Skript Numerik I

von Prof. Dr. Luise Blank im WS14/15

Gesina Schwalbe

6. November 2014

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

oder auch an **gesina.schwalbe@googlemail.com**

Copyright

Bilder oder „IMAGE MISSING“

Leider habe ich mich bisher noch nicht in die (ordentliche) Grafikerstellung in LaTeX-Dokumenten eingearbeitet, weshalb das Skript erstmal nicht mit Grafiken dienen kann – die Fehlstellen sind mit „IMAGE MISSING“ markiert.
Wenn ihr gerne Veranschaulichungen haben möchtet, könnt ihr jederzeit **die entsprechenden Bilder an mich schicken**, bitte mit Kapitelangabe. (Oder mich explizit darum bitten, dass ich mich übergangsweise um das Einscannen, Bearbeiten etc. kümmere). Dann werden sie an die entsprechenden Stellen eingebunden.

Erscheinungsdatum

Ich werde mich bemühen, das Skript jeweils am Vorlesungstag zumindest in unverbesserter Form online zu stellen, so dass v.a. diejenigen, die die Vorlesung nicht besuchen können, einen Überblick über den Stoff bekommen.

Innerhalb einer Woche sollte das Skript aktuell sein.

Anfangsteil

Nachdem ich nicht seit Semesterbeginn mittexe, fehlt noch ein Großteil des ersten Kapitels.
Ich hoffe, es bis Mitte des Semesters nachholen zu können, aber keine Garantie.

06.10.2014 Ziel ist, dass es vor der Vorlesungszeit integriert ist.

1 Einführung

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. Bemerkung:

- Ein verlässlicher Lösungsalgorithmus überprüft dies und behandelt alle Fälle.
- Die Cramersche Regel ist ineffizient (s. Einführung).
- 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:

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

$$\Downarrow$$

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

$$\Downarrow$$

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

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

mit

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

08.10.2014

In jedem Schritt werden die Einträge der k-ten Spalte analog unterhalb der Diagonalen (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:

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

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

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

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
```

2.1.3 Vorsicht

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.
IMAGE MISSING
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.
IMAGE MISSING

2.1.5 Dreieckszerlegung

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

2.1.6 Vorwärtssubstitution

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 Gauß-Elemination zur Lösung von $Ax = b$

1 Dreieckszerlegung

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

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

2.1.8 Rechenaufwand gezählt in „flops“

„flops“ = **f**loating **p**oint **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

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

   Divisionen.
   Damit ergibt sich eine Gesamtanzahl an flops von

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

   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.

2.1.9 Definition: Landau-Symbole

MISSING

2.1.10 Allgemeines zur Aufwandsbetrachtung

13.10.2014 MISSING

2.1.11 Formalisieren des Gauß-Algorithmus

MISSING

2.1.12 Lemma (Eigenschaften der L_k-Matrizen)

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:

$$\begin{aligned} L &= L_1^{-1} \cdot \ldots \cdot L_{n-1}^{-1} \\ &= I + \sum_{i=1}^{n-1} l_i e_i^T \\ &= \begin{pmatrix} 1 & & 0 \\ & \ddots & \\ l_{ij} & & 1 \end{pmatrix} \\ &= I + \widetilde{L} \end{aligned} \tag{2.1.9}$$

Hiermit folgt:

2.1.13 Satz (LR- oder LU-Zerlegung)

Das obige Verfahren ((2.1.2) und (2.1.3)) erzeugt unter der Voraussetzung von nicht-nullwertigen Pivotelementen eine Faktorisierung

$$A = L \cdot R \qquad\qquad IMAGE\ MISSING$$

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
MISSING

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 (=partielle/ halbmaximale Pivotisierung)

Im k-ten Eliminationsschritt ist

IMAGE MISSING (s. Folien)

MISSING

2.2.2 Bemerkung

MISSING

2.2.3 Gauß-Elimination mit Spaltenpivotisierung

Der zugehörige Algorithmus zur Spaltenpivotisierung ist:

$\pi(1:n) = [1:n]$
for $k = 1, \ldots, n-1$
| bestimme Pivotzeile p, so dass
| $|a_{pk}| = \max\{|a_{jk}|, j = k, ..., n\}$
| $\pi(k) \leftrightarrows {}^{1}\pi(p)$
| $A(k, 1:n) \leftrightarrows A(p, 1:n)$
| **if** $a_{kk} \neq 0$
| | $zeile = [k+1:n]$
| | $A(zeile, k) = A(zeile, k)/a_{kk}$
| | $A(zeile, zeile) = A(zeile, zeile) - A(zeile, k)A(k, zeile)$
| **else**
| | „A ist singulär“
| **end**
end

$\leftrightarrows$ bedeutet „vertausche mit“

2.2.4 Satz: 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.

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

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

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

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

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

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

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

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

Damit folgt:

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

und

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

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 $Ax = b$

2.2.6 Bemerkungen

2.2.7 Beispiel zur Pivotisierung

15.10.2014

3 Fehleranalyse

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)

3.1.1 Definition: Gleitkommazahl

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 $e_{min} \leq e \leq e_{max}$ Exponent

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

3.1.2 Bemerkung

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 Beispiel

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: $b_0 = \cdots = b_7 = 1$ sowie $b_0 = \cdots = b_7 = 0$ sind bis auf Ausnahmen keine gültigen Exponenten
- $m = (a_1.a_2 \cdots a_{24}) = 1 + \sum_{\nu=2}^{24} a_\nu 2^{1-\nu}$ stellt die Mantisse dar, $a_1 = 1$ wird nicht abgespeichert.
- Besondere Zahlen per Konvention:

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

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

$x =$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)

Betragsmäßig **größte Zahl**:

| 0 | $01 \cdots 1$ | $1 \cdots\cdots 1$ |

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

Betragsmäßig **kleinste Zahl**:

| 0 | $0 \cdots 0$ | $0 \cdots\cdots 01$ |

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

3.1.4 Verteilung der Maschinenzahlen

ungleichmäßig im Dezimalsystem, z. B.

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

IMAGE MISSING

ist im Dualsystem gleichmäßig verteilt.

3.1.5 Bezeichnungen

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 $b = e + 127$

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

3.1.6 Rundungsfehler

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

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

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

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

Für die **relative Rechengenauigkeit** folgt somit

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

Die **Maschinengenauigkeit** des Rechners ist daher durch

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

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

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

Weiterhin gilt:

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

$$1 + eps$$

b) Eine Maschinenzahl x repräsentiert eine Eingabemenge

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

IMAGE MISSING

3.1.7 Bemerkung

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 $x - y$:

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

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.

3.2.1 Definition: Problem

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.

3.2.2 Definition: absoluter und relativer Fehler

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

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

b) $\frac{\|\widetilde{x}-x\|}{\|x\|}$ 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:

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

3.2.3 Wiederholung: Normen

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

IMAGE MISSING

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

IMAGE MISSING

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

IMAGE MISSING

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

3.2.4 Definition: Matrixnorm

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

$$\begin{aligned}\|A\|_{a,b} &:= \sup_{x\neq 0} \frac{\|Ax\|_b}{\|x\|_a} \\ &= \sup_{\|x\|_a=1} \|Ax\|_b\end{aligned} \tag{3.2.1}$$

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

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

3.2.5 Definition: Frobeniusnorm, p-Norm, Verträglichkeit

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

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

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

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

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

3.2.6 Bemerkungen

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

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

b) Die Norm $\|\cdot\|_{1,1}$ 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].

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

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

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

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

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

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

22.10.2014

3.2a) Normweise Konditionsanalyse

3.2.7 Definition: absolute und relative normweise Kondition

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

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

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

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

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

c) Sprechweise:

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

3.2.8 Lemma

Falls f differenzierbar ist, gilt

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

und für $f(x) \neq 0$

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

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

[4] Beweis: siehe Übungsblatt 3

3.2.9 Beispiel: Kondition der Addition

$f(x_1, x_2) \coloneqq x_1 + x_2,\ f : \mathbb{R}^2 \to \mathbb{R}$.
Wähle l_1-Norm auf $\mathbb{R}^2$ (und $\mathbb{R}$)

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

damit

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

Daraus folgt: Die Addition zweier Zahlen mit gleichem Vorzeichen ergibt

$$\kappa_{rel} = 1$$

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

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

Zum Beispiel in 3.1.8: Es ist

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

also

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

3.2.10 Beispiel: Lösen eines 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

$$f : b \mapsto x = A^{-1}b$$

Berechne dann $\kappa(f, b)$ und löse

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

wobei $\|\cdot\|$ 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. $(x \mapsto Ax \quad \kappa_{abs})$
Ebenso folgt die scharfe Abschätzung

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

Damit

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

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

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

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

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

$$\|\widehat{b}\| = \|A\| \cdot \|\widehat{x}\|$$

und somit

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

b) Betrachte die Störungen in A:
Löse also

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

Sei hierzu

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

und berechne $\kappa(f, A)$ mittels Ableitung $Df(A) : \mathbb{R}^{n\times n} \to \mathbb{R}^n$:

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

Weiterhin gilt

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

da

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

falls $(A + tC)$ invertierbar ist. Für ein genügend kleines t ist das gewährleistet, da A invertierbar ist (s. Lemma 3.2.12).

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

Somit folgt

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

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 $\|A^{-1}\|\cdot\|\Delta A\| < 1$:

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

Beweis: s. Übungsblatt

3.2.11 Definition: Kondition einer Matrix

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

$$\kappa_{\|\cdot\|}(A) = cond_{\|\cdot\|} := \|A\|\cdot\|A^{-1}\|$$

heißt **Kondition der Matrix** bzgl. der Norm $\|\cdot\|$.
Ist $\|\cdot\|$ von einer Vektor-Norm $\|\cdot\|_p$ induziert, bezeichnet $cond_p(A)$ die $cond_{\|\cdot\|_p}(A)$. Wir schreiben $cond(A)$ für $cond_2(A)$.
$cond_{\|\cdot\|}(A)$ schätzt die relative Kondition eines linearen GLS $Ax = b$ für alle möglichen Störungen in b oder in A ab und diese Abschätzung ist scharf.

Es stellt sich nun die Frage:
Wann existiert die Inverse der gestörten invertierbaren Matrix A?
Hierzu werden wir die Relationen benötigen:

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

und mit $C \in \mathbb{R}^{n\times n}$, $\|C\| < 1$

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

27.10.2014

3.2.12 Lemma (Neumannsche Reihe)

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

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

Weiterhin gilt:

$$\|(I-C)^{-1}\| \leq \frac{1}{1-\|C\|}$$

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

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

Daraus folgt bereits, dass $\sum_{k=1}^{\infty} C^k$ existiert (nach Majorantenkriterium).
Weiter gilt dann:

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

□

3.2.13 Bemerkung

a) Für 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 gilt offensichtlich:

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

d) $\kappa(\alpha A) = \kappa(A) \quad$ 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.

3.2.14 Beispiel: Kondition eines nichtlinearen Gleichungssystems

Sei $f : \mathbb{R}^n \to \mathbb{R}^n$ stetig differenzierbar und $y \in \mathbb{R}^n$ gegeben.
Löse

$$f(x) = y$$

Gesucht:

$$\kappa(f^{-1}, y)$$

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

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

Hiermit folgt:

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

[6] Beweis: siehe Übungsblatt 3

Für skalare Funktionen $f : \mathbb{R} \to \mathbb{R}$ 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.
Für $|f'(x)| \gg 0$ ist es eine gute absolute Kondition.

IMAGE MISSING

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

3.2b) Komponentenweise Konditionsanalyse

3.2.15 Beispiel

Falls A Diagonalgestalt hat, sind die Gleichungen unabhängig voneinander (entkoppelt). Die erwartete relative Kondition wäre dann – wie bei skalaren Gleicungen – stets gleich 1. Ebenso sind Störungen nur in der Diagonale zu erwarten. Jedoch:

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

3.2.16 Definition: Komponentenweise Kondition

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

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

Vorsicht:

$$\frac{\|\widetilde{x} - x\|_\infty}{\|x\|_\infty} \neq \max_i \frac{|\widetilde{x}_i - x_i|}{|x_i|}$$

3.2.17 Lemma

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

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

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

$$f_i(\widetilde{x}) - f_i(x) = \left(\frac{\partial f_i}{\partial x_i}(x), \cdots, \frac{\partial f_i}{\partial x_n}(x)\right) \cdot \begin{pmatrix} \widetilde{x}_1 - x_1 \\ \vdots \\ \widetilde{x}_n - x_n \end{pmatrix} + o\left(\|\widetilde{x} - x\|\right)$$

$$\Rightarrow |f_i(\widetilde{x}) - f_i(x)| \leq |Df(x)| \cdot \begin{pmatrix} |x_1| \cdot \frac{\widetilde{x}_1 - x_1}{|x_1|} \\ \vdots \\ |x_n| \cdot \frac{\widetilde{x}_n - x_n}{|x_n|} \end{pmatrix} + o\left(\max_i \frac{\widetilde{x}_i - x_i}{|x_i|}\right) \qquad \text{da } x_i \text{ fest und } \widetilde{x}_i \to x_i$$

$$\leq |Df(x)| \cdot |x| \cdot \max_i \frac{\widetilde{x}_i - x_i}{|x_i|} + o\left(\max_i \frac{\widetilde{x}_i - x_i}{|x_i|}\right)$$

$$\Rightarrow \frac{\|f(\widetilde{x}) - f(x)\|_\infty}{\|f(x)\|_\infty} \leq \frac{\| \, |Df(x)| \cdot |x| \, \|_\infty}{\|f(x)\|_\infty} \cdot \max_i \frac{\widetilde{x}_i - x_i}{|x_i|} + o\left(\max_i \frac{\widetilde{x}_i - x_i}{|x_i|}\right)$$

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

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

und in obiger Rechnung gilt Gleichheit.
Also folgt, dass

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

□

3.2.18 Beispiel

a) Komponentenweise Kondition der Multiplikation

$$f : \mathbb{R}^2 \to \mathbb{R},\ f(x,y) := x \cdot y$$

$$\Rightarrow Df(x,y) = (y,x)$$

$$\Rightarrow \kappa_{rel}(x,y) = \frac{\left\| (|y|,|x|) \cdot \begin{pmatrix} |x| \\ |y| \end{pmatrix} \right\|_\infty}{|x \cdot y|}$$

$$= \frac{2 \cdot |x| \cdot |y|}{|x \cdot y|}$$

$$= 2$$

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

$$f : b \mapsto A^{-1}b$$

$$\kappa_{rel} = \frac{\| \, |A^{-1}| \cdot |b| \, \|_\infty}{\|A^{-1}b\|_\infty}$$

Falls A eine Diagonalmatrix ist, folgt:

$$\kappa_{rel} = 1$$

c) Komponentenweise Kondition des Skalarproduktes:

$$\begin{aligned}
\langle x, y\rangle &:= \sum_{i=1}^{n} x_i y_i = x^T y \\
f : \mathbb{R}^2 \to \mathbb{R},\ f(x,y) &= \langle x, y\rangle \\
\Rightarrow Df(x,y) &= (y^T, x^T) \\
\kappa_{rel} &= \frac{\left\| \left|(y^T, x^T)\right| \cdot \left| \begin{pmatrix} x \\ y \end{pmatrix} \right| \right\|_\infty}{\|\langle x, y\rangle\|_\infty} \\
&= \frac{2 \cdot |y^T| \cdot |x|}{|\langle x, y\rangle|} \\
&= 2 \cdot \frac{\langle |x|, |y|\rangle}{|\langle x, y\rangle|} \\
&= 2 \cdot \frac{\cos(|x|, |y|)}{\cos(x, y)}
\end{aligned}$$

$$\text{da } \cos(x,y) = \frac{\langle y, x\rangle}{\|x\|_2 \cdot \|y\|_2}\,.$$

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

3.3 Stabilität von Algorithmen

Bislang: Kondition eines gegebenen Problems (f, x).
Nun stellt sich die Frage: *Was passiert durch das Implementieren am Rechner?*
Ein „stabiler“ Algorithmus sollte ein gut konditioniertes Problem nicht „kaputt machen“.

IMAGE MISSING

3.3a) Vorwärtsanalyse

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

3.3.1 Bemerkung

Für die Rechenoperationene $+, -, \cdot, /$, kurz ∇, gilt:

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

29.10.2014 mit $|\epsilon|, |\mu| \leq eps$.

3.3.2 Beispiel

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 durch

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

gegeben mit

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

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

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

$$\begin{aligned} \widetilde{f}^{(1)}(x) &= (fl(x_1 \cdot x_2), x_3) \\ &= (x_1 x_2 (1 + \epsilon_1), x_3) \end{aligned}$$

mit $|\epsilon_1| \leq eps$:

$$\begin{aligned} \widetilde{f}(x) &= \widetilde{f}^{(2)}(\widetilde{f}^{(1)}(x)) \\ &= fl(f^{(2)}(x_1 x_2 (1 + \epsilon_1), x_3)) \\ &= \frac{x_1 x_2 (1 + \epsilon_1)}{x_3} \cdot (1 + \epsilon_2) \\ &= f(x) \cdot (1 + \epsilon_1)(1 + \epsilon_2) \end{aligned}$$

mit $|\epsilon_2| \leq eps$:

$$\begin{aligned} \frac{|\widetilde{f}(x) - f(x)|}{|f(x)|} &= |\epsilon_1 + \epsilon_2 + \epsilon_1 \cdot \epsilon_2| \\ &\leq 2eps + eps^2 \end{aligned}$$

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

Daher:

3.3.3 Definition: Elementar ausführbar

Eine Abbildung $\phi : U \subseteq \mathbb{R}^n \to \mathbb{R}^m$ 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

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

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

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

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

3.3.4 Definition: Algorithmus, Implementation

Sei $f : E \subseteq \mathbb{R}^n \to \mathbb{R}^m$ gegeben.
Ein Tupel $\left(f^{(1)}, \cdots, f^{(l)}\right)$ 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 $(\widetilde{f}1^{(1)}, \cdots, \widetilde{f}^{(l)})$ 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.

3.3.5 Lemma (Fehlerfortpflanzung)

Sei $x \in \mathbb{R}^n$ und $\widetilde{x} \in \mathbb{F}^n$ mit $|\widetilde{x}_i - x_i| \leq eps|x_i|$ für alle $i = 1, \cdots, n$. 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 $\widetilde{x}^{(j+1)}$ gilt, falls $x^{(j+1)} \neq 0$ für alle $j = 0, \ldots, (l-1)$ und $\|\cdot\|$ eine beliebige p-Norm ist:

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

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

Beweis

$$\begin{aligned}
\frac{\left\|\widetilde{x}^{(j+1)} - x^{(j+1)}\right\|}{\|x^{(j+1)}\|} &= \frac{\left\|\widetilde{f}^{(j)}(\widetilde{x}^{(j)}) - f^{(j)}(x^{(j)})\right\|}{\|f^{(j)}(x^{(j)})\|} \\
&\leq \frac{\left\|\widetilde{f}(\widetilde{x}) - f(\widetilde{x})\right\|}{\|f(\widetilde{x})\|} \cdot \frac{\|f(\widetilde{x})\|}{\|f(x)\|} + \frac{\|f(\widetilde{x}) - f(x)\|}{\|f(x)\|} \qquad \text{(Index } j \text{ vernachlässigt)} \\
&\leq eps\left(1 + \frac{\|f(\widetilde{x}) - f(x)\|}{\|f(x)\|}\right) + \frac{\|f(\widetilde{x}) - f(x\|)}{\|f(x)\|} \\
&\overset{\text{nach 3.3.3}}{=} eps + (eps+1) \cdot \left(\kappa(j) \cdot \frac{\left\|\widetilde{x}^{(j)} - x^{(j)}\right\|}{\|x^{(j)}\|}\right) + o\left(\frac{\left\|\widetilde{x}^{(j)} - x^{(j)}\right\|}{\|x^{(j)}\|}\right)
\end{aligned}$$

Nach Voraussetzung gilt Gleichung (3.3.4) mit $\mathcal{K}^{(0)} = 1$ für $j = 0$.
Für $j = 1$ folgt nach Voraussetzung mit Gleichung (3.3.3)

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

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

$$\begin{aligned}
\frac{\left\|\widetilde{x}^{(j+1)} - x^{(j+1)}\right\|}{x^{(j+1)}} &\leq eps + (1+eps)\kappa^{(j)}\left[eps\mathcal{K}^{(j-1)} + o(eps)\right] \\
&\qquad + (1+eps) \cdot o\left(eps \cdot \mathcal{K}^{(j-1)} + o(eps)\right) \\
&= eps\left(1 + \kappa^{(j)} \cdot \mathcal{K}^{(j-1)}\right) + o(eps)
\end{aligned}$$

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

Hiermit folgt:

3.3.6 Korollar

Unter der Voraussetzung von Lemma 3.3.5 gilt:

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

3.3.7 Bemerkung

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.

3.3.8 Bemerkung zur Sprechweise

Der Quotient

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

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

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

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**.

3.3.9 Beispiel

Nach Gleichung (3.3.3) gilt für die Elementaroperationen $\mathcal{K} \leq 1$. Da für die Subtraktion zweier annähernd gleich großer Zahlen $\kappa_{rel} \gg 1$ gilt, ist der Stabilitätsfaktor 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 $\widetilde{f}(\widehat{x})$ als exaktes Ergebnis von einer gestörten Eingabe $\widehat{x}$ unter der exakten Abbildung f aufgefasst werden?
Das würde heißen

$$\exists\, \widehat{x} \in \mathbb{R}^n : f(\widehat{x}) = \widetilde{f}(\widetilde{x})\,.$$

Dann schätze den Fehler

$$\|\widehat{x} - x\|$$

bzw. für nicht injektive f

$$\min_{\widehat{x}\in\mathbb{R}^n} \left\{ \|\widehat{x} - x\| \middle| f(\widehat{x}) = \widetilde{f}(\widetilde{x}) \right\}$$

ab.

IMAGE MISSING

3.3.10 siehe Folien

(siehe auch Folien)

IMAGE MISSING

3.4 Beurteilung von Näherungslösungen linearer GLS

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

a) Im Sinne der Vorwärtsanalyse und der Fehlerentwicklung durch das Problem gilt:

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

nach Beispiel 3.2.10, mit dem Residuum

$$\begin{aligned} r(\widetilde{x}) &:= A\widetilde{x} - b \\ &= \widetilde{b} - b \\ &= \Delta b \end{aligned} \tag{3.4.1}$$

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

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

geeigneter.
Für große $cond(A)$ ist dieser Quotient jedoch weiterhin ungeeignet.

b) siehe Folien

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 Lönge 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 $\|\cdot\|$ eine Vektornorm.

4.1.1 Äquilibrierung der Zeilen

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

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

gesetzt wird.

4.1.2 Äquilibrierung der Spalten

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

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

gesetzt wird.

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

4.1.3 Lemma

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 $\widetilde{x}$ trotz Pivotisierung und Äquilibrierung noch sehr ungenau sein.

4.1.4 Nachiteration

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

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

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

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

und setze

$$x^{(1)} \coloneqq \widetilde{x} + \Delta x$$

Wiederhole dies sooft, bis $x^{(i)}$ „genau genug“ ist. Die Lösung $\widetilde{x}$ wird durch Nachiteration meist mit sehr gutem Erfolg verbessert (genaueres in Dahmen/Reusken MISSING (reference)) (4.1.5) wird mit der bereits vorhandenen LR-Zerlegung nur mit der neuen rechten Seite $r(\widetilde{x})$ gelöst, d.h. eine vorwärts und eine Rückwärtssubstitution mit $\mathcal{O}(n^2)$ flops.

4.1.5 Bemerkung (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 $\langle x, Ax\rangle = x^T Ax > 0 \quad$ für alle $x \neq 0$.
(kurs: **spd Matrix**)

4.2.1 Satz (Eigenschaften von symm., pos. def. Matrizen)

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) folgt aus (**??**)

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:

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

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

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

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

Weiterhin gilt:

$$x \neq 0 \Leftrightarrow L_1 x \neq 0$$

da L_1 invertierbar. Also gilt insgesamt:

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

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

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

Induktiv folgt hiermit iv). □ Insbesondere ergibt sich:

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

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

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

4.2.2 Folgerung

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:

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

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

$$IMAGE\ MISSING \tag{4.2.4}$$

Demnach funktioniert spaltenweises und zeilenweises Berechnen.

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

$\frac{1}{6}(n^2 - n)$ Additionen sowie Multiplikationen und

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

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

4.2.5 Bemerkung

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 (**??**)

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.

Index

Literatur

[1] P. Deuflhard und A. Hohmann. Numerische Mathematik. I, Eine algorithmisch orientierte Einführung. 3. Aufl. de Gruyter, 2002.

[2] von G. Golub und J.M. Ortega. Scientific Computing.

[3] K.H. Hoffmann G. Haemmerlin. Numerische Mathematik. Springer Berlin.

[4] W.H. Press u. a. Numerical Recipes in C++. Cambridge University Press.

[5] R.W. Hoppe R.W. Freund. Stoer/Bulirsch: Numerische Mathematik 1. Springer.

[6] J. Stoer und R. Bulirsch. Numerische Mathematik 2. Springer.

[7] W.Dahmen und A. Reusken. Numerik fuer Ingenieure und Naturwissenschaftler. Springer.