2018

Zusammenfassung NMIT1

SEP FS 2019

PASCAL BRUNNER

Inhaltsverzeichnis

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

1 Grundlegendes

Die Numerik beschäftigt sich mit der Konstruktion und Analyse von Algorithmen für kontinuierliche mathematische Probleme, dabei möchte man numerische Resultate erzielen. Die Verfahren werden in zwei Typen unterschieden:

- Direkte, die nach endlicher Zeit bei unendlicher Rechengenauigkeit die exakte Lösung eines Problems liefern (bspw. Gauss'sche Eliminationsverfahren für LGS)
- Näherungsverfahren, die eine Approximation liefern (bspw. Newton Verfahren)

2 Rechenarithmetik

2.1 Maschinenzahlen

Jede Rechenmaschine ist ein endlicher Automat, dementsprechend kann er nicht alle Zahlen exakt darstellen (beschränkte Stellenzahl) und nur endliche viele Operationen ausführen.

➔ Für eine gegebene Basis B € N (B > 1) kann jede reelle Zahl *x* € R aber als *x* = m x B^e dargestellt werden. Wobei m € R = Mantisse und e € Z der Exponent

Computerintern wird üblicherweise die Basis = 2 (Binär) verwendet.

B = 8 → Oktalzahlen

B = 10 → Dezimalzahlen

B = 16 → Hexadezimalzahlen

Definition Maschinenzahlen / Gleitpunktzahlen

- Unter der zusätzlichen Normierungs-Bedingung m_1 != 0 (falls *x* != 0) ergibt sich eine eindeutige Darstellung der sogenannten **maschinendarstellbaren Zahlen M** zur Basis B:
  $M = \{x \epsilon \mathbb{R} | \, x = \pm 0.m_1 m_2 m_3 ... m_n \cdot B^{\pm e_1 e_2 ... e_l}\} \cup \{0\}$
  Dabei gilt m_i, e_i € {0, 1, …, B-1] für i != 0 und B € N, B > 1
- Der **Wert** einer solchen Zahl ist definiert als
  $\sum_{i=1}^{n} m_i B^{e-i}$ und ergibt gerade die (nicht normierte) Darstellung der Zahl im Dezimalsystem. Dabei ist *e* ebenfalls im Dezimalssystem zu nehmen, also
  $e = \sum_{i=1}^{l} e_i B^{l-i}$ und es gilt e € Z, d.h. *e* kann natürlich auch negativ sein. Weiter gibt es eine obere und untere Schranke: e_{min} <= *e* <= e_{max}
- Man redet dann auch von einer **n-stelligen Gleitpunktzahl zur Basis B** (engl. Floating point). Zahlen, die nicht in dieser Menge *M* liegen, müssen durch Rundung in eine maschinendarstellbare Zahl umgewandelt werden

2.2 Umrechnung zwischen den Basen

2.2.1 Umrechnung von einer beliebigen Basis ins Dezimalsystem

Für die Umwandlung einer Gleitkommazahl mit Basis B in die zugehörige Dezimalzahl, empfiehlt es sich den ganzzahligen Anteil (Integer) und den Dezimalteil (nach dem Dezimalpunkt) getrennt zu betrachten. Dabei verwendet man das Horner-Schema.

Für den ganzzahligen Anteil: Der Faktor 2 wird fortlaufend mit den Koeffizienten m_i (i = 1, 2, …, e) multipliziert und von «links nach rechts» aufsummiert (mit Start beim innersten Klammerausdruck)

➔ 11001 = (((1*2 + 1) * 2 + 0) * 2 + 0) * 2 + 1 = 25

Oder analog mit dem Schema

	2^4	2^3	2^2	2^1	2^0
	1	1	0	0	1
$B = 2$	↓	2	6	12	24
	1	3	6	12	**25**

Für den Dezimalanteil: Der Faktor 2^{-1} bzw. ½ bzw. 0.5 wird fortlaufend mit den Koeffizienten m_i (i = *e* + 1, *e* + 2, …, n) multipliziert und von «rechts nach links» aufsummiert (mit dem Start beim innersten Klammerausdruck):

➔ .1011 = ½ * (1 + (½) (0 + (½) (1 + 1 * (½)))) = 0.6875

Oder analog mit dem Schema

	$(\frac{1}{2})^4$	$(\frac{1}{2})^3$	$(\frac{1}{2})^2$	$(\frac{1}{2})^1$	$(\frac{1}{2})^0$
	1	1	0	1	0
$B = \frac{1}{2}$	↓	0.5	0.75	0.375	0.6875
	1	1.5	0.75	1.375	**0.6875**

2.2.1 Umrechnung vom Dezimalsystem in andere Zahlensysteme

Für die Umwandlung von dezimal in ein anderes Zahlensystem, kann das umgekehrte Hornerschema verwendet werden.

Von Dezimal zu Binär

Auch hier wird die Zahl in den ganzzahligen- und Nachkommateil gesplittet.

Für den ganzzahligen Teil geht man wie folgt vor:

1. Teilen Sie die Zahl durch 2 und notieren sich den Rest (0 oder 1)

2. Nehmen Sie das Resultat der vorherigen Division und wiederholen den Vorgang bis die Division durch 2 Null ergibt

3. Die Ziffernfolge für den Rest ergibt von unten nach oben die binäre Darstellung der Zahl

```
1006:2  =  503  Rest:  0 ↑
 503:2  =  251  Rest:  1 |
 251:2  =  125  Rest:  1 |
 125:2  =   62  Rest:  1 |
  62:2  =   31  Rest:  0 |
31 : 2  =   15  Rest:  1 |
15 : 2  =    7  Rest:  1 |
 7 : 2  =    3  Rest:  1 |
 3 : 2  =    1  Rest:  1 |
 1 : 2  =    0  Rest:  1 |
```

Für den Nachkommateil geht man wie folgt vor:

1. Multiplizieren Sie die Zahl mit der Basis 2 und notieren Sie sich die Zahl vor dem Komma

2. Falls diese 1 wird, schneiden Sie sie weg bis der Rest 0 ist, der Rest wiederholt oder die gewünschte Genauigkeit erreicht ist

3. Die Ziffernfolge für den Rest ergibt von oben nach unten die binäre Darstellung der Zahl

$$
\begin{array}{lll}
2 \cdot 0{,}687 = 1{,}374 & \text{Ziffer} & 1 \\
2 \cdot 0{,}374 = 0{,}748 & \text{Ziffer} & 0 \\
2 \cdot 0{,}748 = 1{,}496 & \text{Ziffer} & 1 \\
2 \cdot 0{,}496 = 0{,}992 & \text{Ziffer} & 0 \\
2 \cdot 0{,}992 = 1{,}984 & \text{Ziffer} & 1 \\
2 \cdot 0{,}984 = 1{,}968 & \text{Ziffer} & 1 \\
2 \cdot 0{,}968 = 1{,}936 & \text{Ziffer} & 1 \\
2 \cdot 0{,}936 = 1{,}872 & \text{Ziffer} & 1 \\
2 \cdot 0{,}872 = 1{,}744 & \text{Ziffer} & 1 \\
\vdots & \vdots & \vdots
\end{array}
$$

Die beiden Teilergebnisse werden nun zusammengeführt und allenfalls in der Normalendarstellung aufgeschrieben.

Von Oktal oder Hexadezimal in Binär

Das Vorgehen für diese Zahlensysteme ist identisch, jedoch wird anstatt die 2, die entsprechende Basis verwendet.

Oktal → Division / Multiplikation 8

Hexadezimal → Division / Multiplikation 16

2.1 Approximations- und Rundungsfehler

Die Maschinenzahlen sind nicht gleichmässig verteilt. Es gibt bei jedem Rechner eine grösste (x_{max}) und kleinste positive Maschinenzahl (x_{min}). Dabei gilt für normalisierte Gleitpunktzahlen:

$$
\begin{aligned}
x_{max} &= B^{e_{max}} - B^{e_{max}-n} = (1 - B^{-n})B^{e_{max}} \\
x_{min} &= B^{e_{min}-1}
\end{aligned}
$$

Zahlen, die ausserhalb des Rechenbereichs [$-x_{max}$, x_{max}] liegen, sind im Überlaufbereich (**overflow**) und führen zum Abbruch der Rechnung.

Zahlen ungleich 0, die innerhalb des Bereichs [$-x_{min}$, x_{min}] liegen, führen zu einem Unterlauf (**underflow**)

Offensichtlich ist die Anzahl *n* die Mantissestellen von entscheidender Bedeutung für den Bereich der Zahlen, die abgebildet werden können.

2.1.1 Rundungsfehler und Maschinengenauigkeit

Jede reelle Zahl, die von einem Rechner verwendet werden soll, aber keine Maschinenzahl ist, muss durch eine Maschinenzahl ersetzt werden. Dabei entsteht ein Fehler

---

Definition absoluter / relativer Fehler

- Hat man eine Näherung ~x zu einem exakten Wert x, so ist der Betrag der Differenz |~x – x| der **absolute** Fehler
- Falls x != 0, so ist |~x – x / x | der **relative** Fehler dieser Näherung

➔ In der Numerik ist der relative Fehler der wichtigere

Dabei sollte die Maschinenzahl so gewählt werden, dass es möglichst nahe an der reellen Zahl ist, daher ist ein einfaches Abschneiden nicht der geeignete Weg, sondern es sollte die Zahl mittels Rundung ermittelt werden.

Beim runden einer Zahl x wird eine Näherung unter den Maschinenzahlen gesucht, die einen minimalen absoluten Fehler |rd(x) – x| aufweist. Eine n-stellige dezimale Gleitpunktzahl ~x = $0.m_1m_2m_3...m_n*10^e$ = rd(x), die durch die Rundung eines exakten Wertes x entstanden ist, hat also einen absoluten Fehler von höchstens:

$|rd(x) - x| <= 0.00...05*10^e = 0.5*10^{e-n}$ wobei die 5 an der Stelle n + 1 nach dem Dezimalpunkt auftritt. Für beliebige (gerade) Basis gilt analog:

$$\text{falls } B\text{gerade:} \quad | \quad rd(x) - x \mid \leq 0.\underbrace{00...00}_{n}\frac{B}{2}\cdot B^e = \frac{B}{2}\cdot B^{e-n-1}$$

Für die Berechnung bedeutet dies, dass jedes Zwischenergebnis gerundet wird und nicht erst das Endergebnis. Dies hat zur Folge, dass einzelne Rundungsfehler durch die Rechnung weitergetragen werden und allenfalls das Endergebnis verfälschen können. Für den maximal auftretenden relativen Fehler bei der Rundung ergibt sich bei n-stelliger Gleitpunktarithmetik im Dezimalsystem:

Definition Maschinengenauigkeit

- Die Zahl *eps* := 5 * 10^{-n} heisst **Maschinengenauigkeit**.
  Bei allgemeiner Basis B gilt *eps* := B/2 * B^{-n} → Sie gibt den maximalen relativen Fehler, der durch Rundung entstehen kann, an.
- Alternative Definition: Die Maschinengenauigkeit ist die kleinste positive Maschinenzahl, für die auf dem Rechner 1 + *eps* != 1 gilt

  $$\left| \frac{rd(x) - x}{x} \right| \leq 5 \cdot 10^{-n} \; (da \; x \geq 10^{e-1})$$

➔ Der Begriff Maschinengenauigkeit impliziert nicht, dass der Rechner nicht mit deutlich kleineren Zahlen x < *eps* noch «genau» rechnen kann

2.1.2 Fehlerfortpflanzung bei Funktionsauswertungen / Konditionierung

Wenn eine Funktion f: R → R an der Stelle x ausgewertet werden soll, wird ein zusätzlicher Fehler dadurch geniert, das nicht f(x), sondern f(~x) berechnet wird. Für den fehlerbehafteten Wert ~x können wir den Fehler quantifizieren als $\Delta x = \widetilde{x} - x$ oder $\widetilde{x} = x + \Delta x$

Nun wollen wir den absoluten Fehler |f(~x) – f(x)| und den relativen Fehler |f(~x) – f(x)| / |f(x)| dieser Funktionsauswertung berechnen. Unter der Annahme, dass die Funktion f stetig differenzierbar ist, können wir f(~x) gemäss Taylor entwickeln. Aus der allg. Taylor-Reihe einer Funktion f(x) um den Entwicklungspunkt x_0:

$$f(x) = \sum_{i=0}^{\infty} \frac{f^{(i)}(x_0)}{i!}(x - x_0)^i$$

erhalten wir für die Entwicklung von f(~x) um den Entwicklungspunkt x

$$f(\widetilde{x}) = f(x + \Delta x) = \sum_{i=0}^{\infty} \frac{f^{(i)}(x)}{i!}(\Delta x)^i = f(x) + f'(x)\Delta x + \frac{f''(x)}{2}(\Delta x)^2 + \ldots$$

Wobei wir in der Taylor-Reihe x durch ~x und x_0 durch x ersetzt haben.

Unter der Annahme $\Delta x \ll 1$ können die höheren Fehlerterme $(\Delta x)^n$ für $n \geq 2$ vernachlässigt werden und es ergibt sich die folgende Näherung

$$\begin{aligned} f(\tilde{x}) - f(x) &\approx f'(x)\Delta x \\ &\approx f'(x)(\tilde{x} - x) \end{aligned}$$

bzw. bei beidseitiger Division durch $f(x)$ und rechtsseitiger Multiplikation mit $\frac{x}{x}$:

$$\frac{f(\tilde{x}) - f(x)}{f(x)} \approx \frac{f'(x) \cdot x}{f(x)} \cdot \frac{\tilde{x} - x}{x}$$

---

*Näherung für den **absoluten Fehler bei Funktionsauswertung***

$$| f(\tilde{x}) - f(x) | \approx | f'(x) | \cdot | \tilde{x} - x |$$

---

*Näherung für den **relativen Fehler bei Funktionsauswertung***

$$\frac{| f(\tilde{x}) - f(x) |}{| f(x) |} \approx \frac{| f'(x) | \cdot | x |}{| f(x) |} \cdot \frac{| \tilde{x} - x |}{| x |}$$

---

*Definition **Konditionszahl***

$$K := \frac{| f'(x) | \cdot | x |}{| f(x) |}$$

- Die Konditionszahl K ist wie folgt definiert:
- **gut konditionierte Probleme** → Konditionszahl ist klein
- **schlecht konditionierte Probleme** (ill posed problems) → Konditionszahl ist gross

➔ bei gut konditionierten Problemen wird der relative Fehler durch die Auswertung der Funktion nicht grösser

- der absolute Fehler kann in den Funktionswerten nicht grösser sein als in den x-Werten, sondern eher kleiner (da *|f'(x)| = |cos(x)| <= 1*)
- Die Konditionszahl beim Quadrieren (f(x) = x^2) ist $K = \frac{|2x| \cdot |x|}{|x^2|} = 2$ → der relative Fehler verdoppelt sich ca., ist jedoch noch keine schlechte Konditionierung

Fehlerfortplanzung bei Summation

Wenn x und die Konstante c gleiches Vorzeichen haben, gilt K <= 1, dann haben wir also ein gut konditioniertes Problem. Wenn x und x entgegengesetzte Vorzeichen haben und betragsmässig fast gleich gross sind, dann wird | x + c | sehr klein und K somit sehr gross. Die Addition / Subtraktion ist dann schlecht konditioniert. Dieses Phänomen nennt man auch **Auslöschung**. Es tritt immer dann auf, wenn ungefähr gleich grosse fehlerbehaftete Zahlen voneinander abgezogen werden und das Resultat anschliessend normiert wird.

3 Numerische Lösung von Nullstellenproblemen

In diesem Kapitel wird das Verfahren zur näherungsweisen Lösung von nichtlinearen Gleichungen mit einer Unbekannten behandelt.

3.1 Problemstellung

Gegeben sei eine stetige Funktion f: R → R. Gesucht sei ein Näherungswert für die (bzw. für eine) Nullstelle $\overline{x}$ von f. Die Gleichung der Form g(x) = h(x) ist äquivalent zu $f(x) \equiv g(x) - h(x) = 0.$ geometrisch bedeutet dies, dass f(x) an der Stelle $\overline{x}$ die x-Achse schneidet.

Folgende Fragen sollten geklärt werden, bevor ein solches Problem gelöst werden kann:

1. Gibt es überhaupt eine Nullstelle von f(x), und wenn ja, in welchem Bereich?

2. Gibt es mehrere Nullstellen? Wenn ja, welche davon sollten mit dem Rechner gefunden werden?

→ Ein Polynom hat maximal so viele Nullstellen, wie der Grad des Polynoms.

Satz: Nullstellensatz von Bolzano

Sei f: [a, b] → R stetig mit f(a) <= 0 <= f(b) oder f(a) >= 0 >= f(b). Dann muss f in [a, b] eine Nullstelle besitzen

➔ Wenn man also auf dem Intervall [a, b] einen Vorzeichenwechsel von f feststellt, d.h. f(a) * f(b) < 0, dann besitzt f in diesem Intervall mindestens eine Nullstelle.

3.2 Bisektionsverfahren

Das Bisektionsverfahren nutzt den Umstand aus dem Nullstellensatz von Bolzano.

Als erstes wird x_1 = (a+b) / 2 berechnet und überprüft ob f(x_1) > 0 ist.

Wenn ja, wird [a, x1] als neues Näherungsintervall verwendet

Wenn Nein, dann muss eine Nullstelle in [x1, b] liegen

Das neue Intervall nennen wir [a1, b1]. Die Wiederholung des Verfahrens mit dem neuen Intervall liefert eine Intervallschachtelung, die eine Nullstelle bestimmt. Auf diese Weise kann man eine Nullstelle beliebig genau annähern.

Satz: Bisektionsverfahren

- Gegeben sei eine stetige Funktion f:[a, b] → R mit f(a)*f(b) < 0. In jedem der über die Rekursion für i = 0, 1, … erzeugten Intervall

$$[a_0, b_0] = [a, b];$$

$$[a_{i+1}, b_{i+1}] = \begin{cases} \left[a_i, \frac{a_i+b_i}{2}\right] & \text{falls } f\left(\frac{a_i+b_i}{2}\right) \cdot f(a_i) \leq 0 \\ \left[\frac{a_i+b_i}{2}, b_i\right] & \text{sonst} \end{cases}$$

Befindet sich eine Nullstelle von f und es gilt

$$b_i - a_i = \frac{b-a}{2^i}, \text{ insbesondere also } \lim_{i\to\infty}(b_i - a_i) = 0$$

Es gibt wesentlich schnellere Verfahren eine Nullstelle zu berechnen. Jedoch hat das Bisektionsverfahren einige sehr vorteilhafte Eigenschaften:

- Es funktioniert für allgemeine stetige Funktionen
- Es liefert immer ein Ergebnis, vorausgesetzt, dass man geeignete Startwerte a und b finden kann (man sagt das Verfahren «global konvergiert»)
- Die Anzahl der Schritte, nach der die gewünschte Genauigkeit erreicht ist, hängt nur von a und b aber nicht von f ab

3.3 Fixpunktiteration

Die Fixpunktiteration ist eine weitere Methode zur Nullstellenberechnung. Sie beruht auf die Idee, dass für nichtlineare Gleichungen der Form f(x) = F(x) – x die Bedingung $f(\overline{x})$ = 0 genau dann erfüllt ist, wenn $F(\overline{x}) = \overline{x}$.

3.3.1 Vorgehen:

1. F(x) so umformen, dass wir es in der Form x = … haben
2. Geeigneter Startwert wählen (häufig gem. Abbildung)
3. Berechnungen gemäss Anzahl Schritten durchführen

Definition Fixpunktgleichung / Fixpunkt [1]

- Eine Gleichung der Form F(x) = x heisst **Fixpunktgleichung**
- Ihre Lösungen $\overline{x}$, für die $F(\overline{x}) = \overline{x}$ erfüllt ist, heissen **Fixpunkte** (da die Funktion F die Punkte $\overline{x}$ auf sich selbst abbildet

Anstelle eines Nullstellenproblem, kann man ein dazu äquivalentes Fixpunktproblem betrachten.

Definition Fixpunktiteration [1]

- Gegeben sei F:[a, b] → R, mit x0 € [a,b]. Die rekursive Folge $x_{n+1} \equiv F(x_n), \qquad n = 0, 1, 2, \ldots$ heisst Fixpunktiteration von F zum Startwert x0

Satz zur Fixpunktiteration [1]

- Sei F : [a,] → R mit stetiger Ableitung F’ und $\overline{x}$ € [a, b] ein Fixpunkt von F. Dann gilt für die Fixpunktiteration xn+1 = F(xn)
  - Ist |F’($\overline{x}$)|< 1, so konvergiert xn gegen $\overline{x}$, falls der Startwert x0 nahe genug bei $\overline{x}$ liegt. Der Punkt $\overline{x}$ heisst dann **anziehender Fixpunkt**
  - Ist |F’($\overline{x}$)| > 1, so konvergiert xn für keinen Startwert x0 != $\overline{x}$. Der Punkt $\overline{x}$ heisst dann **abstossender Fixpunkt**

Satz Banachscher Fixpunktsatz

- Sei F: [a, b] → [a, b] (d.h. F bildet [a, b] auf sich selber ab) und es exisiere eine Konstante a mit 0 < a < 1 und

|F(x) – F(y)| <= a |x – y| für alle x, y € [a, b] (d.h. F iist «Lipschitz-stetig» und «kontraktiv», a nennt man auch Lipschitz-Konstante). Dann gilt:

- F hat genau einen Fixpunkt $\overline{x}$ in [a, b]
- Die Fixpunktiteration xn+1 = F(xn) konvergiert gegen $\overline{x}$ für alle Startwerte x0 € [a,b]

a-priori & a-posteriori Abschätzungen

- Es gelten die Fehlerabschätzungen

$$|x_n - \overline{x}| \leq \frac{\alpha^n}{1-\alpha}|x_1 - x_0| \quad \text{a-priori Abschätzung}$$
$$|x_n - \overline{x}| \leq \frac{\alpha}{1-\alpha}|x_n - x_{n-1}| \quad \text{a-posteriori Abschätzung}$$

3.3.2 Vorgehen Banachscher Fixpunktsatz

1. Maximum gemäss Aufgabenstellung bestimmen
2. F(x) & F'(x) bestimmen
3. Maximum gemäss Formel (1) berechnen -> Resultat = *alpha*
   → da gemäss Definiton das Maximum von F'(x) = *alpha* ist
   a. Kontrolle, ob Resultat < 1
4. Intervall-Abbildung definieren
   a. Falls monoton steigend Grenzpunkte berechnen
   b. Grenzpunkte kontrollieren

3.4 Newton-Verfahren

Das Newton-Verfahren konvergiert im Vergleich zu den bereits kennenglernten Verfahren deutlich schneller → Es ist quadratisch konvergent. Für die Berechnung ist sowohl die Funktion *f*, wie auch deren Ableitung *f'* notwendig. Aus diesem Grund muss *f* stetig differenzierbar sein (→ sonst gibt es keine Ableitung von *f*).

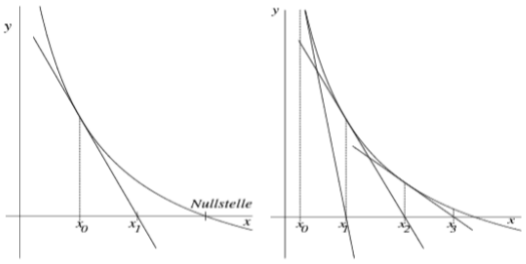
Grundidee: Berechne die Tangente g(x) von *f* im Punkt x_n, d.h. die Gerade $g(x)=f(x_n)+f'(x_n)*(x-x_n)$

Die Nullstelle von g sei x_{n+1}

→ $g(x_{n+1}) = 0 = f(x_n)+f'(x_n)*(x_{n+1}-x_n)$

Die Allgemeine Gleichung erhält man durch auflösen nach x_{n+1} :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad (n = 0, 1, 2, 3, ...)$$



Dies gilt jedoch nur wenn $f'(x_n) \neq 0$ ist. Der Startwert sollte in der Nähe der Nullstelle gewählt werden, um eine schnelle Konvergenz zu erreichen. Wobei die Bedingung für sämtliche Näherungswerte bei $\left|\frac{f(x)\cdot f''(x)}{[f'(x)]^2}\right| < 1$ liegt → Dies sollte zumindest beim Startwert x0 erfüllt sein.

Das Newton-Verfahren ist schnell, jedoch muss man jedes Mal die Ableitung berechnen. Aus diesem Grund gibt es das vereinfachte Newton-Verfahren, sowie das Sekantenverfahren.

3.4.1 Vereinfachtes Newton-Verfahren

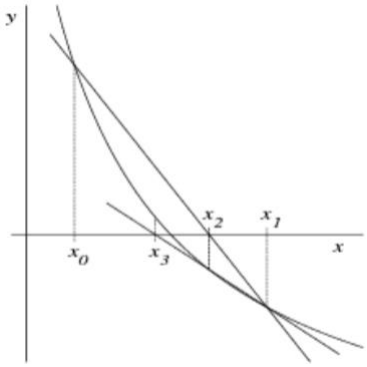
Der Vorteil beim vereinfachten Verfahren ist, dass man nicht für jeden Schritt die Ableitung benötigt. Sondern immer den Wert von $f'(x_0)$ verwenden kann.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_0)} \qquad (n = 0, 1, 2, 3, ...)$$

Der Nachteil bei diesem Verfahren ist, dass es langsamer konvergiert.

3.4.1 Sekantenverfahren

Beim Sekantenverfahren wird nicht der Schnittpunkt der Tangente mit der X-Achse berechnet, sondern der Schnittpunkt von Sekanten («Schneidenden») durch jeweils zwei Punkte $(x_0, f(x_0))$ und $(x_1, f(x_1))$ mit der x-Achse. Statt $f'(x_0)$ wird die Iterationsformel dann die Steigung $\frac{f(x_1) - f(x_0)}{x_1 - x_0}$ der Sekanten eingesetzt und man erhält



$$x_2 = x_1 - \frac{f(x_1)}{\frac{f(x_1)-f(x_0)}{x_1-x_0}} = x_1 - \frac{x_1 - x_0}{f(x_1) - f(x_0)} \cdot f(x_1)$$

Dazu die analoge Iterationsformel

$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} \cdot f(x_n) \qquad (n = 1, 2, 3, ...)$$

3.5 Konvergenzgeschwindigkeit

Die einzelnen Verfahren unterscheiden sich in ihrer Konvergenzgeschwindigkeit. Dies lässt sich durch die Konvergenzordnung vergleichen

---

Definition Konvergenzordnung

Sei (x_n) eine gegen $\overline{x}$ konvergierende Folge. Dann hat das Verfahren die **Konvergenzordnung** q >= 1 wenn es eine Konstante c > 0 gibt mit $| x_{n+1} - \overline{x} | \leq c \cdot | x_n - \overline{x} |^q$ für alle *n*. Falls q = 1 verlangt man noch c < 1. Im Fall q = 1 spricht man von linearer, im Fall q = 2 von quadratischer Konvergenz

3.6 Fehlerabschätzung

Eine einfache Möglichkeit die Näherung auszuwerten, ist zu überprüfen ob ein Vorzeichenwechsel stattfindet. Gemäss dem Nullstellensatz (siehe oben) kann man feststellen ob der Nullpunkt im Intervall liegt. Dieses Verfahren ist auf jedes iterative Verfahren zur Nullstellenbestimmung einer Funktion anwendbar, sofern die Nullstelle eine ungerade Ordnung hat.

Sei xn ein iterativ bestimmter Nährungswert einer extakten Nullstelle ξ (ungerader Ordnung) der stetigen Funktion *f*: R → R und gelte für eine vorgegebene Fehlerschranke / Fehlertoleranz € > 0 $f(x_n - \epsilon) \cdot f(x_n + \epsilon) < 0$ dann muss gemäss dem Nullstellensatz im offenen Intervall (xn - €, xn + €) eine Nullstelle ξ liegen und es gilt die Fehlerabschätzung

4 Numerische Lösung linearer Gleichungssysteme

4.1 Problemstellung

Gesucht ist eine Lösung zu einem linearen Gleichungssystem mit *n* Gleichungen und *n* Unbekannten der Form

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1n}x_n &= b_1 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \ldots + a_{nn}x_n &= b_n \end{aligned}$$

Ein solches Gleichungssystem schreibt man in Matrix-Form als **Ax = b** mit

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \in \mathbb{R}^{n\times n},\ \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^n \text{ und } \boldsymbol{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \in \mathbb{R}^n$$

Matrizen werden in fettgedruckten Grossbuchstaben und Vektoren in fettgedruckten Kleinbuchstaben hervorgehoben.

Für quadratische Matrizen **A** gibt es genau dann eine eindeutige Lösung, wenn die Determinate *det(**A**)* nicht verschwindet (→ **A** ist invertierbar bzw. **A** ist regulär), d.h. wenn eine Matrix $\mathbf{A}^{-1}$ existiert, so dass $\mathbf{A} * \mathbf{A}^{-1} = \mathbf{A}^{-1} * \mathbf{A} = \mathbf{I}$, wobei **I** die *n x n* Einheitsmatrix ist.

Bei numerischen Lösungen von Systemen der unterscheidet man zwischen

- Direkten Verfahren
  - Mit einem direkten Verfahren erhält man mit einer endlichen Zhal von Rechenschritten die exakte Lösung (wenn man Rundungsfehler vernachlässigt)
- Iterativen Verfahren
  - Hier wird eine Folge von Vektoren erzeugt, die gegen die Lösung konvergiert

4.1.1 Direktes Verfahren

Definition Untere Dreiecksmatrix

Eine *n* x *n* Matrix **L** = (l_{ij}) heisst **untere Dreiecksmatrix,** wenn $l_{ij} = 0$ für j > i gilt; sie heisst **normierte untere Dreiecksmatrix,** wenn ausserdem $l_{ij} = 1$ für alle i gilt

$$\boldsymbol{L} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 1 & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 0 \\ l_{n1} & l_{n2} & \cdots & l_{nn-1} & 1 \end{pmatrix}$$

Abbildung 1 untere normierte Dreiecksmatrix

Definition Obere Dreiecksmatrix

Eine *n* x *n* Matrix **R** = (r_{ij}) heisst **obere Dreiecksmatrix,** wenn $r_{ij} = 0$ für i > j gilt; sie heisst **normierte untere Dreiecksmatrix,** wenn ausserdem $r_{ij} = 1$ für alle i gilt

$$\boldsymbol{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & \cdots & r_{1n} \\ 0 & r_{22} & r_{23} & \cdots & r_{2n} \\ 0 & 0 & r_{33} & \cdots & r_{3n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & r_{nn} \end{pmatrix}$$

Abbildung 2 obere Dreiecksmatrix

4.2 Gauss Algorithmus

Der Gauss-Algorithmus basiert auf der Tatsache, dass ein lineares Gleichungssystem **Ax = b** leicht lösbar ist, falls die Matrix **A** als obere Dreiecksmatrix vorliegt, also alle Elemente unterhalb der Diagonalen verschwinden.

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & 0 & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & a_{nn} \end{pmatrix}$$

In diesem Fall kann man für Ax – b mittels der folgenden rekursiven Vorschrift, dem sogenannten Rückwärtseinsetzen, die Komponenten von x berechnen:

$$x_n = \frac{b_n}{a_{nn}},\ x_{n-1} = \frac{b_{n-1} - a_{n-1\,n}\,x_n}{a_{n-1\,n-1}},\ \ldots,\ x_1 = \frac{b_1 - a_{12}x_2 - \ldots - a_{1n}x_n}{a_{11}} \quad \text{bzw.} \quad x_i = \frac{b_i - \sum_{j=i+1}^{n} a_{ij}x_j}{a_{ii}}, \qquad i = n, n-1, \ldots, 1.$$

Mittels dem Gauss-Algorithmus verfolgt man nun die Idee ein beliebiges Gleichungssystem **Ax = b** äquivalent umzuformen ($\tilde{A}$x = b), so dass die Matrix $\tilde{A}$ als obere Dreiecksmatrix vorliegt. Dabei sind folgende Transformationen zulässig:

- Subtraktion eines Vielfaches einer Zeile von einer darunter stehenden Zeile
  $z_j := z_j - \lambda z_i \text{ mit } i < j\ (\lambda \in \mathbb{R})$, wobei z_i die i-te Zeile des Gleichungssystems bezeichnet
- Vertauschen der i-ten und der j-ten Zeile im System z_i → z_j

4.2.1 Programmierung Gauss-Algorithmus

- Zuerst erzeugt man Nullen in der ersten Spalte unterhalb von a_{11} mit der Operation
  $z_j := z_j - \frac{a_{j1}}{a_{11}} z_i$ mit j = 2, … , *n*
  Dies geht nur, falls a_{11} != 0. Ist a_{11} = 0, so vertauschen wir die erste Zeile mit der i-ten Zeile, wobei a_{i1}!= 0 sein muss. Falls alle Zeilen der Matrix in der ersten Zeile eine Null besitzen, funktioniert die Vertauschung nicht. Dann ist allerdings auch die Matrix nicht regulär und die Lösungsmenge kann leer sein oder auch unendlich viele Elemente enthalten.
- Dieser Schritt wird nun wiederholt, in dem man mit der zweiten Spalte fortfährt und unterhalb der diagonalen Nullen erzeugt

➔ Daraus erhält man:

*Gauss-Algorithmus zur Transformation von **Ax = b** auf ein oberes Dreieckssystem*

für $i = 1, ..., n-1$:
erzeuge Nullen unterhalb des Diagonalelementes in der i-ten Spalte

- Falls nötig und möglich, sorge durch Zeilenvertauschung für $a_{ii} \neq 0$:
  falls $a_{ii} \neq 0$: tue nichts
  falls $a_{ii} = 0$: $\begin{cases} \text{falls } a_{ji} = 0 \text{ für alle } j = i+1, ..., n : \\ \quad A \text{ ist nicht regulär; stop;} \\ \text{wenn } a_{ji} \neq 0 \text{ für ein } j = i+1, ..., n : \\ \quad \text{sei } j \geq i+1 \text{ der kleinste Index mit } a_{ji} \neq 0 \\ \quad z_i \longleftrightarrow z_j \end{cases}$
- Eliminationsschritt:
  für $j = i+1, ..., n$ eliminiere das Element a_{ji} durch:

$$z_j := z_j - \frac{a_{ji}}{a_{ii}} \cdot z_i$$

4.3 Fehlerfortpflanzung beim Gauss-Algorithmus und Pivotisierung

Zeilenvertauschung kann auch dazu verwendet werden, um Fehler bspw. durch Gleitpunktoperationen zu minimieren. In jedem Eliminationsschritt wurden die Zeilen bisher mit $\lambda = \frac{a_{ji}}{a_{ii}}$ multipliziert, d.h. der absolute Fehler vergrösserte sich um den Faktor $|\lambda|$

Idealer wäre jedoch wenn man $|\lambda| = |\frac{a_{ji}}{a_{ii}}| < 1$ erreichen würde. Dies ist möglich in dem man vor jedem Eliminationsschritt (bspw. Spaltenvertauschen oder Subtraktion) überprüft, welches Element in der Spalte betragsmässig am grössten ist und die Zeile vertauscht, so dass dieses grösste Element zum Diagonalelement wird. Dies wird Spaltenpivotisierung genannt.

*Gauss-Algorithmus zur Transformation von **Ax = b** mit Spaltenpivotisierung*

für $i = 1, ..., n-1$:
erzeuge Nullen unterhalb des Diagonalelementes in der i-ten Spalte

- Suche das betragsgrösste Element unterhalb der Diagonalen in der i-ten Spalte:
  Wähle k so, dass $|a_{ki}| = \max\{|a_{ji}| \mid j = i, ..., n\}$
  $\begin{cases} \text{falls } a_{ki} = 0 : \ A \text{ ist nicht regulär; stop;} \\ \text{falls } a_{ki} \neq 0 : \ z_k \longleftrightarrow z_i; \end{cases}$
- Eliminationsschritt:
  für $j = i+1, ..., n$ eliminiere das Element a_{ji} durch:

$$z_j := z_j - \frac{a_{ji}}{a_{ii}} \cdot z_i$$

4.4 Dreieckszerlegung von Matrizen

4.4.1 Die LR-Zerlegung

Die Matrix **A** wird in ein Produkt von zwei Matrizen **L** und **R** zerlegt → **A** = **LR**, wobei **R** eine obere Dreiecksmatrix und **L** eine untere normierte Dreieckmatrix ist

$$\boldsymbol{L} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & \cdots & 0 \\ l_{31} & l_{32} & 1 & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 0 \\ l_{n1} & l_{n2} & \cdots & l_{nn-1} & 1 \end{pmatrix}, \boldsymbol{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & \cdots & r_{1n} \\ 0 & r_{22} & r_{23} & \cdots & r_{2n} \\ 0 & 0 & r_{33} & \cdots & r_{3n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & r_{nn} \end{pmatrix}$$

Diese Zerlegung wird als **LR-Faktorisierung** oder **LR-Zerlegung** bezeichnet. Das ursprüngliche Gleichungssystem **Ax = b** lautet dann **LRx = b** ⇔ **Ly = b** und **Rx = y** und lässt sich in zwei Schritten lösen:

1. Zunächst löst man das Gleichungssystem **Ly = b**. Dies kann, analog zum Rückwärtseinsetzen durch Vorwärtseinsetzen geschehen
   $$y_i = \frac{b_i - \sum_{j=1}^{i-1} l_{ij} y_j}{l_{ii}}, , \qquad i = 1, 2, ..., n$$
2. Anschliessend löst man durch Rückwärtseinsetzen das Gleichungssystem **Rx = y**.
   Dann gilt **Ax = LRx = Ly = b** womit das System **Ax = b** gelöst ist.

Im Folgenden gehen wir davon aus, dass Zeilenvertauschungen nicht notwendig sind. Der Gauss-Algorithmus lässt sich dann so erweitern, dass damit eine **LR**-Zerlegung einer invertierbaren Matrix **A** möglich ist. Es gilt:

- **R** ist gerade die durch den Gauss-Algorithmuss auf die obere Dreiecksform gebrachte Matrix $\widetilde{\boldsymbol{A}}$
- Die Elemente l_{ji} von **L** entsprechen gerade den berechneten Faktoren λ aus den Eleminationsschritten $z_j := z_j - \lambda_{ji} z_i$, also $l_{ji} = \lambda_{ji}$

4.4.2 Vorgehen

1. Normale Anwendung des Gauss-Algorithmus für A → man erhält **R (obere Dreiecksmatrix)**
2. Aufschreiben der Einheitsmatrix
3. Angewendete Rechenschritte für Berechnung von **R** für die Einheitsmatrix anwenden, jedoch (**wichtig**) *(-1) gerechnet → + wird zu – und – zu + → man erhält **L (untere normierte Dreiecksmatrix)**

Satz: LR-Zerlegung

Zu jeder regulären *n* x *n* Matrix **A**, für die der Gauss-Algorithmuss ohne Zeilevertauschung durchführbar ist, gibt es *n* x *n* Matrizen **L** und **R** mit den folgenden Eigenschaften:

- **L** ist eine normierte untere Dreiecksmatrix (also mit $l_{ii} = 1$ für *i* = 1, …, n)
- **R** ist eine obere Dreiecksmatrix mit r_{ii} != 0 für *i* = 1, …, n
- **A = L * R** ist die **LR-Zerlegung** von **A**

Aufwand: Die Berechnung der **LR**-Zerlegung mit dem Gauss-Algorithmus benötigt 1/3 (n^3 – n) Punktoperationen

Bemerkungen:

- Die direkte Lösung von **Ax = b** durch die Berechnung der inversen **A**-1 ist nicht praktikabel, da dies die Lösung von *n* linearen Gleichungssystemen erfordern würde und damit erheblich aufwendiger wäre.
- Verwandter Algorithmus von **LR-**Zerlegung wird auch teilweise als Benchmark für die Rechengeschwindigkeit angewendet
- Unteranderem ist die **LR-**Zerlegung eine geschickte Variante, die Zwischenresultate des Gauss-Algorithmus zu speichern

Die LR-Zerlegung mit Zeilenvertauschung

Sind Zeilenvertauschungen notwendig, so lässt in der Regel keine **LR-**Zerlegung erhalten. Allerdings lässt sich die Vertauschung der i-ten und j-ten Zeile in **A** durch eine Multiplikation von links der Form

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

Erreichen, wobei 0 gerade an der i-ten und j-ten Stelle auf der Diagonale steht. Die 1 erscheint dann in der i-ten Zeile und j-ten Spalte sowie der j-ten Zeile und der i-ten Spalte.

4.4.3 Die Cholesky-Zerlegung

Die Cholesky-Zerlegung ist ein Spezialfall der **LR-**Zerlegung. Die Cholesky-Zerlegung funktioniert nur für *symmetrische, positiv definite* Matrizen. Jedoch falls es anwendbar ist, ist es um ca. den Faktor zwei effizienter als die allgemeine **LR-**Zerlegung.

Definition: Symmetrische / positiv definite Matrizen

- Eine Matrix **A** € R^{nxn} heisst symmetrisch falls $\mathbf{A}^T$ = A gilt ($\mathbf{A}^T$ = transponierte Matrix)
- Eine Matrix **A** € R^{nxn} heisst positiv definit, falls für alle **x** € R^n, x != 0 gilt x^T **Ax** > 0

Satz: Cholesky Zerlegung

Für jede positive definite *n* x *n* Matrix **A** gibt es genau eine rechts-obere Dreiecksmatrix **R** mit $r_{ii} > 0$ für i = 1, …, n und $\mathbf{A} = \mathbf{R}^T\mathbf{R}$. Diese Zerlegung heisst **Cholesky-Zerlegung** von **A.**

Der Cholesky-Algorithmus

Gegeben sei eine symmetrische *n* x *n* Matrix **A**. Für i = 1, …, n berechne:

$$S = a_{ii} - \sum_{k=1}^{i-1} r_{ki}^2 \text{ (für } i = 1 \text{ ist also } S = a_{ii})$$

-
- Falls S <= 0 , dann ist **A** nicht positiv definit → STOPP
- Falls S > 0:

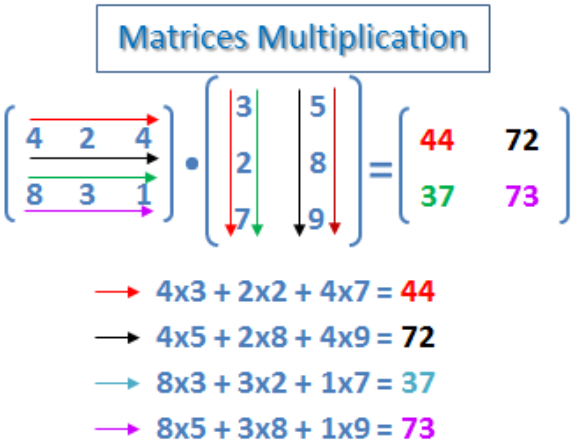
$$r_{ii} = \sqrt{S}$$

$$\text{für } j = i+1, \ldots, n: \; r_{ij} = \frac{1}{r_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj} \right)$$

4.4.4 Zusammengefasstes Vorgehen für 3x3 Matrix A = (a_{ij})

$$\boldsymbol{R} = \begin{pmatrix} \sqrt{a_{11}} & \frac{a_{12}}{r_{11}} & \frac{a_{13}}{r_{11}} \\ 0 & \sqrt{a_{22} - r_{12}^2} & \frac{1}{r_{22}} \left(a_{23} - r_{12} r_{13} \right) \\ 0 & 0 & \sqrt{a_{33} - r_{13}^2 - r_{23}^2} \end{pmatrix}$$

Matrixen-Multiplikation



4.5 Fehlerrechnung und Aufwandabschätzung

4.5.1 Fehlerrechnung bei LGS

Wie aus Kapitel 2 bekannt, kann der Computer nicht alle reelle Zahlen darstellen, deswegen wird intern gerundet. Dadurch entstehen Rundungsfehler. Diese können selbst dann auftreten, wenn der Eingabewert und das Ergebnis eines Algorithmus maschinell darstellbare Zahlen sind → auch die Zwischenergebnisse (potenziell nicht maschinendarstellbare Zahlen) eines Algorithmus werden gerundet.

Aus diesem Grund wird durch einen Algorithmus üblicherweise nicht die exakte Lösung *x* des linearen Gleichungssystems (**Ax = b**) berechnet, sondern nur eine Näherungslösung $\tilde{x}$. Man führt ein «benachbartes» / «gestörtes» Gleichungssystem $A\tilde{x} = b = b + \triangle b$ ein. Wobei $\tilde{x}$ die exakte Lösung ist.

$\triangle b$ nennt man *Residuum* oder *Defekt* der Näherungslösung → Betrag von b ist klein

$\triangle x = \tilde{x} - x$ nennt man den *Fehler der Näherungslösung*

Ziel ist es von $\triangle b$ auf $\triangle x$ zu schliessen.

Definition: Vektornorm

Eine Abbildung ||.|| R^n → R heisst Vektornorm, wenn die folgenden Bedingungen für alle $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, $\lambda \in \mathbb{R}$ erfüllt sind:

- ||**x**|| >= 0 und ||**x**|| = 0 ⇔ *x* = 0
- $\|\lambda \boldsymbol{x}\| = |\lambda| \, \|\boldsymbol{x}\|$
- ||**x+y**|| <= ||**x**|| + ||**y**|| «Dreiecksungleichung»

Dabei gibt es drei gebräuchliche Vektornormen:

- Für **<u>Vektoren</u>** ***x*** = $(x_1, x_2, \ldots, x_n)^T$ € R^n gibt es die folgenden Vektornormen

  1-Norm, Summennorm: $\| \boldsymbol{x} \|_1 = \sum_{i=1}^{n} | x_i |$

  2-Norm, euklidische Norm: $\| \boldsymbol{x} \|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$

  ∞-Norm, Maximum-Norm: $\| \boldsymbol{x} \|_\infty = \max_{i=1,\ldots,n} | x_i |$

- Für eine *n* x *n* **<u>Matrix</u> A** € Rnxn sind mit den Vektornormen die folgenden Matrixnormen verbunden, welche die Eigenschaften der Definition *Vektornorm* ebenfalls erfüllen:

  1-Norm, Spaltensummennorm: $\| \boldsymbol{A} \|_1 = \max_{j=1,\ldots,n} \sum_{i=1}^{n} | a_{ij} |$

  2-Norm, Spektralnorm: $\| \boldsymbol{A} \|_2 = \sqrt{\rho(\boldsymbol{A}^T \boldsymbol{A})}$

  ∞-Norm, Zeilensummennorm: $\| \boldsymbol{A} \|_\infty = \max_{i=1,\ldots,n} \sum_{j=1}^{n} | a_{ij} |$

Satz: Abschätzung für fehlerbehaftete Vektoren

- Sei $\| . \|$ eine Norm, $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ eine reguläre $n \times n$ Matrix und $\boldsymbol{x}, \tilde{\boldsymbol{x}}, \boldsymbol{b}, \tilde{\boldsymbol{b}} \in \mathbb{R}^n$ mit $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ und $\boldsymbol{A}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}}$. Dann gilt für den absoluten und den relativen Fehler in $\boldsymbol{x}$:
  - $\| \boldsymbol{x} - \tilde{\boldsymbol{x}} \| \leq \| \boldsymbol{A}^{-1} \| \cdot \| \boldsymbol{b} - \tilde{\boldsymbol{b}} \|$
  - $\frac{\|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|}{\|\boldsymbol{x}\|} \leq \| \boldsymbol{A} \| \cdot \| \boldsymbol{A}^{-1} \| \cdot \frac{\|\boldsymbol{b} - \tilde{\boldsymbol{b}}\|}{\|\boldsymbol{b}\|}$ falls $\| \boldsymbol{b} \| \neq 0$
- Die Zahl $\mathrm{cond}(\boldsymbol{A}) = \| \boldsymbol{A} \| \cdot \| \boldsymbol{A}^{-1} \|$ nennt man Konditionszahl der Matrix $\boldsymbol{A}$ bzgl. der verwendeten Norm.

→ Für Matrizen, deren Kondition cond(**A**) gross ist, können sich kleine Fehler im Vektor **b** (bzw. Rundungsfehler im Verfahren) zu grossen Fehlern im Ergebnis **x** verstärken. Man spricht in diesem Fall von schlecht konditionierten Matrizen

Satz: Abschätzung für fehlerbehaftete Matrix

Sei $\| \cdot \|$ eine Norm, $\boldsymbol{A}, \widetilde{\boldsymbol{A}} \in \mathbb{R}^{n \times n}$ reguläre $n \times n$ Matrizen und $\boldsymbol{x}, \widetilde{\boldsymbol{x}}, \boldsymbol{b}, \widetilde{\boldsymbol{b}} \in \mathbb{R}^n$ mit $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ und $\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{x}} = \widetilde{\boldsymbol{b}}$. Falls

$$\operatorname{cond}(\boldsymbol{A}) \cdot \frac{\| \boldsymbol{A} - \widetilde{\boldsymbol{A}} \|}{\| \boldsymbol{A} \|} < 1$$

dann gilt:

$$\frac{\| \boldsymbol{x} - \widetilde{\boldsymbol{x}} \|}{\| \boldsymbol{x} \|} \leq \frac{\operatorname{cond}(\boldsymbol{A})}{1 - \operatorname{cond}(\boldsymbol{A}) \cdot \frac{\|\boldsymbol{A} - \widetilde{\boldsymbol{A}}\|}{\|\boldsymbol{A}\|}} \cdot \left(\frac{\| \boldsymbol{A} - \widetilde{\boldsymbol{A}} \|}{\| \boldsymbol{A} \|} + \frac{\| \boldsymbol{b} - \widetilde{\boldsymbol{b}} \|}{\| \boldsymbol{b} \|} \right)$$

➔ Für den Fall, dass **A** exakt gegeben ist, gilt $\frac{\|A - \tilde{A}\|}{\|A\|} = 0$ und der relative Fehler **x** aus Satz 4.4 reduziert sich auf den relativen Fehler in Satz «Abschätzung für fehlerbehaftete Vektoren»

4.5.2 Aufwandabschätzung

Ein wichtiger Aspekt bei der Analyse numerischer Verfahren ist die Abschätzung, wie viel Aufwand diese Verfahren in der Regel benötigen, um zu dem gewünschten Ergebnis zu kommen. Dies hängt massgeblich von der Leistungsfähigkeit des Computers ab → Zeit ist nicht relevant, sondern die Anzahl Rechenoperationen. Man beschränkt sich auf die Gleitkommaoperationen (Addition, Multiplikation etc)

4.6 Iterative Verfahren

Die bisher betrachteten direkten Verfahren besitzen eine Ordnung von $O(n^3)$. Dieser Rechenaufwand ist für grosse Gleichungssystem zu gross. Die iterativen Verfahren haben eine kleinere Ordnung sprich kleinere Aufwand. Dies hat jedoch die Einschränkung, dass man keine exakte Lösung (abgesehen von den Rundungsfehlern) erwarten kann, sondern eine gewisse Ungenauigkeit in Kauf nehmen muss.

Ausgehend von einem Startwert $\mathbf{x}^{(0)}$ berechnet man mittels Rechenvorschrift F: R^n → R^n iterativ eine Folge von Vektoren $\mathbf{x}^{(k+1)} = F(\mathbf{x}^{(k)})$ mit k = 0, 1, 2, … für k → *unendlich* gegen die Lösung **x** des Gleichungssystems **Ax = b** konvergieren. Sobald die gewünschte Genauigkeit erreicht wurde, wird das Verfahren abgebrochen und das Resultat als Näherung angesehen.

Das Ziel ist nun, dass man diese Ausgangslage als Fixpunktiteration behandelt. Die allgemeine Fixpunktgleichung sieht wie folgt aus: F(x) = x, d.h. man muss die ursprüngliche Gleichung **Ax = b** in eine ähnliche Form bringen. Dafür zerlegen wir die Matrix **A** in **A = L + D + R**

Dabei ist **L** eine untere Dreiecksmatrix ($l_{ii} = 0$), **D** eine Dialogsmatrix und **R** eine obere Dreiecksmatrix mit ($r_{ii} = 0$). Die einfachste Form ist **A = L + D + R**

$$= \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ a_{21} & 0 & 0 & \cdots & 0 \\ a_{31} & a_{32} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{n,n-1} & 0 \end{pmatrix} + \begin{pmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & a_{nn} \end{pmatrix} + \begin{pmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & 0 & a_{23} & \cdots & a_{2n} \\ & & & \ddots & \vdots \\ 0 & 0 & 0 & & \\ \vdots & \vdots & \ddots & \ddots & a_{n-1,n} \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

! ACHTUNG: Matrix **L** und **R** sind nicht die gleichen wie bei der LR-Zerlegung

4.6.1 Jacobi-Verfahren

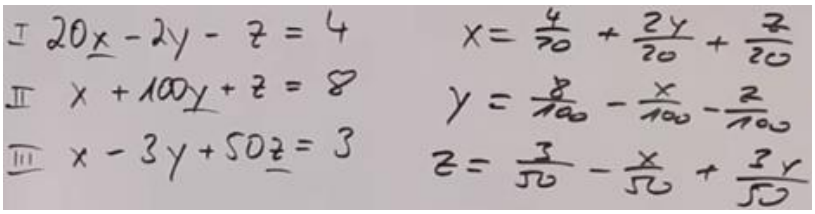
Die obige Zerlegung wird auch Jacobi- oder Gesamtschritt-Verfahren genannt

Das Jacobi-Muster erkennen

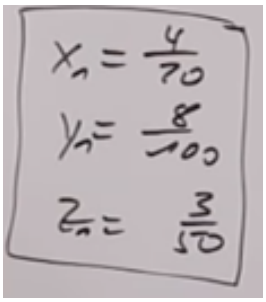
- Anzahl Unbekannte = Anzahl Gleichungen (Zeilen)
- Auf der Hauptdiagonalen sind sehr grosse Werte → Diagonal dominant

Vorgehen

1. Gleichung nach einzelnen Unbekannten auflösen (1 Zeile: x = … / 2. Zeile y = … etc.) → Dadurch, dass wir durch so grosse Werte teilen, stellen wir fest, dass die einzelnen Werten verschwindend klein werden



2. Durch die verschwindend kleine Werte können wir als erste Lösung das erste Paket festlegen → ist natürlich der exakte Wert, jedoch geht man nun iterativ an die Lösung ran



3. Die erhaltenen Werte für x, y, z setzt man nun in das gesamte Paket ein erhält neue Werte
4. Dieser Vorgang widerholt man nun bis man die gewünschte Anzahl an Iterationen durchlaufen hat

---

Definition: Jacobi- bzw. Gesamtschrittverfahren

Zu lösen sei **Ax = b**. Die Matrix **A** = (a_{ij}) sei zerlegt in der Form

$$\boldsymbol{A} = \underbrace{\begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ a_{21} & 0 & 0 & \cdots & 0 \\ a_{31} & a_{32} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn-1} & 0 \end{pmatrix}}_{=:\boldsymbol{L}} + \underbrace{\begin{pmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & a_{nn} \end{pmatrix}}_{=:\boldsymbol{D}}$$

$$+ \underbrace{\begin{pmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & 0 & a_{23} & \cdots & a_{2n} \\ 0 & 0 & 0 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & a_{n-1,n} \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}}_{=:\boldsymbol{R}}$$

Dann heisst die Fixpunktiteration

- $\mathbf{Dx}^{(k+1)} = -(\mathbf{L} + \mathbf{R})\mathbf{x}^{(k)} + b$ bzw.

- $\mathbf{X}^{(k+1)} = -\mathbf{D}^{-1}(\mathbf{L}+\mathbf{R})\mathbf{x}^{(k)} + D^{-1}b$

Gesamtschrittverfahren oder **Jacobi-Verfahren**

Für D ist die inverse D^{-1} sehr einfach zu berechnen, auf der Diagonalen von D^{-1} stehen einfach die Kehrwerte der Diagonalen von D, also

$$\boldsymbol{D} = \begin{pmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & a_{nn} \end{pmatrix} \Rightarrow \boldsymbol{D}^{-1} = \begin{pmatrix} \frac{1}{a_{11}} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{a_{22}} & 0 & \cdots & 0 \\ 0 & 0 & \frac{1}{a_{33}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \frac{1}{a_{nn}} \end{pmatrix}$$

Die Herleitung des Jacobi-Verfahrens folgt direkt aus der Zerlegung von **A**:

$$\begin{aligned} \boldsymbol{A}\boldsymbol{x} &= \boldsymbol{b} \\ (\boldsymbol{L}+\boldsymbol{D}+\boldsymbol{R})\boldsymbol{x} &= \boldsymbol{b} \\ (\boldsymbol{L}+\boldsymbol{R})x + \boldsymbol{D}\boldsymbol{x} &= \boldsymbol{b} \\ \boldsymbol{D}\boldsymbol{x} &= -(\boldsymbol{L}+\boldsymbol{R})x + \boldsymbol{b} \\ \boldsymbol{x} &= -\boldsymbol{D}^{-1}(\boldsymbol{L}+\boldsymbol{R})\boldsymbol{x} + \boldsymbol{D}^{-1}\boldsymbol{b} \equiv F(\boldsymbol{x}) \end{aligned}$$

Womit wir die Fixpunktgleichung bereits aufgestellt haben.

4.6.2 Gauss-Seidel- bzw. Einzelschrittverfahren

Wenn man davon ausgeht, dass das Endresultat nur eine Annäherung ist, so wäre der nächstberechnete Schritt genauer als der vorherige. Aus diesem Grund nimmt setzt man die Komponenten jeweils in die Gleichung ein und erhält dabei eine Gleichung die z.B. wie folgt aussieht.

$$\begin{aligned} x_1^{(k+1)} &= 0.25x_2^{(k)} - 0.25x_3^{(k)} + 1.25 \\ x_2^{(k+1)} &= 0.4x_1^{(k+1)} - 0.2x_3^{(k)} + 2.2 \\ x_3^{(k+1)} &= -0.2x_1^{(k+1)} + 0.4x_2^{(k+1)} + 2.4 \end{aligned}$$

Welche man wiederum in Matrix-Form schreiben kann

$$\boldsymbol{x}^{(k+1)} = \begin{pmatrix} 0 & 0.25 & -0.25 \\ 0 & 0 & -0.2 \\ 0 & 0 & 0 \end{pmatrix} \boldsymbol{x}^{(k)} + \begin{pmatrix} 0 & 0 & 0 \\ 0.4 & 0 & 0 \\ -0.2 & 0.4 & 0 \end{pmatrix} \boldsymbol{x}^{(k+1)} + \begin{pmatrix} 1.25 \\ 2.2 \\ 2.4 \end{pmatrix}$$

Mit den Matrizen **L, D** und **R** wird das zu : $\mathbf{x}^{(k+1)} = \mathbf{D}^{-1}(\mathbf{b} - \mathbf{L}\mathbf{x}^{(k+1)} - \mathbf{R}\mathbf{x}^{(k)})$ oder in der allgemeinen Form

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right) \qquad i = 1, ..., n$$

Durch das Umformen, damit man alle $x^{(k+1)}$ auf der linken Seite hat, erhält man das sogenannte Gauss-Seidel-Verfahren oder auch Einzelschrittverfahren.

Definition: Gauss-Seidel- bzw. Einzelschrittverfahren

Zu lösen sei **Ax = b** Die Matrix **A** = (a_{ij}) sei zerlegt in der Form

$$\boldsymbol{A} = \underbrace{\begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ a_{21} & 0 & 0 & \cdots & 0 \\ a_{31} & a_{32} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn-1} & 0 \end{pmatrix}}_{=:\boldsymbol{L}} + \underbrace{\begin{pmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & a_{nn} \end{pmatrix}}_{=:\boldsymbol{D}}$$

$$+ \underbrace{\begin{pmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & 0 & a_{23} & \cdots & a_{2n} \\ & & & \ddots & \vdots \\ 0 & 0 & 0 & & \\ \vdots & \vdots & \ddots & \ddots & a_{n-1,n} \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}}_{=:\boldsymbol{R}}$$

Dann heisst die Fixpunktiteration

- $(\mathbf{D+L})\mathbf{x}^{(k+1)} = -\mathbf{R}\mathbf{x}^{(k)} + \mathbf{b}$ bzw.
- $\mathbf{x}^{(k+1)} = -(\mathbf{D}+\mathbf{L})^{-1}\mathbf{R}\mathbf{x}^{(k)} + (\mathbf{D}+\mathbf{L})^{-1}\mathbf{b}$

Einzelschrittverfahren oder **Gauss-Seidel-Verfahren**

4.6.3 Konvergenz

Die Kriterien bezüglich der Konvergenz von Fixpunktiterationen können direkt auf die vektoriellen Fixpunktgleichungen des Jacobi- und des Gauss-Seidel-Verfahrens angewandt werden. Es muss dabei nur eine Norm statt des Betragszeichens verwendet werden.

Definition: anziehender / abstossender Fixpunkt

Gegeben sei eine Fixpunktiteration $\mathbf{x}^{(n+1)} = \mathbf{B}\mathbf{x}^{(n)} + b =: F(x^{(n)})$ wobei **B** eine *n x n* Matrix ist und b € R^n. Weiter sei || . || eine der in Kapitel 4.5.1 eingeführten Normen und $\overline{x}$ € R^n erfülle $\overline{x} = \mathbf{B}\,\overline{x} + \mathbf{b} = F(\overline{x})$ Dann heisst

- $\overline{x}$ anziehender Fixpunkt, falls $||\mathbf{B}|| < 1$ gilt
- $\overline{x}$ abstossender Fixpunkt, falls $||\mathbf{B}|| > 1$ gilt

Unter Anwendung des Banachschen Fixpunktsatzes haben wir dann die folgende Fehlerabschätzung zur Verfügung

---

Satz: Abschätzungen

Gegeben sei wie in obiger Definition eine Fixpunktiteration $x^{(n+1)} = \boldsymbol{B}x^{(n)} + \boldsymbol{b} =: F(x^{(n)})$ und $\overline{x}$ € R^n sei eine bezüglich der Norm || . || anziehender Fixpunkt. Dann konvergiert die Fixpunktiteration für alle Startvektoren $x^{(0)}$ € R^n gegen $\overline{x}$ und es gelten die Abschätzungen

$$\| \boldsymbol{x}^{(n)} - \bar{\boldsymbol{x}} \| \leq \frac{\| \boldsymbol{B} \|^n}{1 - \| \boldsymbol{B} \|} \| \boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)} \| \quad \text{a-priori Abschätzung}$$

$$\| \boldsymbol{x}^{(n)} - \bar{\boldsymbol{x}} \| \leq \frac{\| \boldsymbol{B} \|}{1 - \| \boldsymbol{B} \|} \| \boldsymbol{x}^{(n)} - \boldsymbol{x}^{(n-1)} \| \quad \text{a-posteriori Abschätzung}$$

Bemerkung: Der Vergleich mit den Definitionen für das Gesamt- und Einzelschrittverfahren liefert die Matrix **B:**

- Für das Gesamtschrittverfahren (Jacobi) ist $\mathbf{B = -D^{-1}(L + R)}$
- Für das Einzelschrittverfahren (Gauss-Seidel) ist $\mathbf{B = -(D + L)^{-1} R}$

Definition: Diagonaldominanz

A ist eine **diagonaldominate Matrix**, falls eines der beiden folgenden Kriterien gilt:

- Für alle $i = 1, \ldots, n: |\, a_{ii} \,| > \sum_{j=1, j \neq i}^{n} |\, a_{i,j} \,|$ (Zeilensummenkriterium)
- Für alle $j = 1, \ldots, n: |\, a_{jj} \,| > \sum_{i=1, i \neq j}^{n} |\, a_{i,j} \,|$ (Spaltensummenkriterium)

Satz: Konvergenz

Falls **A** diagonaldominant ist, konvergiert das Gesamtschrittverfahren (Jacobi) und auch das Einzelschrittverfahren (Gauss-Seidel) für **Ax = b**

Bemerkungen:

- Die Bedingung $||B|| < 1$ für einen anziehenden Fixpunkt $\bar{x}$ impliziert, dass **A** diagonaldominant ist.
- Diagonaldominanz ist nur ein hinreichendes Kriterium. Es gibt durchaus nicht diagonaldominate Matrizen, für die Verfahren trotzdem konvergieren kann. Ein notwendiges und hinreichendes Kriterium ist, dass der Spektralradius $\rho(\mathbf{B}) < 1$

5 Numerische Lösung nicht linearer Gleichungssysteme

5.1 Funktion mit mehreren Variablen

Für die numerische Lösung von nichtlinearen Gleichungssystemen ist das Verständnis von Funktionen mit mehreren Variablen unabdingbar.

5.1.1 Definition einer Funktion mit mehreren Variablen

Die Definition für Funktionen

$$f: \quad R \rightarrow R$$

$$x \rightarrow y = f(x)$$

mit der abhängigen Variablen y und der unabhängigen Variablen x kennen wir bereits aus der Anaysis und erweitern Sie analog auf Funktionen mit mehreren Variablen

---

Definition: Funktionen mit mehreren Variablen

Unter einer Funktion mit *n* unabhängigen Variablen $x_1, \ldots, x_n$ und einer abhängigen Variablen y versteht man eine Vorschrift, die jedem geordneten Zahlentupel $(x_1, x_2, \ldots, x_n)$ aus einer Definitionsmenge $D \subset \mathbb{R}^n$ genau ein Element y aus einer Wertemenge $W \subset \mathbb{R}$ zuordnet. Die symbolische Schreibweise:

$$\begin{array}{rcl} f: \ D \subset \mathbb{R}^n & \longrightarrow & W \subset \mathbb{R} \\ (x_1, x_2, \ldots, x_n) & \mapsto & y = f(x_1, x_2, \ldots, x_n) \end{array}$$

Da das Ergebnis y € R ein Skalar (eine Zahl) ist, redet man auch von einer **skalarwertigen** Funktion

Bemerkungen:

- Die obige Definition lässt sich einfach erweitern auf beliebige vektorwertige Funktionen, die nicht einen Skalar, sondern einen Vektor als Wert zurückgeben:

$$\boldsymbol{f}: \mathbb{R}^n \longrightarrow \mathbb{R}^m,$$

mit

$$\boldsymbol{f}(x_1, \ldots, x_n) = \begin{pmatrix} y_1 = f_1(x_1, x_2, \ldots, x_n) \\ y_2 = f_2(x_1, x_2, \ldots, x_n) \\ \vdots \\ y_m = f_m(x_1, x_2, \ldots, x_n) \end{pmatrix},$$

  Wobei die *m* Komponenten f_i: $R^n \rightarrow R$ (i = 1, …, m) von f wieder skalarwertige Funktionen sind, entsprechend Defintion oben.
- Wie bei einem Vektor **x** stellen wir zur besseren Unterscheidbarkeit vektorwertige Funktionen **f** fett gedruckt dar, im Gegensatz zu einem Skalar x und einer skalarwertigen Funktion f
- Wir werden uns bei der Lösung nichtlinearere Gleichungssysteme auf vektorwertige Funktion **f:** $R^n \rightarrow R^n$ konzentrieren.

5.1.2 Darstellungsformen

Eine geeignete Darstellungsform ist elementar für die korrekte Interpretation der Resultate.

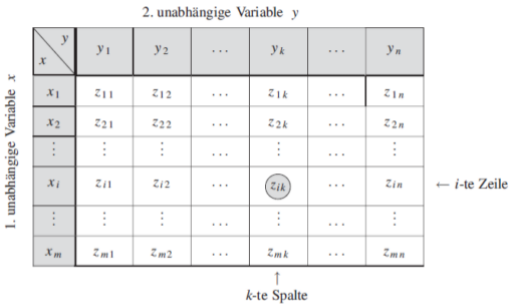
Analytische Darstellung

Die Funktion liegt in Form einer Gleichung vor. Man unterscheidet:

- **Explizite Darstellung:** die Funktionsgleichung ist nach einer Variablen aufgelöst
  $y = f(x_1, x_2, \ldots, x_n)$ Bspw.: $y = 2 \cdot e^{x_1^2 + x_2^2}$
- **Implizite Darstellung:** die Funktionsgleichung ist nicht nach einer Variablen aufgelöst (deshalb handelt es sich hier um eine Funktion mit nur n-1 unabhängigen Variablen)
  $\mathbf{F}(x_1, x_2, \ldots, x_n) = 0$ Bspw. : $x_1^2 + x_2^2 + x_3^2 - 1 = 0$

Darstellung durch Wertetabelle

Betrachten wir den Fall f: $R^2 \rightarrow R$. Setzt man in die als bekannt vorausgesetzte Funktionsgleichung z = f(x, y) für die beiden unabhängigen Variablen x und y der Reihe nach bestimmten Werte ein, so erhält man eine Wertetabelle bzw. Matrix



Grafische Darstellung

In dieser Darstellungsweise beschränken wir uns auf skalarwertige Funktionen mit zwei unabhängigen Variablen f: $R^2 \rightarrow R$ für die es noch anschauliche grafische Darstellungsmöglichkeiten gibt. Dazu betrachten wir die Funktion z = f(x, y) in einem dreidimensionalen kartesischen Koordinatensystem mit den Koordinatenachsen x, y, z

- Darstellung einer Funktion als Fläche im Raum
  Die Funktion f ordnet jedem Punkt(x,y) € D in der Ebene einen Wert z = f(x,y) zu, der als Höhenkoordinate verstanden werden kann. Durch die Anordnung der Punkte (x, y, f(x,y)) im dreidimensionalen Koordinatensystem wird eine über dem Definitionsberech D liegende Fläche ausgezeichnet

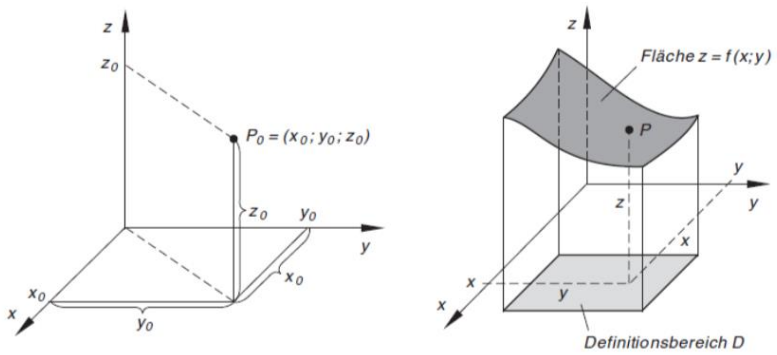


Abbildung 5.3: Links: kartesische Koordinaten eines Raumpunktes. Rechts: Darstellung einer Funktion $z = f(x, y)$ als Fläche im Raum (aus [8]).

- Schnittkurvendiagramm

Wird die Fläche z = f(x,y) bei einer konstanten Höhe z = *const* geschnitten, ergibt sich eine Schnittkurve. Wird diese in die (x,y)-Ebene projiziert, spricht man von einer Höhenlinie bzw. bei der Abbildung von einem Höhenliniendiagramm, wie wird es z.B. von der Wanderkarten her kennen. Natürlich kann man auch andere Schnitte z = *const* (Schnittebene parallel zur (x,y)-Ebene) wählen, z.B. x = *const* (Schnittebene parallel zur (y,z)-Ebene) oder y = *const* (Schnittebene parallel zur (x,z)-Ebene

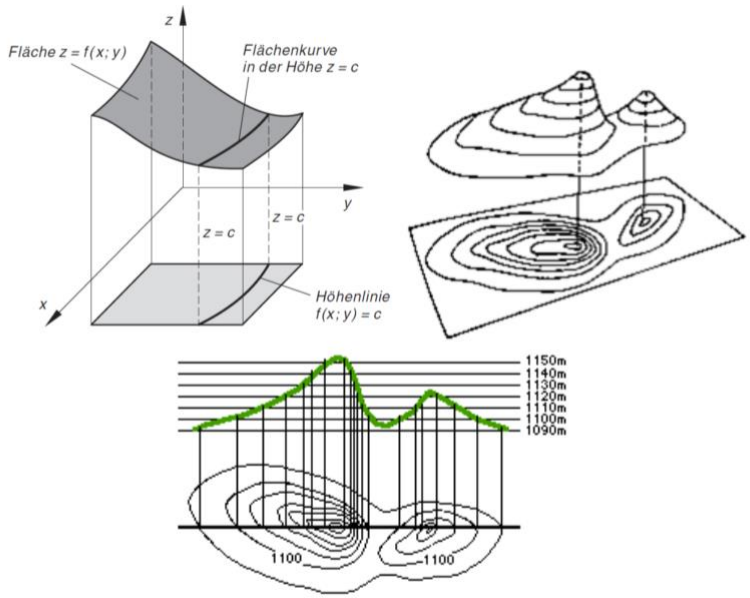


Abbildung 5.4: Zum Prinzip einer Höhenlinie (aus [8]) bzw. eines Höhenliniendiagramms.

5.1.3 Partielle Ableitung

Für eine Funktion f: R→R mit einer Variablen ist die Ableitung an der Stelle x_0 bekanntlich definiert als $f'(x_0) = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}$ aus geometrischer Sicht entsprich dies der Steigung m = f'(x_0) der im Punkt (x_0,f(x_0)) angelegten Kurventangente t mit der Tangentengleichung t(x) = f(x_0) + f'(x_0)(x-x_0)

Bei der Erweiterung der Ableitung mit mehreren unabhängigen Variablen ist die Idee, dass jede unabhängige Variable einzeln zu betrachten ist und sämtliche anderen unabhängigen Variablen «einzufrieren» (→ festgelegter Parameter). Dadurch lässt sich ein n-dimensionales Problem auf n eindimensionale Probleme reduzieren und es kann die obige Ableitung verwendet werden.

Erste Betrachtungsweise

Betrachten wir der Einfachheit halber eine Funktion mit zwei unabhängigen Variablen z = f(x,y) und auf der dadurch definierten Fläche den Punkt P mit den Koordinaten (x_0, y_0, z_0) wobei z_0 = f(x_0, y_0). Wir legen durch den Flächenpunkt P zwei Schnittebenen, die erste verläuft parallel zur (x,z)-Ebene, die zweite zur (y,z)-Ebene.

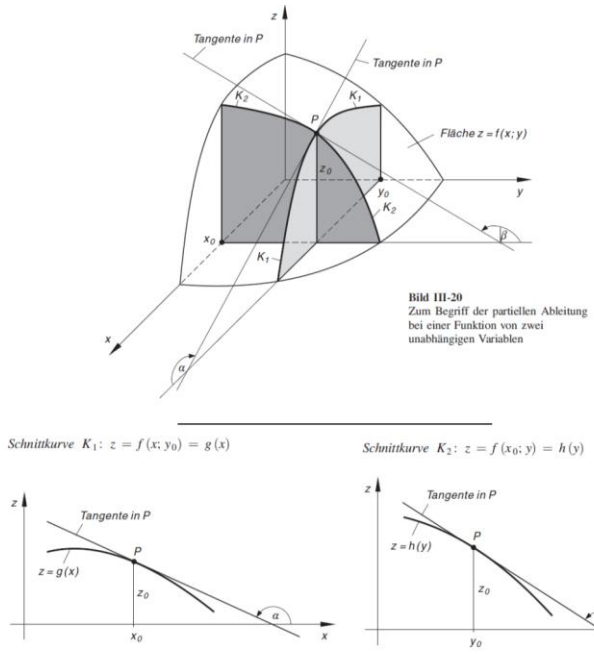


Bild III-20
Zum Begriff der partiellen Ableitung bei einer Funktion von zwei unabhängigen Variablen

Abbildung 5.5: Fläche $z = f(x, y)$ mit dem Punkt $P = (x_0, y_0, z_0)$ und den Schnittflächen sowie den Schnittkurven K_1 und K_2 (aus [8]).

Wir erhalten so durch den Punkt P zwei Schnittkurven K1 und K2 auf der Fläche z= f(x,y). Dbaei hängt die Funktionsgleichung von K_1 nur noch von der Variablen x ab (d.h. für K_1 gilt z = f(x,y_0) =: g(x), denn y_0 ist fixiert). Analog hängt die Funktionsgleichung K_2 nur noch von der Variablen y ab (d.h. für K_2 gilt z = f(x_0, y) =: h(y), denn x_0 ist fixiert). Indem wir nun diese beiden Schnittkurven g(x) = f(x, y_0) und h(y) = f(x_0, y) gemäss unserer bisherigen Definition je einmal an der Stelle x_0 bzw. y_0 ableiten, erhalten wir die Steigung der Tangenten an die Fläche z = f(x,y) im Punkt P, einmal in x-Richtung und einmal in y-Richtung. Konkret berechnen wir die beiden Grenzwerte:

$$g'(x_0) = \lim_{\Delta x \to 0} \frac{g(x_0 + \Delta x) - g(x_0)}{\Delta x} = \lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x, y_0) - f(x_0, y_0)}{\Delta x} =: \frac{\partial f}{\partial x}(x_0, y_0)$$

$$h'(y_0) = \lim_{\Delta y \to 0} \frac{h(y_0 + \Delta y) - h(y_0)}{\Delta y} = \lim_{\Delta y \to 0} \frac{f(x_0, y_0 + \Delta y) - f(x_0, y_0)}{\Delta y} =: \frac{\partial f}{\partial y}(x_0, y_0)$$

Wir bezeichnen diese Grenzwerte als partielle Ableitung 1. Ordnung von f an der Stelle (x_0, y_0)

Definition: Partielle Ableitung 1. Ordnung

Unter der partiellen Ableitungen 1. Ordnung einer Funktion z = f(x,y) an der Stelle (x,y) werden die folgenden Grenzwerte verstanden (falls sie vorhanden sind):

- Partielle Ableitung 1. Ordnung nach x:

$$\frac{\partial f}{\partial x}(x, y) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x, y) - f(x, y)}{\Delta x}$$

- Partielle Ableitung 1. Ordnung nach y:

$$\frac{\partial f}{\partial y}(x, y) = \lim_{\Delta y \to 0} \frac{f(x, y + \Delta y) - f(x, y)}{\Delta y}$$

Evlt. Bemerkungen Seite80 ergänzen

5.1.4 Linearisierung von Funktionen mit mehreren Variablen

Ausgehend von eindimensionalen Fall haben wir für die an eine Funktion y = f(x) im Punkt (x0, f(x0)) eine angelegte Kurventangente g die Tangentengleichung g(x) = f(x0) + f'(x0)(x-x0) . Wir wissen dass wir durch die lineare Tangente die Funktion annähern können. Nun wollen wir die Funktion mit mehreren Variablen erweitern und verwenden dazu die Jacobi-Matrix ein

---

Definition: Jacobi-Matrix / Linearisierung / Tangentialebene

- Sei $\boldsymbol{f} : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ mit $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = \begin{pmatrix} y_1 = f_1(\boldsymbol{x}) \\ y_2 = f_2(\boldsymbol{x}) \\ \vdots \\ y_m = f_m(\boldsymbol{x}) \end{pmatrix}$ und $\boldsymbol{x} = (x_1, x_2, ..., x_n)^T \in \mathbb{R}^n$. Die **Jacobi-Matrix** enthält sämtliche partiellen Ableitung 1. Ordnung von $\boldsymbol{f}$ und ist definiert als

$$\boldsymbol{Df}(\boldsymbol{x}) := \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x) & \frac{\partial f_1}{\partial x_2}(x) & \cdots & \frac{\partial f_1}{\partial x_n}(x) \\ \frac{\partial f_2}{\partial x_1}(x) & \frac{\partial f_2}{\partial x_2}(x) & \cdots & \frac{\partial f_2}{\partial x_n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1}(x) & \frac{\partial f_m}{\partial x_2}(x) & \cdots & \frac{\partial f_m}{\partial x_n}(x) \end{pmatrix}$$

- Die "verallgemeinerte Tangentengleichung"

$$\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}^{(0)}) + \boldsymbol{Df}(\boldsymbol{x}^{(0)}) \cdot (\boldsymbol{x} - \boldsymbol{x}^{(0)})$$

beschreibt eine lineare Funktion und es gilt $\boldsymbol{f}(\boldsymbol{x}) \approx \boldsymbol{g}(\boldsymbol{x})$ in einer Umgebung eines gegebenen Vektors $\boldsymbol{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})^T \in \mathbb{R}^n$. Man spricht deshalb auch von der **Linearisierung** der Funktion $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$ in einer Umgebung von $\boldsymbol{x}^{(0)}$ (ein hochgestellter Index in Klammern $\boldsymbol{x}^{(k)}$ bezeichnet wie bisher einen Vektor aus $\mathbb{R}^n$ nach der $k-$ten Iteration).

- Für den speziellen Fall $f : \mathbb{R}^2 \longrightarrow \mathbb{R}$ mit $y = f(x_1, x_2)$ liefert die linearisierte Funktion

$$g(x_1, x_2) = f(x_1^{(0)}, x_2^{(0)}) + \frac{\partial f}{\partial x_1}(x_1^{(0)}, x_2^{(0)}) \cdot \begin{pmatrix} x_1 - x_1^{(0)} \end{pmatrix} + \frac{\partial f}{\partial x_2}(x_1^{(0)}, x_2^{(0)}) \cdot \begin{pmatrix} x_2 - x_2^{(0)} \end{pmatrix}$$

die Gleichung der **Tangentialebene**. Sie enthält sämtliche im Flächenpunkt $P = (x_1^{(0)}, x_2^{(0)}, f(x_1^{(0)}, x_2^{(0)}))$ an die Bildfläche von $y = f(x_1, x_2)$ angelegten Tangenten.

5.2 Problemstellung zur Nullstellenbestimmung für nichtlineare Systeme

---

Definition: allgemeine Problemstellung zur Nullstellenbestimmung nichtlinearer Gleichungssysteme

---

- Gegeben sei n € N und eine Funktion f: R^n → R^n. Gesucht ist ein Vektor $\bar{x}$ € R^n mit f($\bar{x}$) = 0
- Komponentenweise bedeutet dies: Gegeben sind n Funktionen f_i: R^n → R, die die Komponenten von f bilden. Gesucht ist ein Vektor $\bar{x}$ € R^n mit $f_i(\bar{x})$ = 0 für (i = 1, …, n). Dann heisst $\bar{x}$ € R^n eine Lösung des Gleichungssystems

$$\boldsymbol{f}(x) = \boldsymbol{f}(x_1, ..., x_n) = \begin{pmatrix} f_1(x_1, ..., x_n) \\ f_2(x_1, ..., x_n) \\ \vdots \\ f_n(x_1, ..., x_n) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Bei nichtlinearen Gleichungssysteme gibt es keine einfache Methode, um festzustellen ob und wie viele Lösungen es gibt. Aus diesem Grund ist der Startwert massgeblich entscheidend über Miss- bzw. Erfolg.

5.3 Das Newton-Verfahren für Systeme

In vorherigen Kapiteln haben wir die Nullstellenbestimmung einer Funktion f: R → R mit einer Variablen mit dem Newton-Verfahren hergleitet. Daraus folgte die Iteration

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad (n = 0, 1, 2, 3, ...)$$

Aus der Definition der Jacobi-Matrix und die entsprechende Linearisierung erhalten wir folgende Jacobi-Matrix

$$\boldsymbol{Df}(\boldsymbol{x}) := \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x) & \frac{\partial f_1}{\partial x_2}(x) & \cdots & \frac{\partial f_1}{\partial x_n}(x) \\ \frac{\partial f_2}{\partial x_1}(x) & \frac{\partial f_2}{\partial x_2}(x) & \cdots & \frac{\partial f_2}{\partial x_n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1}(x) & \frac{\partial f_n}{\partial x_2}(x) & \cdots & \frac{\partial f_n}{\partial x_n}(x) \end{pmatrix}$$

Durch die Linearisierung erhalten wir $\boldsymbol{f}(\boldsymbol{x}) \approx \boldsymbol{f}(\boldsymbol{x}^{(n)}) + \boldsymbol{Df}(\boldsymbol{x}^{(n)}) \cdot (\boldsymbol{x} - \boldsymbol{x}^{(n)})$ wobei $x^{(n)}$ den Näherungsvektor nach der n-ten Iteration darstellt. Wenn der Vektor $x^{(n+1)}$ eine Nullstelle von f ist, gilt: $\boldsymbol{f}(\boldsymbol{x}^{(n+1)}) = \boldsymbol{0} \approx \boldsymbol{f}(\boldsymbol{x}^{(n)}) + \boldsymbol{Df}(\boldsymbol{x}^{(n)}) \cdot (\boldsymbol{x}^{(n+1)} - \boldsymbol{x}^{(n)})$ Durch das Auflösung nach $x^{(n+1)}$ erhalten wir dann die Iterationsvorschrift

$$\boxed{\boldsymbol{x}^{(n+1)} = \boldsymbol{x}^{(n)} - \left(\boldsymbol{Df}(\boldsymbol{x}^{(n)})\right)^{-1} \cdot \boldsymbol{f}(\boldsymbol{x}^{(n)})}$$ welche wieder analog zur Vorschrift des eindimensionalen Falls ist.

Wichtig: Es wird nie das Inverse der Jacobi-Matrix berechnet, sondern die obige Gleichung wird zur Lösung eines linearen Gleichungssystems verwendet, in dem man die Substitution $\boldsymbol{\delta}^{(n)} := -\left(\boldsymbol{Df}(\boldsymbol{x}^{(n)})\right)^{-1} \cdot \boldsymbol{f}(\boldsymbol{x}^{(n)})$ als lineares Gleichungssystem aufasst gemäss $\boldsymbol{Df}(\boldsymbol{x}^{(n)})\boldsymbol{\delta}^{(n)} = -\boldsymbol{f}(\boldsymbol{x}^{(n)})$ und so $\boldsymbol{\delta}^{n}$ bestimmen und anschliessend $x^{(n+1)} := x^{(n)} + \boldsymbol{\delta}^{n}$ berechnen kann.

5.3.1 Quadratisch-konvergentes Newton-Verfahren

Das Vorgehen von oben führt zum folgenden Algorithmus

Definition: Newton-Verfahren für Systeme

Gesucht sind Nullstellen von f: R^n → R^n.Sei $x^{(0)}$ ein Startverkor in der Nähe einer Nullstelle. Das Newton-Verfahren zur näherungsweisen Bestimmung dieser Nullstelle lautet:

- Für n = 0, 1, …:
  - Berechne $\boldsymbol{\delta}^{(n)}$ als Lösung des linearen Gleichungssystems $\boldsymbol{Df}(\boldsymbol{x}^{(n)})\boldsymbol{\delta}^{(n)} = -\boldsymbol{f}(\boldsymbol{x}^{(n)})$

- Setze: $x^{(n+1)} := x^{(n)} + \boldsymbol{\delta}^{(n)}$

Bemerkung:

1. mögliche Abbruchkriterien, € > 0

(a) $n \geq n_{max},\ n_{max} \in \mathbb{N}$
(b) $\| \boldsymbol{x}^{(n+1)} - \boldsymbol{x}^{(n)} \| \leq \| \boldsymbol{x}^{(n+1)} \| \cdot \epsilon$
(c) $\| \boldsymbol{x}^{(n+1)} - \boldsymbol{x}^{(n)} \| \leq \epsilon$
(d) $\| \boldsymbol{f}\left(\boldsymbol{x}^{(n+1)}\right) \| \leq \epsilon$

2. Es kann passieren, dass mit dem Newton-Verfahren statt einer Nullstelle von f ein lokales Minimum x_{min} gefunden wird, das ungleich 0 ist. In diesem Falle ist $Df(x_{min})$ aber nicht immer regulär.

Satz: Quadratische Konvergenz des Newton-Verfahrens für Systrem

---

6 Glossar

Fachbegriff	Beschreibung