: 9783540453901. URL: <https://books.google.de/books?id=2aYfBAAQBAJ>.
- [6] Gene Golub und James M. Ortega. *Scientific computing*. Eine Einführung in das wissenschaftliche Rechnen und Parallele Numerik, Übersetzung des englischsprachigen Originals von 1993. B. G. Teubner, Stuttgart, 1996. ISBN: 3-519-02969-3. DOI: [10.1007/978-3-322-82981-8](https://doi.org/10.1007/978-3-322-82981-8).
- [7] Günther Hämmerlin und Karl-Heinz Hoffmann. *Numerische Mathematik*. 4. Aufl. Springer-Lehrbuch. Grundwissen Mathematik. Springer-Verlag, Berlin, 1994. ISBN: 3-540-58033-6. DOI: [10.1007/978-3-642-57894-6](https://doi.org/10.1007/978-3-642-57894-6).
- [8] William H. Press, Saul A. Teukolsky, William T. Vetterling und Brian P. Flannery. *Numerical recipes in C++*. The art of scientific computing, Second edition, updated for C++. Cambridge University Press, Cambridge, 2002. ISBN: 0-521-75033-4.
- [9] Robert Schaback und Holger Wendland. *Numerische Mathematik*. Springer-Lehrbuch. Springer-Verlag Berlin Heidelberg, 2005. ISBN: 9783540267058. URL: <https://books.google.de/books?id=wdgmBAAQBAJ>.
- [10] J.F. Steffensen. *Interpolation*. 1. Aufl. New York, Chelsea, 1965.
- [11] J. Stoer. *Numerische Mathematik 1*. Springer.
- [12] Josef Stoer und Roland Bulirsch. *Numerische Mathematik*. 2. 3. Aufl. Springer-Lehrbuch. Eine Einführung—unter Berücksichtigung von Vorlesungen von F. L. Bauer. Springer-Verlag, Berlin, 1990. ISBN: 3-540-51482-1. DOI: [10.1007/978-3-662-22250-8](https://doi.org/10.1007/978-3-662-22250-8).