Skript Numerik I

von Prof. Dr. Luise Blank im WS14/15

Gesina Schwalbe

5. März 2015

Inhaltsverzeichnis

Vorwort

Skriptfehler

An alle, die gedenken dieses Skript zur Numerikvorlesung im WS2014/15 zu nutzen:
Es wird keinerlei Anspruch auf Richtigkeit, Vollständigkeit und auch sicher nicht Schönheit (ich bin LaTeX-Anfänger) dieses Dokuments erhoben.

Ihr würdet mir aber unglaublich weiterhelfen, wenn ihr jede Anmerkung – das kann alles, von groben inhaltlichen Fehlern über Rechtschreibkorrekturen bis hin zu Wünschen/Anregungen/Tipps zur Typografie, sein – an mich weiterleitet!

Jegliche Anmerkungen bitte gleich und jederzeit an:

gesina.schwalbe@stud.uni-regensburg.de

Bilder oder „IMAGE MISSING“

Ich selber bin leider nur wenig mit (ordentlicher) Grafikerstellung in LaTeXvertraut, aber dank Josef Wimmer stehen inzwischen die inhaltsrelevanten Abbildungen zur Verfügung – an dieser Stelle nochmal herzlichen Dank! Die noch fehlenden Grafiken sind mit „IMAGE MISSING“ markiert und wir sind dankbar über jede Mithilfe beim Füllen der Lücken:
Ihr könnt jederzeit **die entsprechenden Bilder an mich schicken**!
Dann werden sie an die entsprechenden Stellen eingebunden bzw. digital abgezeichnet.
Und gerade bei den **Funktionsplots** (z.B. zum Newton-Verfahren) würde ich mich sehr über Plot-Bilder freuen :-)

Copyright

Danksagung

Vielen Dank an

Kerstin Blomenhofer für die fleißige und ordentliche Mitschrift
(und natürlich auch allen anderen, die mir Notizen zur Verfügung gestellt haben)

Josef Wimmer für die Unterstützung bei der Grafikerstellung

Oliver Rümpelein für den Großteil meiner LaTeX-Kenntnisse und die tatkräftige Unterstützung bei allen Fragen zu allen Zeiten

1 Einführung

06.10.2014

Wozu?

Oft sind Probleme mit der gleichen Struktur zu lösen, z.B.

- $ax^2 + bx + c = 0 \Rightarrow x_{\pm} = -\frac{b}{2a} \pm \frac{1}{2a}\sqrt{b^2 - 4ac}$
- Bestimmung des größten gemeinsamen Teilers zweier Zahlen $\rightsquigarrow$ euklidischer Algorithmus

Hierfür ist ein allgemeiner Algorithmus erwünscht.

Ein weiterer Fall ist, dass ein Problem zwar analytisch gelöst werden kann, es aber zu lange dauert bis das Ergebnis bestimmt ist, z.B. tauchen bei der numerischen Simulation von Strömungen bis zu 1 Million Unbekannte auf. Damit sind Systeme mit $\approx 10^6$ Gleichungen zu lösen, wofür effiziente Algorithmen notwendig sind. Näherungslösung sind bei solchen Problemen häufig ausreichend.

Es gibt auch Probleme, die nicht analytisch gelöst werden können, z.B. bei Differentialgleichungen ist vielleicht Existenz- und Eindeutigkeit gewährleistet, aber keine konstruktive Methode zur Berechnung bekannt. Hierfür ist dann eine Näherungslösung gefragt.

Geschichte

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. (Strömungsprobleme modelliert mit partiellen Differentialgleichungen, es stehen Parallelrechner, Vektorrechner, größere Speicher zur Verfügung, 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 Epedemieentwicklungen, Beschreibung des Blutkreislaufs
- Raffinerie-Industrie
- Kontrolle und Optimierung chemischer und biologischer Prozesse, z.B. Regelung von Wärmezufuhr
- u.s.w.

Fragestellungen der Numerik

Rechengeschwindigkeit & -aufwand

Interessant sind Rechengeschwindigkeit, Rechenaufwand (Anzahl der Rechenoperationen, Rechenzeit auf welchem Rechner...) sowie Komplexität des Algorithmus.

Beispiel: Ineffizienz der Cramerschen Regel

Berechnung der Lösung eines Gleichungssystems $Ax = b$ mit einer $n \times n$-Matrix A

1. Cramersche Regel

$$x_j = \frac{\det(A)_j}{\det A} \quad \text{(ersetze die } j\text{-te Spalte von } A \text{ durch } b\text{)}$$

   wobei $\det A = \sum_\pi \operatorname{sign}(\pi) a_{1m_1} \cdot \ldots \cdot a_{nm_n}$. Dieses Verfahren benötigt etwa $n!$ **Multiplikationen und Additionen**. Bei einer 20×20-Matrix A (was heutzutage klein ist) wären dies etwa $2,5 \cdot 10^{18}$ Operationen. Falls jede arithmetische Operation 10^{-6} Sekunden (also eine Mikrosekunde) benötigt, ist eine Rechenzeit von mehr als **eine Millionen Jahre** nötig!

2. Gaußsches Eliminationsverfahren
   Es benötigt etwa n^3 Operationen, also bei einer Matrix wie oben ungefähr 8000 Operationen und weniger als **0,005 Sekunden** [siehe auch GO96].

3. Verhalten bei Störungen, Stabilität des Verfahrens (Eingabefehler, Rundungsfehler, Diskretisierungsfehler)

Beispiel

Bei kleiner Störung von $\frac{1}{10^{-8}} = 10^8$ im Nenner kann es zu einer großen Störung kommen, z.B. $\frac{1}{2 \cdot 10^{-8}} = 5 \cdot 10^7$

Beispiel

Falls die Gleichung

$$x^2 + 314x - 2 = 0$$

mit der p-q-**Formel** (Mitternachtsformel) gelöst und immer auf 5 signifikante Stellen gerundet wird, ergibt sich mit

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

ein relativer Fehler von etwa 57%.

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

$$x_1 = \frac{1}{2a}(-b - \text{sign}(b)\sqrt{b^2 - 4ac})$$
$$x_2 = \frac{2c}{-b - \text{sign}(b)\sqrt{b^2 - 4ac}} \quad .$$

Die tatsächliche Lösung ist 0.0063693, mit p-q-Formel 0.01, mit der letzten Formel 0.0063692.

Genauigkeit des Verfahrens, Fehleranalyse, Konvergenzgeschwindigkeit, Konvergenzordnung

Beispiel

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

$$f(x \pm h) = f(x) \pm hf'(x) + \frac{h^2}{2} f''(x) + R(x)$$

mit $|R(x)| \leq ch^3$

Vorwärtsgenommener Differenzenquotient

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

konvergiert also mit linearer Abhängigkeit der Schrittweite h

Zentraler Differenzenquotient

$$(D_h^0 f)(x) \coloneqq \frac{f(x+h) - f(x-h)}{2 \cdot h} \approx f'(x)$$
$$\left| f'(x) - \frac{f(x+h) - f(x-h)}{2 \cdot h} \right| \leq ch^2$$

konvergiert mit quadratischer **Ordnung** bei **gleichem Aufwand**!

Some desasters 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 Dharan, 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

- Nutze black box solver oder entwickle Lösungsmethode, 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

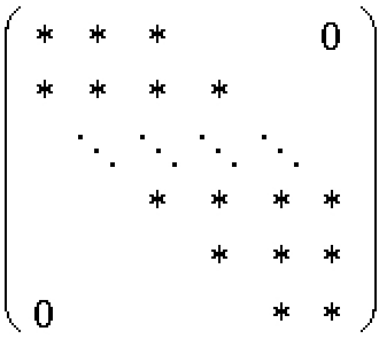
- zuverläßig (fehlerfrei)
- robust (z.B. behandeln Ausnahmesituationen und filtern ungeeignete Daten heraus)
- portierbar auf andere Rechenanlagen
- wartungsfreundlich (leicht zu ändern oder zu erweitern)
- gut dokumentiert
- ausgiebig getestet *(soll in den Übungen trainiert werden)*

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 oder sogar Bandstruktur:

Abb. 1.0.0 *Bandmatrix*



- 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:
   Wir messen den Zusammenhang zwischen Spannung U und Stromstärke I

Abb. 1.0.1 *Einfacher Stromkreis*

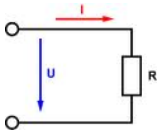
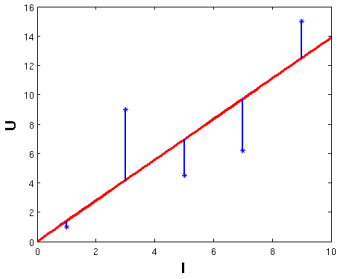


Abb. 1.0.2 *Linearausgleich von Messwerten*



Ohmsches Gesetz: $U = R \cdot I$
Gesucht ist der Widerstand R.
(U_i, I_i) seien die Messdaten mit möglichen Messfehlern.
Finde nun R, sodass $f(R) = \min_r \sum_i (U_i - r\, I_i)^2$

3. **Lösung nichtlinearer Gleichungen**, z.B.
   - Berechnung von Nullstellen $g(x) = 0$

- Berechnung von Fixpunkten $f(x) = x$

4. **Eigenwertwertberechnung** $Ax = \lambda x, \qquad \lambda \in \mathbb{C}$
5. **Interpolation**
   Setze Meßdaten zu einer kontinuierlichen Funktion fort,
   aber wie „glatt“? Z.B. 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 ihre Grundbegriffe werden in einem extra Abschnitt behandelt.

2 Lineare Gleichungssysteme: Direkte Methoden

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

$$A \cdot x = b$$

Weitere Voraussetzungen sind die Existenz und Eindeutigkeit einer Lösung.

Bemerkungen hierzu:

- Ein verlässlicher Lösungsalgorithmus überprüft dies und behandelt alle Fälle.
- Die Cramersche Regel ist ineffizient (s. Einführung 1).
- Das Inverse für $x = A^{-1} \cdot b$ aufzustellen ist ebenso ineffizient, denn es ist keine Lösung für alle $b \in \mathbb{R}^n$ verlangt und der Algorithmus wird evtl. instabil aufgrund vieler Operationen.

$\Rightarrow$ Invertieren von Matrizen vermeiden!!

$\Rightarrow$ Lösen des Linearen Gleichungssystems!!

2.1 Gaußsches Eliminationsverfahren

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

2.1.1 Vorwärtselimination

Das Gaußverfahren gilt der Lösung eines linearen Gleichungssystems der Form

$$Ax = b$$

mit $A = (a_{ij})_{i,j\leq n} \in K^{n\times n}$ Matrix und $b = (b_i)_{i\leq n} \in K^n$ Vektor.
Der zugehörige Algorithmus sieht folgendermaßen aus:

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & & \vdots & & & & \vdots & & \vdots \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array}$$

$$\Downarrow$$

$$(i\text{-te Zeile}) - (1.\text{ Zeile}) \cdot \frac{a_{i1}}{a_{11}} \Rightarrow a_{i1} = 0$$

$$\Downarrow$$

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ & + & a_{22}^{(1)}x_2 & + & \cdots & + & a_{2n}^{(1)}x_n & = & b_2^{(1)} \\ & & \vdots & & & & \vdots & & \vdots \\ & & & & & & a_{nn}^{(1)}x_n & = & b_n^{(1)} \end{array}$$

$$\Downarrow$$
$$\vdots$$

mit

$$a_{ij}^{(1)} = a_{ij} - a_{1j} \cdot \frac{a_{i1}}{a_{11}} \qquad \text{für } i,j = 2,\cdots,n$$
$$b_i^{(1)} = b_i - b_1 \cdot \frac{a_{i1}}{a_{11}} \qquad \text{für } i = 2,\cdots,n$$

In jedem Schritt werden die Einträge der k-ten Spalte analog unterhalb der Diagonalen
08.10.2014 (also $k = 1,\cdots,n-1$) eliminiert:

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

Die Reihe

$$A \to A^{(1)} \to A^{(2)} \to \cdots \to A^{(n-1)}$$

wird bis zum n-ten Schritt fortgeführt, d.h. bis eine obere Dreiecksgestalt eintritt:

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

$$Rx = z \tag{2.1.1}$$

wobei für $i = k+1, \cdots, n$ die Einträge wie folgt aussehen:

$$l_{ik} \coloneqq \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} \tag{2.1.2}$$

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - a_{kj}^{(k-1)} \cdot l_{ik} \qquad \text{für } j = k+1, \cdots, n \tag{2.1.3}$$

$$b_i^{(k)} = b_i^{(k-1)} - b_k^{(k-1)} \cdot l_{ik} \tag{2.1.4}$$

Dieser Prozess wird **Vorwärtselimination** genannt.

Der zugehörige Algorithmus ist:

```
for k = 1, ..., n − 1
|   for i = k + 1, ..., n
|   |   l_ik = a_ik / a_kk
|   |   for j = k + 1, ..., n
|   |   |   a_ij = a_ij − l_ik a_kj
|   |   end
|   |   b_i = b_i − l_ik b_k
|   end
end
```

2.1.2 Rückwärtselimination

Für die Lösung des Gleichungssystems ist dann noch die **Rückwärtssubstitution** nötig:

$$x_n = \frac{b_n^{(n-1)}}{a_{nn}^{(n-1)}} \tag{2.1.5}$$

$$x_{n-1} = \frac{b_{n-1}^{(n-2)} - a_{n-1,n}^{(n-1)} \cdot x_n}{a_{(n-1)(n-1)}^{(n-2)}} \tag{2.1.6}$$

$$x_k = \frac{b_k^{(k-1)} - \sum_{j=k+1}^{n} a_{kj}^{(k-1)} x_j}{a_{kk}^{(k-1)}} \tag{2.1.7}$$

Als Algorithmus:

```
for k = n, n − 1, . . . , 1
|    x_k = b_k
|    for j = k + 1, . . . , n
|    |  x_k = x_k − a_kj x_j
|    end
|    x_k = x_k/a_kk
end
```

Bemerkung 2.1.3. Algorithmen 2.1.1 und 2.1.2 sind nur ausführbar, falls für die sog. **Pivotelemente $\mathbf{a_{kk}^{(k-1)}}$** gilt:

$$a_{kk}^{(k-1)} \neq 0 \quad \text{für } k = 1, \cdots, n$$

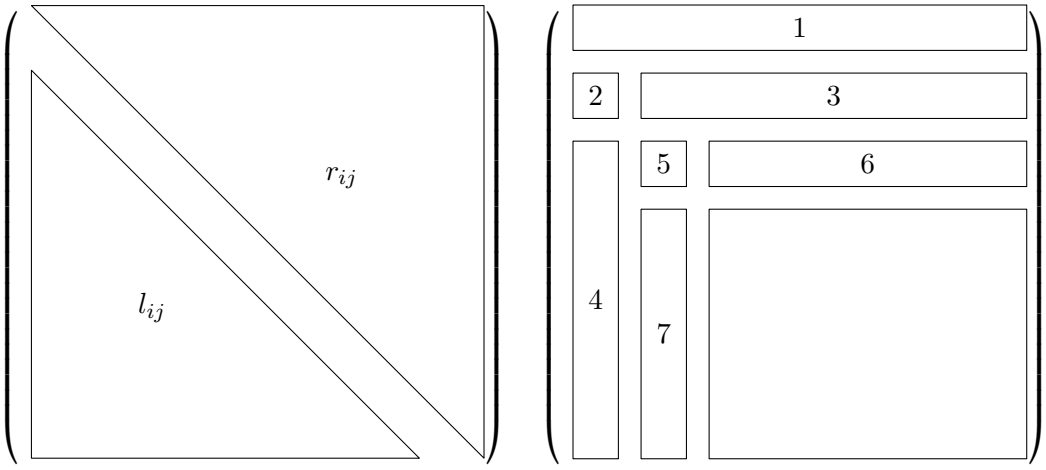
Dies ist auch für invertierbare Matrizen nicht immer gewährleistet.

2.1.4 Weitere algorithmische Anmerkungen

Matrix A und Vektor b sollten möglichst **nie** überschrieben werden! (Stattdessen kann eine Kopie überschrieben werden.) Das Aufstellen von A und b ist bei manchen Anwendungen das teuerste, sie gehen sonst verloren. In 2.1.1 wird das obere Dreieck von A überschrieben. Dies ist möglich, da in (2.1.3) nur die Zeilen $k+1, \cdots, n$ mithilfe der k-ten bearbeitet werden. Am Ende steht R im oberen Dreieck von A und z in b. Die l_{ik} werden spaltenweise berechnet und können daher anstelle der entsprechenden Nullen (in der Kopie) von A gespeichert werden, d.h.:

$$\widetilde{L} \coloneqq (l_{ik}) \tag{2.1.8}$$

und R werden sukzessive in A geschrieben.



Der Vektor z und anschließend der Lösungsvektor x kann in (eine Kopie von) b geschrieben werden. Wird eine neue rechte Seite b betrachtet, muss 2.1.1 nicht komplett neu ausgeführt werden, da sich $\widetilde{L}$ nicht ändert. Es reicht 2.1.4 zu wiederholen.

Definition 2.1.5. Die **Dreieckszerlegung** einer Matrix A entspricht dem Verfahren aus 2.1.1, nur ohne die Zeile (2.1.4).

Definition 2.1.6. Die **Vorwärtssubstitution** entspricht der in 2.1.4 bzw. dem Verfahren aus 2.1.1 ohne die Bestimmung von l_{ik} und R, also nur Schritt (2.1.4).

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

1 Dreieckszerlegung

2 Vorwärtssubstitution $\quad b_i^{(k)} = b_i^{(k-1)} - b_k^{(k-1)} \cdot l_{ik}$

3 Rückwärtssubstitution $\quad x_k = \frac{b_k^{(k-1)} - \sum_{j=k+1}^{n} a_{kj}^{(k-1)} x_j}{a_{kk}^{(k-1)}}$

2.1.8 Rechenaufwand gezählt in „flops“

„flops“ = **fl**oating point **op**eration**s**

1. **Dreieckszerlegung**

   für $j = k+1, \ldots, n$ 1 Addition, 1 Multiplikation für a_{ij}
   für $i = k+1, \ldots, n$ 1 Division zusätzl. für l_{ik}

   Dies ist je für $k = 1, \ldots, n-1$, also ist die Zahl an Additionen und Multiplikationen

$$\begin{aligned}\sum_{k=1}^{n-1} (n-k)^2 &= \sum_{k=1}^{n-1} k^2 \\ &= \frac{(n-1)n(2n-1)}{6} \\ &= \frac{2n^3 - 3n^2 + n}{6}.\end{aligned}$$

   Für große n sind das etwa $\frac{n^3}{3}$ Additionen und Multiplikationen und

$$\sum_{k=1}^{n-1} (n-k) = \frac{n^2 - n}{2} \approx \frac{n^2}{2}$$

Divisionen. Damit ergibt sich eine Gesamtanzahl an flops von

$$2 \cdot \frac{2n^3 - 3n^2 + n}{6} + \frac{n^2 - n}{2} = \frac{2}{3}n^3 - \frac{1}{2}n^2 - \frac{1}{6}n \approx \frac{2}{3}n^3$$

für große n.

2. **Vorwärts- bzw. Rückwärtssubstitution**
   Hier ergeben sich je $\sum_{k=1}^{n-1}(n-k) = \frac{n^2-n}{2} \approx \frac{n^2}{2}$ Multiplikationen und Additionen sowie n Divisionen für die Rückwärtssubstitution und damit insgesamt $n^2 + n$ flops.

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

Definition 2.1.9 (Landau-Symbole)**.** Seien $f, g : D \longrightarrow \mathbb{R}, D \subset \mathbb{R}, -\infty \leq a \leq \infty$ 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 $x \longrightarrow a$, falls

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

(bzw. falls $\lim\limits_{x\to a} \frac{|f(x)|}{|g(x)|} \leq c$)

b) $f(x) = o(g(x))$ für $x \longrightarrow a$, falls $\lim\limits_{x\to a} \frac{|f(x)|}{|g(x)|} = 0$

c) $a_n = \mathcal{O}(b_n)$ für $n \longrightarrow \infty$, 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 und es wird zusätzliche „Kommunikationszeit“ benötigt.

Sortieralgorithmen: Die Indexverwaltung benötigt Zeit, aber keine/kaum Rechenoperationen

If-When-Abfragen: entsprechend

Rechenoperationen liefern jedoch oft eine gute Schätzung.

2.1.11 Formalisieren des Gauß-Algorithmus: LR-Zerlegung

a) **Rückwärtssubstitution:** entspricht $Rx = z$

b) **Vorwärtssubstitution:**

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

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

$$L_k \coloneqq I - l_k e_k^T = \begin{pmatrix} 1 & & & & \\ 0 & \ddots & & 0 & \\ & & 1 & & \\ \vdots & & -l_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & -l_{n,k} & & 1 \end{pmatrix} \tag{2.1.9}$$

dann gilt $b^{(k)} = L_k b^{(k-1)}$, also

$$z = L_{n-1} \cdot L_{n-2} \cdot \ldots \cdot L_1 b$$

Sei

$$L \coloneqq L_1^{-1} \cdot \ldots \cdot L_{n-1}^{-1} \tag{2.1.10}$$

Hiermit folgt dann, dass die Vorwärtssubstitution

$$Lz = b \tag{2.1.11}$$

enspricht.

c) **Dreieckszerlegung:** Wie für die Vorwärtssubstitution ergibt sich

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

und somit $R = L_{n-1} A^{(n-2)} = L_{n-1} \dots L_1 A$ bzw.

$$A = L \cdot R \tag{2.1.12}$$

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:

$$\begin{aligned} L &= L_1^{-1} \cdot \ldots \cdot L_{n-1}^{-1} \\ &= I + \sum_{i=1}^{n-1} l_i e_i^T \\ &= \begin{pmatrix} 1 & & 0 \\ & \ddots & \\ l_{ij} & & 1 \end{pmatrix} \\ &= I + \widetilde{L} \end{aligned} \tag{2.1.13}$$

Hiermit ergibt sich der folgende Satz:

Satz 2.1.13 (LR- oder LU-Zerlegung)**.** *Das obige Verfahren (*(2.1.2) *und* (2.1.3)*) erzeugt unter der Voraussetzung von nicht-nullwertigen Pivotelementen eine Faktorisierung*

$$A = L \cdot R$$

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

Beweis. Zur Eindeutigkeit:

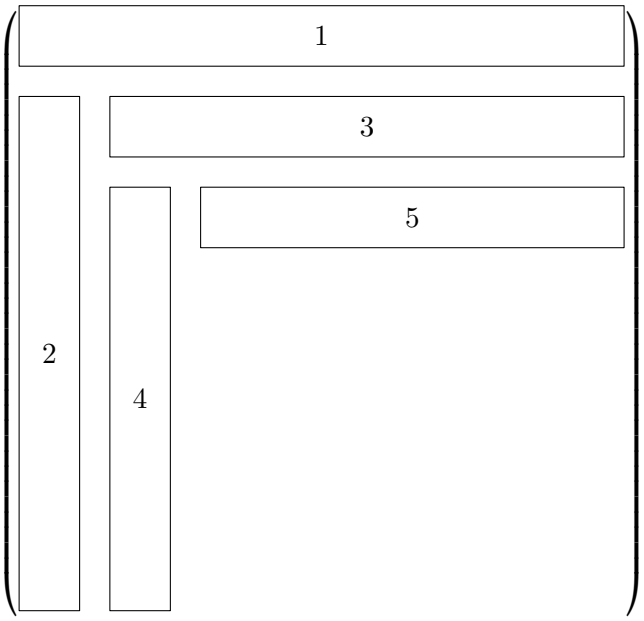
$$A = LR \Leftrightarrow a_{ij} = \sum_{k=1}^{\min j,k} l_{ik} r_{kj}$$

Da $l_{ii} = 1$ gilt, folgt

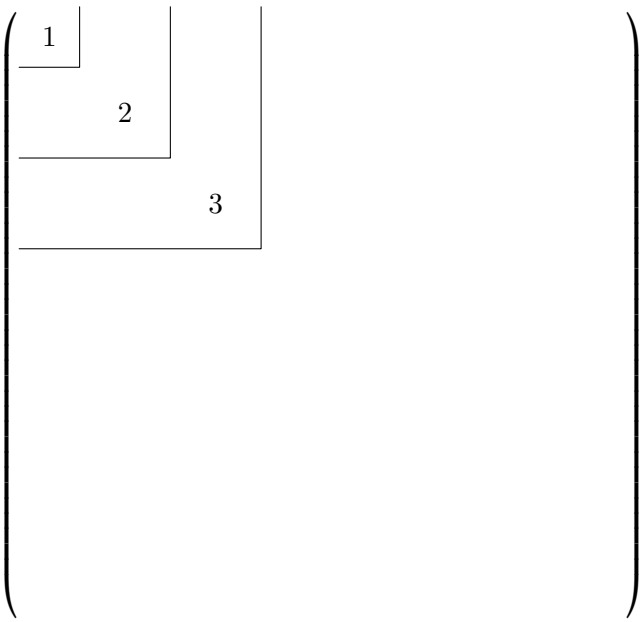
$$\begin{aligned} r_{ij} &= a_{ij} - \sum_{k=1}^{i-1} l_{ik} r_{kj} && \text{für } i \le j \quad \text{und} \\ l_{ij} &= \frac{1}{r_{ij}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} r_{kj} \right) && \text{für } i > j, \end{aligned}$$

da mit A auch R regulär (=invertierbar) ist, und somit $\det(A) = \prod_{j=1}^{n} \neq 0$, also $r_{jj} \neq 0$ gilt. Diese können rekursiv, also eindeutig, berechnet werden, wenn auch die Reihenfolge der Berechnung nicht eindeutig ist. Mögliche Verfahren sind z.B. das **Verfahren von**

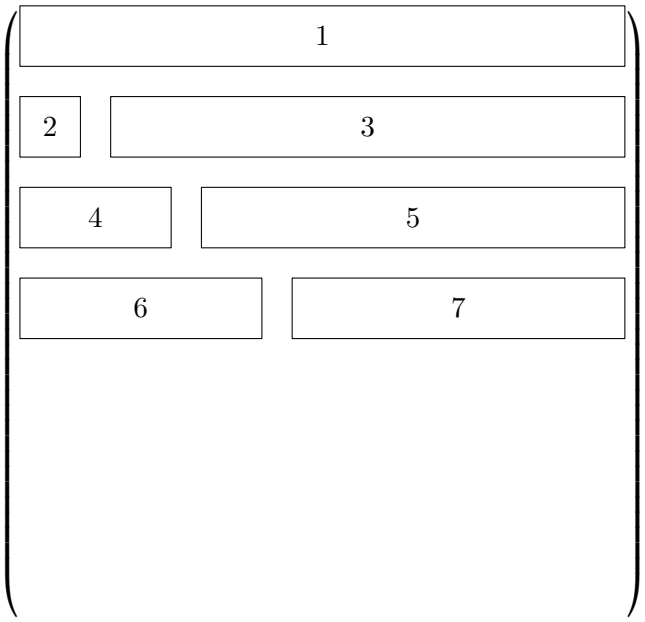
Crout



oder eckweise



oder zeilenweise



oder wie in 2.1.1. □

2.2 Gaußsches Eliminationsverfahren mit Pivotisierung

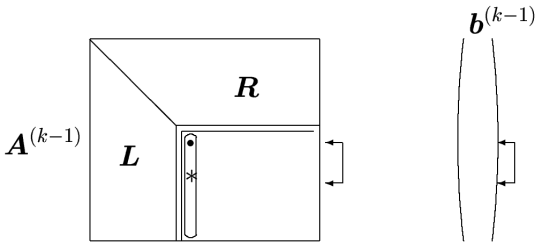
Beispiel Die Matrix $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ ist invertierbar, aber die Gauß-Elimination versagt. Permutiere also die erste mit der zweiten Zeile und der Algorithmus wird anwendbar.

Allgemein Vermeide die Division durch betragsmäßig kleine Zahlen!

2.2.1 Spaltenpivotisierung

Eine Spaltenpivotisierung, auch partielle oder halbmaximale Pivotisierung, erfolgt, indem im k-ten Eliminationsschritt durch Zeilenvertauschen das größte Spaltenelement das Pivotelement stellt:

Abb. 2.2.0 *Ein Schritt der Spaltenpivotisierung*



1. Bestimme das Pivotelement $a_{pk}^{(k-1)}$ als betragsmäßig größtes der „Rest-Spalte“, d.h.
$$|a_{pk}^{(k-1)}| \geq |a_{jk}^{(k-1)}| \qquad \text{für } j = k, \ldots, n$$
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 $|l_{jk}| \ll 1$.

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 unabhängig von der rechten Seite erstellt werden kann, müssen die Permutationen gespeichert werden. Hierfür verwendet man einen sog. **Permutationsvektor** Π, wobei
$$\Pi^{(k-1)}(r) = s$$
bedeutet, dass nach dem $(k-1)$-ten Eliminationsschritt in der r-ten Zeile von $A^{(k-1)}$ die s-te bearbeitete Zeile von A steht, also
$$\begin{aligned} \Pi^{(k)}(k) &= \Pi^{(k-1)}(p) \\ \Pi^{(k)}(p) &= \Pi^{(k-1)}(k) \quad \text{und entsprechend} \\ \Pi^{(k)}(i) &= \Pi^{(k)}(i) \quad \text{für } i \neq k, p \end{aligned}$$

Für die **Permutationsmatrix** P_Π gilt:

$$P_\Pi := (e_{\Pi(1)}, \ldots, e_{\Pi(n)}) \qquad e_j := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \leftarrow j\text{-te Stelle}$$

$$PA = LR$$
$$P^{-1} = P^T \det P_\Pi = \operatorname{sign}(\Pi)$$

2.2.3 Algorithmus: Gauß-Elimination mit Spaltenpivotisierung

Der zugehörige Algorithmus zur Spaltenpivotisierung ist:

```
π(1:n) = [1:n]
for k = 1,...,n-1
 |   bestimme Pivotzeile p, so dass
 |        |a_pk| = max{|a_jk|, j = k,...,n}
 |   π(k) ⇆ [1]π(p)
 |   A(k,1:n) ⇆ A(p,1:n)
 |   if a_kk ≠ 0
 |    |   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
end
```

Satz 2.2.4 (Dreieckszerlegung mit Permutationsmatrix)**.** *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.

$$|l_{ij}| \leq 1 \quad \forall i, j$$

[1] $\leftrightarrows$ bedeutet „vertausche mit“

Beweis. Da $\det A \neq 0$ ist, existiert eine Transposition τ_1, s.d.

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

und

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

Wir erhalten damit

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

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

$$\begin{aligned} \det B^{(1)} &= \frac{1}{\underbrace{a_{11}^{(1)}}_{\neq 0}} \cdot \det A^{(1)} \\ &= \frac{1}{a_{\tau_1,1}^{(1)}} \cdot \det(L_1) \cdot \det(A) \\ &\neq 0\,. \end{aligned}$$

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

Induktiv erhalten wir dann

$$R = A^{(n-1)} = L_{n-1} P_{\tau_{n-1}} \cdot \ldots \cdot L_1 P_{\tau_1} \cdot A$$

Da τ_i nur zwei Zahlen $\geq i$ vertauscht, ist 15.10.2014

$$\Pi_i \coloneqq \tau_{n-1} \circ \ldots \circ \tau_i \quad \text{für } i = 1, \ldots (n-1)$$

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

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

$$P_{\Pi_{i+1}} = (e_1, \ldots e_i, e_{\Pi_{i+1}(i+1)}, \ldots, e_{\Pi_{i+1}(n)}) = \begin{pmatrix} I_i & 0 \\ 0 & P_\sigma \end{pmatrix}$$

Damit folgt:

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

und

$$
\begin{aligned}
R &= L_{n-1} \\
&\quad \cdot (P_{\tau_{n-1}} L_{n-2} P_{\tau_{n-1}}^{-1}) \\
&\quad \cdot (P_{\tau_{n-1}} P_{\tau_{n-2}} L_{n-2} P_{\tau_{n-2}}^{-1} P_{\tau_{n-1}}^{-1}) \\
&\quad \vdots \\
&\quad \cdot (P_{\tau_{n-1}} \cdots P_{\tau_1} L_1 P_{\tau_1} \cdots P_{\tau_{n-1}}) \cdot A \\
&= L_{n-1} \widehat{L}_{n-2} \cdots \widehat{L}_1 P_{\Pi_1} \cdot A
\end{aligned}
$$

Nach Lemma 2.1.12 gilt daher, es existiert eine Permutation Π_1 mit

$$P_{\Pi_1} \cdot A = LR,$$

wobei R obere Dreiecksgestalt hat und

$$
L = \begin{pmatrix} 1 & & & 0 \\ l_{\Pi_2(2),1} & \ddots & & \\ \vdots & \ddots & 1 & \\ l_{\Pi_n(n),1} & \cdots & l_{\Pi_n(n),n-1} & 1 \end{pmatrix} \qquad \text{mit } |l_{ij}| \leq 1
$$

gilt. □

2.2.5 Lösen eines Gleichungssystems $Ax = b$

Das Lösen eines linearen Gleichungssystems der Form $Ax = b$ wird mittels Elimination durch folgende drei Schritte durchgeführt:

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 (allgemeine Formel: $\det(A) = \operatorname{sign}(\Pi) \cdot r_{11} \cdot \ldots \cdot r_{nn}$).

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 $\det(10^{-8}I) = 10^{-8n}$ ist fast 0 für große n, also ist A numerisch singulär.

Beispiel 2.2.7. Wir betrachten die Pivotisierung mit betragsmäßig größtem Spaltenelement und Rundungsfehlern zu

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

Die der Gauß-Elimination mit Rundung auf 3 Dezimalstellen ergibt $l_{21} = 10^4$, denn kleines Pivotelement bedeutet großer Multiplikator.

$$\begin{aligned} r_{22} &= a_{22} - l_{21}a_{12} = 1 - 10^4 \cdot 1 = -999 \approx -10^4 =: \widetilde{r}_{22} \\ b_2^{(1)} &= b_2 - l_{21}b_1 = 2 - 10^4 = -9998 \approx -10^4 =: \widetilde{b}_2^{(1)} \end{aligned}$$

Die Rückwärtssubstitution ergibt

$$\begin{aligned} x_2 &= \frac{-b_2^{(1)}}{r_{22}} = \frac{9998}{9999} \approx 1 \\ \widetilde{x}_2 &= \frac{\widetilde{b}_2}{\widetilde{r}_{22}} = \frac{-10^4}{-10^4} = 1 \end{aligned}$$

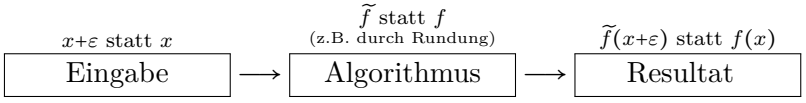
$$\begin{aligned} x_1 &= \frac{1}{r_{11}}(b_1 - r_{12}x_2) = 10^4(1 - 1x_2) = \frac{10^4}{9999} \approx 1 \\ \widetilde{x}_1 &= \frac{1}{\widetilde{r}_{11}}(1 - 1\widetilde{x}_2) = 10^4(1 - 1 \cdot 1) = 0 \end{aligned}$$

Dies führt zu einem starken Fehler.

Mit Spaltenpivotisierung ist $l_{21} = 10^{-4} < 1$ und $\widetilde{r}_{22} = 1$, $b_2^{(2)} = 1$, $\widetilde{x}_2 = 1$, $\widetilde{x}_1 = 1$.
Diese Werte führen auch bei Rundungsfehlern zu besseren Ergebnissen.

3 Fehleranalyse

15.10.2014



Bei der Fehleranalyse liegt das Hauptaugenmerk auf

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 oder die Berechnung von Sinus durch abgebrochene Reihenentwicklung)

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

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

3.1 Zahlendarstellung und Rundungsfehler

Auf (Digital-)Rechnern können nur endlich viele Zahlen realisiert werden. Die wichtigsten Typen sind:

- **ganze Zahlen** (integer):

$$z = \pm \sum_{i=0}^{m} z_i \beta_i \qquad \text{mit} \quad \begin{array}{l} \beta = \text{Basis des Zahlensystems (oft } \beta = 2) \\ z_i \in \{0, \cdots \beta - 1\} \end{array}$$

- **Gleitpunktzahlen** (floating point)

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

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

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

heißt **Gleitkommazahl** mit t Stellen und Exponent e zur Basis b.
Ist $a_1 \neq 0$, so heißt x **normalisierte Gleitkommazahl**.

Bemerkung 3.1.2.

a) 0 ist keine normalisierte Gleitkommazahl, da $a_1 = 0$ ist.

b) $a_1 \neq 0$ stellt sicher, dass die Gleitkommadarstellung eindeutig ist.

c) In der Praxis werden auch nicht-normalisierte Darstellungen verwendet.

d) Heutige Rechner verwenden meist $\beta = 2$, aber auch $\beta = 8, \beta = 16$.

3.1.3 Bit-Darstellung zur Basis 2

Bit-Darstellung nach IEEE-Standard 754 von floating point numbers
Sei die Basis $\beta = 2$.

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

Darstellung im Rechner (Bitmuster) für float:

s	$b_0 \cdots b_7$	$a_2 \cdots\cdots a_{24}$

(Da $a_1 \neq 0$, also $a_1 = 1$ gilt, wird a_1 nicht gespeichert)

Interpretation $(s, b, a_i \in \{0, 1\} \forall i)$

- s Vorzeichenbit: $\sigma = (-1)^s \Rightarrow \begin{matrix} \sigma(0) = 1 \\ \sigma(1) = -1 \end{matrix}$

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

$x = 0$: s bel., $b = 0$, $m = 1$ | s | $0 \ldots 0$ | $0 \ldots 0$ |

$x = \pm\infty$: s bel., $b = 255$, $m = 1$ | s | $1 \ldots 1o$ | $0 \ldots 0$ |

$x =$NaN s bel., $b = 255$, $m \neq 1$

$x = (-1)^s$ s bel., $b = 0$, $m \neq 1$ und x hat die Form $x = (0 + \sum_{\nu=2}^{24} a_\nu \cdot 2^{1-\nu}) \cdot 2^{126}$ („denormalized“ number)

20.10.2014

Betragsmäßig **größte Zahl**:

| 0 | $01 \ldots 1$ | $1 \ldots 1$ |

$$x_{max} = (2 - 2^{-23}) \cdot 2^{127} \approx 3,4 \cdot 10^{38}$$

Betragsmäßig **kleinste Zahl**:

| 0 | $0 \ldots 0$ | $0 \ldots 01$ |

$$x_{min} = 2^{-23} \cdot 2^{-126} = 2^{-149} \approx 1,4 \cdot 10^{-45}$$

3.1.4 Verteilung der Maschinenzahlen

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

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

Abb. 3.1.0 ©

ist im Dualsystem gleichmäßig, jedoch im Dezimalsystem sehr ungleichmäßig verteilt.

Definition 3.1.5.

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

underflow: Entsprechend, betragsmäßig kleiner als die kleinste positive Zahl.

Bsp.: overflow beim integer $b = e + 127$

$$\begin{array}{rrrr} b & = 254 & & 11111110 \\ & + & 3 & 00000011 \\ \hline b + 3 = 257 \bmod 2^8 & = & 1 & \not{1}00000001 \end{array}$$

3.1.6 Rundungsfehler

Habe $x \in \mathbb{R}$ die normalisierte Darstellung

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

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

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

Für die **relative Rechengenauigkeit** folgt somit

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

Die **Maschinengenauigkeit** des Rechners ist daher durch

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

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

$$\tau := \frac{1}{2}\,\text{eps}$$

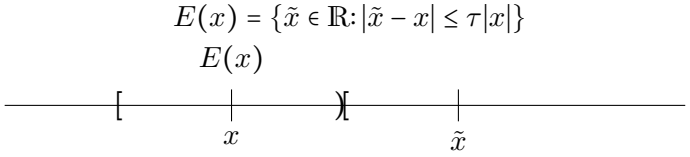
Weiterhin gilt:

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

$$1 + \text{eps}$$

b) Eine Maschinenzahl x repräsentiert eine Eingabemenge

Abb. 3.1.1 ©



Bemerkung 3.1.7. Gesetze der arithmetischen Operationen gelten i.A. nicht, z.B.

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

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

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

3.1.8 Auslöschung von signifikanten Stellen

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

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

Daraus ergibt sich der relative Fehler

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

Der Grund hierfür ist, dass das Problem der Substraktion zweier annähernd gleich großer Zahlen schlecht konditioniert ist.

Zwei Regeln:

1) Umgehbare Substraktion annähernd gleich großer Zahlen vermeiden!

2) Unumgängliche Substraktion möglichst an den Anfang des Algorithmus stellen! (siehe später)

3.2 Kondition eines Problems

Es wird das Verhältnis

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

untersucht.

Definition 3.2.1. Sei $f: U \subseteq \mathbb{R}^n \to \mathbb{R}^m$ mit U offen und sei $x \in U$. 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 $\|\bullet\|$ eine Norm auf $\mathbb{R}^n$.

a) $\|\tilde{x} - x\|$ heißt **absoluter Fehler**

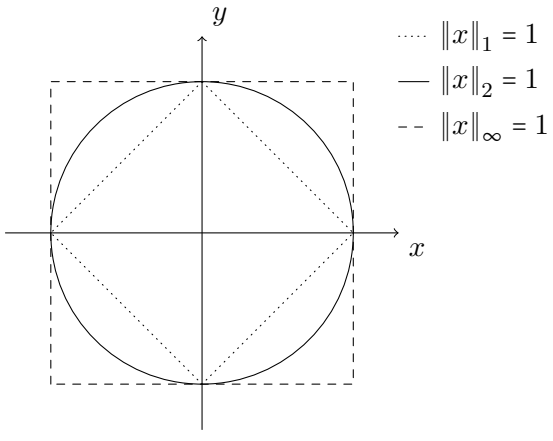
b) $\frac{\|\tilde{x}-x\|}{\|x\|}$ heißt **relativer Fehler**

Da der relative Fehler skalierungsinvariant ist, d.h. unabhänging von der Wahl 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:

$$\begin{aligned} &\text{Für } i = 1, \ldots, n: && |\tilde{x}_i - x_i| \leq \delta && \text{(absolut)} \\ & && |\tilde{x}_i - x_i| \leq \delta |x_i| && \text{(relativ)} \end{aligned}$$

Wiederholung 3.2.3 (Normen)**.**

Abb. 3.2.0 *Sphären mit gleichem Normbetrag*



Summennorm (l_1-Norm): $\|x\|_1 \coloneqq \sum_{i=1}^{n} |x_i|$

Euklidische Norm (l_2-Norm): $\|x\|_2 \coloneqq \sqrt{\sum_{i=1}^{n} |x_i|^2}$

Maximumsnorm (l_∞-Norm): $\|x\|_\infty \coloneqq \max\{|x_i| \mid i = 1, \dots n\}$

Hölder-Norm (l_p-Norm): $\|x\|_p \coloneqq \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}}$

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

$$\|A\|_{a,b} \coloneqq \sup_{x \neq 0} \frac{\|Ax\|_b}{\|x\|_a} = \sup_{\|x\|_a = 1} \|Ax\|_b \tag{3.2.1}$$

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

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

Definition 3.2.5. Sei $A \in \mathbb{R}^{m \times n}$.

a) **Frobeniusnorm** (Schurnorm): $\|A\|_F \coloneqq \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}^2|}$

b) **p-Norm**: $\|A\|_p \coloneqq \|A\|_{p,p}$

c) Eine Matrixnorm heißt **verträglich** mit den Vektornormen $\|\bullet\|_a, \|\bullet\|_b$, falls gilt [1]

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

Bemerkung 3.2.6.

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

$$\|A \cdot B\| \leq \|A\| \cdot \|B\|$$

b) Die Norm $\|\bullet\|_{1,1}$ wird auch **Spaltensummennorm** genannt:

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

Sie ist das Maximum der Spaltensummen[2].

[1] Beachte: $\|A\|_{a,b}$ ist die kleinste Norm im Gegensatz zu $\|A\|$, welche hier beliebig ist.
[2] Beweis: siehe Übungsblatt 3

c) Die Norm $\|\bullet\|_{\infty,\infty}$ wird auch **Zeilensummennorm** genannt[3]:

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

d) Die Frobeniusnorm $\|\bullet\|_F$ ist verträglich mit der euklidischen Norm $\|\bullet\|_2$

e) Die Wurzeln aus den Eigenwerten von A^TA heißen **Singulärwerte** σ_i von A. Mit ihnen kann die $\|\bullet\|_{2,2}$ Norm dargestellt werden[4]:

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

3.2a) Normweise Konditionsanalyse

22.10.2014

Definition 3.2.7. Sei (f,x) ein Problem mit $f{:}U \subset \mathbb{R}^n \to \mathbb{R}^m$ und $\|\bullet\|_a$ auf $\mathbb{R}^n$ und $\|\bullet\|_b$ auf $\mathbb{R}^m$ eine Norm.

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

$$\|f(\tilde{x}) - f(x)\|_b \le \kappa_{abs}(f,x)\,\|\tilde{x} - x\|_a + o(\|\tilde{x} - x\|_a) \tag{3.2.2}$$
$$\Big(f(\tilde{x}) - f(x) = \underbrace{f'(x)(\tilde{x} - x) \pm o(\|\tilde{x} - x\|)}_{Taylorentwicklung} \quad \text{für } \tilde{x} \to x\Big)$$

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

$$\frac{\|f(\tilde{x}) - f(x)\|_b}{\|f(x)\|_b} \le \kappa_{rel}(f,x)\frac{\|\tilde{x} - x\|_a}{\|x\|_a} + o\left(\frac{\|\tilde{x} - x\|_a}{\|x\|_a}\right) \qquad \text{für } \tilde{x} \to x \tag{3.2.3}$$

c) Sprechweise:

- falls κ „klein“ ist, ist das Problem „gut konditioniert“
- falls κ „groß“ ist, ist das Problem „schlecht konditioniert“

[3] Beweis: siehe Übungsblatt 3
[4] Beweis: siehe Übungsblatt 3

Lemma 3.2.8. *Falls f differenzierbar ist, gilt*

$$\kappa_{abs}(f,x) = \|\mathrm{D}\, f(x)\|_{a,b} \tag{3.2.4}$$

und für $f(x) \neq 0$

$$\kappa_{rel}(f,x) = \frac{\|x\|_a}{\|f(x)\|_b} \cdot \|\mathrm{D}\, f(x)\|_{a,b} \tag{3.2.5}$$

wobei $\mathrm{D}\, f(x)$ die Jakobi-Matrix bezeichnet.

Beispiel 3.2.9 (Kondition der Addition)**.** $f(x_1,x_2) \coloneqq x_1 + x_2$, $f\colon \mathbb{R}^2 \to \mathbb{R}$.
Wähle l_1-Norm auf $\mathbb{R}^2$ (und $\mathbb{R}$)

$$\begin{aligned}\mathrm{D}\, f(x_1,x_2) = (\nabla f^T) &= \left(\frac{\partial}{\partial x_1} f, \frac{\partial}{\partial x_2} f\right) \\ &= (1,1) \qquad \text{(Matrix!)}\end{aligned}$$

damit

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

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 Konditionierung:

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

Zum Beispiel in 3.1.8: Es ist

$$\begin{aligned}x &= \begin{pmatrix} 9,995 \\ -9,984 \end{pmatrix} \cdot 10^{-1} \\ \tilde{x} = fl(x) &= \begin{pmatrix} 1 \\ -9,98 \cdot 10^{-1} \end{pmatrix}\end{aligned}$$

also

$$\begin{aligned}\frac{|f(\tilde{x})-f(x)|}{|f(x)|} &= \frac{0,9}{1,1} = 0,\overline{81}\\ &\leq \kappa_{rel}(f,x)\cdot\frac{\|\tilde{x}-x\|_1}{\|x\|_1}\\ &= \kappa_{rel}(f,x)\cdot 4,6\cdot 10^{-4}\end{aligned}$$

Beispiel 3.2.10 (Lösen eines lin. Gleichungssystems)**.** Sei $A\in\mathbb{R}^{n\times n}$ invertierbar und $b\in\mathbb{R}^n$. Es soll

$$Ax = b$$

gelöst werden. Die möglichen Lösungen in A und in b lassen sich folgendermaßen ermitteln:

a) Betrachte die Störungen in b:
Sei hierzu $f\colon b\mapsto x = A^{-1}b$ Berechne dann $\kappa(f,b)$ und löse

$$\begin{aligned}A(x+\Delta x) &= b+\Delta b\\ f(b+\Delta b)-f(b) &= \Delta x\\ &= A^{-1}\cdot\Delta b && \text{da } x = A^{-1}b\\ \Rightarrow \|\Delta x\|_b &= \left\|A^{-1}\Delta b\right\|_b\\ &\leq \left\|A^{-1}\right\|_{a,b}\cdot\|\Delta b\|_b && \forall b,\Delta b\end{aligned}$$

wobei $\|\bullet\|$ auf $\mathbb{R}^{n\times n}$ die dem $\mathbb{R}^n$ zugeordnete Matrix-Norm sei.
Die Abschätzung ist **scharf**, d.h. es gibt ein $\Delta b\in\mathbb{R}^n$, so dass „=“ gilt, nach Definition 3.2.4.
Also gilt[5],

$$\kappa_{abs}(f,b) = \left\|A^{-1}\right\|_{a,b} \tag{3.2.6}$$

unabhängig von b. Ebenso folgt die scharfe Abschätzung

$$\begin{aligned}\frac{\|f(b+\Delta b)-f(b)\|}{\|f(b)\|} &= \frac{\|\Delta x\|}{\|x\|}\\ &= \frac{\left\|A^{-1}\Delta b\right\|}{\|x\|}\\ &\leq \frac{\left\|A^{-1}\right\|\cdot\|b\|}{\|x\|}\cdot\frac{\|\Delta b\|}{\|b\|}\end{aligned}$$

Damit

$$\kappa_{rel}(f,b) = \left\|A^{-1}\right\|\cdot\frac{\|b\|}{\|A^{-1}\cdot b\|} \tag{3.2.7}$$

[5] vgl. auch Lemma 3.2.8: $\kappa_{abs}(f,b) = \|\mathrm{D}\,f(b)\|_{a,b} = \left\|A^{-1}\right\|_{a,b}$

Da $\|b\| \leq \|A\| \cdot \|x\| = \|A\| \cdot \left\|A^{-1}b\right\|$ folgt

$$\kappa_{rel}(f,b) \leq \|A\| \cdot \left\|A^{-1}\right\| \tag{3.2.8}$$

für alle (möglichen rechten Seiten) b.
3.2.8 ist scharf in dem Sinne, dass es ein $\widehat{b} \in \mathbb{R}^n$ gibt mit $\left\|\widehat{b}\right\| = \|A\| \cdot \|\hat{x}\|$ und somit

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

b) Betrachte die Störungen in A:
Löse also

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

Sei hierzu

$$\begin{aligned} f\colon \mathbb{R}^{n\times n} &\to \mathbb{R}^n \\ A &\mapsto x = A^{-1}b \end{aligned}$$

und berechne $\kappa(f,A)$ mittels Ableitung $\mathrm{D}\,f(A)\colon \mathbb{R}^{n\times n} \to \mathbb{R}^n$, die für eine Matrix C die Richtungsableitung $\mathrm{D}\,f(A)C$ liefert:

$$\begin{aligned} C \mapsto \mathrm{D}\,f(A)C &= \frac{\mathrm{d}}{\mathrm{d}\,t}\left((A+tC)^{-1}\cdot b\right)\Big|_{t=0} \\ &= \frac{\mathrm{d}}{\mathrm{d}\,t}\left((A+tC)^{-1}\right)\Big|_{t=0}\cdot b \end{aligned}$$

Weiterhin gilt

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

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

$$\begin{aligned} 0 = \frac{\mathrm{d}}{\mathrm{d}\,t}I &= \frac{\mathrm{d}}{\mathrm{d}\,t}\left((A+tC)(A+tC)^{-1}\right) \\ &= C(A+tC)^{-1} + (A+tC)\cdot\frac{\mathrm{d}}{\mathrm{d}\,t}(A+tC)^{-1} \\ \iff \quad \frac{\mathrm{d}}{\mathrm{d}\,t}(A+tC)^{-1} &= -(A+tC)^{-1}\cdot C\cdot(A+tC)^{-1}, \end{aligned}$$

Für ein genügend kleines t ist die Invertierbarkeit gewährleistet, da A invertierbar ist

(s. Lemma 3.2.12). Also $\mathrm{D}\, f(A)C = -A^{-1}CA^{-1}b$ und es folgt

$$\begin{aligned}
\kappa_{abs}(f,A) &= \|\mathrm{D}\, f(A)\| \\
&= \sup_{\substack{C\in\mathbb{R}^{n\times n}\\ C\neq 0}} \frac{\left\|A^{-1}CA^{-1}b\right\|}{\|C\|} \\
&\leq \sup_{\substack{C\in\mathbb{R}^{n\times n}\\ C\neq 0}} \frac{\left\|A^{-1}\right\|\cdot\|C\|\cdot\left\|A^{-1}b\right\|}{\|C\|} \\
&= \left\|A^{-1}\right\|\cdot\|x\| \\
&\leq \left\|A^{-1}\right\|^2\cdot\|b\| \\
\kappa_{rel}(f,A) &= \frac{\|A\|}{\|f(A)\|}\cdot\|\mathrm{D}\, f(A)\| \\
&\leq \|A\|\cdot\left\|A^{-1}\right\| \qquad (3.2.10)
\end{aligned}$$

c) betrachte Störungen in A und b :

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

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

$$\frac{\|\Delta x\|}{\|x\|} \leq \|A\|\cdot\left\|A^{-1}\right\|\cdot(1-\left\|A^{-1}\right\|\cdot\|\Delta A\|)\cdot\underbrace{\left(\frac{\|\Delta b\|}{\|b\|}+\frac{\|\Delta A\|}{\|A\|}\right)}_{\neq\frac{\|(\Delta A,\Delta b)\|}{\|(A,b)\|}} \qquad (3.2.11)$$

Beweis. s. Übungsblatt □

Definition 3.2.11. Sei $\|\bullet\|$ eine Norm auf $\mathbb{R}^{n\times n}$ und $A\in\mathbb{R}^{n\times n}$ eine reguläre Matrix. Die Größe

$$\kappa_{\|\bullet\|}(A) = \mathrm{cond}_{\|\bullet\|} \coloneqq \|A\|\cdot\left\|A^{-1}\right\|$$

heißt **Kondition der Matrix** bzgl. der Norm $\|\bullet\|$. Ist $\|\bullet\|$ von einer Vektor-Norm $\|\bullet\|_p$ induziert, bezeichnet $\mathrm{cond}_p(A)$ die $\mathrm{cond}_{\|\bullet\|_p}(A)$. Wir schreiben $\mathrm{cond}(A)$ für $\mathrm{cond}_2(A)$. $\mathrm{cond}_{\|\bullet\|}(A)$ 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?

Hierzu werden wir die Resultate aus 3.2.12 und die folgende Relation benötigen:

$$A+\Delta A = A(I+A^{-1}\Delta A)$$

Lemma 3.2.12 (Neumannsche Reihe)**.** *Sei $C \in \mathbb{R}^{n\times n}$ mit $\|C\| < 1$ und mit einer submultiplikativen Norm $\|\bullet\|$, so ist $(I-C)$ invertierbar und es gilt*

$$(I-C)^{-1} = \sum_{k=0}^{\infty} C^k \qquad \textit{sowie} \qquad \left\|(I-C)^{-1}\right\| \leq \frac{1}{1-\|C\|}.$$

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

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

Daraus folgt bereits, dass $\sum_{k=0}^{\infty} C^k$ existiert (nach Majorantenkriterium). Außerdem ist $\|C\|^m$ und damit C^m eine Nullfolge. Weiter gilt dann

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

□

Bemerkung 3.2.13.

a) Für eine symmetrische, positiv definite Matrix $A \in \mathbb{R}^{n\times n}$ gilt[6]

$$\kappa_2(A) = \frac{\lambda_{max}}{\lambda_{min}} \tag{3.2.13}$$

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

$$\kappa(A) := \frac{\max\limits_{\|x\|=1} \|Ax\|}{\min\limits_{\|x\|=1} \|Ax\|} \in [0,\infty] \tag{3.2.14}$$

Diese ist auch für nicht invertierbare und rechteckige Matrizen wohldefiniert. Dann gelten offensichtlich die folgenden Punkte.

[6] Beweis: siehe Übungsblatt 3

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

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

e) $A \neq 0$ und $A \in \mathbb{R}^{n\times n}$ ist genau dann singulär, wenn $\kappa(A) = \infty$. Wegen der Skalierungsinvarianz ist die Kondition zur Überprüfung der Regularität von A besser geeignet als die Determinante.

Beispiel 3.2.14 (Kondition eines nichtlin. Gleichungssystems)**.** Sei $f\colon \mathbb{R}^n \to \mathbb{R}^n$ 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} Ausgabe und y Eingabe.

Sei $\mathrm{D}\, f(x)$ 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

$$D(f^{-1})(y) = (\mathrm{D}\, f(x))^{-1}$$

Hiermit folgt

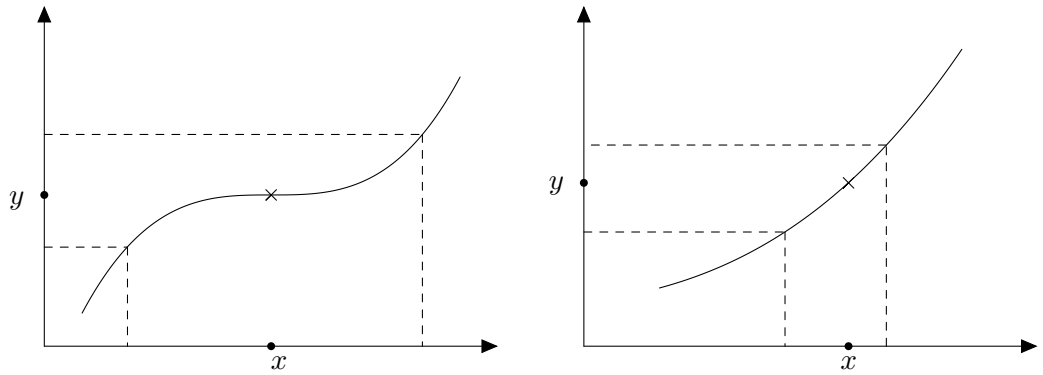
$$\begin{aligned} \kappa_{abs}(f^{-1}, y) &= \left\|(\mathrm{D}\, f(x))^{-1}\right\| \\ \kappa_{rel}(f^{-1}, y) &= \frac{\|f(x)\|}{\|x\|} \cdot \left\|(\mathrm{D}\, f(x))^{-1}\right\| \end{aligned} \tag{3.2.15}$$

Für skalare Funktionen $f\colon \mathbb{R} \to \mathbb{R}$ folgt somit

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

Falls $|f'(x)| \to 0$ ist es eine schlechte absolute Kondition, dagegen für $|f'(x)| \gg 0$ eine gute absolute Kondition.

Abb. 3.2.1 *©Schlechte (links) und Gute (rechts) Kondition im Vergleich*



Links bedeutet eine kleine Störung in y eine große Störung in x.

3.2b) 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 – stets gleich 1. Ebenso sind Störungen nur in der Diagonale zu erwarten. Jedoch

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

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

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

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*

$$\kappa_{rel} = \frac{\| \, |\mathrm{D} f(x)| \cdot |x| \, \|_\infty}{\|f(x)\|_\infty} \tag{3.2.16}$$

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

$$\begin{aligned} f_i(\tilde{x}) - f_i(x) &= \left(\frac{\partial f_i}{\partial x_i}(x), \dots, \frac{\partial f_i}{\partial x_n}(x)\right) \cdot \begin{pmatrix} \tilde{x}_1 - x_1 \\ \vdots \\ \tilde{x}_n - x_n \end{pmatrix} + o(\|\tilde{x} - x\|) \\ \Longrightarrow |f(\tilde{x}) - f(x)| &\leq |\mathrm{D} f(x)| \cdot \begin{pmatrix} |x_1| \cdot \frac{|\tilde{x}_1 - x_1|}{|x_1|} \\ \vdots \\ |x_n| \cdot \frac{|\tilde{x}_n - x_n|}{|x_n|} \end{pmatrix} + o\left(\max_i \frac{|\tilde{x}_i - x_i|}{|x_i|}\right) \qquad \text{da } x_i \text{ fest und } \tilde{x}_i \to x_i \\ &\leq |\mathrm{D} f(x)| \cdot |x| \cdot \max_i \frac{|\tilde{x}_i - x_i|}{|x_i|} + o\left(\max_i \frac{|\tilde{x}_i - x_i|}{|x_i|}\right) \\ \Longrightarrow \frac{\|f(\tilde{x}) - f(x)\|_\infty}{\|f(x)\|_\infty} &\leq \frac{\| \, |\mathrm{D} f(x)| \cdot |x| \, \|}{\|f(x)\|_\infty} \cdot \max_i \frac{|\tilde{x}_i - x_i|}{|x_i|} + o\left(\max_i \frac{|\tilde{x}_i - x_i|}{|x_i|}\right) \end{aligned}$$

Wähle $\tilde{x}_i = x_j + h \cdot \operatorname{sign} \frac{\partial f_i}{\partial x_j}(x)$ mit $h > 0$, dann gilt

$$|\mathrm{D}\, f_i(x)(\tilde{x} - x)| = \mathrm{D}\, f_i(x)(\tilde{x} - x)$$

und in obiger Rechnung gilt Gleichheit. Also folgt, dass

$$\frac{\| \, |\mathrm{D}\, f(x)| \cdot |x| \, \|_\infty}{\|f(x)\|_\infty} = \kappa_{rel}$$

□

Beispiel 3.2.18.

a) Komponentenweise Kondition der Multiplikation

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

b) Komponentenweise Kondition eines linearen Gleichungssystems:
Löse $Ax = b$ mit möglichen Störungen in b, also zu

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

Falls A eine Diagonalmatrix ist, folgt

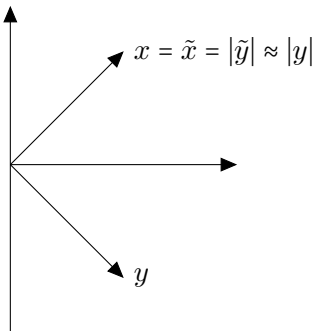
$$\kappa_{rel} = 1$$

c) Komponentenweise Kondition des Skalarproduktes $\langle x, y\rangle := \sum_{i=1}^n x_i y_i = x^T y$ für $f{:}\,\mathbb{R}^2 \to \mathbb{R}$, $f(x,y) = \langle x, y\rangle$:
$\mathrm{D}\, f(x,y) = (y^T, x^T)$, also gilt mit $\cos(x,y) = \frac{\langle x,y\rangle}{\|x\|_2 \cdot \|y\|_2}$

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

Falls x und y nahezu senkrecht aufeinander stehen, kann das Skalarprodukt sehr schlecht konditioniert sein. Zum Beispiel für $x = \tilde{x} = \begin{pmatrix}1\\1\end{pmatrix}$ und $y = \begin{pmatrix}1+10^{-10}\\-1\end{pmatrix}$, $\widetilde{y} = \begin{pmatrix}1\\-1\end{pmatrix}$.

Abb. 3.2.2 *©Vektoren mit nahezu gleichem komponentenweisen Betrag*



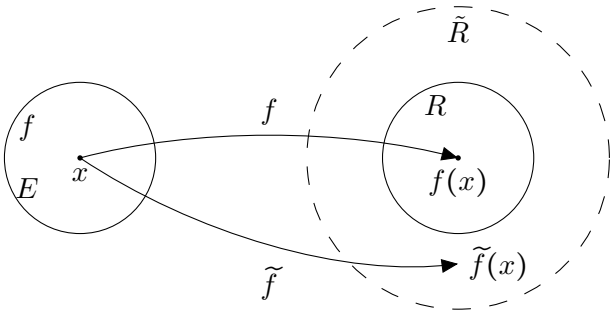
3.3 Stabilität von Algorithmen

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“.

Abb. 3.3.0 *©Bildbereiche eines stabilen und eines instabilen Algorithmus*



$R = f(E)$ stabil
$\tilde{R} = \widetilde{f}(E)$ instabil

3.3a) Vorwärtsanalyse

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

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

$$\begin{aligned} fl(a \nabla b) &= (a \nabla b) \cdot (1 + \varepsilon) \\ &= (a \nabla b) \cdot \frac{1}{1 + \mu} \end{aligned} \tag{3.3.1}$$

29.10.2014

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

Beispiel 3.3.2. Sei $f(x_1, x_2, x_3) \coloneqq \frac{x_1 x_2}{x_3}$ mit Maschinenzahlen x_i und $x_3 \neq 0$ und sei der Algorithmus (siehe Definition 3.3.4) durch $f(x_1, x_2, x_3) = (f^{(2)} \circ f^{(1)})(x_1, x_2, x_3)$ gegeben mit

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

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

Sei $x = (x_1, x_2, x_3)$. Daraus folgt

$$\begin{aligned} \widetilde{f}^{(1)}(x) &= (fl(x_1 \cdot x_2), x_3) \\ &= (x_1 x_2 (1 + \varepsilon_1), x_3) \end{aligned}$$

mit $|\varepsilon_1| \leq \text{eps}$:

$$\begin{aligned} \widetilde{f}(x) &= \widetilde{f}^{(2)}(\widetilde{f}^{(1)}(x)) \\ &= fl(f^{(2)}(x_1 x_2 (1 + \varepsilon_1), x_3)) \\ &= \frac{x_1 x_2 (1 + \varepsilon_1)}{x_3} \cdot (1 + \varepsilon_2) \\ &= f(x) \cdot (1 + \varepsilon_1)(1 + \varepsilon_2) \end{aligned}$$

mit $|\varepsilon_2| \leq \text{eps}$:

$$\begin{aligned} \frac{|\widetilde{f}(x) - f(x)|}{|f(x)|} &= |\varepsilon_1 + \varepsilon_2 + \varepsilon_1 \cdot \varepsilon_2| \\ &\leq 2\,\text{eps} + \text{eps}^2 \end{aligned}$$

Dies ist eine „worst case“ Analyse, da immer der maximale Fehler angenommen wird, und gibt i.d.R. eine starke Überschätzung des Fehlers an. Für qualitative Aussagen sind sie jedoch nicht nützlich. In Computersystemen stehen mehr Operationen wie ∇ zur Verfügung, die mit einer relativen Genauigkeit eps realisiert werden können.

Daher:

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

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

$\widetilde{\phi}$ heißt dann **Realisierung** von ϕ.

Bemerkung
aus (3.3.2) folgt für $1 \leq p \leq \infty$

$$\left\|\widetilde{\phi}(x) - \phi(x)\right\|_p \leq \text{eps} \cdot \|\phi(x)\|_p \quad \forall x \in \mathbb{F}^n \tag{3.3.3}$$

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

$$f^{(i)}: U_1 \subseteq \mathbb{R}^{k_i} \to U_{i+1} \subseteq \mathbb{R}^{k_{i+1}}$$

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

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

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

$$\widetilde{f} = \widetilde{f}^{(l)} \circ \ldots \circ \widetilde{f}^{(1)}$$

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 $i = 1, \ldots, n$. Sei $\left(f^{(1)}, \ldots, f^{(l)}\right)$ ein Algorithmus für f und $(\widetilde{f}^{(1)}, \ldots, \widetilde{f}^{(l)})$ eine zugehörige Implementation. Mit den Abkürzungen*

$$x^{(j+1)} \coloneqq f^{(j)} \circ \ldots \circ f^{(1)}(x)$$
$$x^{(1)} \coloneqq x$$

und entsprechend mit $\tilde{x}^{(j+1)}$ gilt, falls $x^{(j+1)} \neq 0$ für alle $j = 0, \ldots, (l-1)$ und $\|\bullet\|$ eine beliebige p-Norm ist,

$$\frac{\left\|\tilde{x}^{(j+1)} - x^{(j+1)}\right\|}{\left\|x^{(j+1)}\right\|} \leq \text{eps} \cdot \mathcal{K} + o\,(\text{eps}) \tag{3.3.4}$$
$$\mathcal{K}^{(j)} = \left(1 + \kappa^{(j)} + \kappa^{(j)} \cdot \kappa^{(j-1)} + \cdots + \kappa^{(j)} \cdots \kappa^{(1)}\right)$$

wobei $\kappa^{(j)} \coloneqq \kappa_{rel}\left(f^{(j)}, x^{(j)}\right)$ die Kondition der elementar ausführbaren Operationen $f^{(j)}$ ist.

Beweis. Wir zeigen 3.3.3 durch Induktion. Betrachte hierfür vorerst folgende Abschätzung

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

Nach Voraussetzung gilt Gleichung (3.3.4) mit $\mathcal{K}^{(0)} = 1$ für $j = 0$. Für $j = 1$ folgt mithilfe obiger Abschätzung und der Voraussetzung $\frac{\|\tilde{x}-x\|}{\|x\|} = \frac{\|\tilde{x}^{(1)}-x^{(1)}\|}{\|x^{(1)}\|} \leq \text{eps}$

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

Womit der Induktionsanfang gezeigt ist. Für den Induktionsschritt von $j-1$ zu j:

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

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

Hiermit folgt:

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

$$\frac{\left\|\widetilde{f}(\tilde{x}) - f(x)\right\|}{\|f(x)\|} \leq \text{eps} \cdot \left(1 + \kappa^{(l)} + \kappa^{(l)} \cdot \kappa^{(l-1)} + \cdots + \kappa^{(l)} \cdots \kappa^{(1)}\right) + o(\text{eps}) \tag{3.3.5}$$

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. Nach Beispiel 3.2.9 ist die Substraktion zweier annähernd gleicher Zahlen schlecht konditioniert. Deshalb sollte man unvermeidbare Subtraktionen möglichst früh durchführen. Allerdings hängt $\kappa^{(j)}$ nicht nur von $f^{(j)}$, sondern auch vom Zwischenergebnis $x^{(j)}$ ab, welches a priori unbekannt ist.

Bemerkung 3.3.8 (Sprechweise)**.** Der Quotient

$$\frac{\overbrace{\frac{\left\|\widetilde{f}(\tilde{x}) - f(x)\right\|}{\|f(x)\|}}^{\text{Gesamtfehler}}}{\underbrace{\frac{\|f(\tilde{x}) - f(x)\|}{\|f(x)\|}}_{\substack{\text{Fehler}\\ \text{durch}\\ \text{Problem}}} \cdot \underbrace{\frac{\|\tilde{x} - x\|}{\|x\|}}_{\substack{\text{Eingabe-}\\ \text{fehler}}}} \tag{3.3.6}$$

gibt die **Güte des Algorithmus** an. Als Stabilitätsindikator kann also

$$\sigma\left(f, \widetilde{f}, x\right) \coloneqq \frac{\mathcal{K}}{\kappa_{rel}(f, x)} \tag{3.3.7}$$

verwendet werden und es gilt

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

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

Beispiel 3.3.9. Nach Gleichung (3.3.3) gilt für die Elementaroperationen $\mathcal{K} \leq 2$, da sie bzgl. der Maschinenzahlen nahezu exakt sind. Da für die Subtraktion zweier annähernd gleich großer Zahlen $\kappa_{rel} \gg 1$ gilt, ist der Stabilitätsfaktor zweier annähernd gleich großer Zahlen sehr klein und der Algorithmus also stabil, Falls es sich jedoch bei einer

zusammengesetzten Abbildung $f = h \circ g$ bei der zweiten Abbildung h um eine Subtaktion handelt, gilt

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

und die Stabilität ist gefährdet. Genauere Abschätzungen und damit genauere Indikatoren können durch komponentenweise Betrachtungen erhalten werden.

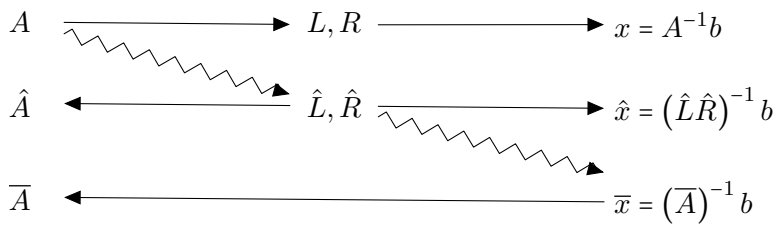
3.3b) Rückwärtsanalyse

Die Fragestellung ist nun:

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

Das würde heißen $\exists \hat{x} \in \mathbb{R}^n : f(\hat{x}) = \widetilde{f}(\tilde{x})$. Dann schätze den Fehler $\|\hat{x} - x\|$ bzw. falls f nicht injektiv ist $\min_{\hat{x} \in \mathbb{R}^n} \left\{ \|\hat{x} - x\| \middle| f(\hat{x}) = \widetilde{f}(\tilde{x}) \right\}$ ab.

Abb. 3.3.1 ©



Ein Anwendungsbeispiel:
Die Eingangsdaten seien Messdaten $\tilde{x}$ mit 1% relativer Genauigkeit. Liefert die Rückwärtsanalyse, dass $\widetilde{f}(\tilde{x})$ 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.

Vorwärtsfehler	$\leq$	Kondition des Problems	$\cdot$	Rückwärtsfehler
$\left\|\widetilde{f}(\tilde{x}) - f(x)\right\|$	$\leq$	$\kappa(f, x)$	$\cdot$	$\|\hat{x} - x\|$

Beispiel: Rückwärtsanalyse der Gauß-Elimination (geht auf Wilkinson zurück)

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*

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

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

Beweis. [siehe SB90]. □

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 $\overline{x}$ mit*

$$\overline{A}\overline{x} = b \qquad \textit{und} \qquad |\overline{A} - A| \leq 2n\,\text{eps}\left\|\hat{L}\right\| \cdot \left\|\hat{R}\right\| + \mathcal{O}(n^2\,\text{eps}^2).$$

Beweis. [siehe DH08]. □

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 $n\left\|\hat{L}\right\|\left\|\hat{R}\right\|$ 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.

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

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

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

Ein Beispiel hierzu ist die Hilbertmatrix $H = \left(\frac{1}{i+j-1}\right)_{i,j=1,\dots,n}$, für die $\operatorname{cond}(H)$ exponentiell mit der Dimension n wächst.

3.4 Beurteilung von Näherungslösungen linearer GLS

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

3.4a) Im Sinne der Vorwärtsanalyse

Im Sinne der Vorwärtsanalyse und der Fehlerentwicklung durch das Problem gilt

$$\frac{\|\tilde{x} - x\|}{\|x\|} \leq \operatorname{cond}(A) \cdot \frac{\|\Delta b\|}{\|b\|}$$

nach Beispiel 3.2.10, mit dem Residuum

$$r(\tilde{x}) \coloneqq A\tilde{x} - b = \widetilde{b} - b = \Delta b \tag{3.4.1}$$

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

$$\frac{\|r(\tilde{x})\|}{\|b\|} \tag{3.4.2}$$

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

3.4b) Im Sinne der Rückwärtsanalyse

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

$$\|r(\tilde{x})\| \leq \varepsilon(\|A\| \cdot \|\tilde{x}\| + \|b\|). \tag{3.4.3}$$

dann existiert eine Matrix $\tilde{A}$ und ein Vektor $\tilde{b}$, so dass $\tilde{A}\tilde{x} = \tilde{b}$ und

$$\left\|\tilde{A} - A\right\| \leq \varepsilon \|A\| \qquad \textit{und} \qquad \left\|\tilde{b} - b\right\| \leq \varepsilon \|b\| \ . \tag{3.4.4}$$

Aufgrund von (3.4.3) *wird der komponentenweise relative Rückwärtsfehler durch*

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

abgeschätzt.

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

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

4 Lineare Gleichungssysteme: Direkte Methoden (Fortsetzung)

4.1 Gaußsches Eliminationsverfahren mit Aquilibrierung 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. Jetzt ist die Frage:

Was ist eine „gute“ Skalierung?

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

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

4.1.1 Äquilibrierung der Zeilen

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

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

gesetzt wird.

4.1.2 Äquilibrierung der Spalten

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

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

gesetzt wird.

Äquilibrierung von Zeilen **und** Spalten führt zu einem nichtlinearen Gleichungssystem und ist i.d.R. aufwendig.

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

$$cond_\infty(A) \leq cond_\infty(DA) \tag{4.1.3}$$

für alle regulären Diagonalmatrizen D.

Beweis. siehe Übungsaufgabe □

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

4.1.4 Nachiteration

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

$$r(\tilde{x}) \coloneqq b - A\tilde{x} = 0 \tag{4.1.4}$$

ansonsten ist $A(x - \tilde{x}) = r(\tilde{x})$. Also löse die Korrekturgleichung

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

und setze $x^{(1)} \coloneqq \tilde{x} + \Delta x$ Wiederhole dies sooft, bis $x^{(i)}$ „genau genug“ ist. Die Lösung $\tilde{x}$ wird durch Nachiteration meist mit sehr gutem Erfolg verbessert [genaueres in DR08].

(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.

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 $\langle x, Ax\rangle = x^T Ax > 0$ für alle $x \neq 0$. (kurs: **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 $i = 1, \ldots, n$*

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) Es gilt $x^T Ax \neq 0 \Rightarrow Ax \neq 0$. Nach ii) ist $Ax \neq 0 \ \forall x \in \mathbb{R}^n \backslash \{0\}$ also $ker(A) = 0$.

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:

$$A^{(1)} := A = \begin{pmatrix} a_{11} & z^T \\ z & B^{(1)} \end{pmatrix}$$

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

$$\Longrightarrow L_1 A^{(1)} L_1^T = \begin{pmatrix} a_{11} & z^T \\ 0 & \\ \vdots & B^{(2)} \\ 0 & \end{pmatrix} \cdot \begin{pmatrix} 1 & -\frac{z}{a_{11}} \\ 0 & \\ \vdots & I \\ 0 & \end{pmatrix}$$

$$= \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & B^{(2)} & \\ 0 & & & \end{pmatrix}$$

Weiterhin gilt $x \neq 0 \Longleftrightarrow L_1 x \neq 0$ da L_1 invertierbar. Also gilt insgesamt:

$$\tilde{x}^T B^{(2)} \tilde{x} = x^T L_1 A^{(1)} L_1^T x \qquad \text{für } x := \begin{pmatrix} 0 \\ \tilde{x} \end{pmatrix}$$

$$= (L_1^T x)^T A (L_1^T x) > 0 \qquad \forall \tilde{x} \neq 0$$

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 $i = 1, \ldots, n$ gilt.

Sei $L := (L_1^{-1} \ldots L_{n-1}^{-1})$ wie in (2.1.8), so ergibt sich:

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

$$A = LDL^T$$

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*

$$A = \bar{L}\bar{L}^T \tag{4.2.1}$$

mit der reellen unteren Dreiecksmatrix

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

heißt ***Cholesky-Zerlegung.*** *.*

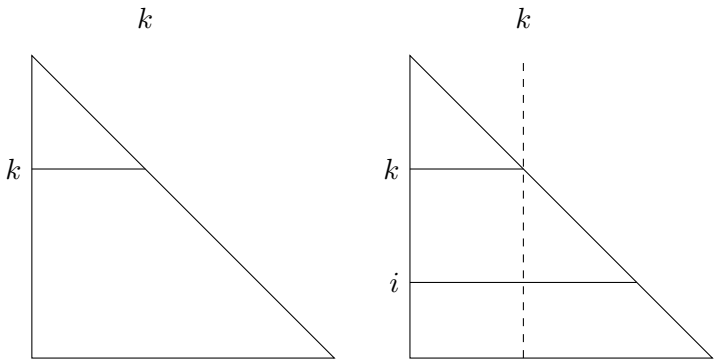
Wegen (4.2.1) *gilt:*

$$a_{kk} = \bar{l}_{k1}^2 + \cdots + \bar{l}_{kk}^2 \tag{4.2.2}$$

$$a_{ik} = \bar{l}_{i1}\bar{l}_{k1} + \cdots + \bar{l}_{ik}\bar{l}_{kk} \tag{4.2.3}$$

Demnach funktioniert spaltenweises und zeilenweises Berechnen.

Abb. 4.2.0 *© Spaltenweise Berechnung der Cholesky-Zerlegung*



Es ergibt sich folgender Algorithmus:

4.2.3 Cholesky-Zerlegung

Der Algorithmus der Cholesky-Zerlegung ist wie folgt:

```
for k = 1, ..., n
|   l_kk = (a_kk − Σ_{j=1}^{k−1} l_kj)^{1/2}
|   for i = k + 1, ..., n
|   |   l_ik = (a_ik − Σ_{j=1}^{k−1} l_ij l_kj)/l_kk
|   end
end
```

4.2.4 Rechenaufwand in flops

Es sind je

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

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

n Quadratwurzeln

also ca. $\frac{2}{3}n^3$ flops für große n notwendig. Im Vergleich zur LR-Zerlegung halbiert sich in etwa der Aufwand.

Bemerkung 4.2.5.

a) Wegen (4.2.2) gilt $|\bar{l}_{kj}| \leq \sqrt{a_{kk}}$, d.h. die Matrizeneinträge können nicht zu groß werden.

b) Für spd Matrizen ist der Cholesky-Algorithmus stabil nach (3.3.13)

c) Da A symmetrisch ist, muss nur die untere Dreiecksmatrix gespeichert werden. In Algorithmen kann $\bar{L}$ in eine Kopie dieser Dreiecksmatrix geschrieben werden.

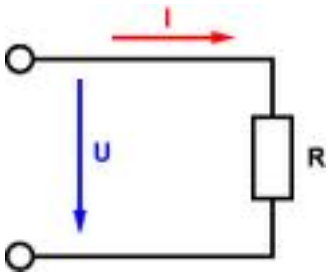
d) Fast singuläre Matrizen können durch die Diagonale erkannt werden.

4.3 Lineare Ausgleichsprobleme

10.11.2014

Beispiel 4.3.1. (s. Einführung) Seien m Messungen (I_i, U_i) für die Stromstärke I und die Spannung U gegeben.

Abb. 4.3.0 *Schaltplan einer einfachen U-I-Messung*

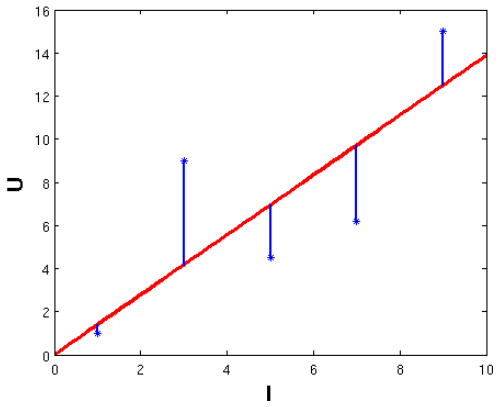


Das Ohmsche Gesetz liefert hierfür:

$$U = R \cdot I$$

Gesucht ist der zugehörige Widerstand R.

Abb. 4.3.1 *Linearausgleich einer U-I-Messung mit Ursprungsgerade als Modellfunktion*



Wird jetzt davon ausgegangen dass die I_i exakt sind, wird das R gesucht, für das RI_i im Mitttel den minimalen Abstand zu U_i hat. Genauer gesagt berechne

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

Vorsicht: Es wird **nicht** die Gerade (bzw. der lineare Untervektorraum) mit minimalem euklidischem Abstand zu (I_i, U_i) gesucht! Dieses Problem ist nichtlinear und aufwendig zu lösen.

4.3.2 Lineares Ausgleichsproblem

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

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

Für exakte Messdaten b_i würde

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

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**:

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

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

Definiere:

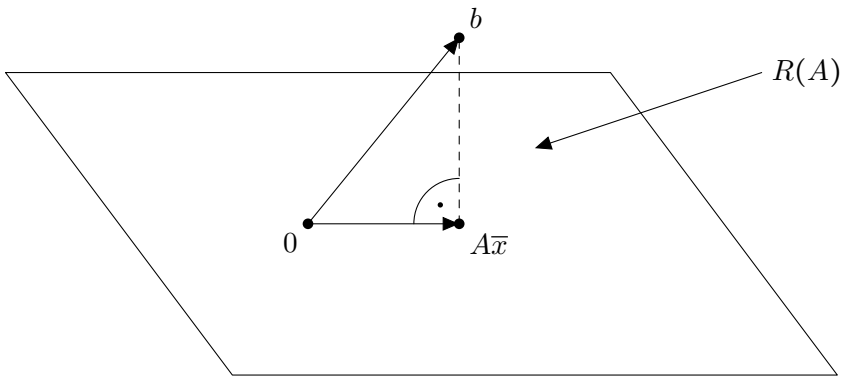
$$b = (b_i)_{i=1,\dots,m} \in \mathbb{R}^m$$
$$x = (x_j)_{j=1,\dots,n} \in \mathbb{R}^n$$
$$A = (a_j(t_i))_{\substack{i=1,\dots,m \\ j=1,\dots,n}} \in \mathbb{R}^{m\times n}$$

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 $\overline{x} \in \mathbb{R}^n$ gesucht mit

$$\|b - A\overline{x}\|_2 = \min_{x\in\mathbb{R}^n} \|b - Ax\|_2 \tag{4.3.2}$$

Das entspricht der „Lösung“ eines überbestimmten, i.A. nicht erfüllbaren GLS $Ax = b$.
Aufgrund der l_2-Norm ist $\overline{x}$ gegeben durch die orthogonale Projektion von b auf den Bildraum $R(A)$, wie gleich gezeigt wird.

Abb. 4.3.2 ©



Satz 4.3.3 (Projektionssatz)**.** *Sei V ein reeller Vektorraum mit einem Skalarprodukt $\langle\bullet,\bullet\rangle$ und der induzierten Norm $\|v\| := \sqrt{\langle v, v\rangle}$. Sei $U \subset V$ 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 $\overline{u} \in U$, so dass $v - \overline{u} \in U^\perp$, d.h. $\langle v - \overline{u}, u\rangle = 0 \ \forall u \in U$ Dies definiert die **orthogonale Projektion** $P: V \to U, v \mapsto \overline{u} = Pv$.*

2) Zu jedem $v \in V$ bestimmt $P \cdot v$ die eindeutige Lösung $\|v - Pv\| = \min_{u \in U} \|v - u\|$ Also gilt mit einem eindeutigen $\overline{u} = Pv$, dass

$$\|v - \overline{u}\| = \min_{u \in U} \|v - u\| \Longleftrightarrow \langle v - \overline{u}, u \rangle = 0 \quad \forall u \in U \tag{4.3.3}$$

Beweis.

1) Sei $\{u_1, \ldots, u_n\}$ eine Orthonormalbasis von U und $\overline{u} \in U$. Daraus folgt

$$\exists! (\alpha_i)_{i=1,\ldots,n} \subset \mathbb{R} : \overline{u} = \sum_{i=1}^{n} \alpha_i u_i$$

Damit gilt

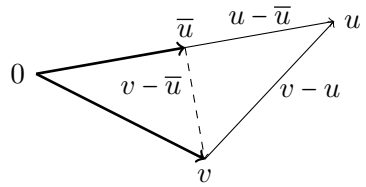
$$\begin{aligned} & 0 = \langle v - \overline{u}, u \rangle && \forall u \in U \\ \Longleftrightarrow \quad & 0 = \langle v - \sum_{i=1}^{n} \alpha_i u_i, u_i \rangle && \forall j = 1, \ldots, n \\ \Longleftrightarrow \quad & \langle v, u_j \rangle = \sum_{i=1}^{n} \alpha_i \langle u_i, u_j \rangle =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

Setze also

$$P \cdot v = \overline{u} = \sum_{i=1}^{n} \langle v, u_i \rangle u_i \in U \tag{4.3.4}$$

dann ist $\overline{u}$ die eindeutig bestimmte Lösung für $v - \overline{u} \in U^{\perp}$

2) **Abb. 4.3.3** *Geometrische Konstruktion von $\overline{u}$*



Sei $u \in U$. Dann gilt

$$\begin{aligned} \|v - u\|^2 &= \|v - \overline{u} + \overline{u} - u\|^2 \\ &= \|v - \overline{u}\|^2 + 2 \underbrace{\langle v - \overline{u}, \overbrace{\overline{u} - u}^{\in U} \rangle}_{=0} + \|u - \overline{u}\|^2 \\ &= \|v - \overline{u}\|^2 + \|u - \overline{u}\|^2 \end{aligned} \tag{4.3.5}$$

(Dies ist anschaulich der Satz des Pythagoras.) Damit ist $\|v - u\|$ für ein $u \in U$ genau dann minimal, wenn $\|u - \overline{u}\|$ minimal ist, also wenn $u = \overline{u}$.

□

Satz 4.3.4. *Der Vektor $\overline{x} \in \mathbb{R}^n$ ist genau dann Lösung des linearen Ausgleichsproblems $\min_{x\in\mathbb{R}^n} \|b - Ax\|_2$, falls er die Normalengleichung*

$$A^T A\overline{x} = A^T b \tag{4.3.6}$$

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

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

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

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

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 $\operatorname{cond}(A)$ ab.

4.3.5 Lösung der Normalgleichung

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

1. $A^T A$ zu berechnen:

   Aufwand ca. $\frac{1}{2}n^2 m$ Multiplikationen

   Kondition häufig schlecht, da $\frac{1}{2}n^2$ Skalarprodukte berechnet werden

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

   Aufwand ca. $\frac{1}{6}n^3$ Multiplikationen

   Kondition Für $A \in \mathbb{R}^{m\times n}$ mit $m \ge n$ und $\operatorname{rang}(A) = n$ gilt (siehe Übungsaufgabe 19)

$$\operatorname{cond}_2(A^T A) = \operatorname{cond}_2(A)^2 \tag{4.3.7}$$

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 für Normalgleichungen ungeeignet**.

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

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

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

$$\begin{pmatrix} \overline{b}_1 \\ \overline{b}_2 \end{pmatrix} := Q^T \cdot b \tag{4.3.8}$$

dann ist

$$\overline{x} = R^{-1}\overline{b}_1 \tag{4.3.9}$$

die Lösung des linearen Ausgleichsproblems und

$$\left\|\overline{b}_2\right\| = \|b - A\overline{x}\| = \min_{x\in\mathbb{R}^n} \|b - Ax\|$$

Zur Erinnerung: Q orthogonal $:\Leftrightarrow QQ^T = I \Leftrightarrow Q^{-1} = Q^T$
Weiterhin ist Q längenerhaltend, d.h. $\|Qv\|_2 = \|v\|_2$, und somit folgt $\|Q\|_2 = \left\|Q^{-1}\right\|_2 = 1$ sowie

$$\operatorname{cond}_2(Q) = 1 \tag{4.3.10}$$

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

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

wird minimal für $R\overline{x} = \overline{b}_1$. □

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

$$\operatorname{cond}_2(A) = \operatorname{cond}_2(R)$$

Die auftretende Kondition entspricht also der des Ausgleichsproblems.

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

$$A = Q \cdot R$$

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

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

4.4 Orthogonalisierungsverfahren

12.11.2014

Konstruiere eine QR-Zerlegung

$$A = Q \cdot \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{4.4.1}$$

durch einen Eliminationsprozess:

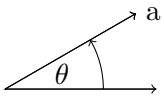
$$A \longrightarrow Q^{(1)}A \longrightarrow Q^{(2)}Q^{(1)}A \longrightarrow Q^{(p)}\cdots Q^{(1)}A = \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{4.4.2}$$

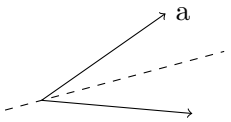
mit orthogonalen Matrizen $Q^{(i)}$. Dann gilt

$$Q = Q^{(1)T}\cdots Q^{(p)T} \tag{4.4.3}$$

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 Anschauungen, nämlich:

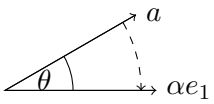
a) Drehung 

b) Spiegelung 

4.4a) Givens-Rotation

Es wird eine Drehung auf den 1. Einheitsvektor durchgeführt:

Abb. 4.4.0 *Drehung auf den Einheitsvektor*



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

d.h. Elimination von a_2 mit $\|\alpha e_1\|_2 = \|a\|_2$. Also gilt $\alpha = \pm\|a\|_2$. Drehungen werden beschrieben durch

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

und es muss gelten $Qa = \begin{pmatrix} \alpha \\ 0 \end{pmatrix}$. Hiermit folgt für $\|a\| = 0$, dass $c = 1, s = 0$, und für $\|a\| \neq 0$, dass $c = \frac{a_1}{\alpha}, s = \frac{a_2}{\alpha}$ mit

$$\alpha = \pm\sqrt{a_1^2 + a_2^2} \tag{4.4.4}$$

Im Folgenden wird dies kurz mit $[c, s] = \text{givens}(a_1, a_2)$ bezeichnet. Als **Givens-Rotation** wird eine Matrix der Form

$$\Omega_{k,l} = \begin{pmatrix} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ & & & \mathbf{c} & & & \mathbf{s} & & \\ & & & & 1 & & & & \\ & & & & & \ddots & & & \\ & & & & & & 1 & & \\ & & & -\mathbf{s} & & & \mathbf{c} & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{pmatrix} \begin{matrix} \\ \\ \\ \leftarrow k\text{-te Zeile} \\ \\ \\ \\ \leftarrow l\text{-te Zeile} \\ \\ \\ \end{matrix} \tag{4.4.5}$$

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

$$\begin{aligned} \Omega_{kl} A &= \widetilde{A} \quad \text{mit} \\ \widetilde{a}_{ij} &= a_{ij} \qquad \text{für } i \neq k, l \\ \widetilde{a}_{kj} &= c a_{kj} + s a_{lj} \\ \widetilde{a}_{lj} &= -s a_{kj} + c a_{lj} \end{aligned}$$

Demnach werden nur die k-te und l-te Zeile werden verändert.
Falls nun $[c, s] = \text{givens}(x_k, x_l)$ gilt

$$\Omega_{k,l} \cdot x = \begin{pmatrix} x_1 \\ \vdots \\ x_{k-1} \\ \alpha \\ x_{k+1} \\ \vdots \\ x_{l-1} \\ 0 \\ x_{l+1} \\ \vdots \\ x_n \end{pmatrix} \qquad \text{mit} \quad \alpha = \pm \left\| \begin{pmatrix} x_k \\ x_l \end{pmatrix} \right\|_2 \tag{4.4.6}$$

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

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

gilt, sind $p = \sum_{j=1}^{n}(m-j)$ 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.

Für eine 3x4-Matrix sieht das Verfahren folgendermaßen aus:

$$A = \begin{pmatrix} * & \\ * & * \\ * & \\ * & \end{pmatrix} \xrightarrow{\Omega_{3,4}} \begin{pmatrix} * & \\ * & * \\ * & \\ 0 & \end{pmatrix} \xrightarrow{\Omega_{1,2}} \begin{pmatrix} * & \\ * & * \\ 0 & \\ 0 & \end{pmatrix}$$

$$\xrightarrow{\Omega_{2,3}} \begin{pmatrix} * & \\ 0 & * \\ 0 & \\ 0 & \end{pmatrix} \xrightarrow{\Omega_{3,4}} \begin{pmatrix} * & * \\ 0 & * \\ 0 & * \\ 0 & 0 \end{pmatrix} \xrightarrow{\Omega_{2,3}} \begin{pmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & * \end{pmatrix}$$

$$\xrightarrow{\Omega_{3,4}} \begin{pmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} R \\ 0 \end{pmatrix}$$

Es ergibt sich:

4.4.1 Givens-QR-Algorithmus

```
for j = 1, ..., n
|    for i = m, m-1, ..., j+1
|    |    % setze a_ij auf 0
|    |    [c, s] = givens(a_{i-1,j}, ..., a_ij)
|    |    speichere c und s für a_ij
|    |    A(i-1:j, j:h) = ( c  s ) * A(i-1:j, j:n)
|    |                    ( -s c )
|    end
end
```

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 ρ gespeichert werden muss. Hiermit wird auch das Vorzeichen von α festgelegt:

   $a_2 = 0$**:** Setze $c = 1, s = 0, \alpha = a_1$, merke $\rho = 1$.

   $|a_2| > |a_1|$**:** Setze $\tau = \frac{a_1}{a_2}, s = \frac{1}{\sqrt{1+\tau^2}}, c = s \cdot \tau, \alpha = a_2\sqrt{1+\tau^2}$, merke $\rho = \frac{c}{2}$.

   $|a_1| \geq |a_2|$**:** Setze $\tau = \frac{a_2}{a_1}, c = \frac{1}{\sqrt{1+\tau^2}}, s = c \cdot \tau, \alpha = a_1\sqrt{1+\tau^2}$, merke $\rho = \frac{2}{s}$.

d) Aufgrund von c) muss nur ρ gespeichert werden:

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

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

   $|\rho| > 1$**:** 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 Aufand des Givens-QR-Algorithmus

a) $m \approx n$
   $\longrightarrow$ ca. $\frac{4}{3}n^3$ Multiplikationen und $\frac{1}{2}n^2$ Quadratwurzeln nötig
   Die Givens-QR-Zerlegung ist somit ungefähr viermal so aufwändig wie die Gauß-Elimination, dafür jedoch stabil.

b) $m \gg n$
   $\longrightarrow$ ca. $2n^2m$ 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

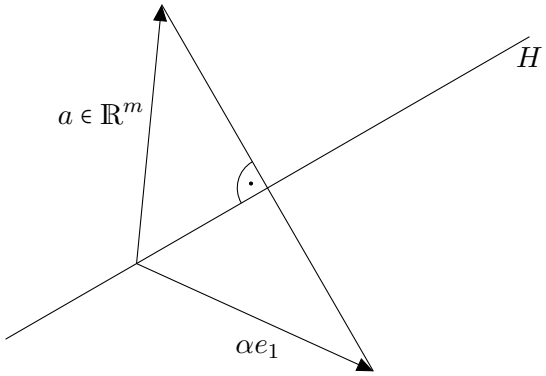
$$A = \begin{pmatrix} * & \cdots & & * \\ * & * & & \\ & \ddots & \ddots & \\ & & & \\ 0 & & & * \end{pmatrix}, \tag{4.4.7}$$

   also $a_{ik} = 0 \forall i < k+1$, sind nur $(n-1)$ Givens-Rotationen auszuführen. Diese Matrizen tauchen z.B. bei Eigenwertberechnungen auf und sind dort ein wichtiger Bestandteil der Verfahren.

4.4b) Householder-Reflexion

Es sei H eine Hyperebene im $\mathbb{R}^m$ und zusätzlich ein Vektor $a \in \mathbb{R}^m$ gegeben.

Abb. 4.4.1 © *Householder-Reflexion*



Gesucht ist nun die Reflexion Q, so dass

$$Qa = \alpha e_1 = \begin{pmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{pmatrix} \qquad \text{mit } \alpha = \pm \|a\|$$

wobei

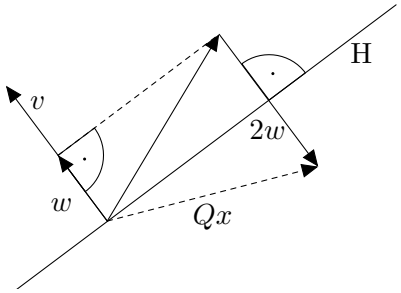
$$v = a - \alpha e_1 \tag{4.4.8}$$

gegeben ist, welches senkrecht zu H steht. Damit Stellenauslöschungen in v, d.h. in v_1, vermieden werden, wähle ein entsprechendes Vorzeichen für α, also

$$\alpha = -\operatorname{sign}(a_1)\, \|a\|_2 \tag{4.4.9}$$

Die zugehörige Reflexion Q ist gegeben durch

Abb. 4.4.2 ©



wobei

$$w = \langle \frac{v}{\|v\|}, x \rangle \cdot \frac{v}{\|v\|}$$

$$Qx = x - 2w = x - 2\frac{v^T x}{v^T v} v = (I - 2\frac{vv^T}{v^T v})x \tag{4.4.10}$$

$$Q = I - 2\frac{vv^T}{v^T v} \qquad vv^T \in \mathbb{R}^{n\times n},\ v^T v \in \mathbb{R} \tag{4.4.11}$$

und heißt **Householder Reflexion** (wurde 1958 von Householder eingeführt).
Für die spezielle Wahl (4.4.8) mit (4.4.9) vom Vektor v folgt

$$\begin{aligned} vv^T = \|v\|^2 &= \|a\|^2 - 2\alpha\langle a, e_1\rangle + \alpha^2 \\ &= -2\alpha(a_1 - \alpha) \\ &= -2\alpha v \end{aligned} \tag{4.4.12}$$

Bemerkung 4.4.4.

a) Q ist symmetrisch

b) Q ist orthogonal

c) Q ist involutorisch, d.h. $Q^2 = I$ (bzw. gilt $Q^{-1} = Q^T = Q$)

Die Householder Reflexion setzt nicht nur eine Null, sondern im Vektor gleich alle gewünschten Nullen

Rotation: $\begin{pmatrix} * \\ * \\ * \end{pmatrix} \to \begin{pmatrix} * \\ * \\ 0 \end{pmatrix} \to \begin{pmatrix} * \\ 0 \\ 0 \end{pmatrix}$

Reflexion: $\begin{pmatrix} * \\ * \\ * \end{pmatrix} \to \begin{pmatrix} * \\ 0 \\ 0 \end{pmatrix}$

Um die erste Spalte in A auf die gewünschte Gestalt zu bringen, bestimme $Q^{(1)}$ wie oben, indem die erste Spalte als a gewählt wird:

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

In der k-ten Spalte sollen nun die $(k-1)$-ten Zeilen und die $(k-1)$-ten Spalten bleiben und die Restmatrix verändert werden.

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

$$\uparrow (k-1)\text{-te Spalte}$$

Setze also

$$Q^{(k)} = \begin{pmatrix} I_{k-1} & 0 \\ 0 & \overline{Q}^{(k)} \end{pmatrix}, \tag{4.4.13}$$

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

$$a = (a_{i,k}^{k-1})_{i=k,\dots,m} \subset \mathbb{R}^{m+1-k} \tag{4.4.14}$$

bestimmt wird. Dann gilt

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

Nach insgesamt

$$p = \min(m-1, n) \tag{4.4.15}$$

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

$$Q^T A = Q^{(p)} \cdots Q^{(1)} A = \begin{pmatrix} R \\ 0 \end{pmatrix}, \tag{4.4.16}$$

wobei Bemerkung 4.4.4 auch für $Q^T = Q^{(p)} \cdots Q^{(1)}$ und somit auch für

$$Q = Q^{(1)} \cdots Q^{(p)} \tag{4.4.17}$$

gilt.

4.4.5 Speicherung

17.11.2014

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:

$$A \longrightarrow \begin{pmatrix} | & & & R \\ | & | & & \\ v^{(1)} | & v^{(2)} | & \ddots & \\ | & | & | & v^{(p)} \end{pmatrix} \quad \text{und} \quad \begin{pmatrix} \alpha^{(1)} \\ \vdots \\ \alpha^{(p)} \end{pmatrix}$$

Wohlgemerkt kann so auch $A \in \mathbb{R}^{m \times n}$ mit $m < n$ bearbeitet werden, dann wird A zu:

Abb. 4.4.3 *Zerlegung einer $m \times n$-Matrix für $m < n$*

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

4.4.6 Householder QR-Algorithmus

for $j = 1, \ldots, \min(m-1, n)$
| // berechne α für $a = A(j:m, j)$
| $\alpha(j) = -\operatorname{sign}(A(j,j))\sqrt{A(j:m,j)^T A(j:m,j)}$ siehe (4.4.9)
| // berechne $A(j:m, j) = v = a - \alpha e_1$
| $A(j,j) = A(j,j) - \alpha(j)$ siehe (4.4.8)
| // berechne $-\langle v, v\rangle = \alpha(a_1 - \alpha) = \alpha v_1$
| $\beta = \alpha(j)A(j,j)$ siehe (4.4.12)
| // berechne $\overline{Q}^{(j)} T^{(j)}$ aber nicht mehr die erste Spalte, welche $\alpha(j)e_1$ ist
| **for** $l = j + 1 : n$
| | // setze $v_x = -2(v^T x)(v^T v)^{-1}$
| | $v_x = A(j:m, j)^T A(j:m, l) \cdot \frac{1}{\beta}$ siehe (4.4.10)
| | // berechne $\overline{Q}^{(j)} x = x + v_x \cdot v$
| | $A(j:m, l) = A(j:m, l) + v_x A(j:m, l)$ siehe (4.4.10)
| **end**
end

4.4.7 Berechnung von $Q^T b$

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

for $j = 1, \ldots, \min\{m-1, n\}$
| // beachte (4.4.13), also $b(1:j-1)$ bleibt gleich
| // setze $v_x = -2(v^T b)(v^T v)^{-1}$
| $v_x = \big(A(j:m,j)^T b(j:m)\big) \cdot \big(\alpha(j) A(j,j)\big)^{-1}$
| // berechne $\overline{Q}^{(j)} b = b + v_x v$
| $b(j:m) = b(j:m) + v_x A(j:m,j)$
end

4.4.8 Aufwand für den Householder-QR-Algorithmus

a) Falls $m \approx n$ sind ungefähr $\frac{2}{3}n^3$ Multiplikationen notwendig und ist somit ungefähr doppelt so teuer wie die LR-Zerlegung, ist aber stabil.

b) Falls $m \gg n$ sind ungefähr $2mn^2$ Multiplikationen notwendig. Der Aufwand ist daher ungefähr so hoch wie beim Cholesky-Verfahren für Normalgleichungen, aber stabil.

5 Numerische Lösung nichtlinearer Gleichungssysteme

Beispiel:

linear: $Ax = b$

nichtlinear: $f(x) = \sin(x) + x^3 - 4 = 0$

5.1 Einführung

Beispiel 5.1.1.

1) $f(x) = x^2 - c = 0 \Leftrightarrow x = \pm\sqrt{c}$: Berechnung der Wurzel

2) Sei p ein Polynom: Nullstellenbestimmung

3) Löse das nichtlineare Randwertproblem $-\Delta u = f(u)$ in $\Omega = (0,1) \times (0,1)$ mit $u = 0$ auf $\partial\Omega$. Mit dem Differenzenverfahren[1] ergibt sich

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

ein System nichtlinearer Gleichungen.

Nullstellenbestimmung

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

Gesucht $x^* \in D$ mit $f(x^*) = 0$

Fixpunktgleichung

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

Gesucht $x^* \in D$ mit $g(x^*) = x^*$

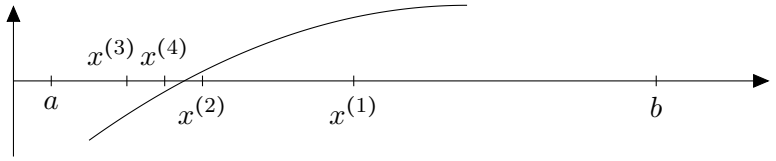
Falls m=n ist dies äquivalent zur Nullstellenbestimmung.

[1] s. Übungsaufgabe 2)

5.1.2 Das Bisektionsverfahren

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

Abb. 5.1.0 ©



Eine Idee zur Annäherung ist, eine Folge von Intervallen $[a^{(i)}, b^{(i)}] \subset [a^{(i-1)}, b^{(i-1)}]$, die eine Nullstelle enthalten zu generieren und mit $b^{(i)} - a^{(i)} \to 0$. Definiere hierfür

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

und

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

Für jedes $i \geq 1$ gilt somit

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

und es existiert eine Nullstelle x^* in $[a^{(i)}, b^{(i)}] \in [a^{(i-1)}, b^{(i-1)}]$ für alle i. Damit folgt

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

Also $\lim_{i\to\infty} x^{(i)} = x^*$.

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

Bemerkung 5.1.4.

a) $x^{(i)}$ wird als Intervallmitte, also unabhängig von $f(x^{(i)})$ gewählt. die Konvergenzgeschwindigkeit hängt von der Länge des Intervalls $[a,b]$ und der Lage von x^* bezüglich der Intervallhalbierung ab. Die Konvergenz kann demnach sehr langsam sein.

b) Ein Vorteil ist, dass keine Differenzierbarkeitsvoraussetzungen nötig sind.

c) Das Verfahren ist nicht für $f: D \to \mathbb{R}^n$ anwendbar.

5.2 Fixpunktiteration

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

$$x^* = g(x^*) \tag{5.2.1}$$

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

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

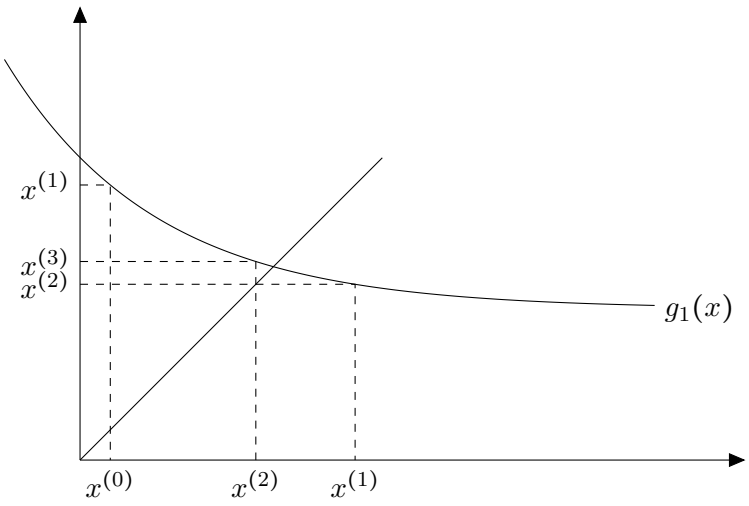
Es bedarf noch der Voraussetzung, dass $x^{(k)} \in D \ \forall k$. Falls $x^{(k)}$ konvergiert, ist der Grenzwert x^* ein Fixpunkt, denn für stetiges g gilt:

$$x^* = \lim_{k\to\infty} x^{(k+1)} = \lim_{k\to\infty} g(x^{(k)}) \overset{g \text{ stetig}}{=} g(\lim_{k\to\infty} x^{(k)}) = g(x^*) \tag{5.2.3}$$

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

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

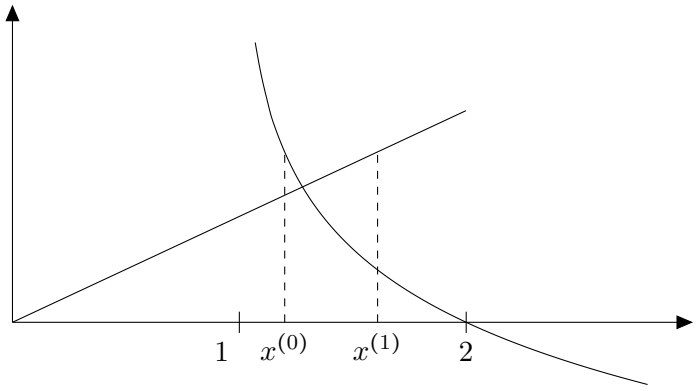
Abb. 5.2.0 © *Konvergenz der Fixpunktiteration für* $x = 1 + e^{-x}$



$\longrightarrow$ Konvergenz

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

Abb. 5.2.1 *©Versagen der Fixpunktiteration für* $x = -ln(x-1)$



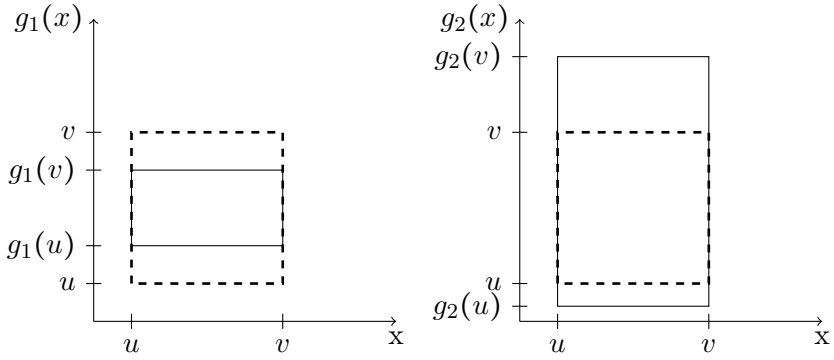
$\longrightarrow g(x^{(2)})$ nicht definiert!

Definition 5.2.2. Sei $D \subseteq \mathbb{R}^n$ abgeschlossen und $\|\bullet\|$ eine Norm auf dem $\mathbb{R}^n$. Eine Abbildung $g\colon D \to \mathbb{R}^n$ heißt **Kontraktion** bezüglich $\|\bullet\|$, 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 κ heißt Kontraktionszahl von g.

Abb. 5.2.2 © *Veranschaulichung von Kontraktion:* g_1 *Kontraktion,* g_2 *nicht*



Offensichtlich ist jede auf D kontrahierende Abbildung stetig.

Lemma 5.2.3. *Sei* $D = \overline{\Omega}$ *mit* $\Omega \subseteq \mathbb{R}^n$ *offen und konvex und* $\|\bullet\|$ *eine Norm auf dem* R^n. *Falls* $g\colon D \to D$ *eine stetig differenzierbare Funktion ist und bezüglich der zugeordneten Matrixnorm* $\sup_{x\in\Omega} \|Dg(x)\| < 1$ *gelte, so ist* g *kontrahierend bezüglich* $\|\bullet\|$.

Beweis. Mit $u, v \in D$ gilt $u + t(v-u) \in D$, da D konvex ist. Somit ist $h\colon [0,1] \to \mathbb{R}^n$ mit $h(t) \coloneqq g(u + t(v-u))$ wohldefiniert und stetig differenzierbar. Mit dem Hauptsatz der

Differenzial- und Integralrechnung folgt:

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

□

Satz 5.2.4 (Banachscher Fixpunktsatz)**.** *Sei $D \subset \mathbb{R}^n$ abgeschlossen und die Abbildung $g: D \to \mathbb{R}^n$ eine Kontraktion. Dann gilt:*

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

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

$$x^{(k+1)} := g(x^{(k)}) \overset{k \to \infty}{\longrightarrow} x^* \qquad (5.2.5)$$

3) Es gelte die a posteriori Fehlerabschätzung

$$\left\| x^{(k)} - x^* \right\| \leq \frac{\kappa}{1-\kappa} \left\| x^{(k)} - x^{(k-1)} \right\| \qquad (5.2.6)$$

und die a priori Fehlerabschätzung

$$\left\| x^{(k)} - x^* \right\| \leq \frac{\kappa^k}{1-\kappa} \left\| x^{(1)} - x^{(0)} \right\| \qquad (5.2.7)$$

19.11.2014

Beweis. **zu 2)** Sei $x_0 \in D$ beliebig. (5.2.5) ist wohldefiniert, da $g(D) \subset D$. $(x^{(k)})_{k \in \mathbb{N}}$ bildet eine Cauchyfolge, da

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

Daraus folgt, dass $\lim_{k\to\infty} x^{(k)} = x^*$ existiert und $x^* \in D$, da D abgeschlossen ist und somit vollständig.

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

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

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

zu 3) Betrachte

$$\left\|x^* - x^{(k)}\right\| = \lim_{l\to\infty} \left\|x^{(k+l)} - x^{(k)}\right\| \le \frac{\kappa^k}{1-\kappa} \left\|x^{(1)} - x^{(0)}\right\|$$

bzw.

$$\begin{aligned}\lim_{l\to\infty} \left\|x^{(k+l)} - x^{(k)}\right\| &\le \lim_{l\to\infty} \sum_{i=0}^{l-1} \left\|x^{(k+i+1)} - x^{(k+i)}\right\| \\ &\le \lim_{l\to\infty} \sum_{i=0}^{l-1} \kappa^{i+1} \left\|x^{(k)} - x^{(k-1)}\right\| \\ &\le \frac{\kappa}{1-\kappa} \left\|x^{(k)} - x^{(k-1)}\right\|\end{aligned}$$

□

Bemerkung 5.2.5.

1) Als Voraussetzung wäre bereits ausreichend: D ist vollständiger metrischer Raum mit Metrik d. Dann ersetze die Norm durch die Metrik d.

2) Im Allgemeinen ist der Nachweis $g(D) \subset D$ schwierig.

Folgerung 5.2.6. *Sei $x^* \in \mathbb{R}^n$, so dass $g(x^*) = x^*$, und sei g in einer Umgebung von $\overline{B_\varepsilon(x^*)} = \{x \in \mathbb{R}^n | \|x - x^*\| \le \varepsilon\}$ stetig differenzierbar und es gelte $\|g'(x)\| < 1$ für $x \in \overline{B_\varepsilon(x^*)}$, 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** $p \ge 1$ gegen x^*, falls $\lim_{k\to\infty} x^{(k)} = x^*$ und falls es ein $C > 0$ sowie $N \in \mathbb{N}$ gibt, so dass

$$\left\|x^{(k+1)} - x^*\right\| \le C \left\|x^{(k)} - x^*\right\|^p \qquad \forall k \ge N$$

Im Fall $p = 1$ ist zusätzlich $C < 1$ und man spricht von **linearer Konvergenz**. Für $p = 2$ heißt es **quadratische Konvergenz**. Gilt

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

so konvergiert die Folge **superlinear**.

Bemerkung 5.3.2. Die Fixpunktiteration konvergiert unter der Voraussetzung in 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.

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 $U\backslash\{x^*\}$ die erzeugte Folge mit Ordnung p konvergiert.

b) Das Verfahren heißt **lokal konvergent**, falls es für alle Startwerte in einer Umbegung 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

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

hat die Folge genau die Konvergenzordnung 1. Weiter gilt mit $A_k = \frac{x^{(k)} - x^{(k-1)}}{x^{(k-1)} - x^{(k-2)}}$

$$\lim_{k\to\infty} \frac{A_k}{1 - A_k} \cdot \frac{x^{(k)} - x^{(k-1)}}{x^* - x^{(k)}} = 1 \qquad \textit{sowie} \qquad \lim_{k\to\infty} A_k = A \tag{5.3.2}$$

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

$$\lim_{k\to\infty} \frac{x^{(k)} - x^{(k-1)}}{x^* - x^{(k)}} = 1 \tag{5.3.3}$$

***zu Def.** 5.3.1: Im Fall $p = 1$ ist zusätzlich $C < 1$ verlangt.*

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

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

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

□

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

a) Für $p = 1$, große k und A_k in etwa konstant gilt

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

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

b) Für $p > 1$ und für große k gilt

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

Bemerkung 5.3.7. Für Folgen im $\mathbb{R}^n$ 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, d.h.

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

Beweis. Nutze $\lim_{k\to\infty} \frac{\left\|e^{(k+1)}\right\|}{\left\|e^{(k)}\right\|} = 0$ und

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

□

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

$$p \approx \frac{\log(\left\|x^{(k+2)} - x^{(k+1)}\right\|)}{\log(\left\|x^{(k+1)} - x^{(k)}\right\|)}$$

Beweis. Siehe Übungsaufgabe. □

5.4 Newton-Verfahren für skalare Gleichung

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

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

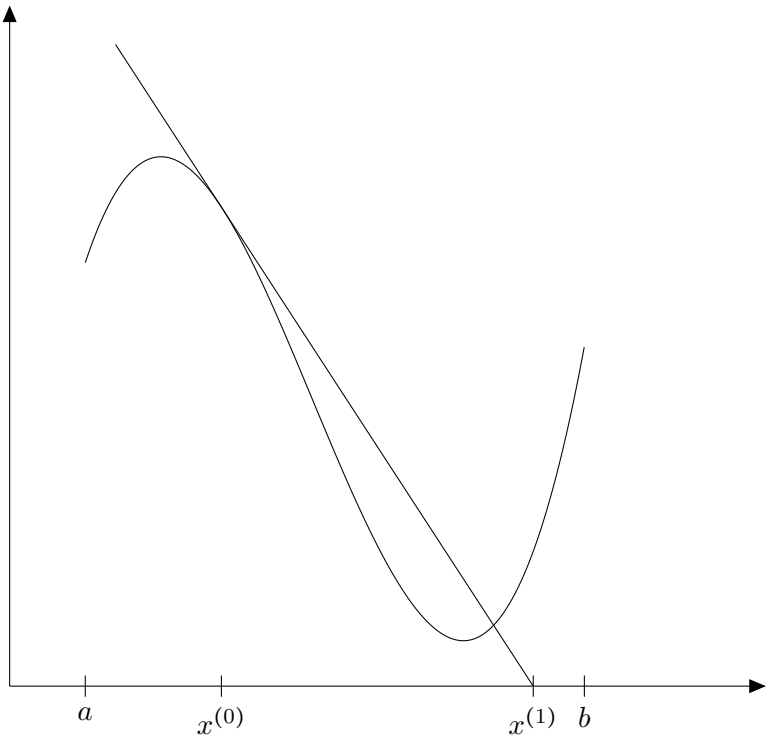
d.h. f kann lokal gut durch die Tangente approximiert werden.

Betrachte für das Newton-Verfahren nun die Nullstellengleichung $f(x^*) = 0$ und bestimme iterativ die Nullstelle der Tangentengleichung

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

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

Abb. 5.4.0 *©Newton-Verfahren bei einem Funktionsgraphen*



5.4.1 Iterationsschritt des Newton(-Kantorowitsch)-Verfahrens

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

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

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 $\varepsilon > 0$, s.d. 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. Gleichung (5.4.1) definiert eine Fixpunktiteration mit $g(x) = x - \frac{f(x)}{f'(x)}$. Für $f \in C^2(a,b)$ gilt

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

Da $f(x^*) = 0$ und $f'(x^*) \neq 0$ gilt $g'(x^*) = 0$. Weiterhin gibt es eine Umgebung U_0 von x^*, in der $f(x) \neq 0$ $\forall x \in U_0$, da f stetig ist. In U_0 ist somit g' stetig. Da $g'(x^*) = 0$ ist, existiert 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 $g(\overline{B_\varepsilon(x^*)}) \subset \overline{B_\varepsilon(x^*)}$. Der Banachsche Fixpunktsatz liefert Konvergenz für alle $x^{(0)} \in \overline{B_\varepsilon(x^*)}$.

Die quadratische Konvergenz folgt aus

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

aufgrund der Taylorentwicklung und da $x^{(k)} \in \overline{B_\varepsilon(x^*)}$ $\forall k \in \mathbb{N}$ (da g Kontraktion).

Für $f \in C^1$ siehe [HH94]. □

Bemerkung 5.4.3.

a) Mehrfache Nullstellen können im Allgemeinen nicht mit (5.4.1) bestimmt werden.

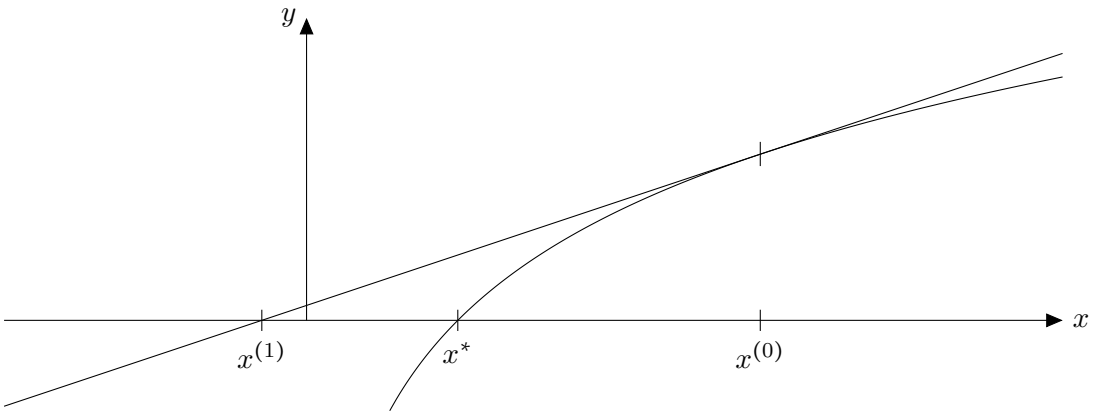
b) Die Ableitung f' muss analytisch (als Funktion) gegeben sein.

c) Die Lage und Größe des Konvergenzintervalls ist a priori unbekannt.
(Hierfür könnte z.B. das Bisektionsverfahren Anwendung finden.)

24.11.2014 **Beispiel 5.4.4** (Newton-Verfahren ohne Konvergenz)**.**

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

Abb. 5.4.1 *©Fehlschlagen des Newton-Verfahrens: außerhalb des Definitionsbereichs*



- $|x^* - x^{(1)}| \not< |x^* - x^{(0)}|$ ist keine Kontraktion, da die Konvergenz nicht gesichert ist.

5.4.5 Newton-Verfahren: Iterativer Linearisierungsprozess

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

Beispiel 5.4.6. Es ist die Lösung von $x - e^{-\frac{1}{2}x} = 0$ mit $x^{(0)} = 0,8$ gesucht.

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

$$x^{(10)} = \mathbf{0,7034}7017 \qquad \text{auf 4 Stellen exakt}$$

b) Mit dem Newton-Verfahren

$x^{(3)} = 0,70346742$	bis auf 17 Stellen exakt
$x^{(4)}$	bis auf Maschinengenauigkeit exakt

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

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

Damit ergibt sich

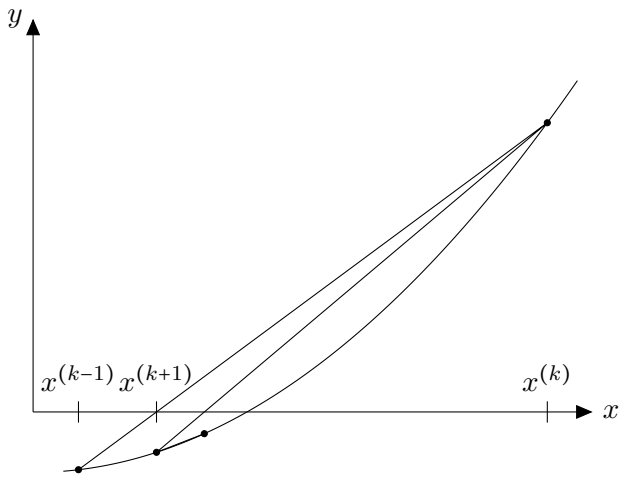
$$x^{(k+1)} = x^{(k)} - f(x^{(k)}) \frac{x^{(k)} - x^{(k-1)}}{f(x^{(k)}) - f(x^{(k-1)})}$$

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

5.4.7 Iterationsschritt des Sekantenverfahrens

$$x^{(k+1)} = \frac{x^{(k-1)} f(x^{(k)}) - x^{(k)} f(x^{(k-1)})}{f(x^{(k)}) - f(x^{(k-1)})} \tag{5.4.2}$$

Abb. 5.4.2 *©Geometrische Veranschaulichung Sekantenverfahren*



Satz 5.4.8 (Konvergenz des Sekantenverfahrens)**.** *Sei $f \in C^2([a,b])$ und $x^* \in (a,b)$ eine einfache Nullstelle. Dann konvergiert das Sekantenverfahren in einer Umbegung von x^* superlinear mit Ordnung*

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

Beweis. Siehe z.B. [HH94; SB90, Zwischenwertsatz, Fibonacci-Folge] □

zu Beispiel 5.4.6: Das Sekantenverfahren benötigt einen zweiten Startwert, z.B.

$$\begin{aligned} & x^{(1)} = 0,7 \\ \Rightarrow\ & x^{(3)} = 0,7034674 && \text{auf 7 Stellen exakt} \\ & x^{(6)} && \text{bis auf Maschinengenauigkeit exakt} \end{aligned}$$

Bemerkung 5.4.9.

a) Das Verfahren ist keine Fixpunktiteration. Es benötigt $x^{(k)}$ und $x^{(k-1)}$ für $x^{(k+1)}$ (**Mehrschrittverfahren**)

b) Die Berechnung von $f(x)$ und $f'(x)$ ist im Allgemeinen sehr teuer. Das Sekanten-Verfahren benötigt pro Iteration nur eine Funktionsauswertung, das Newton-Verfahren hingegen zwei. 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 $p^2 = 2,618\ldots$ für $x^{(k)} \to x^{(k+2)}$ (siehe auch Beispiel 5.4.6).

   Beispiel: Sei $f : \mathbb{R}^n \to \mathbb{R}$ und $f(x)$ die erste Komponente von $A^{-1}x$. Diese n-dimensionale Funktionsauswertung benötigt $\mathcal{O}(n^3)$ flops.

c) 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 $f: \Omega \subseteq \mathbb{R}^n \to \mathbb{R}^n$ linearisiert

$$f(\overline{x}) \approx f(x) + \mathrm{D}\, f(x)(\overline{x} - x) \tag{5.5.1}$$

mit

$$\mathrm{D}\, f(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x) & \ldots & \frac{\partial f_1}{\partial x_n}(x) \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1}(x) & \ldots & \frac{\partial f_n}{\partial x_n}(x) \end{pmatrix} \qquad \text{(genannt: die Jacobi-Matrix von f)}$$

Falls nun die Jacobi-Matrix $\mathrm{D}\, f(x)$ invertierbar ist und $f(\overline{x}) = 0$ gilt, folgt

$$\overline{x} = x - [\mathrm{D}\, f(x)]^{-1} \cdot f(x)$$

5.5.1 Iterationsschritt des Newton-Verfahrens

$$x^{(k+1)} = x^{(k)} - [\mathrm{D}\, f(x^{(k)})]^{-1} \cdot f(x^{(k)}) \tag{5.5.2}$$

5.5.2 Newton-Verfahren

setze Startwert x
$i = 0$
$fx = f(x)$
while „Abbruchkriterium“
| $\mathrm{D}\, fx = \mathrm{D}\, f(x)$
| Löse[2] $\mathrm{D}\, fx \cdot d = -fx$
| $x = x + d$
| $fx = f(x)$
| $i = i + 1$
end

Bemerkung 5.5.3. Ein Newton-Iterationsschritt (5.5.2) wird also aufgeteilt in Berechnung der sogenannten **Newton-Korrektur**

$$\mathrm{D}\, f(x^{(k)})\Delta x^{(k)} = -f(x^{(k)}) \tag{5.5.3}$$

und dem **Korrekturschritt**

$$x^{(k+1)} = x^{(k)} + \Delta x^{(k)} \tag{5.5.4}$$

5.5.4 Aufwand pro Iteration

n eindimensionale Funtionsauswertungen für $f(x)$

n^2 eindimensionale Funtionsauswertungen für $\mathrm{D}\, f(x)$

$\mathcal{O}(n^2)$ flops (i.d.R.) zum Lösen eines GLS

26.11.2014

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 $\widetilde{f}(x) \coloneqq A \cdot f(x) = 0$ mit regulärem $A \in \mathbb{R}^{n\times n}$ gelöst wird. Dies gilt, da

$$\begin{aligned}[D\widetilde{f}(x)]^{-1} \cdot \widetilde{f}(x) &= [A \cdot \mathrm{D}\, f(x)]^{-1} \cdot (A \cdot f(x)) \\ &= [\mathrm{D}\, f(x)]^{-1} \cdot f(x)\end{aligned}$$

und damit ist die Newton-Korrektur $\Delta x^{(k)}$ affin-invariant.

Satz 5.5.6. *Sei $\Omega \in \mathbb{R}^n$ offen und $f{:}\Omega \to \mathbb{R}^n$ in $C^2(\Omega)$. Sei $x^* \in \Omega$ eine Nullstelle f mit einer invertierbaren Jacobi-Matrix $\mathrm{D}\, f(x^*)$. Dann existiert eine Umgebung von x^*, so dass das Newton-Verfahren für jeden Startwert $x^{(0)}$ in dieser Umgebung quadratisch gegen x^* konvergiert.*

[2] entspricht $Ax = b$

Beweis. Der Beweis kann wie im eindimensionalen durchgeführt werden. Aber Vorsicht: $D^2 f(x)$ ist eine **bilineare Abbildung** in $\mathcal{L}(\mathbb{R}^n, \mathcal{L}(\mathbb{R}^n, \mathbb{R}^n))$.

Wir zeigen die Behauptung induktiv über die quadratische Konvergenz. Da $\mathrm{D} f(x^*)$ 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 $\mathrm{D} f(x)$ invertierbar und stetig ist. Sei

$$c \coloneqq \sup_{x \in B_\varepsilon(x^*)} \|[\mathrm{D} f(x)\|^{-1}]$$

und

$$w \coloneqq \sup_{x \in B_\varepsilon(x^*)} \left\|D^2 f(x)\right\|$$

Für $x^{(k)} \in B_\varepsilon(x^*)$ ist $x^{(k)} + t(x^* - x^{(k)}) \in B_\varepsilon(x^*)$ für $t \in [0,1]$ und

$$h^{(k)}(t) \coloneqq f(x^{(k)} + t(x^* - x^{(k)})) \qquad \forall t \in [0,1]$$

ist wohldefiniert und in $C^2([0,1], \mathbb{R}^n)$. Wie in 5.4.2 folgt

$$\begin{aligned} x^{(k+1)} - x^* &= [\mathrm{D} f(x^{(k)})]^{-1} \left(f(x^*) - f(x^{(k)}) - \mathrm{D} f(x^{(k)})(x^* - x^{(k)}) \right) \\ &= [\mathrm{D} f(x^{(k)})]^{-1} \left(h^{(k)}(1) - h^{(k)}(0) - Dh^{(k)}(0) \cdot 1 \right) \\ &= [\mathrm{D} f(x^{(k)})]^{-1} \int_0^1 D^2 h^{(k)}(1-t) dt \qquad \text{(Restglieddarst. der Taylorentw.)} \end{aligned}$$

Das Ziel ist nun zu zeigen, dass $\left\|x^{(k+1)} - x^*\right\| \le c \cdot \left\|x^{(k)} - x^*\right\|^2$. Mit den Definitionen von oben wird die Ungleichung zu

$$\begin{aligned} \left\|x^{(k+1)} - x^*\right\| &\le c \cdot \frac{1}{2} \sup_{t \in [0,2]} \left\|D^2 h^{(k)}(t)\right\| \\ &\le c \cdot \frac{1}{2} w \left\|x^{(k)} - x^*\right\|^2 \end{aligned} \tag{5.5.5}$$

und zwar für alle $x^{(k)} \in B_\varepsilon(x^*)$, wie noch gezeigt wird. Sei nun $\delta \le \varepsilon$, so dass $\frac{1}{2} \cdot c \cdot w < 1$ gilt, so folgt induktiv für $x^{(0)} \in B_\delta(x^*)$ mit (5.5.5)

$$\begin{aligned} \left\|x^{(k+1)} - x^*\right\| &\le \frac{1}{2} w \delta^2 < \delta \\ &\Rightarrow x^{(k+1)} \in B_\delta(x^*) \subseteq B_\varepsilon(x^*) \end{aligned}$$

Auf $x^{(k+1)}$ ist der nächste Iterationsschritt anwendbar und mit (5.5.5) folgt quadratische Konvergenz. Es bleibt zu zeigen:

$$\left\|D^2 h(t)\right\| \le w \left\|x^{(k)} - x^*\right\|^2 \qquad \forall t \in [0,1],\ h : [0,1] \to \mathbb{R}^n,\ h = \begin{pmatrix} h_1 \\ \vdots \\ h_n \end{pmatrix}$$

Hierfür betrachte

$$Dh_i^{(k)}(t) = \underbrace{\mathrm{D}\, f_i\left(x^{(k)} + t(x^* - x^{(k)})\right)}_{\in \mathbb{R}^{1\times n}} \cdot (x^* - x^{(k)}) \in \mathbb{R}$$

$$D^2 h_i^{(k)}(t) = (x^* - x^{(k)})^T \cdot \underbrace{D^2 f_i\left(x^{(k)} + t(x^* - x^{(k)})\right)}_{\in \mathbb{R}^{n\times n}} \cdot (x^* - x^{(k)}) \in \mathbb{R}$$

□

Unter genaueren Voraussetzungen kann die Existenz von x^* gezeigt und eine Umgebung $B_r(x^*)$ explizit angegeben werden.

Dies liefert folgender Satz:

Satz (Satz von Kantorowitsch)**.** *Sei $f: \Omega_0 \to \mathbb{R}^n$, $\Omega_0 \subset \mathbb{R}^n$ konvex, f stetig differenzierbar auf Ω_0 und erfülle für ein $x^{(0)} \in \Omega_0$ folgendes:*

a) $\|\mathrm{D}\, f(x) - \mathrm{D}\, f(y)\| \le \gamma \|x - y\|$ *für alle* $x, y \in \Omega_0$

b) $\left\|[\mathrm{D}\, f(x^{(0)})\right\|^1] \le \beta$

c) $\left\|[\mathrm{D}\, f(x^{(0)})\right\|^1 f(x^{(0)})] \le \alpha$

mit den Konstanten $h = \alpha\beta\gamma$, $r_\pm = \frac{1\pm\sqrt{1-2h}}{h}\alpha$. Dann hat f, falls $h \le \frac{1}{2}$ und $\overline{B_{r_-}(x^{(0)})} \subset \Omega$, genau eine Nullstelle x^* in $\Omega_0 \cap B_{r_+}(x^{(0)})$. Weiterhin bleibt die Folge der Newton-Iterierten in $B_{r_-}(x^{(0)})$ und konvergiert gegen x^*.

Beweis. z.B. in Ortega/Rheinhold (2000) □

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 bleibt.

3) Breche ab, wenn das Ergebnis genau genug ist, d.h. der Fehler $e^{(k)} := \left\|x^* - x^{(k)}\right\|$ 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 $\kappa \in (0,1)$ für alle k geben mit

$$\begin{aligned}\left\|\Delta x^{(k)}\right\| &= \left\|x^{(k+1)} - x^{(k)}\right\| \\ &= \left\|g(x^{(k)}) - g(x^{(k-1)})\right\| \\ &\leq \kappa \left\|x^{(k)} - x^{(k-1)}\right\| = \kappa \left\|\Delta x^{(k-1)}\right\| \end{aligned} \tag{5.6.1}$$

Als Abbruchkriterium für eine (mögliche) Divergenz des Verfahrens wähle z.B. $\kappa = \frac{1}{2}$ und breche ab, falls

$$\left\|\Delta x^{(k)}\right\| > \frac{1}{2}\left\|\Delta x^{(k-1)}\right\| \tag{5.6.2}$$

Um im Mehrdimensionalen eine vielleicht unnötig (teure) Berechnung von $\mathrm{D}\, f(x^{(k)})$ bzw. von $\Delta x^{(k)}$ zu vermeiden, kann $\Delta x^{(k)}$ durch

$$\overline{\Delta x}^{(k)} = -[\mathrm{D}\, f(x^{(k-1)})]^{-1} \cdot f(x^{(k)}) \tag{5.6.3}$$

approximiert werden. $\mathrm{D}\, f(x^{(k-1)})$ und eine 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) kann dann auch auf

$$\left\|\overline{\Delta x}^{(k)}\right\| \geq \frac{1}{2}\left\|\Delta x^{(k-1)}\right\| \tag{5.6.4}$$

getestet werden.

5.6.2 Kriterium für erreichte Konvergenz

Es ist $f(x^*)$ gesucht, also teste hierauf. Das residuumbasierte Kriterium

$$\left\|f(x^{(k)})\right\| \leq \mathrm{tol} \tag{5.6.5}$$

ist nur bedingt anwendbar, denn nach 5.5.5 ist das Verfahren affin-invariant. Demnach bleibt $(x^{(k)})_{k\in\mathbb{N}}$ gleich, ob nun $f(x)$ oder $\widetilde{f}(x) = \alpha f(x)$ betrachtet wird. Aber für $\widetilde{f}$ bricht (5.6.5) das Verfahren ab, falls $|\alpha| \cdot \left\|f(x^{(k)})\right\| \leq \mathrm{tol}$. Affin-invariant ist dagegen der Ansatz

$$\left\|\Delta x^{(k)}\right\| = \left\|x^{(k+1)} - x^{(k)}\right\| = \left\|[\mathrm{D}\, f(x^{(k)})]^{-1} f(x^{(k)})\right\| \leq \mathrm{tol}\,. \tag{5.6.6}$$

(5.6.6) kann aufgrund der quadratischen Konvergenz (nur) für große k auch mit (5.3.5) der Approximation des Fehlers $\left\|x^* - x^{(k)}\right\|$ motiviert werden.

5.7 Varianten des Newton-Verfahrens

$\mathrm{D}\, f(x^{(k)})$ steht nicht immer analytisch zur Verfügung. Die exakte Jacobi-Matrix wird häufig durch eine andere Matrix B approximiert, z.B. durch Differenzenquotienten oder sogenanntes „automatisches Differenzieren“. Der Iterationsschritt lautet dann

$$\begin{aligned} \text{löse} \quad B^{(k)} d^{(k)} &= -f(x^{(k)}) \\ x^{(k+1)} &= x^{(k)} + d^{(k)} \end{aligned} \tag{5.7.1}$$

Um den Aufwand zu verringern kann $\mathrm{D}\, f(x^{(k)})$ durch $\mathrm{D}\, f(x^{(0)})$ approximiert werden.

5.7.1 Iterationsschritt des vereinfachten Newton-Verfahrens

$$x^{(k+1)} = x^{(k)} - [\mathrm{D}\, f(x^{(0)})]^{-1} f(x^{(k)}) \tag{5.7.2}$$

Das Verfahren konvergiert nur noch lokal linear. Der Aufwand je Iteration ist jedoch erheblich geringer.

5.7.2 Das Broyden-Verfahren

01.12.2014

Das Broyden-Verfahren ist eine Verallgemeinerung des Sekantenverfahrens auf $n > 1$. $\mathrm{D}\, f(x^{(k)})$ wird durch den „Differenzenquotienten“ approximiert, d.h.

$$B^{(k)} (\underbrace{x^{(k)} - x^{(k-1)}}_{:= p^{(k-1)}}) = \underbrace{f(x^{(k)}) - f(x^{(k-1)})}_{:= q^{(k-1)}} \tag{5.7.3}$$

$B^{(k)}$ ist jedoch nicht eindeutig durch (5.7.3) festgelegt. Das Broyden-Verfahren bestimmt $B^{(k)}$ rekursiv durch eine Aufdatierung mit einer Rang-1-Matrix, auch „rang-1-update“ ($C_{\text{neu}} = C_{\text{alt}} + M$ mit $\operatorname{rang}(M) = 1$).

Ein Iterationsschritt des Broyden-Verfahrens ist

$$\begin{aligned} d^{(k)} &= -[B^{(k)}]^{-1} f(x^{(k)}) \\ x^{(k+1)} &= x^{(k)} + d^{(k)} \\ p^{(k)} &:= d^{(k)} \qquad \text{nach (5.7.1)} \\ q^{(k)} &:= f(x^{(k+1)}) - f(x^{(k)}) \end{aligned}$$

$$B^{(k+1)} = B^{(k)} + \frac{1}{{p^{(k)}}^T \cdot p^{(k)}} \cdot \left(q^{(k)} - \underbrace{B^{(k)} p^{(k)}}_{\substack{= -f(x^{(k)}) \\ \text{nach (5.7.1)}}} \right) {p^{(k)}}^T \tag{5.7.4}$$

Hierfür muss $x^{(0)}$ und $B^{(0)}$ gegeben sein. Unter bestimmten Voraussetzungen konvergiert das Verfahren lokal superlinear [siehe SB90, dortige Referenzen]. Der fleißige Leser vergewissere sich, dass für (5.7.4) auch (5.7.3) gilt.

5.7.3 Das gedämpfte Newton-Verfahren

Es gilt

$$f(x^*) = 0 \qquad \Longleftrightarrow \qquad f(x^*) = \min_{x \in \mathbb{R}^n} \frac{1}{2} \|f(x)\|_2^2 \tag{5.7.5}$$

Betrachte nun die Funktion $\Phi: \mathbb{R}^n \to \mathbb{R}$ mit

$$\Phi(x) \coloneqq \frac{1}{2} \|f(x)\|_2^2 = \frac{1}{2} f(x)^T f(x)$$

Für Φ ist die Newton-Korrektur $d^{(k)} \coloneqq \Delta x^{(k)} \coloneqq -[\mathrm{D}\, f(x)]^{-1} f(x)$ in $x^{(k)}$ eine **Abstiegsrichtung**, d.h. für $\mu > 0$ klein genug gilt

$$\Phi(x^{(k)} + \mu d^{(k)}) < \Phi(x^{(k)}) \tag{5.7.6}$$

denn für $f(x) \neq 0$

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}\mu} \Phi(x + \mu d)\Big|_{\mu=0} &= \left[f(x + \mu d)^T \,\mathrm{D}\, f(x + \mu d) d \right]_{\mu=0} \\ &= -f(x)^T f(x) \\ &< 0 \end{aligned}$$

Die Idee ist nun, statt $\mu = 1$ wie im Newton-Verfahren ein „geeignetes“ $\mu \in (0, 1]$ zu wählen und

$$x^{(k+1)} = x^{(k)} + \mu d^{(k)} \tag{5.7.7}$$

entsprechend zu setzen, d.h. dämpfe d mit Schrittweite μ. Ein möglicher „Eignungstest“ ist

$$\left\| f(x^{(k)} + \mu d^{(k)}) \right\| \leq (1 - \frac{1}{2}\mu) \left\| f(x^{(k)}) \right\| \tag{5.7.8}$$

Und eine mögliche Strategie um μ zu bestimmen ist

1. Setze $\mu = 1$.
2. Halbiere μ rekursiv solange, bis (5.7.8) gilt.

Es sind allerdings effektivere Dämpfungsstrategien bekannt!

Es gibt also eine äußere Iteration (k) um x^* zu bestimmen und eine Innere, um für jedes (k) ein geeignetes μ zu berechnen. Die innere Schleife ist mit n eindimensionalen Funktionsauswertungen „billig“. Unter bestimmten Voraussetzungen ist **globale** Konvergenz gewährleistet.

6 Interpolation

Es seien diskrete Werte $f_i = f(t_i)$ und eventuell $f_i^{(j)} = f^{(j)}(t_i)$ an den Punkten t_i gegeben (z.B. Messdaten). Gesucht ist nun eine zugehörige Funktion φ.

Anwendungsfeld solcher Probleme ist z.B. CAD (computer aided design).

Eine naheliegende Forderung ist die **Interpolationseigenschaft**

$$\varphi^{(j)}(t_i) = f_i^{(j)} \qquad \text{gegeben für alle } i, j,$$

d.h. φ und f sollen an den **Knoten** oder **Stützstellen** t_i übereinstimmen. Die $f_i^{(j)}$ heißen Stützwerte. φ soll zusätzlich meist leicht an einer beliebigen Stelle t auswertbar sein. Es bietet sich z.B. an

- Polynome: $\varphi(t) = a_0 + \cdots + a_n t^n$
- rationale Funktion: $\varphi(t) = \frac{p(t)}{q(t)}$ (insbesondere, falls Pole vorliegen)
- Splines, d.h. stückweise Polynome
- trigonometrische Funktionen, Fouriertransformationen: $\varphi(t) = \sum_{j=1}^{\infty} a_j e^{-2\pi i j t}$ (insbesondere für periodische Funktionen)

6.1 Polynom-Iterpolation

Wir definieren als

$$\mathcal{P}_n \coloneqq \left\{ p \in \mathbb{R}[x] \,\middle|\, p = \sum_{i=0}^{n} a_i x^i \text{ mit } a_i \in \mathbb{R} \right\} \tag{6.1.1}$$

den Raum aller Polynome mit $\deg(p) \leq n$. Die Monome $1, x, \ldots, x^n$ bilden eine Basis von $\mathcal{P}_n$ und es gilt $\dim \mathcal{P}_n = n + 1$.

6.1a) Lagrangesche Interpolationsformel und Lemma von Aitken

Satz 6.1.1. *Zu beliebigen $n+1$ Stützpunkten $(x_i, f_i)_{i=0,\dots,n}$ mit $x_i \neq x_k$ für $i \neq k$, gibt es genau ein $p \in \mathcal{P}_n$ mit $p(x_i) = f_i$ für $i = 0, \dots, n$. p heißt **Interpolationspolynom** und wird mit $p(f \mid x_0, \dots, x_n)$ bezeichnet.*

Beweis.

Eindeutigkeit Seien $p_1, p_2 \in \mathcal{P}_n$ mit $p_1(x_i) = p_2(x_i) = f_i$ für $i = 0, \dots, n$. Dann ist $p \coloneqq p_1 - p_2 \in \mathcal{P}_n$ mit mindestens $n+1$ Nullstellen. Daraus folgt bereits, dass $p \equiv 0$, also $p_1 \equiv p_2$.

Existenz Betrachte die Polynome $L_i \in \mathcal{P}_n$ für $i = 0, \dots, n$ mit

$$L_i(x_k) = \delta_{ik} \quad \text{für } i, k = 0, \dots, n \tag{6.1.2}$$

Diese sind gegeben durch

$$L_i(x) \coloneqq \frac{(x - x_0)\cdots(x - x_{i-1})(x - x_{i+i})\cdots(x - x_n)}{(x_i - x_0)\cdots(x_i - x_{i-1})(x_i - x_{i+i})\cdots(x_i - x_n)} \tag{6.1.3}$$

und sind nach a) eindeutig. Sie heißen **Lagrange-Polynome**. Die **Lagrange-Interpolationsformel**

$$p(x) = \sum_{i=0}^{n} f_i L_i(x) \tag{6.1.4}$$

bestimmt somit $p(f \mid x_0, \dots, x_n)$, da $p \in \mathcal{P}_n$ und $p(x_i) = f_i$ für $i = 0, \dots, n$.

□

Mit (6.1.4) folgt weiterhin

a) p hängt linear von den Stützwerten f_i ab.

b) Die Lagrange-Polynome bilden eine Basis des $\mathcal{P}_n$.

Die Lagrange-Interpolationsformel ist, wenn auch für theoretische Fragen günstig, für praktische Zwecke zu rechenaufwändig.

Zur Auswertung des Interpolationspolynoms $p(f \mid x_0, \dots, x_n)$ an einer festen Stelle $\overline{x}$, d.h. um den Wert $p(\overline{x})$ zu berechnen, benötigt man:

Lemma 6.1.2 (Lemma von Aitken)**.** *Für das Interpolationspolynom $p(f \mid x_0, \dots, x_n)$ gilt für $n > 0$ die Rekursionsformel*

$$p(f \mid x_0, \dots, x_n)(x) = \frac{(x_0 - x)p(f \mid x_1, \dots, x_n)(x) - (x_n - x)p(f \mid x_0, \dots, x_{n-1})(x)}{(x_0 - x_n)} \tag{6.1.5}$$

Beweis. Sei die rechte Seite von (6.1.5) definiert als $\varphi(x)$. Dann ist $\varphi \in \mathcal{P}_n$ und

$$\varphi(x_i) = \frac{(x_0 - x_i)f_i - (x_n - x_i)f_i}{x_0 - x_n} = f_i \quad \text{für } i = 1, \dots, n-1$$
$$\varphi(x_0) = \frac{0 - (x_n - x_0)f_0}{x_0 - x_n} = f_0 \quad \text{für } i = 0$$
$$\varphi(x_n) = \frac{(x_0 - x_n)f_n - 0}{x_0 - x_n} = f_n \quad \text{für } i = n$$

Damit ist $p(f \mid x_0, \dots, x_n) = \varphi$ aufgrund der Eindeutigkeit. □

Weiterhin gilt 03.12.2014

$$p(f \mid x_i)(x) = f_i \quad \forall x \in \mathbb{R} \tag{6.1.6}$$

Definiere nun für ein festes $\overline{x}$

$$P_{ik} \coloneqq p(f \mid x_{i-k}, \dots, x_i)(\overline{x}) \tag{6.1.7}$$

Dann lässt sich der Wert

$$p(\overline{x}) = p(f \mid x_0, \dots, x_n)(\overline{x}) = P_{nn}$$

nach (6.1.5) wie folgt berechnen:

6.1.3 Schema von Neville

Setze $P_{i0} = f_i$ für $i = 0, \dots, n$ und

$$P_{ik} = P_{i,k-1} + \frac{\overline{x} - x_i}{x_i - x_{i-k}} \cdot \left(P_{i,k-1} - P_{i-1,k-1}\right) \quad \text{für } 1 \le k \le i \le n \tag{6.1.8}$$

$x_i \backslash k$	0		1		...		
x_0	f_0	$\searrow$					
x_1	f_1	$\rightarrow$	P_{11}	$\searrow$			
$\vdots$	$\vdots$		P_{21}	$\rightarrow$	P_{22}		
$\vdots$	$\vdots$					$\ddots$	
x_n	f_n		P_{n1}		...		P_{nn}

(6.1.8) benötigt weniger Multiplikationen als (6.1.5) und ist deutlich billiger als (6.1.4). Pro Auswertung an einer Stelle $\overline{x}$ sind $\frac{n(n+1)}{2}$ Multiplikationen und Divisionen notwendig. Soll das Interpolationspolynom an mehreren Stellen ausgewertet werden, ist das Schema von Neville zu teuer.

6.1b) Newtonsche Interpolationsformel

Betrachte folgende Basisdarstellung eines Polynoms $p \in \mathcal{P}_n$

$$p(x) = \sum_{i=0}^{n} a_i \prod_{j=0}^{i-1} (x - x_j) \tag{6.1.9}$$

Damit wird p zu

$$p(x) = \Bigg(\dots \Big(\big(a_n(x - x_{n-1}) + a_{n-1} \big) \cdot (x - x_{n-2}) + a_{n-2} \Big) \cdot (x - x_{n-3}) + a_{n-3} \Big) \quad \dots \Bigg) \tag{6.1.10}$$

Es ergibt sich für bekannte $a_i \in \mathbb{R}$:

6.1.4 Das Horner-Schema zur Auswertung p(x)

$p = a_n$
for $k = n - 1 : -1 : 0$
| $p = p \cdot (x - x_k) + a_k$
end

Pro Auswertung benötigt es n Multiplikationen. Zur Bestimmung der notwendigen Koeffizienten $a_0, \dots, a_n$ von $p(f \mid x_0, \dots, x_n)$ kann folgende Vorwärtssubstitution verwendet werden

$$\begin{aligned} f_0 &= p(x_0) = a_0 \\ f_1 &= p(x_1) = a_0 + a_1(x_1 - x_0) \\ &\vdots \\ f_n &= p(x_n) = a_0 + \dots + a_n(x_n - x_0) \cdots (x_n - x_{n-1}) \end{aligned}$$

Es geht jedoch noch billiger, wie gleich zu sehen ist.

Definition 6.1.5.

a) Die folgenden Polynome heißen **Newton-Polynome**

$$\begin{aligned} w_0(x) &:= 1 \\ w_i(x) &:= \prod_{j=0}^{i-1} (x - x_j) \qquad \text{für } i \geq 1 \end{aligned}$$

b) Die n-te dividierte Differenz $[x_0, \dots, x_n]f$ ist für $x_n \neq x_i$ definiert durch

$$\begin{aligned} [x_i]f &\coloneqq f_i \\ [x_i, \dots, x_n]f &\coloneqq \frac{[x_{i+1}, \dots, x_n]f - [x_i, \dots, x_{n-1}]f}{x_n - x_i} \end{aligned} \tag{6.1.11}$$

Satz 6.1.6. *Sei* $p(f \mid x_0, \dots, x_n)(x) = \sum_{i=0}^{n} a_i w_i(x) = a_n x^n + \sum_{i=0}^{n-1} c_i x^i$, *dann gilt*

$$a_n = [x_0, \dots, x_n]f$$

und für jede Permutation $\sigma \in S_n$

$$[x_0, \dots, x_n]f = [x_{\sigma(0)}, \dots, x_{\sigma(n)}]f$$

Somit ist $[x_0, \dots, x_n]f$ *wie* p *unabhängig von der Reihenfolge der Knoten und es gilt für* $x_l \neq x_j$

$$[x_i, \dots, x_k]f = \frac{[x_i, \dots, \widehat{x_j}, \dots, x_k]f - [x_i, \dots, \widehat{x_l}, \dots, x_k]f}{x_l - x_j} \tag{6.1.12}$$

wobei $\widehat{\bullet}$ *bedeutet, dass dieser Knoten weggelassen wird.*

Beweis. Wir zeigen den Satz per Induktion.

Induktionsanfang mit $n = 0$: Es ist $p(f \mid x_0)(x) = f_0 = a_0 = [x_0]f$.

Induktionsschritt $n \to n+1$: Nach dem Lemma von Aitken gilt (6.1.5), also

$$p(f \mid x_0, \dots, x_{n+1})(x) = \frac{(x_0 - \boldsymbol{x})\boldsymbol{p}(f \mid x_1, \dots, x_{n+1})(x) - (x_{n+1} - \boldsymbol{x})\boldsymbol{p}(f \mid x_0, \dots, x_n)(x)}{(x_0 - x_{n+1})}$$

Nach Induktionsvoraussetzung gilt also für den Koeffizienten zu x^{n+1}, welcher a_{n+1} ist,

$$\begin{aligned} a_{n+1} &= \frac{-[x_1, \dots, x_{n+1}]f + [x_0, \dots, x_n]f}{x_0 - x_{n+1}} \\ &= [x_0, \dots, x_{n+1}]f \end{aligned}$$

Die Permutationsinvarianz der dividierten Differenzen folgt direkt aus $[x_0, \dots, x_n]f = a_n$, da die Interpolationspolynome $p(f \mid x_i, \dots, x_k)$ permutationsinvariant sind. □

Satz 6.1.7 (Newtonsche Darstellung)**.** *Es gilt*

$$p(f \mid x_0, \dots, x_n)(x) = \sum_{i=0}^{n} [x_0, \dots, x_i]f \cdot w_i(x) \tag{6.1.13}$$

und $\{w_k \mid k \in \{0, \dots, n\}\}$ *bildet Basis von* $\mathcal{P}_n$.

Beweis. Wir zeigen (6.1.10) per Induktion:

$n = 0$ ist klar.

$n-1 \to n$: Sei $p_n(x) := p(f \mid x_0, \ldots, x_n)(x) = [x_0, \ldots, x_n]f \cdot w_n(x) + q_{n-1}(x)$ mit $q_{n-1} \in \mathcal{P}_{n-1}$ und $q_{n-1}(x_i) = p_{n-1}(x_i) = f_i$ für $i = 0, \ldots, n-1$, da $w_n(x_i) = 0$. Damit folgt

$$q_{n-1} = p(f \mid x_0, \ldots, x_{n-1}) = \sum_{i=0}^{n-1} [x_0, \ldots, x_i]f \cdot w_i(x)$$

nach Induktionsvoraussetzung. Es folgt Gleichung (6.1.13). $\{w_0, \ldots, w_n\}$ sind linear unabhängig und jedes Polynom $\varphi \in \mathcal{P}_n$ lässt sich wegen $q(x) = p(q \mid x_0, \ldots, x_n)(x)$ durch (6.1.13) darstellen. □

Entsprechend zum Schema von Neville für $p(\overline{x})$ ergibt sich zur Berechnung der Koeffizienten $a_i = [x_0, \ldots, x_i]f$ in (6.1.10) folgendes Schema:

6.1.8 Das Schema der dividierten Differenzen

$$\begin{array}{llllllll}
x_0 & \boldsymbol{f_0 = [x_0]f} & \searrow & & & & & \\
x_1 & f_1 = [x_1]f & \to & \boldsymbol{[x_0, x_1]f} & \searrow & & & \\
\vdots & \vdots & & \vdots & \to & \boldsymbol{[x_0, x_1, x_2]f} & & \\
\vdots & \vdots & & \vdots & & \vdots & \ddots & \\
x_n & f_n = [x_n]f & \to & [x_{n-1}, x_n]f & \to & \cdots & \to & \boldsymbol{[x_0, \ldots, x_n]f}
\end{array}$$

(markiert sind die benötigten Werte) Der Aufwand zur Berechnung aller Koeffizienten a_i beträgt $\frac{n(n+1)}{2}$ Divisionen. Kombiniert mit dem Horner-Schema ergibt sich zur Auswertung von p an m Stellen ein Aufwand von

$$\frac{n(n+1)}{2} \text{ Divisionen } + m \cdot n \text{ Multiplikationen}$$

Das Schema von Neville benötigt dagegen

$$m \cdot \frac{n(n+1)}{2} \text{ Multiplikationen und Divisionen}$$

Bemerkung 6.1.9. Da $p(f \mid x_0, \ldots, x_n)$ linear von den Stützwerten f_i abhängt, d.h. $p(\alpha f + \beta g \mid x_0, \ldots, x_n) = \alpha p(f \mid x_0, \ldots, x_n) + \beta p(g \mid x_0, \ldots, x_n)$ gilt, folgt somit auch

$$[x_i, \ldots, x_k](\alpha f + \beta g) = \alpha [x_i, \ldots, x_k]f + \beta [x_i, \ldots, x_k]g$$

Satz 6.1.10 (Leibnizregel für dividierte Differenzen)**.** *Seien $x_0, \ldots, x_n$ paarweise verschieden. Dann gilt*

$$[x_0, \ldots, x_n](g \cdot h) = \sum_{i=0}^{n} [x_0, \ldots, x_i]g \cdot [x_i, \ldots, x_n]h \tag{6.1.14}$$

Beweis. Es seien

$$G(x) \coloneqq \sum_{i=0}^{n} [x_0, \dots, x_i] g \cdot w_i(x) = p\,(g \mid x_0, \dots, x_n)\,(x)$$
$$H(x) \coloneqq \sum_{j=0}^{n} [x_j, \dots, x_n] h \cdot \widetilde{w}_j(x) = p\,(h \mid x_0, \dots, x_n)\,(x)$$

mit $\widetilde{w}_j(x) = \prod_{l=j+1}^{n}(x - x_l) \in \mathcal{P}_{n-j}$.
Dann ist $(G \cdot H)(x_i) = g(x_i)h(x_i)\ \forall i \le n$ und

$$G \cdot H = \sum_{i,j=0}^{n} [x_0, \dots, x_i] g \cdot [x_j, \dots, x_n] h \cdot w_i \widetilde{w}_j \in \mathcal{P}_{2n}\,.$$

Wegen der Relation

$$w_i(x_k)\widetilde{w}_j(x_k) = \prod_{l=0}^{i-1}(x_k - x_l) \prod_{l=j+1}^{n}(x_k - x_l) = 0 \quad \forall k, i \ge j+1$$

interpoliert auch

$$P \coloneqq \sum_{0 \le i \le j \le n} [x_0, \dots, x_i] g \cdot [x_j, \dots, x_n] h \underbrace{w_i \widetilde{w}_j}_{\in \mathcal{P}_{n+i-j}} \in \mathcal{P}_n$$

die Werte $g(x_k), h(x_k)$ für $k = 0, \dots, n$ und es gilt $P \in \mathcal{P}_n$. Aufgrund der Eindeutigkeit gilt

$$p\,(g \cdot h \mid x_0, \dots, x_n) = P = \sum_{i=0}^{n} [x_0, \dots, x_i] g \cdot [x_i, \dots, x_n] h \cdot x^n + \sum_{l=0}^{n-1} c_l x^l$$

und Gleichung (6.1.14) folgt. □

6.1c) Hermite-Interpolation

08.12.2014

Gegeben seien die Knoten $x_0 < x_1 < \dots < x_m$ und die Werte $f_i^{(k)}$ für $k = 0, \dots, n_i - 1$ mit $i = 1, \dots m$. Dann sind $n + 1 \coloneqq \sum_{i=0}^{m} n_i$ Werte gegeben.

Definition 6.1.11. Ein Polynom p mit $p \in \mathcal{P}_n$ und der Interpolationseigenschaft

$$p^{(k)}(x_i) = f_i^{(k)} \tag{6.1.15}$$

für $i = 0, \dots m$ und $k = 0, \dots, n_i - 1$ heißt **Hermite-Interpolationspolynom** p.

Satz 6.1.12. *Das Hermite Interpolationspolynom existiert und ist eindeutig.*

Beweis.

Eindeutigkeit: Seien $p_1, p_2 \in \mathcal{P}_n$ mit (6.1.15), dann gilt $q = p_1 - p_2 \in \mathcal{P}_n$ und q hat mit Vielfachheit gezählt mindestens $(n+1)$ Nullstellen. Damit folgt bereits $q \equiv 0$.

Existenz: Die Existenz kann wie in Satz 6.1.1 mit verallgemeinerten Lagrangepolynomen gezeigt werden. Alternativ: Für $p(x) = c_0 + c_1 x + \cdots + c_n x^n$ ergibt sich mit (6.1.15) ein lineares GLS mit $(n+1)$ Unbekannten und $(n+1)$ Gleichungen. Da dieses aufgrund der Eindeutigkeit injektiv ist, ist das GLS nicht singulär und damit existiert für beliebige $f_i^{(k)}$ eine Lösung.

□

Führe nun virtuelle Knoten ein:

$$\overbrace{\underbrace{x_0}_{=:t_0} = \cdots = x_0}^{n_0} < \overbrace{\underbrace{x_1}_{=:t_{n_0}} = \cdots = x_1}^{n_1} < \overbrace{\underbrace{x_m}_{=:t_{n-n_m+1}} = \cdots = \underbrace{x_m}_{=:t_n}}^{n_m} \tag{6.1.16}$$

und ersetze die Bedingung (6.1.15) durch

$$p^{(s_j)}(t_j) = f^{(s_j)}(t_j) \quad \text{für } j = 0, \ldots, n \tag{6.1.17}$$

dann gilt

$$s_j = \max\{r \in \mathbb{N} \mid t_j = t_{j-r}\} \tag{6.1.18}$$

Beispiel 6.1.13.

$$\begin{array}{llllll} x_0 = 0 & f_0^{(0)} = -1 & f_0^{(1)} = -2 & & \Rightarrow & n_0 = 2 \\ x_1 = 1 & f_1^{(0)} = 0 & f_1^{(1)} = 10 & f_1^{(2)} = 40 & \Rightarrow & n_1 = 3 \end{array} \Longrightarrow n = 2 + 3 - 1 = 4$$

$$t_0 = t_1 = x_0 = 0, \quad t_2 = t_3 = t_4 = x_1 = 1$$
$$s_0 = 0, \quad s_1 = 1, \quad s_2 = 0, \quad s_3 = 1, \quad s_4 = 2$$

Bezeichne wiederum mit $p(f \mid t_0, \ldots, t_n)$ das Interpolationspolynom in $\mathcal{P}_n$, welches (6.1.17) erfüllt.

Lemma 6.1.14 (Lemma von Aitken)**.** *Es gelten die Formeln*

a) falls $t_i = \cdots = t_{i+k} = x_l$:

$$p(f \mid t_i, \ldots, t_{i+k})(x) = \sum_{r=0}^{k} \frac{f^{(r)}(x_l)}{r!}(x - x_l)^r \tag{6.1.19}$$

b) falls $t_i < t_{i+k}$:

$$p(f \mid t_i, \ldots, t_{i+k})(x) = \frac{(x - t_i) \cdot p(f \mid t_{i+1}, \ldots, t_{i+k})(x) - (x - t_{i+k}) \cdot p(f \mid t_k, \ldots, t_{i+k-1})(x)}{t_{i+k} - t_i} \tag{6.1.20}$$

Beweis. In beiden Fällen wird nachgewiesen, dass die rechte Seite das zugehörige Hermite-Interpolationsproblem löst. Aufgrund der Eindeutigkeit von $p \in \mathcal{P}_k$ folgt dann die Behauptung. □

Definition 6.1.15. Die verallgemeinerte **dividierte Differenz** ist definiert durch

$$[t_i, \dots, t_{i+k}]f := \frac{1}{k!} f^{(k)}(x_l) \quad \text{für } t_i = \cdots = t_{i+k} = x_l$$

und wie in (6.1.11) d.h.

$$[t_i, \dots, t_{i+k}]f = \frac{[t_{i+1}, \dots, t_{i+k}]f - [t_i, \dots, t_{i+k-1}]f}{t_{i+k} - t_i} \quad \text{für } t_i < t_{i+k}$$

Wie in 6.1.6 und 6.1.7 folgt aus dem Lemma von Aitken die **Newtonsche-Darstellung** (6.1.13)

$$p(f \mid t_0, \dots, t_n)(x) = \sum_{i=0}^{n} [t_0, \dots, t_i]f \cdot w_i(x)$$

mit $w_i(x) = \prod_{j=0}^{i-1}(x - t_j)$. Ebenso folgt die Leibnizformel für $t_0 \le t_1 \le \cdots \le t_n$ sowie $g, h \in C^n(I)$ aufgrund der Stetigkeit der dividierten Differenzen bzgl. t_i. Letzteres wäre noch zu zeigen.

Beispiel 6.1.16 (Fortsetzung von 6.1.131)**.**

	x_i	$[t_i]f$	$[t_i, t_{i+1}]f$	$\dots$		
t_0	0	**−1**				
t_1	0	−1	**−2**			
t_2	1	0	1	**3**		
t_3	1	0	10	9	**6**	
t_4	1	0	10	10	11	**5**

Damit folgt für p

$$\begin{aligned} p(f \mid t_0, \dots, t_4)(x) = &-1 \\ &- 2(x-0) \\ &+ 3(x-0)(x-0) \\ &+ 6(x-0)(x-0)(x-1) \\ &+ 5(x-0)(x-0)(x-1)(x-1) \\ = &-1 - 2x + 3x^2 + 6x^2(x-1) + 5x^2(x-1)^2 \end{aligned}$$

6.1d) Fehlerabschätzung bei Polynominterpolation

Der Fehler $f(x) - p(x)$ kann nicht abgeschätzt werden, falls nicht mehr als die Funktionswerte f_i von f bekannt sind.

Satz 6.1.17 (Fehlerdarstellung mit dividierten Differenzen)**.** *Es gilt die Fehlerdarstellung für alle f*

$$f(\overline{x}) - p(f \mid t_0, \ldots, t_n)(\overline{x}) = [t_0, \ldots, t_n, \overline{x}]f \cdot w_{n+1}(\overline{x}) \tag{6.1.21}$$

Beweis.

$$f(\overline{x}) - p(f \mid t_0, \ldots, t_n)(\overline{x}) = p(f \mid t_0, \ldots, t_n)(\overline{x}) + [t_0, \ldots, t_n, \overline{x}]f \cdot w_{n+1}(\overline{x})$$

aufgrund der Newtonschen Darstellung (6.1.13). □

Satz 6.1.18 (Restglieddarstellung)**.** *Sei f $(n+1)$-mal stetig differenzierbar, so gibt es zu jedem $\overline{x}$ eine Zahl $\xi(\overline{x})$ aus dem kleinsten Intervall I, welches $x_0, \ldots, x_m$ enthält, so dass*

$$f(\overline{x}) - p(f \mid t_0, \ldots, t_n)(\overline{x}) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \cdot w_{n+1}(\overline{x}) \tag{6.1.22}$$

Beweis. Sei $\overline{x} \neq x_i$ (sonst trivial)

Abb. 6.1.0 *Veranschaulichung zur Restglieddarstellung*

10.12.2014

Definiere

$$F(x) \coloneqq f(x) - p(f \mid t_0, \ldots, t_n)(x) - K w_{n+1}(x) \tag{6.1.23}$$

mit $K = \text{const}$ so gewählt, dass $F(\overline{x}) = 0$. Aufgrund von $p^{(k)}(x_i) = f^{(k)}(x_i)$ und $w_{n+1}^{(k)}(x_i) = 0$ wegen $w_{n+1}(x) = (x - t_0) \cdots (x - t_n) = \prod_{l=0}^{m} (x - x_l)^{n_l}$ gilt

$$F^{(k)}(x_i) = 0 \quad \text{für } i = 0, \ldots, m$$

sowie

$$F(\overline{x}) = 0$$

Da $F(x_i) = 0$ für $i = 0, \ldots, m$ und $F(\overline{x}) = 0$, folgt

$$\exists\, \rho_0^{(1)}, \ldots, \rho_m^{(1)} \in I, : F'(\rho_i^{(1)}) = 0$$

nach dem Satz von Rolle mit paarweise verschiedenen $\rho_i^{(1)}$. F' hat also $(m+1) + \#\{i \mid n_i > 1\}$ paarweise verschiedene Nullstellen. Nach dem Satz von Rolle hat F'' also $n + \#\{i \mid n_i > 1\}$ paarweise verschiedene Nullstellen in I, die ungleich zu den x_i sind für die $n_i > 2$. Weiterhin gilt $F''(x_i) = 0$, falls $n_i > 2$. Sukzessive folgt damit für $F^{(n+1)}$ die Existenz von

$$m + 1 - n + \underbrace{\#\{i \mid n_i > 1\} + \#\{i \mid n_i > 2\} + \cdots + \#\{i \mid n_i > n\}}_{=n-m}$$

paarweise verschiedene Nullstellen in I. Also existiert mindestens eine Nullstelle ξ von $F^{(n+1)} = f^{(n+1)} - K(n+1)!$. Damit ist $K = \frac{f^{(n+1)}(\xi)}{(n+1)!}$. Mit $F(\overline{x}) = 0$ folgt die Behauptung. □

Folgerung 6.1.19. *Für $f \in C^n(I)$ mit $I = [\min_{i\le m} x_i, \max_{i\le m} x_i]$ gibt es ein $\xi \in I$ mit*

$$[t_0, \ldots, t_n]f = \frac{f^{(n)}(\xi)}{n!} \tag{6.1.24}$$

Beweis. Für $f \in C^{n+1}(I)$ folgt aus Satz 6.1.15 und (6.1.17) $[t_0, \ldots, t_n, \overline{x}]f = \frac{f^{(n+1)}(\xi)}{(n+1)!}$, womit (6.1.24) folgt. □

Wie in der obigen Fehlerdarstellung zu sehen ist, spielt nicht nur $f^{(n+1)}$ bzw. $[t_0, \ldots, t_n, \overline{x}]f$ eine Rolle für den Fehler, sondern mit $w_{n+1}(\overline{x})$ auch die Verteilung der Knoten. Falls die Knoten frei wählbar sind und $\|f - p(f \mid t_0, \ldots, t_n)\|_{C([a,b])}$ klein sein soll, wird gewünscht, dass

$$\|w_{n+1}\|_{C([a,b])} \coloneqq \|w_{n+1}\|_\infty = \max_{x\in[a,b]} |w_{n+1}(x)|$$

minimal ist, d.h. man sucht nach

$$\min_{t_0,\ldots,t_n\in\mathbb{R}} \max_{x\in[a,b]} |(x - t_0)\cdots(x - t_n)|$$

Falls $t_i = x_i$, kann gezeigt werden [z.B. DH08; FH07], dass für $[a, b] = [-1, 1]$ das zugehörige w_{n+1} das sogenannte **Tschebyscheff-Polynom**

$$T_{n+1}(x) \coloneqq \cos\big((n+1)\arccos(x)\big) \in \mathcal{P}_{n+1} \tag{6.1.25}$$

ist. Dessen Nullstellen, die **Tschebyscheff-Punkte**, sind

$$x_i = \cos\left(\frac{2i+1}{2n+2}\pi\right) \quad \text{für } i = 0, \ldots, n \tag{6.1.26}$$

Weiterhin ist der führende Koeffizient von T_{n+1} genau 2^n, so dass

$$\|w_{n+1}\|_{C([-1,1])} = 2^{-n} \tag{6.1.27}$$

für die Tschebyscheff-Punkte gilt. Für andere Intervalle $[a, b]$ wird die lineare Transformation

$$x = \frac{1}{2}\big(t(b-a) + (b+a)\big)$$

von $[-1, 1]$ auf $[a, b]$ durchgeführt.

Wird die Fehlerabschätzung bzgl. der $\| \bullet \|_2$-Norm, d.h.

$$\|f\|_2 \coloneqq \left(\int_a^b |f(t)|^2 dt \right)^{\frac{1}{2}}$$

betrachtet, erweisen sich auf $[-1, 1]$ die Nullstellen des **Legendre-Polynoms** P_{n+1} als minimal [HH94], wobei

$$P_n(x) = \frac{n!}{(2n)!} \frac{\mathrm{d}^n}{\mathrm{d}\, x^n} \left((x^2 - 1)^n \right)$$

Achtung: Die Folge der Interpolationspolynome, die sich bei gleichmäßiger Knotenverteilung ergeben, konvergieren **nicht** gegen f mit wachsender Anzahl der Knoten! Ein Gegenbeispiel ist $f(x) = |x|$ in $[-1, 1]$: Die Folge $\{p(f \mid x_0, \ldots, x_n)(x)\}_{n \in \mathbb{N}}$ mit x_i gleichmäßig verteilt, divergiert für alle Werte $0 < |x| < 1$.

Abb. 6.1.1 *Gegenbeispiel zur Konvergenz: äquidistante Gitter für Betragsfunktion*

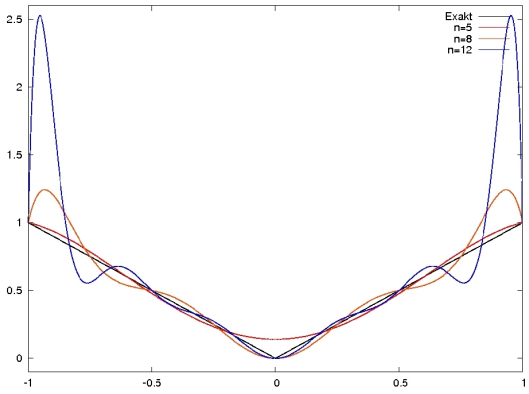
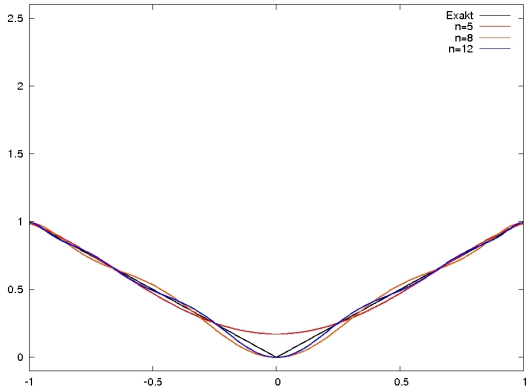


Abb. 6.1.2 *Gegenbeispiel zur Konvergenz: Tschebyscheff-Punkte für Betragsfunktion*



Ein weiteres Gegenbeispiel ist $f(x) = \frac{1}{1+x^2}$ in $[-5, 5]$ (siehe Übungsaufgabe).

Es gilt sogar:

Satz 6.1.20 (Faber)**.** *Zu jeder Folge von Intervalleinteilungen von $[a, b]$ gibt es eine stetige Funktion f , so dass die Interpolationspolynome auf $[a, b]$ nicht gleichmäßig gegen f konvergiert (bzgl. $\| \bullet \|_\infty$).*

Basierend auf dem Satz von Jackson gilt:

Satz 6.1.21 (Konvergenz bei Tschebyscheff-Knoten)**.** *Sei $f \in C([-1, 1])$ Lipschitz-stetig. Dann konvergieren die Interpolationspolynome zu den Tschebyscheff-Knoten gleichmäßig gegen f .*

Andererseits gilt [siehe HH94]:

Satz 6.1.22 (Marcinkiewicz)**.** *Zu jeder Funktion $f \in C([a, b])$ kann eine Folge von Intervalleinteilungen angegeben werden, so dass die Folge der Interpolationspolynome gleichmäßig gegen f konvergiert.*

Im Gegensatz zur $\| \bullet \|_\infty$-Norm (auch Tschebyscheff-Norm genannt) gilt für die $\| \bullet \|_2$-Norm [siehe HH94]:

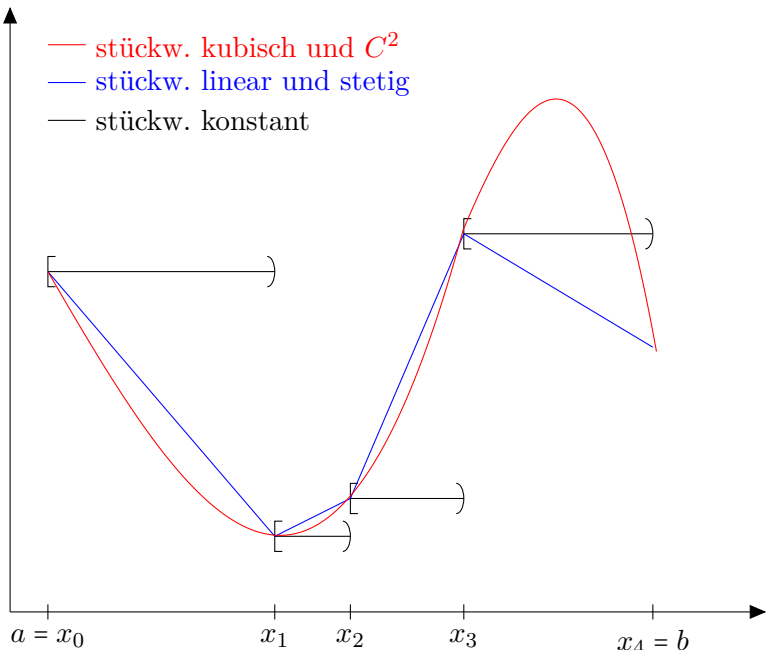
Satz 6.1.23. *Sei $f \in C([-1, 1])$ und seien $x_0, \ldots, x_n$ die Nullstellen des Legendre-Polynoms L_{n+1}. Dann gilt*

$$\lim_{n \to \infty} \| f - p(f \mid x_0, \ldots, x_n) \|_2 = 0$$

6.2 Stückweise polynomiale Approximation durch Splines

Eine hohe Anzahl von Knoten garantiert keine „gute“ Approximation durch Interpolationspolynome. In der Regel treten dann hohe Schwingungen auf, die unerwünscht sind. Die Idee der stückweisen polynomialen Approximation ist, jeweils einige wenige Stützwerte polynomial zu interpolieren und diese „glatt“ zusammenzusetzen.

Abb. 6.2.0 © *stückweise polynomiale Approximation*



Stückweise linear und stetig zu interpolieren ist am einfachsten, wobei aber stückweise kubisch und global zweimal stetig differenzierbar zur graphischen Aufarbeitung am geeignetsten ist, da das Auge noch Unstetigkeiten in der Krümmung erkennt.

6.2a) Splines und zwei verschiedene Basen

Definition 6.2.1. Sei $\Delta = \{x_0, \dots, x_n\}$ ein Gitter mit paarweise verschiedenen Knoten $a = x_0 < \cdots < x_n = b$. Ein (Polynom-)**Spline von Ordnung** $k \geq 2$ (d.h. der Grad der Teilstückpolynome ist maximal $(k-1)$) ist eine Funktion s mit $s \in C^{(k-2)}([a,b])$ und $s \in \mathcal{P}_{k-1}([x_i, x_{i+1}])$ für $i = 0, \dots, n-1$. Der Raum der Splines von Ornung k bzgl. Δ sei mit $S_{k,\Delta}$ bezeichnet. Für $k = 1$ ist der Raum definiert durch

$$S_{1,\Delta} \coloneqq \left\{ s : [a,b] \to \mathbb{R} \,\middle|\, s|_{[x_i, x_{i+1}]} \in \mathcal{P}_{k-1} \text{ für } 0 \leq i \leq n \right\}$$

Für $k = 2$ ergeben sich **lineare Splines** und für $k = 4$ die **kubischen Splines**. Offensichtlich gilt

a) $S_{k,\Delta}$ ist ein linearer Vektorraum.

b) $\mathcal{P}_{k-1} \subseteq S_{k,\Delta}$

Natürlich lässt sich ein Spline stückweise als Polynom darstellen, es hat also $n \cdot k$ Koeffizienten. Gewünscht ist aber eine Basisdarstellung, welche dann $n+k-1$ Koeffizienten hat.

Satz 6.2.2. *Die Monome x^l und die* ***abgebrochenen Potenzen***

$$(x - x_i)_+^l \coloneqq \begin{cases} (x - x_i)^l & x \geq x_i \\ 0 & x < x_i \end{cases} \tag{6.2.1}$$

bilden eine Basis $\mathcal{B} = \left\{1, x, \ldots, x^{k-1}, (x - x_1)_+^{k-1}, \ldots, (x - x_{n-1})_+^{k-1}\right\}$ des Splineraumes $S_{k,\Delta}$. Insbesondere gilt

$$\dim S_{k,\Delta} = k + n - 1 \tag{6.2.2}$$

Abb. 6.2.1 *Dimension eines Splineraumes*

Beweis. siehe Übungsaufgabe □

6.2.3 Nachteile der Splineraumbasis

a) Basisfunktionen haben keine lokalen Träger, d.h. zur Auswertung von $s(\overline{x})$ müssen alle Basisfunktionen ausgewertet werden, was relativ teuer ist.

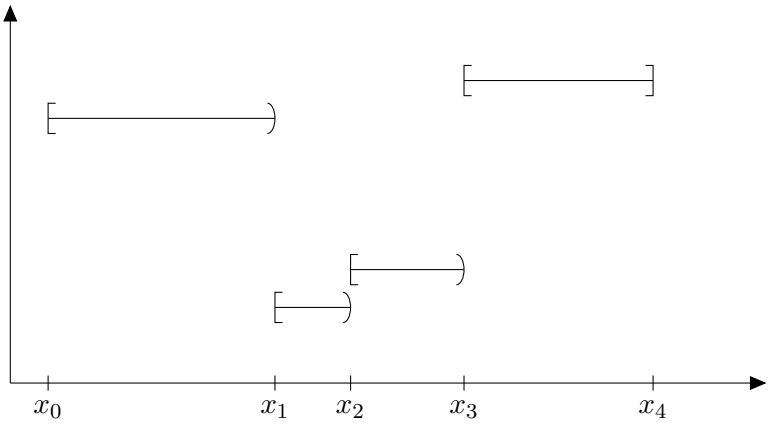
b) Falls $x_i \approx x_{i+1}$ sind $(x - x_i)_+^{k-1}$ und $(x - x_{i+1})_+^{k-1}$ „nahezu“ linear abhängig. Damit ist die Basis schlecht konditioniert bzgl. Störungen in b_i.

Das Ziel ist nun eine bessere Basis zu konstruieren.

Beispiel 6.2.4.

a) $S_{1,\Delta}$ sind stückweise konstante Funktionen

Abb. 6.2.2 © *Basis eines $S_{1,\Delta}$-Splines aus stückweise konstanten Elementen*



Die zugehörige Basis wie oben beschrieben wäre $\{1, (x - x_i)_+^0\}$. Eine in diesem Fall geeignetere Basis ist

$$N_{i,1}(x) := \chi_{[x_{i-1},x_i)}(x) = \begin{cases} 1 & \text{falls } x \in [x_{i-1}, x_i) \\ 0 & \text{sonst} \end{cases},$$

womit die Darstellung eines $S_{1,\Delta}$-Splines zu

$$s(x) = \sum_{j=1}^{n+1} f_{j-1} N_{j,1}(x)$$

wird.

b) $S_{2,\Delta}$ sind stückweise lineare, global stetige Funktionen. Eine Basis wie in 6.2.2 ist $\{1, x^1, (x - x_i)_+^1\}$

Abb. 6.2.3 *Basis eines $S_{2,\Delta}$-Splines mit Hutfunktion (siehe auch 6.2.6 für allg. Hutfunktionen)*

Hier kann eine Basis wie oben durch sog. **Hutfunktionen** konstruiert werden

$$N_{i,2}(x_{j-1}) = \delta_{i,j}$$

$$N_{i,2}(x) = \begin{cases} \frac{1}{x_{i-1}-x_{i-2}}(x - x_{i-2}) & x \in [x_{i-2}, x_{i-1}] \\ -\frac{1}{x_{i-1}-x_{i-2}}(x - x_{i-1}) + 1 & x \in [x_{i-1}, x_i] \end{cases}$$

Abb. 6.2.4 *allgemeine Hutfunktion am Punkt x_i eines $S_{2,\Delta}$-Splines*

Ein Spline, welches f in x_i interpoliert hat dann die Form

$$s(x) = \sum_{j=1}^{n+1} f_{j-1} N_{j,2}(x)$$

Definition 6.2.5. Sei $t_1 \leq \cdots \leq t_m$ eine beliebige Folge von Knoten x_i. Dann sind die **B-Splines** $N_{j,l}$ für $l = 1, \ldots, m$ und $j = 1, \ldots, m - l$ rekursiv erklärt durch

$$\begin{aligned} N_{j,1}(x) &:= \chi_{[t_j, t_{j+1})}(x) \text{ für } t_{j+1} \neq t_m \\ N_{p,1}(x) &= \chi_{[t_p, t_m]}(x) \text{ für } p = \min\{i \mid t_{i+1} = t_m\} \end{aligned} \tag{6.2.3}$$

sowie für $l > 1$

$$N_{j,l}(x) = \frac{x - t_j}{t_{j+l-1} - t_j} N_{j,l-1}(x) + \frac{t_{j+l} - x}{t_{j+l} - t_{j+1}} N_{j+1,l-1}(x) \tag{6.2.4}$$

Hierbei gilt die Konvention $\frac{0}{0} = 0$, denn für $t_j = t_{j+l-1}$ gilt $N_{j,1} \equiv 0$ und auch $N_{j,l-1} \equiv 0$.

Ziel ist nun u.a. mit diesen eine Basis von $S_{k,\Delta}$ anzugeben, die zudem gut konditioniert ist. Vorher jedoch werden noch einige Eigenschaften der B-Splines untersucht.

6.2.6 Gestalt der B-Splines

Bei einem B-Spline $N_{j,k}$ entspricht j dem Ort und $k - 1$ entspricht dem Polynomgrad.

Abb. 6.2.5

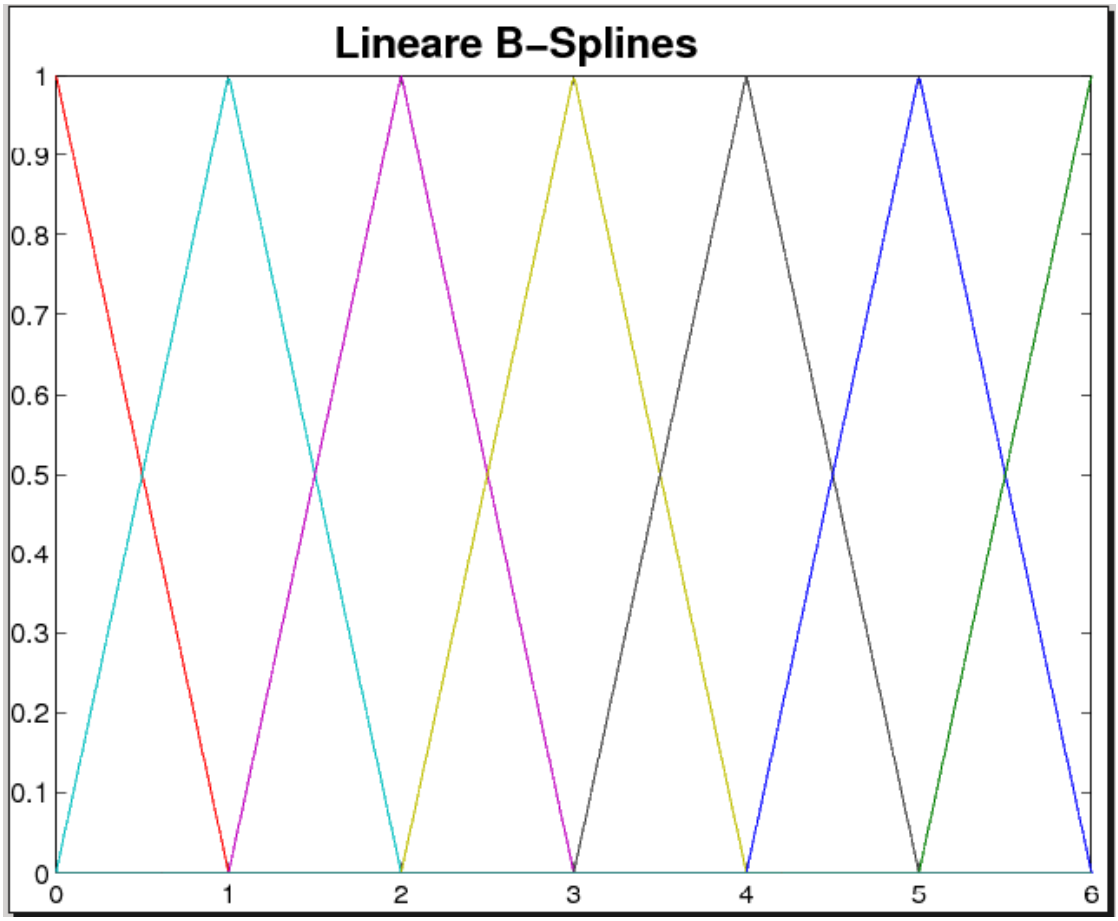


Abb. 6.2.6

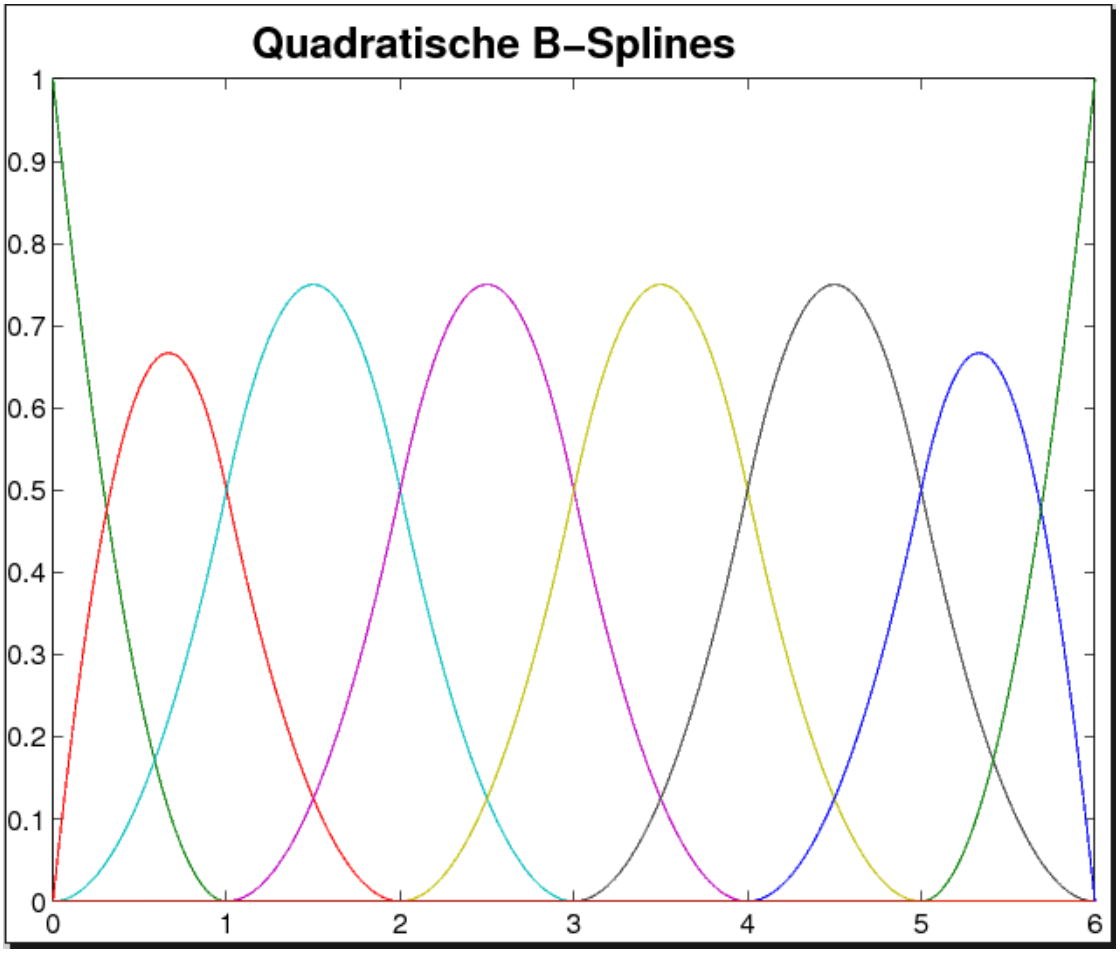
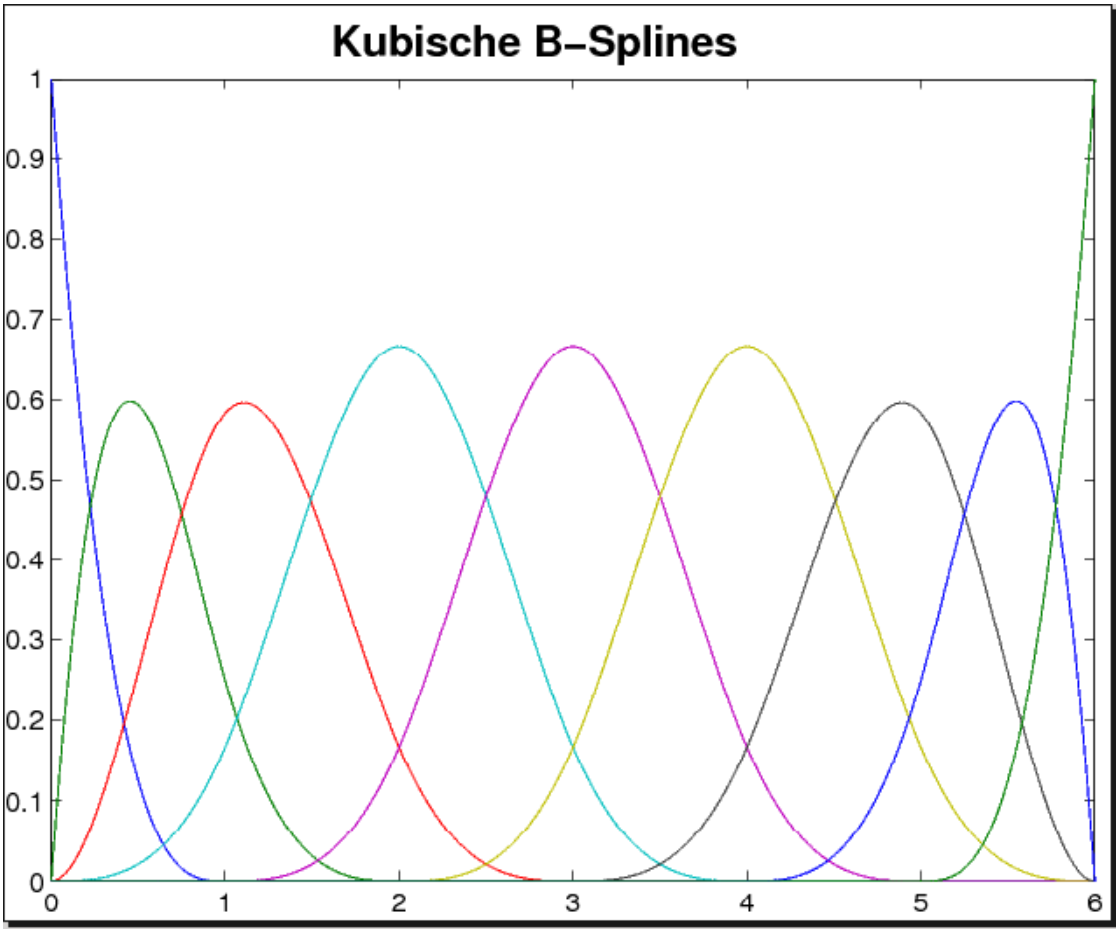


Abb. 6.2.7



Korollar 6.2.7. *Es gilt*

a) $\operatorname{supp} N_{j,k} \coloneqq \overline{\left\{x \in [a,b] \,\middle|\, N_{j,k}(x) \neq 0\right\}} \subset [t_j, t_{j+k}]$ *(lokaler Träger)*

b) $N_{j,k} > 0 \;\; \forall x \in (t_j, t_{j+1})$ *(nicht negativ)*

c) $N_{j,k}|_{[t_l, t_{l+1})} \in \mathcal{P}_{k-1}$

Beweis. siehe Übungsaufgabe □

6.2.8 Auswertung von s an der Stelle $\overline{x}$

Sei $\overline{x} \in [t_j, t_{j+1}]$ und $s(x) = \sum_{i=1}^{N} c_i N_{i,k}(x)$. Da nur $N_{j,1}(\overline{x}) \neq 0$ ist, können aufgrund von (6.2.4) auch nur $N_{j-k+1,k}(\overline{x}), \ldots, N_{j,k}(\overline{x})$ verschieden von Null sein, da

$j\backslash l$	t_j	1		2		3		k
1	t_1	$N_{1,1}(\overline{x}) = 0$		0		0		0
		0		$\vdots$		$\vdots$		$\vdots$
		$\vdots$						0
$j-k+1$	t_{j-k+1}	$\ldots$		$\ldots$		$\ldots$		$\boldsymbol{N_{j-k+1,k}(\overline{x})}$
$\vdots$	$\vdots$							$\vdots$
				0		$\boldsymbol{N_{j-2,3}(\overline{x})}$		$N_{j-2,k}(\overline{x})$
$j-1$	t_{j-1}	0	$\longrightarrow$	$\boldsymbol{N_{j-1,2}(\overline{x})}$	$\longrightarrow$	$N_{j-1,3}(\overline{x})$		$N_{j-1,k}(\overline{x})$
j	t_j	$\boldsymbol{N_{j,1}(\overline{x}) = 1}$	$\longrightarrow$	$\boldsymbol{N_{j,2}(\overline{x})}$	$\longrightarrow$	$\boldsymbol{N_{j,3}(\overline{x})}$	$\longrightarrow \ldots$	$\boldsymbol{N_{j,k}(\overline{x})}$
		0		0				$\vdots$
		$\vdots$		$\vdots$				
	t_{j+k}	0		0				0

Daraus folgt, dass $N_{i,k}(\overline{x}) \neq x$ für höchstens $i = j-k+1, \ldots, j$. Nach der Rekursionsformel baut $N_{i,k}$ auf $N_{i,1}, \ldots, N_{i+(k-1),1}$ auf und benutzt $t_i, \ldots, t_{i+k}$. Da in der Rekursionsformel nur nichtnegative Vielfache nichtnegativer Zahlen addiert werden, ist das Verfahren zur Bestimmung der $N_{i,k}(\overline{x})$ numerisch sehr stabil.

Lemma 6.2.9. *Falls $t_j < t_{j+k}$ und $x \neq t_m$, so gilt*

$$N_{j,k}(\overline{x}) = (t_{j+k} - t_j) \underset{[t_j,\ldots,t_{j+k}]f \text{ mit } f(t) := (t-x)_+^{k-1}}{[t_j, \ldots, t_{j+k}](\bullet - x)_+^{k-1}} \tag{6.2.5}$$

Beweis. kein Beweis.

Idee: Per Induktion über k. Nutze hierfür $(t-x)_+^k = (t-x)(t-x)_+^{k-1}$, $k \geq 1$, sowie $[t_j, \ldots, t_i](\bullet - x) = 0$ für $i > j+1$ und die Leibnizregel. □

Zu gegebenem Splineraum $S_{j,\Delta}$ mit

$$\Delta : a = x_0 < x_1 < \cdots < x_n = b \tag{6.2.6}$$

setzt man nun

$$T : t_1 = \cdots = t_k < t_{k+1} < \cdots < t_{k+n} = t_{n+2k+1}\,, \tag{6.2.7}$$

dann lässt sich folgender Satz zeigen:

Satz 6.2.10. 17.12.2014

a) Es gilt $\mathcal{P}_{k-1} \subset \operatorname{span}\left\{N_{1,k}, \ldots, N_{n+k-1,k}\right\}$

b) Die B-Splines bilden eine Zerlegung der Eins auf $[a,b]$*, d.h. für alle* $x \in [a,b]$ *ist* $1 \equiv \sum_{j=1}^{n+k-1} N_{j,k}(x)$

c) Weiterhin sind die B-Splines (lokal) linear unabhängig, d.h. falls

$$\sum_{l=1}^{n+k-1} c_l N_{l,k}(x) = 0 \quad \forall x \in (c,d) \subset [a,b]\,,$$

folgt $c_j = 0$ *für alle* j *mit* $(c,d) \cap \underbrace{(t_j, t_j+k)}_{=\operatorname{supp} N_{j,k}} \neq \varnothing$.

Abb. 6.2.8 *Lineare Unabhängigkeit der B-Splines*

Beweis.

a), b) folgen aus der Marschen-Identität [siehe z.B. DH08]

c) O.B.d.A. enthalte (c,d) keine Knoten t_i (sonst zerlege (c,d) in Teilintervalle). Dann folgt $(c,d) \subseteq (t_l, t_{l+1})$. Nach a) lässt sich jedes Polynom $p \in \mathcal{P}_{k-1}$ durch die B-Splines $N_{l,k}$ darstellen. Auf (c,d) sind nur $k = \dim \mathcal{P}_{k-1}$ B-Splines von Null verschieden. Daher müssen diese linear unabhängig sein.

□

Folgerung 6.2.11. *Die B-Splines* $\{N_{1,k}, \ldots, N_{n+k-1,k}\}$ *bilden eine Basis von* $S_{k,\Delta}$.

Die Koeffizienten c_i *der Basisdarstellung von* $s \in S_{k,\Delta}$

$$s(x) = \sum_{i=1}^{n+k-1} c_i N_{ik}(x)$$

heißen ***de Boor-Punkte*** *von* s*. Die Funktionswerte* $s(x)$ *sind somit eine Konvexkombination der de Boor-Punkte* c_i.

Bemerkung 6.2.12. Die B-Spline Basis ist eine gut konditionierte Basis, d.h. die Auswertung von Splines mittels ihrer B-Spline Darstellung ist gut konditioniert. Es gilt mit $N \coloneqq n + k - 1$

$$D_k \max_{j=1,\ldots,N} |c_j| \leq \left\| \sum_{j=1}^{N} c_j N_{jk} \right\|_\infty \leq \max_{j=1,\ldots,N} |c_j|$$

Die Konstante D_k hängt nur von der Ordnung k ab, nicht von der Lage der Knoten.

6.2b) Splineinterpolation

6.2.13 Splineinterpolation allgemein

Gegeben seien $N = n + k - 1 = \dim S_{k,\Delta}$ Stützpunkte (ξ_i, f_i) mit $\xi_1 < \cdots < \xi_N$. Gesucht sind nun Splines $s \in S_{k,\Delta}$ mit $s(\xi_i) = f_i$ für alle $i = 1, \ldots, N$. Das ist äquivalent zu

$$\begin{aligned} \sum_{j=1}^{N} c_j N_{j,k}(\xi_i) &= f_i && \text{für } i = 1, \ldots, N \\ \Leftrightarrow Ac &= f && \text{mit } A = (N_{j,k}(\xi_i))_{i,j=1,\ldots,N} \end{aligned} \tag{6.2.8}$$

6.2.14 Lineare B-Splines

Sei $k = 2$, $N = n + 1$, $S_{2,\Delta}$ der Splineraum zu einem Δ, welches konstruiert werden soll.

Abb. 6.2.9 *Stützstellen für eine konstruierte Zerlegung* Δ

Sei $x_0 \coloneqq \xi_1 < x_1 \coloneqq \xi_2 < \cdots < x_n \coloneqq \xi_{n+1}$, dann folgt $N_{i,2}(\xi_j) = N_{i,2}(t_{j+1}) = \delta_{i,j}$. Also ist in diesem Fall die Matrix $A = (N_{j,k}(\xi_i))_{i,j \le n} = I$ also $c_i = f_i = f(\xi_i) = f(x_{i-1})$. Der zugehörige Spline s hat dann die Form

$$s(x) = \sum_{i=1}^{N} f_i N_{i,2}(x) = \sum_{j=0}^{N} f(x_j) N_{j+1,2}(x)$$

Zur Auswertung von s wird 6.2.8 verwendet. A sei definiert als

$$A \coloneqq \left(N_{j,k}(\xi_i)\right)_{i,j=1,\ldots,N}$$

und hat folgende Eigenschaften

- A besitzt Bandstruktur, da ξ_i höchstens in k Intervallen $[t_j, t_{j+k}]$ liegen kann.
- A ist regulär, also $N_{ik}(\xi_i) \neq 0$ für $i = 1, \ldots, N$ (nach dem Satz von Schoenberg und Whitney, 1953).
- A ist total positiv, d.h. für alle quadratischen Untermatrizen B gilt $\det(B) \geq 0$. (nach Karlin, 1968)

Daraus kann man folgern, dass die Gaußelimination für Bandmatrizen ohne Pivotsuche durchführbar ist, falls $N_{ik}(\xi_i) \neq 0$ für $i = 1, \ldots, N$.

6.2.15 Kubische B-Spline-Interpolation

Sei $k = 4$ und entsprechend $N = n + 3$ und seien die Stützpunkte zur Interpolation wie in 6.2.14 gegeben durch

$$\xi_i = x_{i-1} \quad \text{für } i = 1, \ldots, n+1\,.$$

(Daraus folgt $N \neq n + 3$ und 6.2.13 ist nicht anwendbar.) Zur eindeutigen Bestimmung von $s \in S_{4,\Delta}$ fehlen zwei Bedingungen. Typischerweise werden zusätzlich zu

$$s(x_i) = f(x_i) \quad \text{für } i = 0, \ldots, n$$

eine der folgenden Randbedingungen gefordert:

i) $s'(a) = f'(a)$ und $s'(b) = f'(b)$: **vollständige** oder **Hermitesche kubische Spline-Interpolation**

ii) $s''(a) = s''(b) = 0$ („natürliche“ Endbedingungen): **natürliche kubische Spline-Interpolation** (d.h. s kann linear außerhalb von $[a, b]$ glatt fortgesetzt werden)

iii) $s'(a) = s'(b)$ und $s''(a) = s''(b)$, falls f periodisch mit Periode $b - a$ ist: **periodische kubische Spline-Interpolation**.

Die drei Randbedingungen werden aufgrund physikalischer Eigenschaften gefordert. Hierzu ein Beispiel:

Es beschreibe $y(t)$ die Lage eines homogenen, isotropen Stabes (z.B. eine dünne Holzlatte). Dann misst

$$E(y) = c \int_a^b \left(\frac{y''(t)}{(1 + y'(t)^2)^{\frac{3}{2}}} \right)^2 dt = c \int_a^b (\kappa(t))^2 \, dt$$

(wobei κ die Krümmung der Kurve y in der Ebene ist) die Biegeenergie. Aufgrund des **Hamiltonschen Prinzips** stellt sich der Stab so ein, dass $E(y)$ minimiert wird. Unter der Annahme $|y'(t)| \ll 1$ für $t \in [a, b]$ wird E linearisiert zu $c \int_a^b y''(t)^2 dt$. Also wird annähernd eine Funktion $y \in C^2$ gesucht, welche $\|y''\|_2^2$ minimiert.

Obige Splines haben gerade diese Eigenschaft, denn:

Satz 6.2.16. *Sei s ein interpolierender, kubischer Spline von f in den Knoten x_i und $y \in C^2$ eine beliebige Funktion von f, so dass*

$$\left[s''(t) \cdot (y'(t) - s'(t)) \right]_{t=a}^{b} = 0$$

Dann gilt $\|s''\|_2 \leq \|y''\|_2$.

Beweis. Es gilt

$$\|y''\|_2^2 = \|s'' + (y'' - s'')\|_2^2 = \|s''\|_2^2 + 2\underbrace{\int_a^b s''(y'' - s'')dt}_{\substack{=0 \\ \text{wie gezeigt wird}}} + \|y'' - s''\|_2^2$$

da

$$\int_a^b s''(y'' - s'')dt = \sum_{i=1}^{N}\left(\left[s''(y' - s')\right]_{x_{i-1}}^{x_i} - \int_{x_{i-1}}^{x_i} s'''(y' - s')dt\right)$$

$(S_{4,\Delta} \subseteq C^2([a,b]),\ s'''(x) \equiv d_{i-1} = \text{const, da } s \in \mathcal{P}_3([x_{i-1}, x_i])\ )$

$$\begin{aligned}
&= \underbrace{\left[s''(y' - s')\right]_a^b}_{\substack{=0 \\ \text{nach Voraussetzung}}} - \sum_{i=1}^{N} d_{i-1}\int_{x_{i-1}}^{x_i}\big(y'(t) - s'(t)\big)\,dt \\
&= -\sum_{i=1}^{N} d_{i-1}\Big(\big(y(x_i) - s(x_i)\big) - \big(y(x_{i-1}) - s(x_{i-1})\big)\Big) \\
&= 0
\end{aligned}$$

da y,s Interpolationen von f sind. Mit den Randbedingungen i), ii) und iii) ist die Voraussetzung von 6.2.16 erfüllt. □

Korollar 6.2.17. *Es existiert genau ein interpolierender kubischer Spline $s \in S_{4,\Delta}$, welcher die Randbedingungen aus 6.2.16 erfüllt. Weiterhin gilt für jede interpolierende Funktion $y \in C^2([a,b])$, die derselben Randbedingung genügt, $\|s''\|_2 \le \|y''\|_2$.*

Beweis. Die Anzahl der Forderungen ist $N = \dim S_{4,\Delta}$. Es ergibt sich also ein quadratisches lineares Gleichungssystem und es genügt Injektivität zu zeigen. Sei $f \equiv 0$, dann erfüllt $y \equiv 0$ alle Forderungen. Für alle Lösungen $s \in S_{4,\Delta}$ gilt daher

$$\|s''\|_2 \le \|y''\|_2 = 0 \quad \text{nach 6.2.16}$$

Also gilt $s'' \equiv 0$ und s ist stückweise kubische Funktion mit $s \in C^2([a,b])$ und $s(x_i) = 0$. Damit folgt $s \equiv 0$, was die Injektivität zeigt. □

6.2.18 Berechnung der natürlichen Splines mittels B-Splines

22.12.2014

$$A \coloneqq \begin{pmatrix}
\frac{x_2+x_1-2x_0}{x_2-x_0} & -\frac{x_1-x_0}{x_2-x_0} & & & & & \\
N_{2,4}(x_1) & N_{3,4}(x_1) & N_{4,4}(x_1) & & & & 0 \\
& \ddots & \ddots & \ddots & & & \\
& & N_{i+1,4}(x_i) & N_{i+2,4}(x_i) & N_{i+3,4}(x_i) & & \\
& & & \ddots & \ddots & \ddots & \\
0 & & & N_{n,4}(x_{n-1}) & N_{n+1,4}(x_{n-1}) & N_{n+2,4}(x_{n-1}) \\
& & & & -\frac{x_n-x_{n-1}}{x_n-x_{n-2}} & \frac{2x_n-x_{n-2}-x_{n-1}}{x_n-x_{n-2}}
\end{pmatrix}$$

Die Werte $N_{j,4}(x_i)$ lassen sich nach 6.2.8 berechnen.

Insbesondere ergibt sich für äquidistantes Gitter

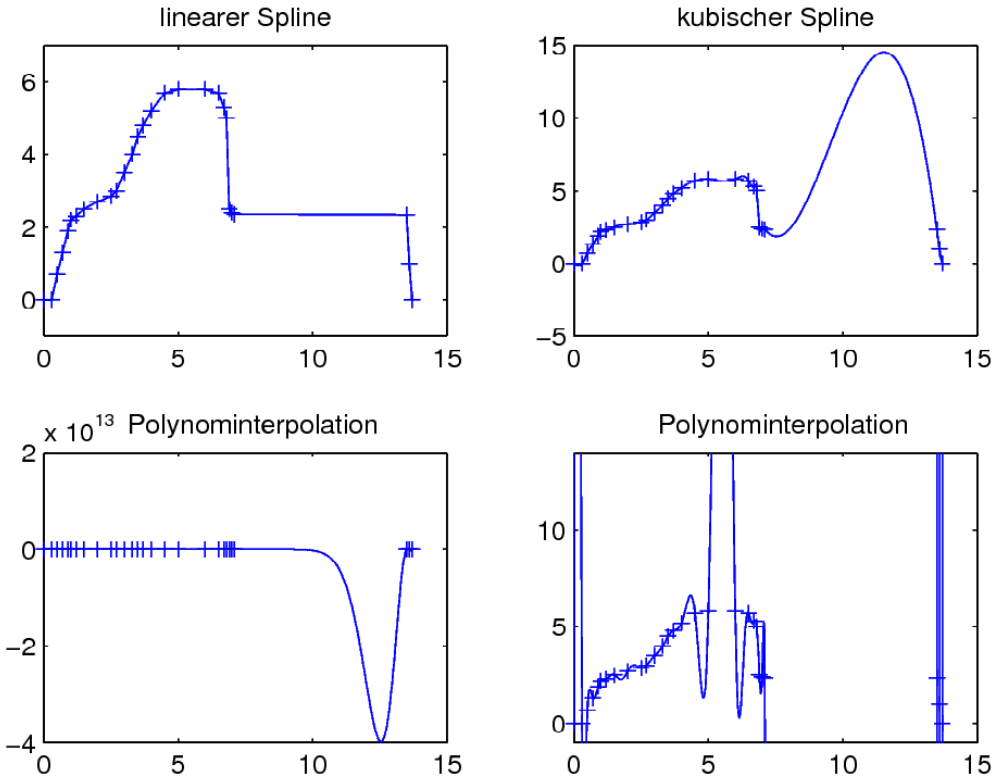
$$\frac{1}{12}\begin{pmatrix} 18 & -6 & & & & & \\ 3 & 7 & 2 & & & & \\ & 2 & 8 & 2 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 2 & 8 & 2 & \\ & & & & 2 & 7 & 3 \\ & & & & & -6 & 18 \end{pmatrix}\begin{pmatrix} c_2 \\ \vdots \\ c_{n+2} \end{pmatrix} = \begin{pmatrix} f_0 \\ \vdots \\ f_n \end{pmatrix}$$

(siehe Übung).

Beispiel 6.2.19 (Laster)**.** Das folgende Beispiel zeigt, dass sowohl die Glattheit der Funktion wie auch die Knotenverteilung einen Einfluß hat, da die zugehörige Funktion nicht differenzierbar ist, bzw. die Ableitungen betragsmäßig sehr groß sind.

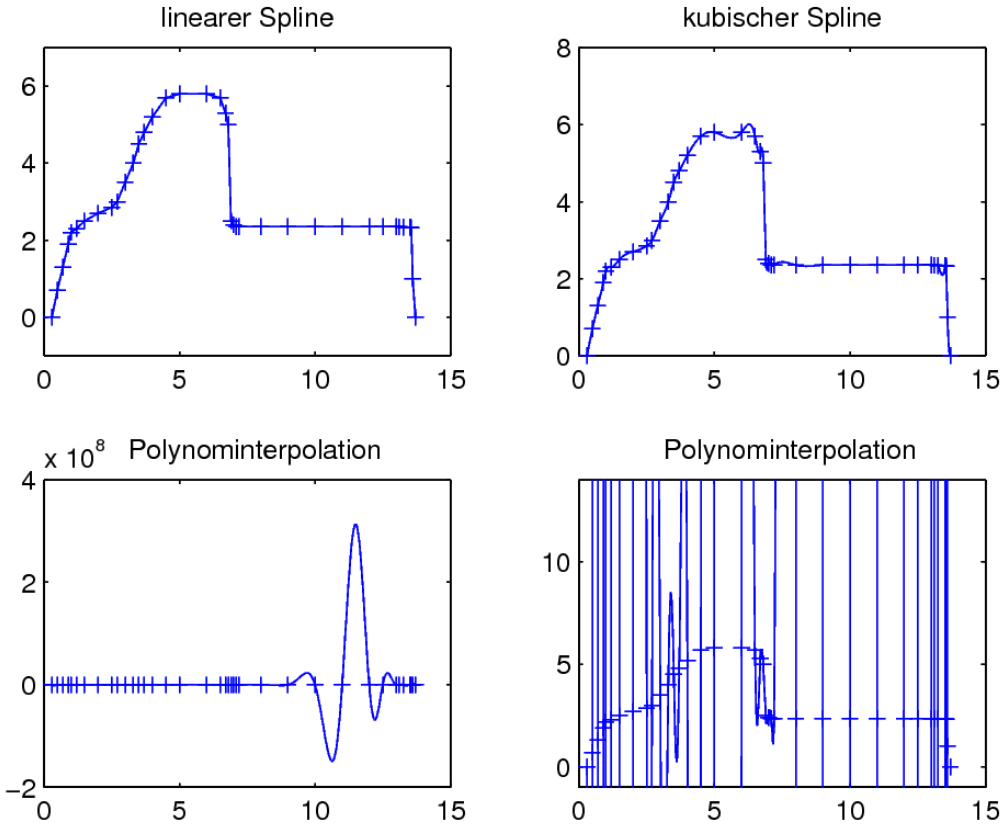
Natürliche Spline-Interpolation versus polynomiale Interpolation:

Abb. 6.2.10 *Splines versus Polynominterpolation*



Mit mehr Stützstellen:

Abb. 6.2.11 *Splines versus Polynominterpolation mit vielen Stützstellen*



Satz 6.2.20 (Fehlerabschätzung für kubische Splines)**.** *Sei $f \in C^4([a,b])$ und $s \in S_{4,\Delta}$ der interpolierende, vollständige, kubische Spline zu äquidistanten Knoten mit $h = x_{i+1} - x_i$. Dann gilt für $k = 0,1,2,3$*

$$\left\| f^{(k)} - s^{(k)} \right\|_\infty \leq c_k h^{4-k} \left\| f^{(4)} \right\|_\infty$$

mit $c_0 = \frac{5}{384}$, $c_1 = \frac{1}{24}$, $c_2 = \frac{3}{8}$, $c_3 = 1$, wobei c_0 und c_1 optimal sind.

Beweis. [siehe z.B. SB90, (in allgemeiner Form, d.h. auch für nicht gleichmäßiges Gitter)] Die optimalen c_k wurden von Hall und Meyer (1976) bewiesen. Oft wird mit Stetigkeitsmodulen gearbeitet. □

Für eine wachsende Anzahl der Knoten ergibt sich also Konvergenz, anders als bei polynomialer Interpolation.

Satz 6.2.21 (Fehlerabschätzung für lineare Splines)**.** *Sei* $f \in C^2([a,b])$ *und* $s \in S_{2,\Delta}$ *der interpolierende Spline an den Stützstellen* $a = x_0 < x_1 < \cdots < x_n = b$*. Dann gilt mit* $h = max_{i=0,\ldots,n-1}|x_{i+1} - x_i|$

$$\|f - s\|_\infty \leq \frac{h^2}{8} \left\|f''\right\|_\infty .$$

Beweis. siehe Übungsaufgabe □

Weitere Literatur zu Splines: [Boo01]

7 Numerische Integration/Quadratur

7.1 Einführung und einfache Quadraturformeln

Häufig sind Integrale

$$I(f) \coloneqq \int_a^b f(t)\,\mathrm{d}\,t$$

nicht analytisch zu berechnen oder die benötigte analytische Darstellung des unbestimmten Integrals ist kompliziert. Ziel ist $I(f)$ durch eine **Numerische Quadratur** $\hat{I}(f)$ zu approximieren.

Der Begriff numerische Integration ist allgemeiner und beschreibt das numerische Lösen von

$$y'(t) = f\big(t, y(t)\big) \qquad \text{mit } y(a) = c$$

Die numersiche Quadratur $\hat{I}(f)$ sollte

a) effizient ausführbar sein,

b) viele Eigenschaften der Integration beibehalten und

c) $I(f)$ gut approximieren.

Zur Wiederholung einige Eigenschaften des Riemann-Integrals:
Seien $f, g \in C([a,b])$, dann ist $I\colon C([a,b]) \to \mathbb{R}$ eine positive Linearform, d.h. I ist linear und

$$f \geq 0 \quad \Rightarrow \quad I(f) \geq 0 \tag{7.1.1}$$

Weiterhin gilt

$$\int_a^b f(t)\,\mathrm{d}\,t + \int_b^c f(t)\,\mathrm{d}\,t = \int_a^c f(t)\,\mathrm{d}\,t \tag{7.1.2}$$

Diese sollen nun möglichst beibehalten werden.

Definition 7.1.1. Ein Verfahren heißt konvergent von der Ordnung p (hat die Konvergenzordnung p), falls der Fehler (hier $|I(f) - \hat{I}(f)|$) mit $\mathcal{O}(h^p)$ gegen Null geht für $h \to 0$.

7.1.2 Mittelpunktregel

Abb. 7.1.0 *Veranschaulichung der Mittelpunktregel*

Zerlege $[a,b]$ in Teilintervalle $[x_i, x_{i+1}]$ für $i = 0,\dots,n-1$ der Länge $h_i = x_{i+1} - x_i$, d.h. $a = x_0 < x_1 < \cdots < x_n = b$, und setze zur Approximation von $I(f)$

$$\hat{I}_M(f) \coloneqq \sum_{i=0}^{n-1} f\left(\frac{x_{i+1}+x_i}{2}\right)\cdot(x_{i+1}-x_i) \tag{7.1.3}$$

Für ein äquidistantes Gitter ergibt sich mit $h = \frac{b-a}{n}$

$$\hat{I}_M(f) = h\sum_{i=0}^{n-1} f\left(x_i + \frac{h}{2}\right) \tag{7.1.4}$$

Die **Fehlerschranke** (Konvergenzordnung) für $f \in C^2([a,b])$

$$\left|I(f) - \hat{I}_M(f)\right| \le \frac{h^2}{24}(b-a)\left\|f''\right\|_\infty \tag{7.1.5}$$

ergibt sich aus der Taylorentwicklung

$$f(t) = f\left(x_i+\frac{h}{2}\right) + f'\left(x_i+\frac{h}{2}\right)\cdot\left(t-\left(x_i+\frac{h}{2}\right)\right) + \frac{1}{2}f''\left(\xi_i(t)\right)\cdot\left(t-\left(x_i+\frac{h}{2}\right)\right)^2$$

mit einer Zwischenstelle $\xi_i(t)$ zwischen t und $\left(x_i+\frac{h}{2}\right)$, denn

$$\begin{aligned}
\left|I(f) - \hat{I}_M(f)\right| &\le \sum_{i=0}^{n-1}\left|\int_{x_i}^{x_{i+1}} f(t)\,\mathrm{d}\,t - h\cdot f\left(x_i+\frac{h}{2}\right)\right| \\
&= \sum_{i=0}^{n-1}\left|\int_{x_i}^{x_{i+1}}\left(f(t) - f\left(x_i+\frac{h}{2}\right)\right)\mathrm{d}\,t\right| \\
&= \sum_{i=0}^{n-1}\left|\int_{x_i}^{x_{i+1}} f'\left(x_i+\frac{h}{2}\right)\cdot\left(t-\left(x_i+\frac{h}{2}\right)\right) + \int_{x_i}^{x_{i+1}} f''\left(\xi_i(t)\right)\cdot\left(t-\left(x_i+\frac{h}{2}\right)\right)^2\mathrm{d}\,t\right|
\end{aligned}$$

da $\int_{x_i}^{x_{i+1}} \left(t - \left(x_i + \frac{h}{2}\right)\right) \mathrm{d}\, t = 0$

$$
\begin{aligned}
&= \sum_{i=0}^{n-1} \left| \int_{x_i}^{x_{i+1}} \frac{1}{2} f''\big(\xi_i(t)\big) \cdot \left(t - \left(x_i + \frac{h}{2}\right)\right)^2 \mathrm{d}\, t \right| \\
&\leq \left\| f'' \right\|_\infty \cdot \frac{1}{2} \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} \left(t - \left(x_i + \frac{h}{2}\right)\right)^2 \mathrm{d}\, t \\
&= \left\| f'' \right\|_\infty \cdot \frac{1}{2} \sum_{i=0}^{n-1} \frac{1}{3} \left[x_{i+1} \left(\underbrace{x_{i+1} - x_i}_{h} - \frac{h}{2} \right)^2 - x_i \left(x_i - x_i - \frac{h}{2} \right)^2 \right] \\
&= \left\| f'' \right\|_\infty \frac{h^2}{24} \cdot \sum_{i=0}^{n-1} (x_{i+1} - x_i) \\
&= \frac{h^2}{24} (b-a) \left\| f'' \right\|_\infty
\end{aligned}
$$

Für andere Interpolationspunkte als den Mittelpunkt (z.B. einer der Eckpunkt) wird $\mathcal{O}(h^2)$, d.h. die Ordnung 2, nicht erreicht.

7.1.3 Trapezsumme

Für die Trapezsumme wird stückweise stückweise linear interpoliert.

Abb. 7.1.1 *Veranschaulichung Trapezsumme*

Mit der **Trapezregel** $(x_{i+1} - x_i) \cdot \frac{f(x_i)+f(x_{i+1})}{2}$ ergibt sich

$$\hat{I}_T(f) = \frac{1}{2} \sum_{i=0}^{n-1} \big(f(x_i) + f(x_{i+1})\big) \cdot (x_{i+1} - x_i) \tag{7.1.6}$$

und für äquidistante Gitter wird dies zur **Trapezsumme**

$$\hat{I}_T(f) = h \cdot \left(\frac{1}{2} f(x_0) + \sum_{i=0}^{n-1} f(x_i) + \frac{1}{2} f(x_n) \right) \tag{7.1.7}$$

Die Konvergenz folgt durch die Riemannschen Unter- bzw. Obersummen $R_u^{(k)}(f)$ und $R_o^{(k)}(f)$ wegen

$$I(f) \longleftarrow R_u^{(k)}(f) \leq \hat{I}_T(f) \leq R_o^{(k)}(f) \longrightarrow I(f) \qquad \text{für } h \to 0$$

Später zeigen wir noch die Konvergenzordnung für $f \in C^2([a,b])$

$$\left|I(f) - \hat{I}_T(f)\right| \leq \frac{h^2}{12}(b-a)\left\|f''\right\|_\infty \tag{7.1.8}$$

Offensichtlich erfüllen $\hat{I}_M(f)$ und $\hat{I}_T(f)$ die Eigenschaften aus 7.1.1. Weiterhin sind sie für lineare Funktionen exakt, d.h. berechnen die genauen Integralwerte.

7.2 Interpolatorische Integrationsformeln

(auch Newton-Cotes-Formeln)
Die Idee ist nun, f auf $[a,b]$ polynomial zu interpolieren an verschiedenen Stellen $a \leq x_0 \leq x_1 \leq \cdots \leq x_n < b$ und dann das Integral zu berechnen, d.h.

$$\begin{aligned}\hat{I}(f) &:= I\big(p\,(f \mid x_0, \ldots, x_n)\,\big) \\ &= \sum_{i=0}^{n} f(x_i) \cdot \int_a^b L_i(t)\,\mathrm{d}\,t \\ &= \sum_{i=0}^{n} w_i f(x_i)\end{aligned} \tag{7.2.1}$$

mit den Gewichten

$$w_i := \int_a^b L_i(t)\,\mathrm{d}\,t\,. \tag{7.2.2}$$

Offensichtlich sind diese Quadraturformeln exakt für Polynome vom Grad weniger oder gleich n.

Es gilt sogar

Lemma 7.2.1. *Zu $n+1$ paarweise verschiedenen Knoten $x_0, \ldots, x_n$ gibt es genau eine Quadraturformel*

$$\hat{I}(f) = \sum_{i=0}^{n} w_i f(x_i)$$

mit $w_i \in \mathbb{R}$, welche für alle Polynome $p \in \mathcal{P}_n$ exakt ist.

Beweis. $L_i \in \mathcal{P}_n$, also

$$I(L_i) = \hat{I}(L_i) = \sum_{j=0}^{n} w_j L_i(x_j) = \sum_{j=0}^{n} w_j \delta_{ij} = w_i \quad \text{für } i = 0, \ldots, n$$

Die Gewichte sind damit eindeutig bestimmt. □

Definition 7.2.2. $\hat{I}$ heißt **abgeschlossene Quadraturformel** , falls $a = x_0$ und $b = x_n$.
$\hat{I}$ heißt **offen**, falls $a < x_0$ und $x_n < b$.
($\hat{I}_M$ ist also eine offene Quadraturformel.)

7.2.3 Newton-Cotes-Formeln

Abgeschlossene, interpolatorische Quadraturformeln mit äquidistanter Stützstellenverteilung $x_0, \dots, x_n$ heißen **Newton-Cotes-Formeln der Ordnung** n. Mit $h = \frac{b-a}{n}$, $x_i = a + i \cdot h$ und Substitution $z \coloneqq \frac{t-a}{h}$ gilt

$$w_i \coloneqq h \cdot \int_0^n \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{z-j}{i-j} dz \,. \tag{7.2.3}$$

Diese sind unabhängig von der Lage der Stützstellen und liegen in Form von

$$w_i = \frac{b-a}{n \cdot s} \sigma_i$$

tabellarisch vor. Es gilt $n \cdot s$, $\sigma_i \in \mathbb{Z}$ und

$$\hat{I}_n(f) = \frac{(b-a)}{n \cdot s} \cdot \sum_{i=0}^{n} \sigma_i f(x_i) \,. \tag{7.2.4}$$

Da sie exakt sind für $p \in \mathcal{P}_n$, können $\frac{\sigma_i}{n \cdot s}$ z.B. aus dem linearen Gleichungssystem für $[a, b] = [0, 1]$

$$\int_0^1 x^k \, \mathrm{d}\, x = \frac{1}{n \cdot s} \sum_{i=0}^{n} \sigma_i x_i^k \qquad \text{für } k = 0, \dots, n$$

7.2.4 Tabelle (Newton-Cotes-Gewichte)

$n \backslash \sigma_i$	σ_1	$\dots$				$n \cdot s$	
1	1	1				2	Trapezregel
2	1	4	1			6	Simpson-Regel
3	1	3	3	1		8	$\frac{3}{8}$-Regel
4	7	32	12	32	7	90	Milne-Regel
$\vdots$							
8			$\dots$	-4540	$\dots$		

Positive Funktionen haben kein positives Integral ab einem n_0!! 07.01.2015

Daraus folgt, dass die Monotonie (Positivität) bei abgeschlossenen Newton-Cotes Formeln i.A. nicht gewährleistet ist, wenn Polynome von hohem Grad exakt integriert werden sollen. Diese sind dann daher ungeeignet.

Satz 7.2.5. *Sei $\hat{I}_n(f)$ die abgeschlossene Newton-Cotes-Formel der Ordnung n. Dann gilt*

a) $\sum_{i=0}^{n} w_i = b - a$

b) $w_i = w_{n-i}$

c) Ist n *gerade, so ist* $\hat{I}_n$ *exakt für Polynome bis zum Grad* $n+1$ *nicht nur* n*.*

Beweis. siehe Übungsblatt □

Satz 7.2.6. *Sei* $\hat{I}(f) \coloneqq \sum_{i=0}^{n} w_i f(x_i)$ *mit* $w_i \in \mathbb{R}$ *und* $a \le x_0 < x_1 < \cdots < x_n \le b$*. Dann ist der Integrationsfehler* R *mit*

$$R(f) \coloneqq \hat{I}(f) - \int_a^b f(x)\,\mathrm{d}x \tag{7.2.5}$$

eine lineare Abildung, für die folgende Darstellungsformel gilt: Falls $R(p) = 0$ *für alle* $p \in \mathcal{P}_l$*, dann gilt für alle* $f \in C^{l+1}([a,b])$

$$R(f) = \int_a^b f^{(l+1)}(t) K(t)\,\mathrm{d}t \tag{7.2.6}$$

mit

$$\begin{aligned} K(t) &\coloneqq \frac{1}{l!} R\left((\bullet - t)_+^l\right) \\ &= \frac{1}{l!}\left(\sum_{i=0}^{n} w_i (x_i - t)_+^l - \int_a^b (x-t)_+^l\,\mathrm{d}x\right) \end{aligned} \tag{7.2.7}$$

$K(t)$ *heißt* ***Peano-Kern des Funktionals*** R*.*

Beweis. Es gilt

$$f(x) = f(a) + f'(a)(x-a) + \cdots + \frac{f^{(l)}(a)}{l!}(x-a)^l + r_l(x)$$

mit Restglied

$$\begin{aligned} r_l(x) &= \frac{1}{l!}\int_a^x f^{(l+1)}(t)\cdot(x-t)^l\,\mathrm{d}t \\ &= \frac{1}{l!}\int_a^b f^{(l+1)}(t)\cdot(x-t)_+^l\,\mathrm{d}t \end{aligned}$$

Da $R(p) = 0$ für alle $p \in \mathcal{P}_l$ folgt

$$\begin{aligned} R(f) = R(r_l) &= \frac{1}{l!} R\left(\int_a^b f^{(l+1)}(t)\cdot(\bullet - t)_+^l\,\mathrm{d}t\right) \\ &= \frac{1}{l!}\left\{\sum_{i=0}^{n} w_i \int_a^b f^{(l+1)}(t)\cdot(x_i - t)_+^l\,\mathrm{d}t - \int_a^b \int_a^b f^{(l+1)}(t)\cdot(x-t)_+^l\,\mathrm{d}t\,\mathrm{d}x\right\} \end{aligned}$$

Mit dem Satz von Fubini gilt

$$\begin{aligned} R(f) &= \frac{1}{l!}\int_a^b f^{(l+1)}(t)\left\{\sum_{i=0}^{n} w_i\cdot(x_i - t)_+^l - \int_a^b (x-t)_+^l\,\mathrm{d}x\right\}\mathrm{d}t \\ &= \frac{1}{l!}\int_a^b f^{(l+1)}(t)\cdot R\left((\bullet - t)_+^l\right)\mathrm{d}t \end{aligned}$$

□

Folgerung 7.2.7. *Falls der Peano-Kern K auf $[a,b]$ ein konstantes Vorzeichen hat, gilt nach dem Mittelwertsatz der Integralrechnung*

$$R(f) = f^{(l+1)}(\xi) \int_a^b K(t)\,\mathrm{d}\,t \quad \text{für ein } \xi \in (a,b) \tag{7.2.8}$$

Wähle nun ein g, für das $R(g)$ und $g^{(l+1)}$ bekannt oder leicht berechenbar sind, z.B. $g(x) \coloneqq x^{l+1}$. Hier ist

$$\int_a^b K(t)\,\mathrm{d}\,t = \frac{R(g)}{(l+1)!} = \frac{R(x^{l+1})}{(l+1)!}$$

Dann gilt für ein beliebiges $f \in C^{l+1}([a,b])$

$$R(f) = \frac{R(x^{l+1})}{(l+1)!} f^{(l+1)}(\xi) \quad \text{für ein } \xi \in (a,b) \tag{7.2.9}$$

Satz 7.2.8 (Approximationsfehler)**.** *Im Fall der Newton-Cotes-Formeln besitzen die Peano-Kerne konstante Vorzeichen. Somit gilt für $f \in C^{n+j+1}$ mit $j = \begin{cases} 0 & \text{für } n \text{ ungerade} \\ 1 & \text{für } n \text{ gerade} \end{cases}$*

$$R_n(f) = \hat{I}_n(f) - \int_a^b f(x)\,\mathrm{d}\,x = \frac{R(x^{n+j+1})}{(n+j+1)!} f^{(n+j+1)}(\xi) \tag{7.2.10}$$

Beweis. Der Beweis des konstanten Vorzeichens gilt [siehe Ste65]. Nach Satz 7.2.5c) gilt $R_n(p) = 0 \ \forall p \in \mathcal{P}_{n+j}$. Mit (7.2.9) und $l = n + j$ folgt dann die Aussage (7.2.10) □

7.2.9 Tabelle: Fehler für Newton-Cotes-Formeln

n		$R_n(f)$
1	Trapezregel	$(b-a)^3 \cdot \frac{1}{12} f^{(2)}(\xi)$
2	Simpsonregel	$\frac{(b-a)^5}{2} \cdot \frac{1}{90} f^{(4)}(\xi)$
3	$\frac{3}{8}$-Regel	$\frac{(b-a)^5}{3} \cdot \frac{3}{80} f^{(4)}(\xi)$
4	Milne-Regel	$\frac{(b-a)^7}{4} \cdot \frac{8}{945} f^{(6)}(\xi)$

Die Erhöhung des Polynomgrades n ergibt also i.A. keine bessere Approximation. Dies hängt auch von der Länge des Intervalls $[a,b]$ ab. Stattdessen werden die Newton-Cotes-Formeln auf Teilintervalle angewendet, d.h. sie bilden die Grundlage für stückweise polynomiale interpolatorische Integrationsformeln.

7.2.10 Iterierte (wiederholte) Newton-Cotes-Formeln

Betrachte die Intervallunterteilung $a = t_0 < t_1 < \cdots < t_N = b$ und approximiere

$$I(f) = \sum_{l=0}^{N-1} \int_{t_l}^{t_{l+1}} f(t)\,\mathrm{d}\,t$$

durch wiederholtes Anwenden einer Newton-Cotes-Formel $\hat{I}_n(f)$ auf die Teilintervalle $[t_l, t_{l+1}]$ (die Formel hier sei $\hat{I}_n^{(l)}(f)$), d.h.

$$\hat{I}(f) = \sum_{l=0}^{N-1} \hat{I}_n^{(l)}(f) = \sum_{l=0}^{N-1} \frac{t_{l+1} - t_l}{n \cdot s} \sum_{i=0}^{n} \sigma_i f(t_l + i\widetilde{h}_l) \tag{7.2.11}$$

mit $\widetilde{h}_l \coloneqq (t_{l+1} - t_l) \cdot \frac{1}{n}$. Für äquidistante Knoten $t_i = a + ih$ mit $h = \frac{b-a}{N}$ ergibt die Anwendung der Trapezregel ($n = 2$) die Trapezsumme $\hat{I}_T(f) = h\left(\frac{1}{2}f(t_0) + \sum_{i=1}^{N-1} f(t_i) + \frac{1}{2}f(t_N)\right)$ nach (7.1.7). Ferner lässt sich die Konvergenzordnung (7.1.8) zeigen.

Lemma 7.2.11. *Sei $f \in C^2([a,b])$, dann existiert ein $\tau \in [a,b]$, so dass*

$$\hat{I}_T(f) - I(f) = \frac{b-a}{12} h^2 f''(\tau) \tag{7.2.12}$$

Beweis. Es gilt $t_{l+1} - t_l = h = \frac{b-a}{N}$. Daraus folgt

$$\hat{I}_T^{(l)}(f) - I^{(l)}(f) = (t_{l+1} - t_l)^3 \cdot \frac{1}{12} f''(\xi_l)$$

nach Tabelle 7.2.9 und mit $\xi_l \in [t_l, t_{l+1}]$. Weiterhin gilt dann

$$\begin{aligned}
\hat{I}_T(f) - I(f) &= h^3 \frac{1}{12} \sum_{i=0}^{N-1} f''(\xi_i) \\
&= \frac{b-a}{12} h^2 \cdot \underbrace{\frac{1}{N} \sum_{i=0}^{N-1} f''(\xi_i)}_{\coloneqq d} \\
&= \frac{b-a}{12} h^2 f''(\tau)
\end{aligned}$$

da mit $\min_{z\in[a,b]} f''(z) \le d \le \max_{z\in[a,b]} f''(z)$ aus dem Zwischenwertsatz die Existenz eines $\tau \in [a,b]$ folgt mit $f''(\tau) = d$. □

Die Trapezsumme ist also ein Quadraturverfahren zweiter Ordnung (Polynome vom Grad < 2 werden exakt integriert) und der Integrationsfehler konvergiert mit der Ordnung 2 (wegen h^2) gegen 0 für $N \to \infty$.

7.2.12 Iterierte (oder summierte) Simpsonregel (Keplersche Fassregel)

$$\hat{I}_S(f) = \frac{1}{6}\sum_{l=0}^{N-1}(t_{l+1}-t_l)\cdot\left(f(t_l)+4f\left(\frac{t_l+t_{l+1}}{2}\right)+f(t_{l+1})\right) \tag{7.2.13}$$

Für äquidistantes Gitter ergibt sich daher für $t_l = a + l \cdot h$

$$\begin{aligned}\hat{I}_S(f) = \frac{1}{6}h\cdot\Bigg\{&f(t_0)+4f\left(\frac{t_1+t_0}{2}\right)+2f(t_1)+4f\left(\frac{t_2+t_1}{2}\right)\\ &+\cdots+2f(t_{N-1})+4f\left(\frac{t_{N-1}+t_N}{2}\right)+f(t_N)\Bigg\}\end{aligned} \tag{7.2.14}$$

und wie eben die Fehlerabschätzung für $f \in C^4([a,b])$

$$\hat{I}_S(f) - I(f) = \frac{b-a}{2880}h^4 f^{(4)}(\tau) \quad \text{mit } \tau \in [a,b] \tag{7.2.15}$$

Zu beachten ist, dass aufgrund von $f\left(\frac{t_l+t_{l+1}}{2}\right)$ doppelt so viele Funktionsauswertungen nötig sind wie bei der Trapezsumme, d.h. $2N+1$ insgesamt!

Bei einer geraden Anzahl N von Intervallen kann mit der Simpsonregel jedoch eine iterierte Regel aufgestellt werden, welche wie die Trapezregel $N+1$ Funktionsauswertungen benötigt. Wende hierfür die Simpsonregel auf $[t_{2l}, t_{2l+2}]$ an bzw. setze $h = 2\cdot\frac{b-a}{N}$. Es ergibt sich mit $\overline{h} = \frac{b-a}{N}$

$$S(f) = \frac{\overline{h}}{3}\left[f(t_0)+4f(t_1)+2f(t_2)+4f(t_3)+\cdots+2f(t_{N-2})+4f(t_{N-1})+f(t_N)\right] \tag{7.2.16}$$

und es gilt die Fehlerabschätzung für $f \in C^4([a,b])$ wegen $\overline{h} = 2h$

$$S(f) - I(f) = \frac{b-a}{180}\cdot\overline{h}^4 f^{(4)}(\tau) \quad \text{mit } \tau \in [a,b] \tag{7.2.17}$$

7.3 Extrapolationsmethode und klassische Romberg-Quadratur

Sie basiert auf der asymptotischen Entwicklung des Approximationsfehlers der Trapezsumme:

Satz 7.3.1 (Euler-Maclaurinsche Summenformel)**.** *Sei $f \in C^{2n+2}([a,b])$ und $\hat{I}_T^h(f)$ die Trapezsumme für äquidistante Knoten $x_l = a + lh$ mit $h = \frac{b-a}{N}$, d.h.*

$$\hat{I}_T^h(f) = h\left\{\frac{1}{2}f(a)+\sum_{i=1}^{N-1}f(a+lh)+\frac{1}{2}f(b)\right\} \tag{7.3.1}$$

Dann gilt

$$T(h) \coloneqq \hat{I}_T^h(f) = \tau_0 + \tau_1 h^2 + \tau_2 h^4 + \cdots + \tau_n h^{2n} + R_{n+1}(h) h^{2n+1} \tag{7.3.2}$$

wobei

$$\tau_0 = I(f) = \int_a^b f(x)\,\mathrm{d}\,x \tag{7.3.3}$$

das gesuchte Integral ist, und

$$\tau_k = \frac{B_{2k}}{(2k)!}\left(f^{(2k-1)}(b) - f^{(2k-1)}(a)\right) \quad k = 1, \ldots, n$$

mit den Bernoulli-Zahlen B_{2k} sind, und

$$R_{n+1}(h) = \frac{B_{2n+2}}{(2n+2)!}(b-a) f^{(2n+2)}(\xi(h))$$

mit $a < \xi(h) < b$ eine in h gleichmäßige, beschränkte Funktion ist.

Beweis. Der Beweis ist in [SB90] oder [HH94] zu finden. □

7.3.2 Idee der Extrapolation

12.01.2015

Gesucht ist $T(0) = \tau_0 = I(f)$. Berechne hierfür für paarweise verschiedene $h_0, \ldots, h_m$ die Werte $T(h_i)$, bestimme dann das Interpolationspolynom

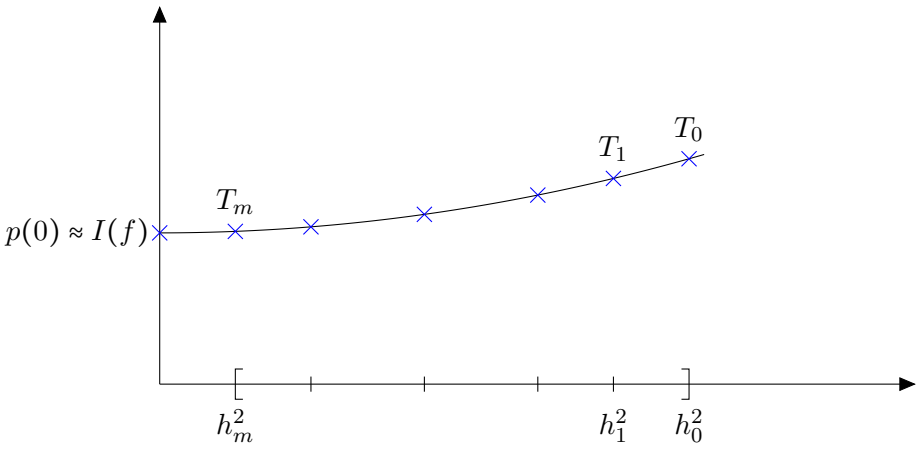
$$p(x) = a_0 + a_1 x + \cdots + a_m x^m \tag{7.3.4}$$

mit

$$p(h_i^2) = T(h_i) = \hat{I}_T^{h_i}(f) \quad \text{für } i = 0, \ldots, m \tag{7.3.5}$$

Dann ist $p(h^2)$ eine Approximation der Funktion $T(h)$ auf $[\min_i h_i, \max_i h_i]$. „Extrapoliere“ nun von diesem Intervall auf die Stelle $h = 0$ und nutze den extrapolierten Wert $p(0)$ als Näherung für $T(0) = I(f)$.

Abb. 7.3.0 *©Veranschaulichung von Extrapolation*



Da das Interpolationspolynom nur an einer Stelle, nämlich $h = 0$, berechnet werden soll, wird $p(0)$ mittels des Schemas von Neville berechnet.

$$\begin{aligned} T_{i0} &= T(h_i) && 0 \le i \le m \\ T_{ik} &= T_{i,k-1} + \frac{T_{i,k-1} - T_{i-1,k-1}}{\left(\frac{h_{i-k}}{h_i}\right)^2 - 1} && \text{für } 1 \le k \le i \le m \end{aligned} \tag{7.3.6}$$

$$\begin{array}{llllllll} i\backslash k & & & & & & & \\ h_0^2 & T_{0,0} = T(h_0) = \hat{I}_T^{h_0}(f) & \searrow & & & & & \\ h_1^2 & T_{1,0} = T(h_1) & \longrightarrow & T_{1,1} & & & & \\ h_2^2 & T_{2,0} & \longrightarrow & T_{2,1} & \longrightarrow T_{2,2} & & & \\ \vdots & & \vdots & & & \ddots & & \\ h_i^2 & T_{i,0} & \longrightarrow & T_{i,1} & \longrightarrow & \dots & T_{i,i} & \\ \vdots & & \vdots & & & & & \ddots \\ h_m^2 & T_{m,0} = T(h_m) & \longrightarrow & T_{m,1} & \longrightarrow & \dots & & T_{m,m} = p(0) \end{array}$$

wobei $T_{m,m}$ das gewünschte Ergebnis $p(0)$ liefert.

Bemerkung 7.3.3. Jedes $T_{i,k} = p(T \mid h_{i-1}^2, \dots, h_i^2)(0)$ stellt eine lineare Integrationsformel dar

$$T_{i,k} = \sum_{l=0}^{m} \alpha_l f(t_l) \quad \text{mit } t_l \in [a,b]$$

7.3.4 Aufwand

Es sind $\mathcal{O}(m^2)$ flops für das Schema von Neville nötig und die Berechnung von $\hat{I}_T^{h_0}(f), \dots, \hat{I}_T^{h_m}(f)$, also die Funktionsauswertungen $f(a + lh_i)\ \forall l, h_i$.

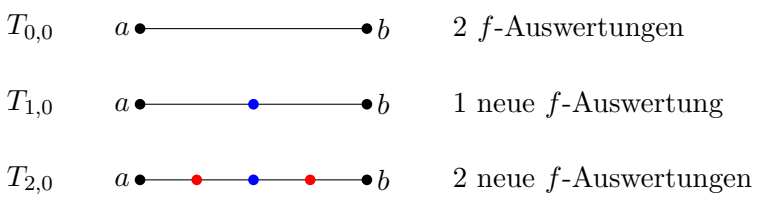
7.3.5 Klassische Romberg-Folge zur Romberg-Quadratur

Wähle

$$h_0 \coloneqq b - a, h_1 \coloneqq \frac{h_0}{2}, \dots, h_i \coloneqq \frac{h_{i-1}}{2} \tag{7.3.7}$$

d.h. $N_i = 2^i, H = b - a$ und $h_i = \frac{H}{N_i}$.

Abb. 7.3.1 *Auswertungspunkte h_i der Romberg-Folge*



Pro Halbierung $(h_{j-1} \to h_j)$ sind 2^{j-1} neue f-Auswertungen nötig. Für $T_{i,j}$ sind folglich $2 + \sum_{j=1}^{i} 2^{j-1} = 2^i + 1$ f-Auswertungen notwendig und

$$T_{i+1,0} = T(h_{i+1}) = T\left(\frac{1}{2}h_i\right) = \frac{1}{2}\underbrace{T(h_i)}_{=T_{i,0}} + h_{i+1}\big(\ldots f(a+h_{i+1}) + f(a+3h_{i+1}) + \cdots + f(b-h_{i+1})\big) \tag{7.3.8}$$

7.3.6 Bulirsch-Folge

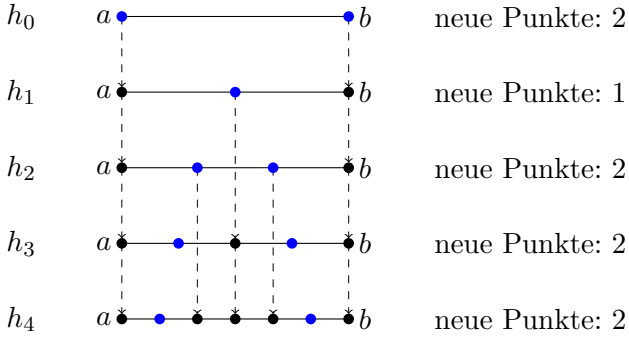
Sie wurde von Bulirsch 1964 vorgeschlagen. Wähle

$$H = h_0 = b-a, h_1 = \frac{h_0}{2}, h_2 = \frac{h_0}{3} \text{und } h_i = \frac{h_{i-2}}{2} \text{für } i \geq 3. \tag{7.3.9}$$

d.h.

$$N_i = \begin{cases} 1 & i = 0 \\ 2^l & i = 2l-1 \\ 3 \cdot 2^{l-1} & i = 2l \end{cases}$$

Abb. 7.3.2 *Auswertungspunkte h_i der Bulirsch-Folge*



Es sind ungefähr die Hälfte der f-Auswertungen der Romberg-Folge nötig. Ein Nachteil ist jedoch, dass die $T_{j,k}$ entsprechenden Quadraturformeln negative Gewichte α_i enthalten können, während sie bei der Romberg-Folge positiv bleiben.

7.3.7 Verfahren und Abbruch

In der Regel ist für f kein genaues m mit $f \in C^{2(m+1)}([a,b])$ bekannt, gebe stattdessen ein m_{max} von f vor. Beachte weiterhin, dass $T_{j,m_{max}} = p\left(T \mid h_{j-m_{max}}^2, \ldots, h_j^2\right)(0)$ für

$j > m_{max}$ ebenfalls τ_0 approximiert und für $h_0 \to 0$ besser wird (später). Daher gehe in der Tabelle von Neville zeilenweise vor.

$$\begin{array}{llllll} h_0^2 & T_{0,0} & & & & \\ h_1^2 & T_{1,0} & T_{1,1} & & & \\ h_2^2 & T_{2,0} & T_{2,1} & T_{2,2} & & \\ & \vdots & & & \ddots & \\ h_{m_{max}}^2 & T_{m_{max},1} & & \dots & & T_{m_{max},m_{max}} \end{array}$$

Abb. 7.3.3 *Veranschaulichung der Bulirsch-Folge*

Es wird fortgefahren, bis $T_{i,m_{max}}$ „genau genug“ ist. Z.B. wird $m_{max} = 7$ gesetzt und abgebrochen, falls

a) i zu groß wird

b) genügend viele Ziffern „stehen“ oder $T_{i,m_{max}} \approx T_{i+1,m_{max}}$, was z.B. durch

$$|T_{i,m_{max}} - T_{i+1,m_{max}}| \leq \varepsilon |s| \tag{7.3.10}$$

mit grobem Schätzwert s von τ_0 und einer relativen Genauigkeitstoleranz ε getestet wird.

Zum Konvergenzbeweis nutze folgende Aussage:

Lemma 7.3.8. *Für die Lagrange-Funktionen L_i zu den Stützstellen $t_0, \dots, t_k$ gilt*

a) $\sum_{j=0}^{k} L_j(0) t_j^l = p(x^l \mid t_0, \dots, t_k)(0) = \begin{cases} 1 & l = 0 \\ 0 & 1 \leq l \leq k \\ (-1)^k t_0 \cdots t_k & l = k+1 \end{cases}$

b) Falls $t_j = h_j^2$ mit $0 < \frac{h_{j+1}}{h_j} \leq const < 1$ gilt $\sum_{j=0}^{k} |L_j(0)| t_j^{k+1} \leq c_k t_0 \cdots t_k$

Beweis.

a) Übungsaufgabe.

b) Allgeiner Fall siehe [SB90], für geometrische Folgen $h_j = h_0 c^3$ und $0 < c < 1$ siehe [Sto].

□

Satz 7.3.9. *Sei $T(h)$ wie in (7.3.7) und gelte $0 < \frac{h_{j+1}}{h_j} \leq const < 1$. Dann gilt für $i \geq k$ und $k < m$*

$$T_{i,k} - \tau_0 = (-1)^k h_{i-k}^2 \cdots h_2^2 \left(\tau_{k+1} + \mathcal{O}(h_{i-k}^2)\right) \tag{7.3.11}$$

und für $i \geq m$

$$|T_{i,m} - \tau_0| \leq c_m h_{i-m}^2 \cdots h_i^2 \tag{7.3.12}$$

Korollar 7.3.10.

a) Für festes $k \leq m$ gilt asymptotisch für $i \to \infty$ mit $h_{i-k} \to 0$

$$T_{i,k} - \tau_0 = \mathcal{O}\left(h_{i-k}^{2(k+1)}\right) \tag{7.3.13}$$

d.h. die Elemente $T_{i,k}$ in der $(k+1)$-ten Spalte der Tabelle konvergieren gegen τ_0 mit der Ordnung $2(k+1)$.

b) Pro Spalte $(k \to k+1)$ gewinnt das Verfahren zwei Ordnungen.

Beweis. zu 7.3.9 Sei $k = 0$, dann gilt

$$T_{j,0} = T(h_j) = \tau_0 + \tau_1 h_j^2 + \cdots + \tau_m h_j^{2m} + \mathcal{O}\left(h_j^{2(m+1)}\right)$$

Daraus folgt

$$T_{j,0} - \tau_0 = h_j^2 \left(\tau_1 + \mathcal{O}(h_j^2)\right)$$

Da $T_{i,k} = p(T \mid h_{i-k}^2, \ldots, h_i^2)(0) = p(T \mid \widetilde{h}_0^2, \ldots, \widetilde{h}_k^2)(0) =: \widetilde{T}_{k,k}$, reicht es $T_{k,k}$ zu betrachten. Sei nun $k < m$ und $i = k$

$$\begin{aligned}
T_{k,k} = p(0) &= \sum_{j=0}^{k} L_j(0) T_{j,0} \\
&= \sum_{j=0}^{k} L_j(0) \left(\sum_{l=0}^{k+1} \tau_l h_j^{2l} + \mathcal{O}\left(h_j^{2(k+1)}\right) \right) \\
&= \sum_{l=0}^{k+1} \tau_l \sum_{j=0}^{k} L_j(0) \left(h_j^2\right)^l + \underbrace{\sum_{j=0}^{k} L_j(0) \cdot \mathcal{O}\left(h_j^{2(k+2)}\right)}_{\substack{= \mathcal{O}\left(\sum_{j=0}^{k} |L_j(0)| \cdot h_j^{2(k+1)}\right) \\ = \mathcal{O}\left(h_0^2 \sum_{j=0}^{k} |L_j(0)| \cdot h_j^{2(k+1)}\right)}} \\
&\overset{\text{nach } 7.3.8}{=} \tau_0 + (-1)^k h_0^2 \cdots h_k^2 \tau_{k+1} + c_k h_0^2 \cdots h_k^2 \mathcal{O}\left(h_0^2\right)
\end{aligned}$$

Für $i = k = m$:

$$T_{m,m} = \underbrace{\sum_{l=0}^{m} \tau_l \sum_{j=0}^{m} L_j(0) h_j^{2l}}_{= \tau_0} + \sum_{j=0}^{m} L_j(0) \mathcal{O}\left(h_j^{2(m+1)}\right)$$

Also ist nach 7.3.8 $|T_{m,m} - \tau_0| = \mathcal{O}\left(h_0^2 \cdots h_m^2\right)$. □

14.01.2015

7.4 Gauß-Quadratur und Orthogonalsysteme

Bei der Konstruktion der Newton-Cotes-Formeln sind die $(n+1)$ Knoten x_i fest (äquidistant) gegeben. Die Gewichte w_i sind dann so bestimmt, dass

$$\hat{I}_n(f) = \sum_{i=0}^{n} w_i f(x_i)$$

zumindest für alle $f \in \mathcal{P}_n$ exakt sind. Für die Gauß-Quadratur werden sowohl $(n+1)$ Knoten x_i wie auch die Gewichte w_i so bestimmt, dass $\hat{I}_n$ das Integral mit möglichst hoher Ordnung approximiert. Da $(2n+2)$ Parameter frei sind, ist maximal eine Ordnung $(2n+2)$ zu erwarten, d.h. $\hat{I}_n$ ist für $p \in \mathcal{P}_{2n+1}$ exakt. Da x_i nicht linear in $\hat{I}_n$ eingeht, ist u.a. nicht offensichtlich, dass dies erreicht wird. Der Ansatz für Gauß-Quadratur ist noch allgemeiner: Betrachtet werden gewichtete Integrale

$$I(f) \coloneqq \int_a^b w(x) f(x) \,\mathrm{d}\, x \tag{7.4.1}$$

wobei a und b auch $-\infty$ oder $+\infty$ sein können.

7.4.1 Voraussetzungen an w

a) w ist auf (a,b) stetig und nicht negativ

b) Alle **Momente**

$$\mu_k \coloneqq \int_a^b x^k w(x) \,\mathrm{d}\, x \quad k = 0, 1, \ldots$$

sind endlich.

c) Für jedes Polynom p mit $p(x) \geq 0$ für $x \in [a,b]$ und $\int_a^b w(x)p(x) \,\mathrm{d}\, x = 0$ gilt $p \equiv 0$.

Zu einem gegebenen w führen wir das Skalarprodukt ein

$$(f,g) \coloneqq \int_a^b w(x) f(x) g(x) \,\mathrm{d}\, x \tag{7.4.2}$$

mit $\|f\|^2 \coloneqq (f,f)$. Zwei Funktionen f und g heißen orthogonal, falls $(f,g) = 0$ gilt.

Lemma 7.4.2. *Es gibt keine reellen Zahlen x_i, w_i für $i = 0, \ldots, n$, so dass die zugehörige Quadratur $\hat{I}_n$ exakt ist für alle Polynome in $\mathcal{P}_{2n+2}$.*

Beweis. Angenommen es exisieren x_i, w_i, so dass die Quadratur für alle Polynome in $\mathcal{P}_{2n+2}$ exakt ist. Wähle nun $\overline{p}(x) \coloneqq \prod_{j=0}^{n} (x - x_j)^2 \in \mathcal{P}$. Dann folgt ein Widerspruch, da

$$0 < I(\overline{p}) = \hat{I}_n(\overline{p}) = \sum_{i=0}^{n} w_i \overline{p}(x_i) = 0$$

□

Das Ziel ist es, zu einem gegebenen w und n die Werte x_i und w_i zu finden. Ein Mittel hierzu sind die Orthogonalpolynome zur Bestimmung der x_i.

Lemma 7.4.3. *Falls $\hat{I}_n$ für alle $p \in \mathcal{P}_{2n+1}$ exakt ist, ist*

$$P_{n+1}(x) \coloneqq (x - x_0)\cdots(x - x_n) \in \mathcal{P}_{n+1}$$

orthogonal bzgl. $(\bullet, \bullet)$ zu allen $p \in \mathcal{P}_n$.

Beweis. Sei $p \in \mathcal{P}_n$, dann gilt $p \cdot P_{n+1} \in \mathcal{P}_{2n+1}$, also

$$(p, \mathcal{P}_{n+1}) = I(p \cdot P_{n+1}) = \hat{I}_n(pP_{n+1}) = \sum_{i=0}^{n} w_i p(x_i) P_{n+1}(x_i) = 0$$

□

Es stellen sich noch folgende Fragen:

1. Existiert ein (eindeutiges) P_{n+1}, welches orthogonal ist zu $\mathcal{P}_n$ und normalisiert?
2. Sind die Nullstellen von P_{n+1} einfach und reell?
3. Wie sind die w_i zu wählen?
4. Ist dann $\hat{I}_n(p)$ exakt für alle $p \in \mathcal{P}_{2n+1}$?

Satz 7.4.4 (Existenz von Orthogonalpolynomen)**.** *Es gibt für $j = 0, 1, \ldots$ eindeutig bestimmte, normalisierte Polynome $p_j \in \mathcal{P}_j$ mit*

$$(p_j, p_k) = 0 \quad \forall j \neq k$$

Diese Polynome genügen der (Drei-Term-)Rekursionsformel mit $p_{-1} \coloneqq 0, \gamma_0 \coloneqq 1$

$$\begin{aligned} &1. \quad p_0(x) = 1 \\ &2. \quad p_{i+1}(x) = (x - \delta_i)p_i(x) - \gamma_i p_{i-1}(x) \quad \textit{für } i \geq 0 \end{aligned} \tag{7.4.3}$$

und

$$\begin{aligned} \delta_i &= \frac{(xp_i, p_i)}{(p_i, p_i)} && \textit{für } i \geq 0 \\ \gamma_i &= \frac{(p_i, p_i)}{(p_{i-1}, p_{i-1})} && \textit{für } i \geq 1 \end{aligned} \tag{7.4.4}$$

Beweis. Wir beweisen per Induktion.

$i = 0$**:** $p_0 \in \mathcal{P}_0$, normalisiert: klar.

$i \Rightarrow i+1$**:** Es ist $xp_i \in \mathcal{P}_i$ und damit $p_{i+1} = xp_i - \sum_{l=0}^{i} c_l p_l \in \mathcal{P}_{i+1}$ und p_{i+1} ist normalisiert. Mit Induktionsvoraussetzung gilt $(p_l, p_j) = 0$ für $0 \le l \ne j \le i$. Also $0 = (p_{i+1}, p_j) = (xp_i - c_j p_j, p_j)$ für $j = 0, \dots, i$ mit entsprechend

$$c_j = \frac{(xp_i, p_j)}{(p_j, p_j)} = \frac{(p_i, xp_j)}{(p_j, p_j)}$$

Weiterhin folgt nach Induktionsvoraussetzung für $1 \le j+1 \le i$

$$p_{j+1}(x) = (x - \delta_j)p_j(x) - \gamma_j p_{j-1}(x)$$

Demnach ist $c_0 = \cdots = c_{i-2} = 0$ und $\gamma = c_{i-1} = \frac{(p_i,p_i)}{(p_{i-1},p_{i-1})}$ aufgrund der Orthogonalität und $\delta = c_i = \frac{(xp_i,p_i)}{(p_i,p_i)}$. Mit c_i ist auch p_{i+1} eindeutig.

□

Damit folgt:

Korollar 7.4.5. *Es gilt für p_{n+1} aus 7.4.4 $(p, p_{n+1}) = 0 \ \forall p \in \mathcal{P}_n$. Das eindeutige p_{n+1} erfüllt somit die Bedingungen von Lemma 7.4.3.*

Satz 7.4.6. *Die Nullstellen $x_0, \dots, x_{n-1}$ von p_n sind reell, einfach und liegen im offenen Intervall (a, b).*

Beweis. Seien $a < x_0 < \cdots < x_l < b$ die Nullstellen von p_n, an denen p_n das Vorzeichen wechselt, d.h. die reellen Nullstellen ungerader Vielfachheit in (a, b).

Es ist zu zeigen, dass $l = n - 1$. Dazu sei angenommen, dass $l < n - 1$, dann gilt für das normalisierte Polynom $q(x) \coloneqq \prod_{j=0}^{l}(x - x_j) \in \mathcal{P}_l$ die Gleichung $(p_n, q) = 0$. Andererseits ändert $p_n \cdot q \not\equiv 0$ auf (a, b) das Vorzeichen nicht, also würde gelten

$$(p_n, q) = \int_a^b w(x)p_n(x)q(x)\,\mathrm{d}\,x \underset{\text{Vor. an } w}{\overset{\uparrow}{\ne}} 0$$

Dies ist ein Widerspruch. □

Beispiel 7.4.7. Im Folgenden eine Tabelle mit orthogonalen Polynomsystemen für verschiedene Gewichtsfunktionen und Intervalle:

$w(x)$	$[a, b]$	
1	$[-1, 1]$	Legendre-Polynome P_n
$\frac{1}{\sqrt{1-x^2}}$	$[-1, 1]$	Tschebyscheff-Polynome T_n
e^{-x}	$[0, \infty]$	Laguerre-Polynome L_n
e^{-x^2}	$[-\infty, \infty]$	Hermite-Polynome H_n

Die **Gewichte** w_i für die Gauß-Quadratur $\hat{I}_n(f) = \sum_{i=0}^n w_i f(x_i)$ sind nun für die bestimmten Orthogonalpolynome und deren Nullstellen $x_0, \dots, x_n$ eindeutig durch

$$w_i = \int_a^b w(x) L_i(x) \, \mathrm{d}\, x \tag{7.4.5}$$

mit den zu $x_0, \dots, x_n$ gehörigen Lagrange-Polynomen gegeben, damit $\hat{I}_n$ für $p \in \mathcal{P}_n$ exakt ist. (Berechnet werden die Gewichte i.A. anders.) Für einige Gewichtsfunktionen sind sie in [Sto] oder [HH94] zu finden, weitere in [AS64]. Folgende (äquivalente) Bestimmung der w_i liefert mehr Informationen:

Satz 7.4.8.

1) Seien $x_0, \dots, x_n$ die Nullstellen von p_{n+1} und $(w_0, \dots, w_n)$ die Lösung des linearen Gleichungssystem

$$\sum_{i=0}^n w_i p_k(x_i) = \begin{cases} (p_0, p_0) = \int_a^b w(x) \, \mathrm{d}\, x & \text{falls } k = 0 \\ 0 & \text{falls } k = 1, \dots, n \end{cases}. \tag{7.4.6}$$

Dann gilt $w_i > 0$ für $i = 0, \dots, n$ sowie

$$\int_a^b w(x) p(x) \, \mathrm{d}\, x = \sum_{i=0}^n w_i p(x_i) \quad \forall p \in \mathcal{P}_{2n+1} \tag{7.4.7}$$

2) Gilt umgekehrt (7.4.7) für gewisse reelle Zahlen x_i, w_i für $i = 0, \dots, n$, so sind die x_i die Nullstellen von p_{n+1} und die w_i erfüllen (7.4.6).

Beweis. 1) Übungsaufgabe, 2) z.B. [Sto]. □

Satz 7.4.9 (Approximationsfehler)**.** *Für $f \in C^{2n+2}([a,b])$ gilt für ein $\xi \in (a,b)$*

$$\int_a^b w(x) f(x) \, \mathrm{d}\, x - \sum_{i=0}^n w_i f(x_i) = \frac{f^{(2n+2)}(\xi)}{(2n+2)!} (p_{n+1}, p_{n+1}) \tag{7.4.8}$$

Beweis. Betrachte die Hermite-Interpolierende $h \in \mathcal{P}_{2n+1}$ mit $h(x_i) = f(x_i)$ und $h'(x_i) = f'(x_i)$ für $i = 0, \dots, n$. Da $h \in \mathcal{P}_{2n+1}$ folgt $I(h) = \sum_{i=0}^n w_i h(x_i) = \hat{I}_n(f)$ und daraus

$$I(f) - \hat{I}_n(f) = I(f) - I(h) = \int_a^b w(x) \cdot (f(x) - h(x)) \, \mathrm{d}\, x$$

Aufgrund der Restglieddarstellung bei Polynominterpolation gilt

$$\begin{aligned} f(x) - h(x) &= \frac{f^{(2n+2)}(\xi(x))}{(2n+2)!} (x - x_0)^2 \cdots (x - x_n)^2 \\ &= \frac{f^{(2n+2)}(\xi(x))}{(2n+2)!} p_{n+1}^2(x) \end{aligned}$$

$\xi(x)$ liegt im kleinsten Intervall, welches x und alle x_i enthält. Mit f, h, p_{n+1} ist auch $f^{(2n+2)}(\xi)$ stetig von x abhängig. Da $w(x)p_{n+1}^2(x) \geq 0$ konstantes Vorzeichen hat, folgt mit dem Mittelwertsatz der Integralrechnung

$$I(f) - \hat{I}_n(f) = \frac{f^{(2n+2)}(\xi)}{(2n+2)!} \cdot \int_a^b w(x)p_{n+1}(x)p_{n+1}(x)\,\mathrm{d}\,x$$

□

7.4.10 Vergleich und Bemerkungen

Für $w(x) \equiv 1$ auf $[-1,1]$ ergibt sich die Gauß-Legendre-Quadratur für die gilt [siehe HH94]

$$(p_{n+1}, p_{n+1}) = \frac{(n+1)!^4}{(2n+2)!^2} \cdot \frac{2^{2n+3}}{(2n+3)}$$

Die Transformation auf das Intervall $[a,b]$ ergibt dann

$$\int_a^b f(x)dx - \hat{I}_n(f) = \left(\frac{b-a}{n}\right)^{2n+3} \cdot \underbrace{\frac{(n+1)!^4 \cdot n^{2n+3}}{(2n+2)!^3(2n+3)}}_{\leq 1} f^{(2n+2)}(\xi)$$

Im Vergleich hierzu liefert die Newton-Cotes-Formel

$$\left|I(f) - \hat{I}_n(f)\right| \leq \left(\frac{b-a}{n}\right)^{n+2+I} K_{n,b,a} f^{(n+1+I)}(\xi) \qquad \text{mit } I = \begin{cases} 0 & \text{für } n \text{ ungerade} \\ 1 & \text{für } n \text{ gerade} \end{cases}$$

In beiden Fällen sind $n+1$ **Funktionsauswertungen** nötig.

Beispielsweise für $n = 4$:

Gauß-Legendre: $\underbrace{\frac{1024}{618866325}}_{\approx 1{,}654 \cdot 10^{-6}} \cdot \left(\frac{b-a}{4}\right)^{11} f^{(10)}(\xi)$

Newton-Cotes: $\underbrace{\frac{8}{145}}_{=0{,}0552} \cdot \left(\frac{b-a}{4}\right)^{7} f^{(6)}(\xi)$

Newton-Cotes hat für große n **negative Gewichte** und wird daher fur große n kaum angewendet. Um einen kleinen Approximationsfehler zu erhalten, muss Newton-Cotes iteriert, d.h. auf Teilintervalle angewendet werden.

Gauß-Legendre hat nur **positive Gewichte** und besitzt die höchste Ordnung (auch im Vergleich zum Romberg-Verfahren). Wenn zu einer vorgegebenen Fehlertoleranz die Gauß-Quadratur angewendet werden soll, ist die Schwierigkeit, das entsprechende n zu wählen, da i.d.R. keine vernünftigen Abschätzungen für $f^(m)(\xi)$ vorliegen.

Gewöhnlich werden die Gauß-Formeln für wachsende n solange angewendet, bis die Näherungswerte bis auf die gewünschte Genauigkeit übereinstimmen. In jedem Schritt müssen jedoch n neue Funktionswerte berechnet werden, da sich die Knoten ändern. (Es gibt Verfahren, die diesen Nachteil ausbessern, [siehe Sto]).

Das ist beim Romberg-Verfahren nicht der Fall, welches deswegen in den meisten Fällen „überlegen“ ist. Desweiteren kann beim Romberg-Verfahren $h \to 0$ gehen, ohne höhere Ableitung vorauszusetzen.

Die Gauß-Legendre-Quadratur kann – wie Newton-Cotes – auf Teilintervalle angewendet werden. Dann kann für festes n $h \to 0$ gehen.

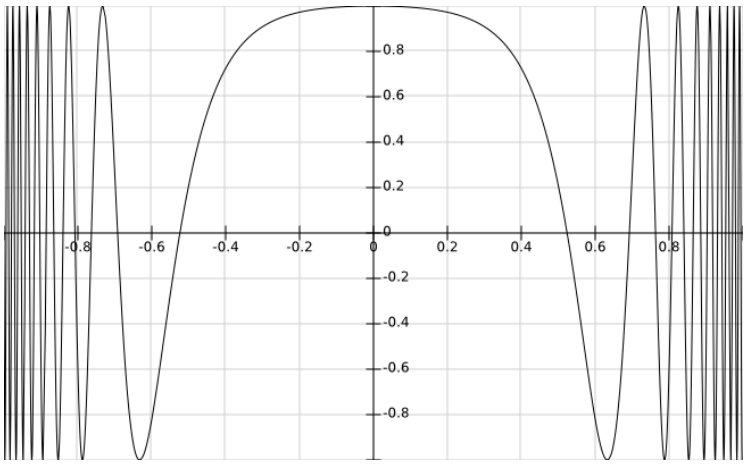
Einige große Vorteile der Gauß-Quadratur sind:

a) Sie ist anwendbar auch für Gewichtsfunktionen $w \not\equiv 1$, d.h. insbesondere für schwach singuläre Funktionen (f integrierbar, aber $f^(k)$ existiert nicht auf $[a,b]$) wie z.B. $\frac{1}{\sqrt{1-x^2}}$, für die die Trapezregel nicht anwendbar ist.

b) Sie kann auch Integrale über unendliche Intervalle approximieren (z.B. $\int_0^\infty e^{-x}\,\mathrm{d}\,x$). Um die Approximationsgüte zu verbessern, muss dann jedoch n vergrößert werden. Eine Zerlegung in Teilintervalle ist nicht mehr möglich.

Integrale mit Singularitäten müssen meist vorher umgeschrieben werden, z.B. Intervallunterteilung, Substitutionen, Reihenentwicklung, ... Mögliche Methoden sind in [Sto] beschrieben. Bei manchen Integralen ist die numerische Quadratur „machtlos“ (z.B. bei stark oszillierenden Funktionen wie $f(t) = \cos(te^{4t^2})$).

Abb. 7.4.0 *Funktionsplot von $f(t) = \cos(te^{4t^2})$:*
Stark oszillierende Funktion, bei der numerische Quadratur misslingt



8 Eigenwertabschätzung

8.1 Eigenwertabschätzungen

Die Eigenwerte einer Matrix sind stetige Funktionen der Elemente der Matrix. Für Eigenvektoren gilt dies i.A. nicht.

Satz 8.1.1 (Gerschgorin, 1931)**.** *Die Eigenwerte λ einer Matrix $A = (a_{ij})_{i,j} \in \mathbb{C}^{n\times n}$ liegen in der Vereinigung aller* ***Gerschgorin-Kreise***

$$K_i := \{z \in \mathbb{C} \mid |z - a_{ii}| \leq r_i\}$$

deren Radien durch $r_i := \sum_{\substack{k=1 \\ k\neq i}}^{n} |a_{ik}|$ gegeben sind.

Beweis. Es gilt per Definition $Av = \lambda v \Leftrightarrow (\lambda - a_{ll})v_l = \sum_{k\neq l} a_{lk}v_k$. Sei nun l so, dass $v_l = \|v\|_\infty$. Dann folgt $|\lambda - a_{ll}| \leq \sum_{k\neq l} |a_{lk}| = r_k$. □

Obige Aussage lässt sich verschärfen durch die Betrachtung von A^T oder $D^{-1}AD$ etc. Je weniger sich A von einer Diagonalmatrix unterscheidet, desto schärfer ist die Abschätzung.

Satz 8.1.2. *Bilden k Gerschgorin-Kreise eine Menge G, die zu den restlichen K_i disjunkt ist, dann liegen in G genau k Eigenwerte der Matrix A.*

Beweis. Betrachte

$$A(t) = \begin{pmatrix} a_{11} & & 0 \\ & \ddots & \\ 0 & & a_{nn} \end{pmatrix} + t(A - D)$$

Dann ist $A(0) = D$ und $A(1) = A$. Für $t = 0$ gilt $\lambda_i(0) = a_{ii}$ und es liegen genau k Eigenwerte in G und $n - k$ in den restlichen K_j. Weiterhin ist $K_i(t) \subseteq K_i$ für $t \in [0, 1]$. Da $\lambda_i(t)$ stetig von t abhängt und $\lambda_i(t) \subseteq \bigcup_{l=1}^{n} K_l$ und die restlichen K_j disjunkt zu G sind, gilt die Aussage des Satzes für alle $t \in [0, 1]$. □

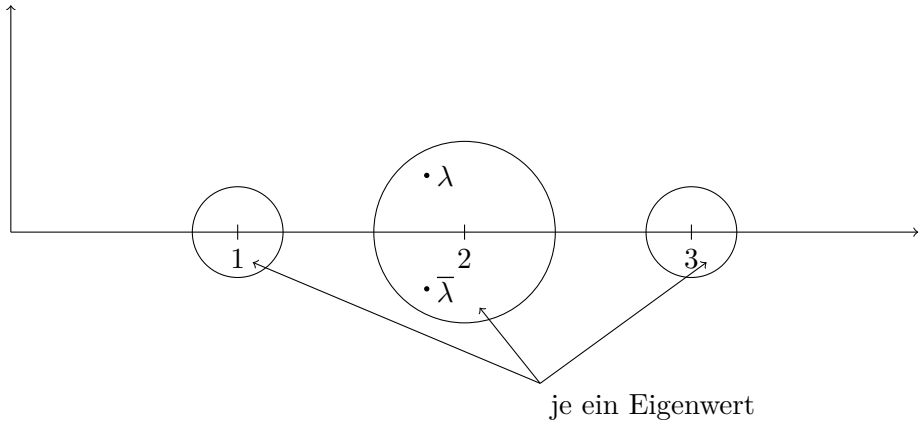
Beispiel 8.1.3. Sei

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

Dann ist

$$K_1 = \{\mu \mid |\mu - 1| \leq 0,2\}$$
$$K_2 = \{\mu \mid |\mu - 2| \leq 0,4\}$$
$$K_3 = \{\mu \mid |\mu - 3| \leq 0,2\}$$

Abb. 8.1.0 *Gerschgorin-Kreise*



da mit λ auch $\overline{\lambda}$ Eigenwert ist, müssen alle Eigenwerte reell sein:

$$\lambda_1 \approx 0,9862$$
$$\lambda_2 \approx 2,0078$$
$$\lambda_3 \approx 3,0060$$

Solche Abschätzungen werden u.a. für die numerische Approximation von Eigenwerten benutzt.

8.2 Potenzmethode (Vektoriteration, power method)

8.2.1 Voraussetzung für die Potenzmethode

Sei $A \in \mathbb{R}^{n \times n}$ eine in $\mathbb{C}$ diagonalisierbare Matrix und sei der dominante Eigenwert λ_1 einfach, d.h. $|\lambda_1| > |\lambda_2| \geq |\lambda_1| \geq \cdots \geq |\lambda_n|$ mit den bzgl. $\| \bullet \|_2$ normierten Eigenvektoren v_i. Weiter sei ein Startvektor $x^{(0)} \in \mathbb{R}^n$ gegeben, welcher keinen Nullanteil in v_1 hat.

8.2.2 Vektoriteration

Die Vektoriteration ist gegeben durch

$$x^{(k+1)} \coloneqq Ax^{(k)} \quad \text{für } k \geq 0 \tag{8.2.1}$$

Hierfür folgt: Da A reell ist, muss $\lambda_1 \in \mathbb{R}$ und $v_1 \in \mathbb{R}^n$ gelten. Da A in $\mathbb{C}$ diagonalisierbar ist, gibt es eine Basis des $\mathbb{C}^n$ aus normierten Eigenvektoren $\{v_1, \ldots, v_n\}$ und es gilt $x^{(0)} = \sum_{i=1}^n \alpha_i v_i$. Nach Voraussetzung ist $\alpha_1 \neq 0$ und $\alpha_1 \in \mathbb{R}$.

$$\begin{aligned} x^{(k)} &= A^k x^{(0)} \\ &= \sum_{i=1}^n \alpha_i \lambda_i^k v_i \\ &= \alpha_1 \lambda_1^k \left(v_1 + \sum_{i=2}^n \frac{\alpha_i}{\alpha_1} \cdot \left(\frac{\lambda_i}{\lambda_1} \right)^k v_i \right) \\ &= \alpha_1 \lambda_1^k \left(v_1 + r^{(k)} \right) \end{aligned} \tag{8.2.2}$$

mit $r^{(k)} \coloneqq \sum_{i=1}^n \frac{\alpha_i}{\alpha_1} \cdot \left(\frac{\lambda_i}{\lambda_1} \right)^k v_i$.

Da $\left| \frac{\lambda_i}{\lambda_1} \right| \leq \left| \frac{\lambda_2}{\lambda_1} \right| < 1$ gilt, gilt $\left\| r^{(k)} \right\| \leq \left| \frac{\lambda_2}{\lambda_1} \right|^k \sum_{i=2}^n \left| \frac{\alpha_i}{\alpha_1} \right|^k \underbrace{\|v_i\|_2}_{=1}$ also

$$\left\| r^{(k)} \right\|_2 = \mathcal{O}\left(\left| \frac{\lambda_2}{\lambda_1} \right|^k \right) \overset{k \to \infty}{\longrightarrow} 0$$

Für große k verhält sich $x^{(k)}$ daher wie $x^{(k)} \approx \alpha_1 \lambda_1^k v_1$ und „konvergiert“ gegen ein Vielfaches des Eigenvektors v_1, bzw. genauer

$$\frac{x^{(k)}}{\lambda_1^k} = \alpha_1 v_1 + \alpha_1 r^{(k)} \overset{k \to \infty}{\longrightarrow} \alpha_1 v_1$$

Da $\left\| x^{(k)} \right\|_2 \longrightarrow \infty$ für $|\lambda_1| > 1$ und $\left\| x^{(k)} \right\|_2 \longrightarrow 0$ für $|\lambda_1| < 1$, ist es zweckmäßig $x^{(k)}$ zu normieren. Setze also

$$v^{(k)} \coloneqq \frac{x^{(k)}}{\left\| x^{(k)} \right\|} \in \mathbb{R}^n \tag{8.2.3}$$

so folgt, da mit $\lambda_1 \in \mathbb{R}$, $\operatorname{sign}(\lambda_1) = \frac{|\lambda_1|}{\lambda_1} \in \{-1, 1\}$

$$\operatorname{sign}(\lambda_1)^k v^{(k)} = \frac{\alpha_1}{|\alpha_1|} \cdot \frac{v_1 + r^{(k)}}{\left\| v_1 + r^{(k)} \right\|} = \operatorname{sign}(\alpha_1) v_1 + \mathcal{O}\left(\left| \frac{\lambda_2}{\lambda_1} \right|^k \right) \tag{8.2.4}$$

Sei nun

$$\lambda^{(k)} = {v^{(k)}}^T A v^{(k)} \tag{8.2.5}$$

so folgt

$$\begin{aligned}\lambda^{(k)} &= \underbrace{\left(\operatorname{sign}(\alpha_1)\operatorname{sign}(\lambda_1)^k v_1^T + \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)\right)}_{v^{(k)^T}} A v^{(k)} \\ &= v_1^T A v_1 + \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right) \\ &= \lambda_1 + \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)\end{aligned}$$

da v_1 normierter Eigenvektor zu λ_1.

8.2.3 Direkte Potenzmethode/Vektoriteration

Wähle $x^{(0)} \in \mathbb{R}^n$
$v^{(0)} := x^{(0)} / \left\|x^{(0)}\right\|_2$
for $k = 0, 1, 2, \ldots,$
| $\widetilde{x}^{(k+1)} = A v^{(k)}$
| $\lambda^{(k)} = {v^{(k)}}^T \widetilde{x}^{(k+1)}$
| $v^{(k+1)} = \widetilde{x}^{(k+1)} / \left\|\widetilde{x}^{(k+1)}\right\|_2$
end

Eine Bemerkung hierzu: $\widetilde{x}^{(k)}$ ist ein Vielfaches von $x^{(k)}$. Durch die Normierung zu $v^{(k)}$ spielt das keine Rolle.

Es folgt

Satz 8.2.4. *Unter den Voraussetzungen 8.2.1 approximiert die Vektoriteration 8.2.3 den dominanten Eigenwert $\lambda_1 \in \mathbb{R}$ und einen zugehörigen normierten Eigenvektor $v_i \in \mathbb{R}^n$. Weiterhin gilt*

$$\left|\lambda_1^{(k)} - \lambda_1\right| = \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right) \qquad \textit{und} \qquad \left\|\operatorname{sign}(\lambda_1)^k v^{(k)} - v_1\right\| = \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$$

Die Effizienz hängt also von $\left|\frac{\lambda_2}{\lambda_1}\right|$ ab.

Bemerkung 8.2.5.

a) Falls $(v_1)_l \neq 0$ ist, gilt

$$\lim_{k\to\infty} \frac{x^{(k)}}{x_l^{(k)}} = \frac{1}{(v_1)_l} \cdot v_1 \quad \text{und} \quad \lim_{k\to\infty} \frac{x_l^{(k)}}{x_l^{(k-1)}} = \lambda_1 \tag{8.2.6}$$

b) Für eine mehrfache Nullstelle $\lambda_1 = \cdots = \lambda_m$ gilt weiterhin (8.2.6) und $v^{(k)}$ approximiert einen Eigenvektor.

c) Falls A symmetrisch ist, sind die Eigenvektoren v_i orthogonal. Hiermit lässt sich zeigen, dass sogar gilt

$$\left|\lambda^{(k)} - \lambda_1\right| = \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}\right)$$

Die Nachteile sind allerdings

1) Es kann nur der betragsmäßig größte Eigenwert mit Eigenvektor erhalten werden.

2) Falls $|\lambda_1| \approx |\lambda_2|$, ist die Konvergenzgeschwindigkeit langsam.

8.3 Inverse Vektoriteration

Sei A im Folgenden invertierbar mit Eigenwerten λ_i. Sei v Eigenvektor zu A mit Eigenwert λ und sei $\widetilde{\lambda} \in \mathbb{R}$ beliebig. Dann gilt

$$(A - \widetilde{\lambda}I)v = (\lambda - \widetilde{\lambda})v \quad \Leftrightarrow \quad (A - \widetilde{\lambda}I)^{-1}v = \frac{1}{\lambda - \widetilde{\lambda}}v$$

Falls nun $\widetilde{\lambda}$ ein guter Schätzwert von λ_i ist, d.h.

$$|\widetilde{\lambda} - \lambda_i| < |\widetilde{\lambda} - \lambda_j| \quad \forall j \neq i$$

dann ist $\frac{1}{\lambda_i - \widetilde{\lambda}}$ der maximale Eigenwert von $(A - \widetilde{\lambda}I)^{-1}$ mit Eigenvektor v. 21.01.2015

Die direkte Potenzmethode angewandt auf $(A - \widetilde{\lambda}I)^{-1}$ liefert dann $\overline{\lambda} = \frac{1}{\lambda_i - \widetilde{\lambda}}$ und somit $\lambda_i = \frac{1}{\overline{\lambda}} + \widetilde{\lambda}$ und einen zugehörigen Eigenvektor.

8.3.1 Inverse Vektoriteration mit Spektralverschiebung

Hier wird $\widetilde{x}^{(k+1)}$ anders als in 8.2.3 schrittweise berechnet durch

$$(A - \widetilde{\lambda}I)\widetilde{x}^{(k+1)} = v^{(k)} \tag{8.3.1}$$

Der zugehörige Algorithmus ergibt sich als

Wähle $x^{(0)}$
$v^{(0)} := \frac{x^{(0)}}{\|x^{(0)}\|_2}$
for $k = 0, 1, 2, \cdots$
| löse $(A - \widetilde{\lambda} I)\widetilde{x}^{(k+1)} = v^{(k)}$
| $\lambda^{(k)} = \frac{1}{(v^{(k)})^T \widetilde{x}^{(k+1)}} + \widetilde{\lambda}$
| $v^{(k+1)} = \widetilde{x}^{(k+1)} / \|\widetilde{x}^{(k+1)}\|_2$
end

In 8.2.3 ist nur eine Matrix-Vektor Multiplikation für $\widetilde{x}$ nötig. In 8.3.1 muss dagegen je Iteration ein lineares Gleichungssystem gelöst werden! Wird einmal eine LR- oder QR-Zerlegung von $A - \widetilde{\lambda} I$ berechnet (Aufwand $\mathcal{O}(n^3)$), erfordert (8.3.1) nur noch das Lösen (bzw. Vorwärts- und Rückwärtssubstitution) für verschiedene rechte Seiten (Aufwand $\mathcal{O}(n^2)$). Für eine gute Schätzung $\widetilde{\lambda}$ können z.B. die Gerschgorinkreise benutzt werden. Die Konvergenzgeschwindigkeit hängt ab von $\frac{|\lambda_i - \widetilde{\lambda}|}{\min_{j \neq i} |\lambda_j - \widetilde{\lambda}|}$.

Bemerkung 8.3.2. [siehe DH08] $A - \widetilde{\lambda} I$ ist für $\widetilde{\lambda} \approx \lambda$ schlecht konditioniert, und somit auch das Problem der Bestimmung von $\widetilde{x}^{(k)}$. Jedoch bleibt die Bestimmung von $v^{(k)}$ weiterhin gut konditioniert.

9 Lineare Gleichungssysteme: Iterative Methoden

9.1 Einführung

Beispiel 9.1.1 (Approximation der Poisson-Gleichung)**.** Betrachte

$$\Delta u = u_{xx} - u_{yy} =: f \tag{9.1.1}$$

in $\Omega = (0,1)^2$ mit $u = 0$ auf $\partial\Omega$. Es gibt N^2 Unbekannte, $N^2 \times N^2$. Also $A \in \mathbb{R}^{n\times n}$ mit $n = N^2$.

$$\begin{aligned}&(x_0,y_0),(x_1,y_0),(x_2,y_0),\dots,(x_{N+1},y_0),(x_0,y_1),(x_1,y_1),\dots,(x_{N+1},y_{N+1})\\ &\Rightarrow 0\dots0\ \underbrace{(-1)\ 0\dots0\ (-1)}_{N}\ 4\ \underbrace{(-1)\ 0\dots0\ (-1)}_{N}\ 0\dots0 \qquad i\text{-te Zeile von } A\end{aligned}$$

Damit sind nur maximal 5 Elemente verschieden von Null je Zeile und Au benötigt ca. $5n = 5N^2$ Multiplikationen (statt $n^2 = N^4$).

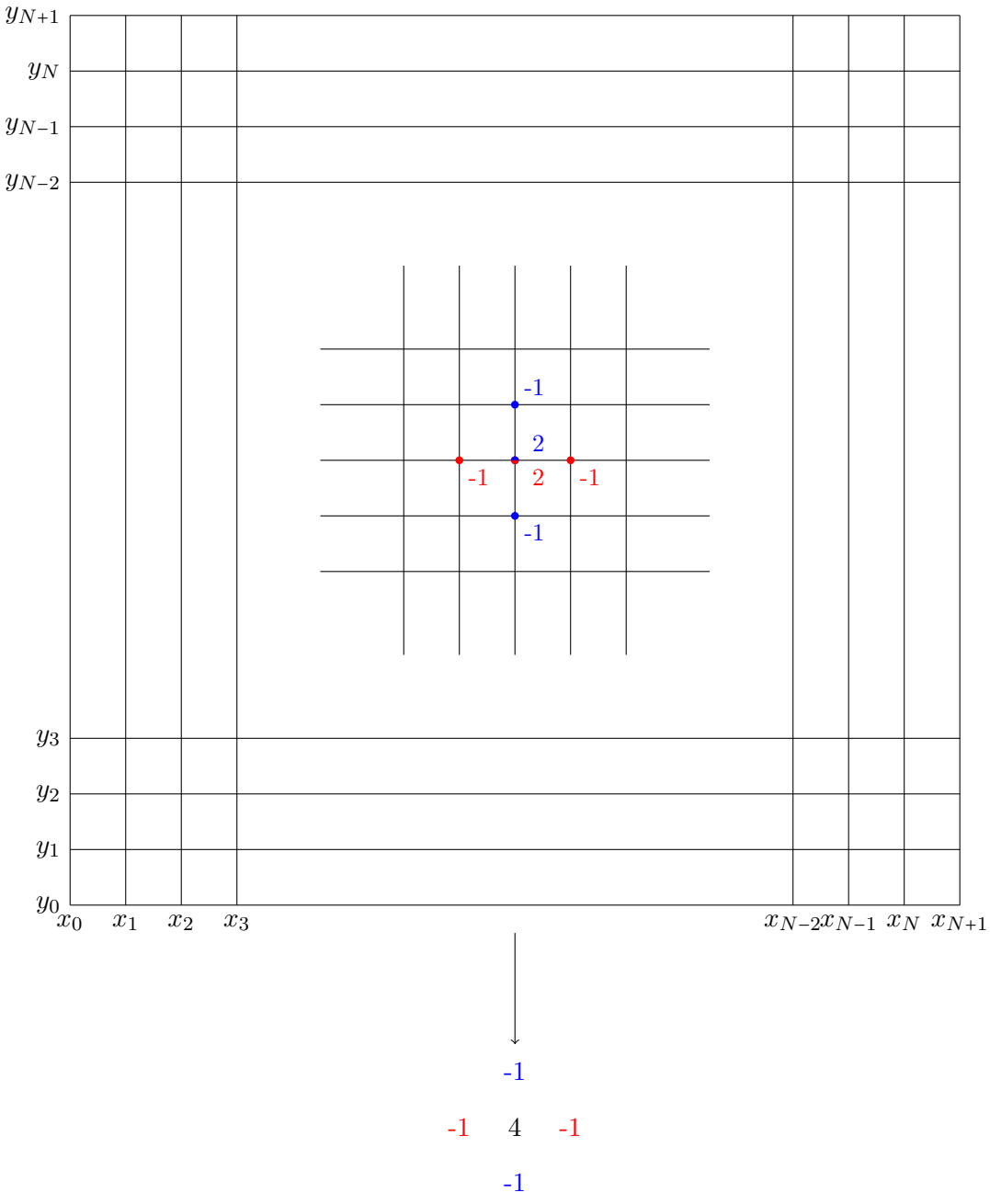
$$A = \begin{pmatrix} A_{11} & -I & & 0\\ -I & \ddots & \ddots & \\ & & & -I\\ 0 & & -I & A_{nn}\end{pmatrix} \quad \text{mit } A_{ii} = \begin{pmatrix} 4 & -1 & & \\ -1 & \ddots & \ddots & \\ & & & -1\\ & & -1 & 4\end{pmatrix} \in \mathbb{R}^{n\times n} \tag{9.1.2}$$

9.1.2 Typische Aufgabenstellung

Zu lösen ist $Ax = b$ mit

1. $A \in \mathbb{R}^{n\times n}$ mit sehr großem n,
2. die meisten Komponenten a_{ij} sind 0, d.h. A ist dünn besetzt (sparse, $\mathcal{O}(n)$ Nicht-Null-Einträge),
3. A hat eine Struktur wie z.B. die Blockstruktur in (9.1.2),
4. A ist das Resultat einer Diskretisierung, d.h. x approximiert eine kontinuierliche Lösung,
5. Die Eingabedaten sind zu b verarbeitet worden, d.h. i.A. steht b nicht „exakt“ zur Verfügung, sondern ist eine Näherung.

Abb. 9.1.0 *Gitter mit Unbekannten zu Beispiel 9.1.1*



9.1.3 Konsequenzen

1.: Die Komplexität $\mathcal{O}(n^3)$ der direkten Verfahren für vollbesetzte Matrizen ist zu groß.

2. und 3. können i.d.R. nicht effizient durch direkte Methoden ausgenutzt werden oder die Struktur bleibt nicht erhalten.

Wegen 4. und 5. ist die Lösung x nicht exakt erforderlich, sondern deren Genauigkeit sollte im Rahmen des Diskretisierungsfehlers oder der Eingabefehler liegen.

Bemerkung 9.1.4. Ziel einer Iterationsvorschrift

$$x^{(k+1)} = \Phi(x^{(0)}, \ldots, x^{(k)})$$

ist, dass

a) die Folge $\{x^{(k)}\}$ möglichst schnell gegen x konvergiert und

b) $x^{(k+1)}$ mit möglichst geringem Aufwand berechnet wird.

Typischerweise wird ein Aufwand von nicht wesentlich mehr als einer Realisierung einer Matrix-Vektor-Multiplikation Ay pro Iteration erwünscht. Für dünnbesetztes A entspricht dies pro Iteration einem Aufwand von $\mathcal{O}(n)$.

Anstelle von $\mathcal{O}(n^3)$ bei einem direkten „black-box Löser“ ergibt sich damit für k Iterationen ein Aufwand von $k \cdot \mathcal{O}(n)$ flops.

9.2 Klassische (stationäre) Verfahren

Betrachte

$$Ax = b \tag{9.2.1}$$

Wähle nun für A eine **Aufspaltung** $A = M - N$ mit einer einfach invertierbaren Matrix M, s.d.

$$Mx = Nx + b \tag{9.2.2}$$

Die **Iterationsvorschrift** ist dann: Löse zu einem gegebenen Startwert $x^{(0)}$ als Schätzung für x für $k = 0, 1, 2, \ldots$

$$Mx^{(k+1)} = Nx^{(k)} + b \tag{9.2.3}$$

Es ergeben sich folgende Äquivalenzen

$$(9.2.3) \Leftrightarrow x^{(k+1)} = M^{-1}Nx^{(k)} + M^{-1}b \tag{9.2.4}$$

$$\Leftrightarrow x^{(k+1)} = x^{(k)} + M^{-1}r^{(k)} \tag{9.2.5}$$

mit $r^{(k)} = b - Ax^{(k)}$ (Residuum im k-ten Schritt).

Als **Abbruchkriterium**

a) sollte immer eine maximale Iterationszahl angegeben werden und

b) wird oft für eine gegebene Toleranz tol $\geq \left\|r^{(k)}\right\|$ oder wie diskutiert tol $\cdot \|b\| \geq \left\|r^{(k)}\right\|$ verwendet.

Nach (9.2.4) ergibt sich also die **Fixpunktiteration**

$$x^{(k+1)} = Gx^{(k)} + d =: g(x^{(k)}) \tag{9.2.6}$$

zur **Fixpunktgleichung** $x^* = Gx^* + d = g(x^*)$ mit der **Iterationsmatrix**

$$G = M^{-1}N = I - M^{-1}A = g'(x) \tag{9.2.7}$$

und $d = M^{-1}b$. Weiterhin folgt

$$x^{(k+1)} - x^* = G(x^{(k)} - x^*) = G^{k+1}(x^{(0)} - x^*)$$

und hiermit gilt elementweise (falls $x^{(0)} - x^* \notin \ker(G^{k+1})$)

$$\lim_{k\to\infty} x^{(k)} = x^* \quad \Leftrightarrow \quad \lim_{k\to\infty} G^k = 0$$

Satz 9.2.1 (Konvergenzkriterien)**.** *Sei $G \in \mathbb{R}^{n\times n}$. Dann sind folgende Aussagen äquivalent:*

i) Die Iteration (9.2.6) *konvergiert für jeden Startwert $x^{(0)} \in \mathbb{R}^n$*

ii) Es gilt $\lim_{k\to\infty} G^k = 0$

iii) Für den Spektralradius $\rho(G) := \max_i |\lambda_i|$ mit $\lambda_i \in \mathbb{C}$ Eigenwert zu G gilt

$$\rho(G) < 1 \tag{9.2.8}$$

Beweis. **i)⇔ ii):** siehe oben.

ii)⇒ iii): Ist $\rho(G) \geq 1$ dann existiert ein Eigenwert $\lambda \in \mathbb{C}$ mit $|\lambda| > 1$ und Eigenvektor $v \neq 0$ zu G. Damit ist $G^k v = \lambda^k v$. Mit $\lim_{k\to\infty} \lambda^k \neq 0$ (falls existent) folgt $\lim_{k\to\infty} G^k \neq 0$.

iii)⇒ ii): Sei $\rho(G) < 1$. Da $\left(TGT^{-1}\right)^k = TG^kT^{-1}$ für eine invertierbare Matrix T, reicht es, für die Jordansche Normalform $J = TGT^{-1}$ von G zu zeigen, dass $\lim_{k\to\infty} J^k = 0$ gilt. Da $J^k = \begin{pmatrix} J_1^k & & 0 \\ & \ddots & \\ 0 & & J_m^k \end{pmatrix}$ mit den Jordankästchen $J_i = \begin{pmatrix} \lambda_i & 1 & & 0 \\ & \ddots & \ddots & \\ & & & 1 \\ 0 & & & \lambda_i \end{pmatrix} =: \lambda_i I + S_i$

mit $S_i = \begin{pmatrix} 0 & 1 & & 0 \\ & \ddots & \ddots & \\ & & & 1 \\ 0 & & & 0 \end{pmatrix}$ ist, ist nur $\lim_{k\to\infty} J_i^k$ zu untersuchen.

$$\begin{aligned} J_i^k &= (\lambda_i I + S_i)^k \\ &= \sum_{l=0}^{k} \binom{k}{l} \lambda_i^{k-l} S_i^l \\ &= \sum_{l=0}^{n-1} \binom{k}{l} \lambda_i^{k-l} S_i^k \end{aligned}$$

für $k \geq n$, da für $S_i \in \mathbb{R}^{r\times r}$ gilt $S_i^r = 0$ und mit $r \leq k$ ist dann $\lim_{k\to\infty} S_i^k = 0$. Weiterhin gilt $\binom{k}{l} \leq k^l$ und wegen $|\lambda_i| < 1$

$$\left|\binom{k}{l}\right| \cdot \left|\lambda_i^{k-l}\right| \leq \left|\lambda_i^k\right| \cdot \left|\frac{k}{\lambda_i}\right|^l \leq |\lambda_i|^k \cdot \left|\frac{k}{\lambda_i}\right|^n \overset{k\to\infty}{\longrightarrow} 0$$

Also $J_i^k \to 0$ für $k \to \infty$.

□

Lemma 9.2.2. *Für jede von einer Vektornorm induzierten Matrixnorm $\|\bullet\|$ gilt*

$$\rho(G) \leq \|G\| \tag{9.2.9}$$

Beweis. Sei λ Eigenwert von G zum Eigenvektor v. Dann gilt $\frac{\|Gv\|}{\|v\|} = |\lambda|$ und daraus folgt direkt

$$\|G\| \geq \sup_{v\in V} \frac{\|Gv\|}{\|v\|} \geq |\lambda| \quad \forall \lambda$$

□

$\|G\| < 1$ ist meist leichter zu prüfen als $\rho(G) < 1$. Desweiteren gilt die Fehlerabschätzung

$$\left\|x^{(k)} - x^*\right\| \leq \|G\|^k \left\|x^{(0)} - x^*\right\| \tag{9.2.10}$$

für submultiplikative Matrixnormen. 24.01.2015

9.3 Gesamt- und Einzelschritt-Verfahren

In (9.2.3) soll M leicht „invertierbar“ sein.

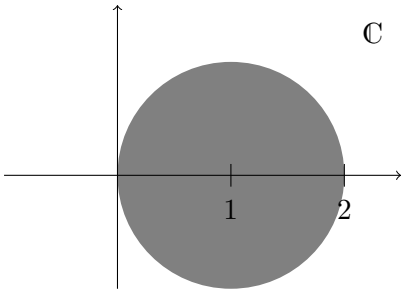
9.3.1 Richardson-Verfahren

Es sei $M = I$, also $G = I - A$ und $x^{(k+1)} = x^{(k)} + \left(b - Ax^{(k)}\right) = x^{(k)} + r^{(k)}$ konvergiert für alle Startwerte genau dann, wenn

$$\rho(G) = \max_i |1 - \lambda_i| < 1 \tag{9.3.1}$$

mit den Eigenwerten λ_i zu A. Es muss also gelten $\rho(A) < 2$, was selten der Fall ist.

Abb. 9.3.0 *Mögliche Werte der λ_i für $M = I$*



9.3.2 Gesamtschritt- oder Jacobi-Verfahren

Es sei $M_G \coloneqq D$ die Diagonale von A.

Vorsicht: Falls $a_{ii} = 0$ für ein i, permutiere die Zeilen oder Spalten. Für ein reguläres A gibt es eine Permutation, so dass D anschließend invertierbar ist.

Es gilt die Rekursionsvorschrift

$$x^{(k+1)} = x^{(k)} + D^{-1}r^{(k)} \tag{9.3.2}$$

Oft wird die Schreibweise mit $A = L + D + R =$  benutzt

$$Dx^{(k+1)} = -(L + R)x^{(k)} + b \tag{9.3.3}$$

bzw. für $i = 1, \ldots, n$

$$a_{ii}x_i^{(k+1)} = -\sum_{j \neq i} a_{ij}x_j^{(k)} + b_i \tag{9.3.4}$$

Die Iterationsmatrix ist $G_G \coloneqq -D^{-1}(L + R)$.

9.3.3 Einzelschritt- oder Gauß-Seidel-Verfahren

Falls in (9.3.4) die Schleife von $i = 1$ bis $i = n$ sukzessive durchlaufen wird, stehen bei der Berechnung von $x_i^{(k+1)}$ die Werte $x_1^{(k+1)}, \dots, x_{i-1}^{(k+1)}$ bereits zur Verfügung und können genutzt werden. Also gilt für $i = 1, \dots, n$

$$a_{ii}x_i^{(k+1)} = -\sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=1+1}^{n} a_{ij}x_j^{(k)} + b_i \tag{9.3.5}$$

$$\iff \quad Dx^{(k+1)} = -Lx^{(k+1)} - Rx^{(k)} + b$$

$$\iff \quad (D+L)x^{(k+1)} = -Rx^{(k)} + b \tag{9.3.6}$$

$$\iff \quad x^{(k+1)} = x^{(k)} + (D+L)^{-1}r^{(k)}$$

Daraus folgt, dass $M_G = D + L$, $N = -R$ und die zugehörige Iterationsmatrix ist

$$G_E = -(D+L)^{-1}R = -(D+L)^{-1}A + I$$

Beispiel 9.3.4. Ohne zusätzliche Voraussetzungen an A lässt sich nicht voraussagen, welches der beiden Verfahren konvergiert.

$$A = \begin{pmatrix} 1 & -2 & 2 \\ -1 & 1 & -1 \\ -2 & -2 & 1 \end{pmatrix} \qquad \Longrightarrow \rho(G_G) = 0, \qquad \rho(G_E) = 2(1+\sqrt{2})$$

$$A = \frac{1}{2}\begin{pmatrix} 2 & 1 & 1 \\ -2 & 2 & -2 \\ -1 & 1 & 2 \end{pmatrix} \qquad \Longrightarrow \rho(G_G) = \frac{1}{2}\sqrt{5}, \qquad \rho(G_E) = \frac{1}{2}$$

Definition 9.3.5.

a) $A \in \mathbb{R}^{n\times n}$ erfüllt die **starke Zeilensummenbedingung** , bzw. $A \in \mathbb{R}^{n\times n}$ ist strikt diagonaldominant, wenn

$$|a_{ii}| > \sum_{j\neq i} |a_{ij}| \quad 1 \le i \le m \tag{9.3.7}$$

b) A erfüllt die **schwache Zeilensummenbedingung** , bzw. $A \in \mathbb{R}^{n\times n}$ ist schwach diagonaldominant, wenn

$$|a_{ii}| \ge \sum_{j\neq i} |a_{ij}| \quad 1 \le i \le m \tag{9.3.8}$$

und für zumindest ein i die starke Bedingung erfüllt ist.

Definition 9.3.6. $A \in \mathbb{R}^{n\times n}$ heißt **zerfallend**, wenn es $J \subsetneq \{1, \dots, n\}$ gibt, so dass $a_{ij} = 0$ für $i \in J, j \notin J$.

Bemerkung 9.3.7. Falls A zerfallend ist, gibt es eine Permutationsmatrix P, so dass

$$P^T A P = \begin{pmatrix} \widetilde{A}_{11} & 0 \\ \widetilde{A}_{21} & \widetilde{A}_{22} \end{pmatrix}$$

mit $\widetilde{A}_{11} \in \mathbb{R}^{p\times p}$, $\widetilde{A}_{22} \in \mathbb{R}^{q\times q}$, $p, q > 0$, $p + q = m$. Das Gleichungssystem $Ax = b$ zerfällt somit in zwei kleinere

$$\begin{aligned} \widetilde{A}_{11}\widetilde{x}_1 &= \widetilde{b}_1 \\ \widetilde{A}_{22}\widetilde{x}_2 &= \widetilde{b}_2 - \widetilde{A}_{21}\widetilde{x}_1 \end{aligned}$$

Beispiel 9.3.8. Die Matrix A in Beispiel 9.1.1 zur Approximation der Poisson-Gleichung ist schwach und nicht stark diagonal dominant und ist nicht zerfallend.

Satz 9.3.9 (Zeilensummenkriterium)**.** *Das Jacobi- sowie Gauß-Seidel-Verfahren konvergiert, falls*

a) A die starke Zeilensummenbedingung erfüllt.

b) A die schwache Zeilensummenbedingung erfüllt und nicht zerfallend ist.

In beiden Fällen gilt

$$\|G_E\|_\infty \le \|G_G\|_\infty = \max_i \frac{1}{|a_{ii}|} \sum_{j\neq i} |a_{ij}| =: \overline{a} \le 1 \tag{9.3.9}$$

Beweis. Es ist $G_G := D^{-1}(L + R) = D^{-1}(A - D)$ und damit $\overline{a} = \|G_G\|_\infty \le 1$ aufgrund der Diagonaldominanz. Für das Einzelschrittverfahren betrachte $y := G_E x = -(D + L)^{-1}Rx$, wofür gilt $Dy = -Ly - Rx$. Dann

$$\begin{aligned} |(G_E x)_1| = |y_1| &\le \frac{1}{|a_{11}|}\left(\sum_{j=2}^{n} |a_{1j}| \cdot |x_j|\right) \\ &\le \|x\|_\infty \frac{1}{|a_{11}|}\left(\sum_{j=2}^{n} |a_{1j}|\right) \\ &\le \|x\|_\infty \overline{a} \\ &\le \|x\|_\infty \end{aligned}$$

Rekursiv folgt aus $|y_j| \le \|x\|_\infty$ für $j \le i - 1$

$$\begin{aligned} |(G_E x)_i| = |y_i| &\le \frac{1}{|a_{ii}|}\left(\sum_{j=1}^{i-1} |a_{ij}| \cdot \underbrace{|y_j|}_{\le \|x\|_\infty} + \sum_{j=i+1}^{n} |a_{ij}| \cdot \underbrace{|x_j|}_{\le \|x\|_\infty}\right) \\ &\le \|x\|_\infty \frac{1}{|a_{ii}|}\left(\sum_{j=1}^{i-1} |a_{ij}|\right) \\ &= \|x\|_\infty \overline{a} \\ &\le \|x\|_\infty \end{aligned} \tag{9.3.10}$$

Woraus $\|G_E\|_\infty \leq \overline{a} = \|G_G\|_\infty$ folgt.

Zu a): Aus der starken Zeilensummenbedingung folgt $\overline{a} < 1$ und

$$\rho(G) \leq \|G\|_\infty \leq \overline{a} < 1$$

und mit Satz 9.2.1 dann die Behauptung.

Zu b): Die schwache Zeilensummenbedingung ergibt $\rho(G) \leq 1$ für beide Verfahren. Es bleibt zu zeigen, dass kein Eigenwert λ mit $|\lambda| = 1$ existiert. Angenommen G hat einen Eigenwert λ mit $|\lambda| = 1$. Sei v ein zugehöriger Eigenvektor mit $\|v\|_\infty = 1$. Definiere $N_1 \coloneqq \{l \in \mathbb{N} \mid |v_l| = 1\}$ und $N_2 \coloneqq \{k \mid |v_k| < 1\}$. Aus der Definition von G_G bzw. der Ungleichung (9.3.10) für G_E folgt mit $|(Gv)_i| = |\lambda v_i| = |v_i|$

$$\begin{aligned} |v_i| = |(Gv)_i| &\leq \frac{1}{|a_{ii}|}\left(\sum_{j\neq i} |a_{ij}| \cdot |v_j|\right) \\ &= \frac{1}{|a_{ii}|}\left(\sum_{\substack{j\in N_1 \\ j\neq i}} |a_{ij}| + \sum_{\substack{j\in N_2 \\ j\neq i}} |a_{ij}| \cdot \underbrace{|v_j|}_{<1}\right) \\ &< 1 \end{aligned}$$

Falls für ein i ein $j \in N_2$ mit $i \neq j$ und $a_{ij} \neq 0$ existiert, folgt

$$|v_i| < \frac{1}{|a_{ii}|}\left(\sum_{j\neq i} |a_{ij}|\right) \leq \overline{a} \leq 1$$

Da $|v_i| = 1$ für $i \in N_1$ muss $a_{ij} = 0$ gelten für alle $j \in N_2$ also $a_{ij} = 0$ für $i \in N_1$, $j \in N_2$, was ist ein Widerspruch zum nicht zerfallenden A ist. □

28.01.2015

Satz 9.3.10. *Das Gauß-Seidel-Verfahren konvergiert für jede symmetrische, positiv definite (spd) Matrix.*

Beweis. Sei A spd, dann ist $\langle x, x\rangle \coloneqq x^T Ax$ ein Skalarprodukt. Dieses Skalarprodukt $\langle \bullet, \bullet\rangle$ induziert eine Norm $\|\bullet\|$ auf dem $\mathbb{R}^N$. Ist $\|\bullet\|$ auf dem $\mathbb{R}^{N\times N}$ die zugeordnete Matrixnorm, so folgt $\rho(G) \leq \|G\|$.

Zeige nun $\|Gx\|^2 \leq c^2 \|x\|^2$ für $x \neq 0$ mit $0 \leq c < 1$, dann gilt $\|G\| \leq c < 1$ und das Verfahren

konvergiert.

$$\begin{aligned}
\|x\|^2 - \|G_E x\|^2 &= \langle x, x\rangle - \langle Gx, Gx\rangle \\
&= x^T Ax - x^T G^T AGx \\
&= x^T (A - G^T AG)x \\
&\overset{\text{s.u.}}{=} \left(x^T A^T (D+L)^{-T} D^{\frac{1}{2}}\right) \cdot \left(D^{\frac{1}{2}} (D+L)^{-1} Ax\right) \\
&= \left\| D^{\frac{1}{2}} (D+L)^{-1} Ax \right\|_2^2 \\
&\geq \underbrace{\left\| \left[D^{\frac{1}{2}} (D+L)^{-1} A \right]^{-1} \right\|_2^{-2}}_{=:k>0} \cdot \|x\|_2^2 && \text{da } \|Bx\| \geq \frac{1}{\|B^{-1}\|} \|x\| \\
&\geq kd^2 \|x\|^2 && \text{da alle Normen auf } \mathbb{R}^N \text{ äquivalent} \\
&> 0
\end{aligned}$$

Damit gilt zusammengefasst

$$\underbrace{(1 - kd^2)}_{c<1} \|x\|^2 \geq \|G_E x\|^2$$

$$\begin{aligned}
A - G_E^T AG_E &= A - \left[(D+L)^{-1}A - I\right]^T A \left[(D+L)^{-1}A - I\right] \\
&= A \cdot \left((D+L)^{-T} + (D+L)^{-1} - (D+L)^{-T} A (D+L)^{-1}\right) \cdot A \\
&= A \cdot \big((D+L)^{-T}(D+L)(D+L)^{-1} \\
&\qquad + (D+L)^{-T}(D+L)^T (D+L)^{-1} \\
&\qquad - (D+L)^{-T}(D+L+L^T)(D+L)^{-1}\big) \cdot A \\
&= A(D+L)^{-T} \cdot \big(D(D+L)^{-1} + L(D+L)^{-1} \\
&\qquad + D(D+L)^{-1} + L^T (D+L)^{-1} \\
&\qquad - D(D+L)^{-1} - L(D+L)^{-1} - L^T (D+L)^{-1}\big) \cdot A \\
&= A(D+L)^{-T} D (D+L)^{-1} A \\
&= \left(D^{\frac{1}{2}} (D+L)^{-1} A^T\right)^T \left(D^{\frac{1}{2}} (D+L)^{-1} A\right)
\end{aligned}$$

und $D^{\frac{1}{2}}$ existiert, da D mit A spd nur positive Einträge hat. □

9.4 Relaxationsverfahren

Sei $\widetilde{x}_i^{(k+1)}$ eine Hilfsgröße gegeben durch einen Schritt des Einzelschritt-Verfahrens, d.h.

$$a_{ii} \widetilde{x}_i^{(k+1)} = -\sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} + b_i \tag{9.4.1}$$

Dann definiere

$$x_i^{(k+1)} := (1-w)x_i^{(k)} + w\widetilde{x}_i^{(k+1)} \tag{9.4.2}$$

mit einem sogenannten **Relaxationsparameter** w. Es heißt

Überrelaxation falls $w > 1$,

Unterrelaxation falls $w < 1$.

Das **SOR-(successive overrelaxation)-Verfahren** lautet:

$$a_{ii}x_i^{(k+1)} = w\left(-\sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)} + b_i\right) - (w-1)a_{ii}x_i^{(k)} \quad \text{für } i = 1,\dots,n \tag{9.4.3}$$

bzw.

$$\begin{aligned} Dx^{(k+1)} &= w(-Lx^{(k+1)} - Rx^{(k)} + b) - (x-1)Dx^{(k)} \\ \Leftrightarrow (D+wL)x^{(k+1)} &= -\left[w(R+D) - D\right]x^{(k)} + wb \end{aligned}$$

Also $M_{SOR} = D + wL$ und $G_{SOR} = -(D+wL)^{-1}(w(D+R) - D)$ Für spezielle Fälle kann ein optimales w berechnet werden. Im Allgemeinen jedoch nicht oder es ist zu aufwändig.

Satz 9.4.1 (Konvergenz des SOR-Verfahrens)**.**

a) Für beliebige Matrizen kann das SOR-Verfahren nur konvergieren, falls 0<w<2 gilt.

b) Für spd Matrizen konvergiert das Verfahren genau dann, wenn 0<w<2 gilt.

Beweis.

a) [SB90]

b) (Satz von Ostrowski, Reich)

□

Die Relaxationsidee ist allgemein anwendbar

$$\begin{aligned} x^{(k+1)} &= w(Gx^{(k)} + d) + (1-w)x^{(k)} \\ &= G_w x^{(k)} + wd \end{aligned} \tag{9.4.4}$$

mit der Iterationsmatrix

$$G_w = wG + (1-w)I \tag{9.4.5}$$

Die Kunst besteht nun darin, w so zu wählen, dass $\rho(G_w)$ möglichst klein ist. Der Aufwand pro Iteration bleibt fast identisch zum Originalverfahren.

Wird w optimal gewählt, so konvergiert das gedämpfte Richardson-Verfahren

Heutige wichtigste Anwendungsgebiete sind die sogenannten Mehrgitterverfahren (Multigrid methods).

9.4.2 Symmetrisches SOR-Verfahren (SSOR)

Läuft die Schleife in (9.3.4) rückwärts, d.h. $i = n, \dots, 1$, kann entsprechend ein Einzelschritt-Verfahren genutzt werden für $i = n, n-1, \dots, 1$

$$\begin{aligned} a_{ii}x_i^{(k+1)} &= -\sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k+1)} + b_i \\ \Leftrightarrow (D+R)x^{(k+1)} &= -Lx^{(k)} + b \end{aligned}$$

Eine Iteration des **SSOR-Verfahrens** ist gegeben durch eine relaxierte Vorwärts-Einzelschritt-Iteration und eine relaxierte Rückwärts-Einzelschritt-Iteration, d.h.

$$\begin{aligned} Dx^{(k+\frac{1}{2})} &= w(-Lx^{(k+\frac{1}{2})} - Rx^{(k)} + b) - (w-1)Dx^{(k)} \\ Dx^{(k+1)} &= w(-Lx^{(k+\frac{1}{2})} - Rx^{(k+1)} + b) - (w-1)Dx^{(k+\frac{1}{2})} \end{aligned} \tag{9.4.6}$$

Vorteile von SSOR gegenüber SOR

a) k SSOR-Schritte „entsprechen“ $2k$ SOR-Schritten, können aber mit einem Aufwand umgesetzt werden, der $(k + \frac{1}{2})$ SOR-Schritten entspricht.

b) Falls A spd ist, so ist dies auch

$$M_{SSOR}^{-1} = w(1-w)(D+wR)^{-1}D(D+wL)^{-1} \tag{9.4.7}$$

Beim SOR-Verfahren geht die Symmetrie für M_{SOR} verloren, diese wird jedoch oft benötigt.

Index

Literatur

[AS64] M. Abramowitz und I. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. 1964.

[Boo01] Carl de Boor. *A Practical Guide to Splines*. Applied Mathematical Sciences. Springer New York, 2001. ISBN: 9780387953663.

[DH08] Peter Deuflhard 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.

[DR08] Wolgang Dahmen und Arnold Reusken. *Numerik für Ingenieure und Naturwissenschaftler*. 2. Aufl. Springer-Lehrbuch. Springer-Verlag, Berlin, 2008. ISBN: 978-3-540-76493-9.

[FH07] 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=2aYfBAAAQBAJ.

[GO96] 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.

[HH94] 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.

[Pre+02] William H. Press u. a. *Numerical recipes in C++*. The art of scientific computing, Second edition, updated for C++. Cambridge University Press, Cambridge, 2002. ISBN: 0-521-75033-4.

[SB90] 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.

[Ste65] J.F. Steffensen. *Interpolation*. 1. Aufl. New York, Chelsea, 1965.

[Sto] J. Stoer. *Numerische Mathematik 1*. Springer.

[SW05] Robert Schaback und Holger Wendland. *Numerische Mathematik*. Springer-Lehrbuch. Springer-Verlag Berlin Heidelberg, 2005. ISBN: 9783540267058. URL: https://books.google.de/books?id=wdgmBAAAQBAJ.