Einstieg mit 4 Beispielen

Die Numerische Mathematik behandelt Methoden zur Lösung vielfältiger Probleme aus der Mathematik, Statistik, Informatik, den Natur- und Ingenieurswissenschaften, aber auch anderen Disziplinen wie Medizin, Wirtschafts- und Sozialwissenschaften. Sie verwendet Kenntnisse aus den Grundvorlesungen Lineare Algebra und Analysis. Häufig stellen sich die Probleme in der Form, dass zu gegebenen Daten (z.B. Messwerten) ein neues mathematisches Objekt bestimmt werden soll. Beispiele sind

- Interpolation von (x, y)-Daten durch ein Polynom,
- Lösung eines linearen Gleichungssystem, z.B. bei den Kirchhoffschen Regeln der Elektrizitätslehre,
- Regressionsaufgabe zu statistischen Daten, z.B. Berechnung von Ausgleichsgeraden,
- Lösung von Differentialgleichungen mit gegebenen Koeffizientenfunktionen.

Besondere Bedeutung wird den folgenden Untersuchungen zukommen:

I. Numerische Algorithmen zur Lösung des mathematischen Problems:
   (a) Entwicklung der Algorithmen und Implementierung in einer Programmierumgebung
   (b) Bei näherungsweiser Berechnung der Lösung: Fehleranalyse, Abbruchkriterien für Iterationsverfahren
   (c) Komplexität der Algorithmen (bezogen auf Computerarchitektur)
   (d) Kondition der Algorithmen (= numerische Stabilität)

II. Kondition (= natürliche Stabilität) des mathematischen Problems: welchen Einfluss haben Änderungen der gegebenen Daten auf die Lösung

Oft werden zu einem mathematischen Problem mehrere Algorithmen bereitgestellt und zu jedem Algorithmus gibt es verschiedene Implementierungen. Man beachte, dass das numerische Ergebnis jeweils vom Algorithmus und der Implementierung abhängt.

Numerische Algorithmen sind erforderlich.

Am Beispiel der Nullstellenberechnung eines Polynoms wollen wir veranschaulichen, dass numerische Methoden unverzichtbar sind. Aus den Grundvorlesungen ist die folgende Aussage bekannt.

Fundamental-Satz der Algebra
Jedes Polynom $p(z) = a_n z^n + \cdots + a_1 z + a_0$ mit Koeffizienten $a_0, \ldots, a_n \in \mathbb{C}$ und $a_n \neq 0$ hat genau n komplexe Nullstellen unter Berücksichtigung der Vielfachheit.

Die verschiedenen Lösungsmethoden werden am Beispiel des Polynoms

$$p(z) = z^3 + z^2 - 2$$

veranschaulicht.

- Lösungsmethode der Schule: Raten der Nullstelle $z = 1$ und anschließende Faktorisierung mit dem *Hornerschema*: $p(z)$ wird mit Hilfe des Distributivgesetzes geschrieben in der Form

  $$p(z) = ((z + 1)z + 0)z - 2.$$

  Schematisch wird die Berechnung für $z = 1$ dargestellt in der Form

	1	1	0	-2	$\leftarrow$ Koeffizienten von p
$z = 1$		1	2	2	$\leftarrow$ $z\cdot$ Zahl links unterhalb
Summe	1	2	2	**0**	$= p(1)$

  Dabei stehen in der ersten Zeile die Koeffizienten von p, in der dritten Zeile die Summe der beiden darüberstehenden Zahlen, und in der mittleren Zeile das Produkt der Zahl links unterhalb mit der Auswertungsstelle $z = 1$. Der letzte Eintrag rechts unten im Schema ist der Funktionswert $p(1)$, also ist $z = 1$ eine Nullstelle von p.

  Damit wissen wir auch, dass p durch den Linearfaktor $z - 1$ teilbar ist. Die Einträge der letzten Zeile im Hornerschema liefern hierzu die Koeffizienten der Faktorisierung

  $$p(z) = (z - 1)(z^2 + 2z + 2).$$

  Zwei weitere (komplexe) Nullstellen von p erhalten wir sodann mit der pq-Formel, nämlich $-1 + i$ und $-1 - i$. Die komplette Zerlegung von p in Linearfaktoren lautet

  $$p(z) = (z - 1)(z + 1 + i)(z + 1 - i).$$

- Diese Methode versagt bereits für das Polynom

  $$q(x) = x^3 + x^2 - 2.1,$$

weil wir keine Nullstelle mehr einfach erraten können. Man könnte nun, ähnlich zur pq-Formel für quadratische Polynome, die Formel von Cardano[1] benutzen. Solche Formeln gibt es für Polynome 3. und 4. Grades, aber nicht mehr für höhere Grade. Daher soll exemplarisch auf andere Methoden zurückgegriffen werden.

Wir wissen aus dem Zwischenwertsatz und der Monotonie von p ($\rightarrow$ Analysis), dass genau eine reelle Nullstelle im Intervall $[1, 2]$ existiert. Ein Näherungsverfahren zur Berechnung dieser Nullstelle ist das *Newtonverfahren* ($\rightarrow$ Numerik). Hierbei werden nacheinander immer bessere Näherungswerte mit einem "Iterationsverfahren" berechnet. Wir geben sowohl die Notation in einem Pseudo-Code (links) als auch eine Matlab/Octave-Notation (rechts) an:

```
Wähle Startwert   x_0
Für k = 0,1,2,...:
 berechne  x_{k+1} := x_k - q(x_k)/q'(x_k)
bis k > k_max oder |q(x_{k+1})| ≤ tol.
```

```
kmax=10; tol=1e-12; format long
k=0; x=1; q=x^3+x^2-2.1;
while (k <= kmax) && (abs(q) > tol)
 qs=3*x^2+2*x;
 x=x-q/qs;
 k=k+1; q=x^3+x^2-2.1;
end
```

Zum Startwert $x_0 = 1$ erhalten wir schon in wenigen Schritten sehr gute Näherungswerte:

$$x_1 = 1.02,\ \ x_2 = 1.019688444547780,\ \ x_3 = 1.019688368159693,\ \ x_4 = 1.019688368159688.$$

Der letzte Wert x_4 ist bis auf 15 Nachkommastellen genau.

Numerische Ergebnisse müssen richtig interpretiert werden.

Wir diskutieren den Begriff der "natürlichen Stabilität" anhand eines einfachen linearen Gleichungssystems.

- Die eindeutige Lösung des linearen Gleichungssystems

$$\begin{bmatrix} 2.0001 & 1.9999 \\ 1.9999 & 2.0001 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

  ist $[x_1, x_2]^T = [0.25,\ 0.25]^T$.

- Die Änderung der rechten Seite auf $[1.0001,\ 0.9999]^T$, also um 0.1 Promille in jeder Komponente, liefert die neue Lösung $[x_1, x_2]^T = [0.75,\ -0.25]^T$. Dies ist eine relative Änderung der Komponenten des Lösungsvektors um 200 Prozent.

Man nennt dieses mathematische Problem "schlecht konditioniert" oder "natürlich instabil", weil geringe Änderungen der Daten (hier die Matrix-Koeffizienten und die Komponenten der rechten Seite) zu großen Änderungen der Lösung führen. Ein wichtiger Bereich der Numerik beschäftigt sich mit der Umformulierung schlecht konditionierter Probleme, Stichwort *Vorkonditionierung*.

[1] Gerolamo Cardano, 1501-1576

Numerische Algorithmen im Vergleich: Stabilität.

Anhand der Stellenauslöschung bei der Subtraktion soll erklärt werden, dass auch für natürlich stabile Probleme instabile Algorithmen gewählt werden können. Dies beobachtet man häufig bei der "numerischen Grenzwertbildung" durch Einsetzen. Als Beispiel betrachten wir

$$\lim_{x\to 0} \frac{\tan x - x}{x^3} = \frac{1}{3}.$$

- Den Grenzwert 1/3 rechnet man z.B. mit der Regel von de l'Hospital aus.
- Versucht man, den Grenzwert durch Einsetzen von kleinen positiven Werten für x zu "erraten", also z.B.

  $$x = 10^{-k} \quad \text{mit} \quad k = 1:9,$$

  ergeben sich mit Matlab/Octave die Werte

0.33467208545054,	0.33334666720702,	0.33333346673159,
0.33333333651891,	0.33333287573298,	0.33330746474457,
0.33087224502121,	0,	0

  Das Einsetzen kleiner x-Werte in den gegebenen Funktionsterm ist also völlig unbrauchbar, wie das Rechenergebnis 0 für $x = 10^{-8}$ bereits belegt.
- Verwendung der ersten 4 Glieder der Taylor-Reihe des Tangens ergibt

  $$\frac{\tan x - x}{x^3} = \frac{1}{x^3}\left(x + \frac{1}{3}x^3 + \frac{2}{15}x^5 + \frac{17}{315}x^7 + O(x^9) - x\right) = \frac{1}{3} + \frac{2}{15}x^2 + \frac{17}{315}x^4 + O(x^6),$$

  und die Summe der ersten 3 Summanden ist für positive $x \le 10^{-3}$ bereits im Bereich der Rechengenauigkeit von Matlab/Octave.

Fazit: Das *mathematische* Problem, den Funktionswert für kleine positive x zu berechnen, ist natürlich stabil (gut konditioniert), aber die Berechnung über den Funktionsterm ist *numerisch instabil*.

Numerische Algorithmen im Vergleich: Komplexität.

Für ein mathematisches Problem werden häufig verschiedene Algorithmen angeboten, die sich hinsichtlich der Komplexität (Laufzeit auf dem Computer, Anzahl der Rechenoperationen) drastisch unterscheiden. Als Beispiel dient die Determinantenberechnung einer $n \times n$-Matrix A.

- Leibnizregel:

  $$\det A = \sum_{\sigma \in S_n} \epsilon_\sigma \, a_{1,\sigma_1} \cdots a_{n,\sigma_n}$$

  mit $\epsilon_\sigma = 1$ oder -1 für die Permutation σ gerader oder ungerader Ordnung (z.B. Regel von Sarrus für $n = 3$).

  Aufwands-Betrachtung: $n!$ Summanden, die jeweils $n-1$ Multiplikationen erfordern. IMMENS!

- rekursive Verwendung des Entwicklungssatzes von Laplace:

$$\det A = \sum_{k=1}^{n} (-1)^{1+k} a_{1,k} \det A_{1,k},$$

wobei $A_{1,k}$ durch Streichen der 1. Zeile und der k-ten Spalte von A entsteht.

Aufwands-Betrachtung: Viel geringer, wenn bei der Rekursion die Determinante der Teilmatrizen, die mehrfach auftreten, nur einmal berechnet werden (z.B. in Maple).

- A wird mit dem Gauß-Algorithmus in Dreiecksgestalt überführt:

$$A \longrightarrow R = \begin{bmatrix} r_{1,1} & \cdots & r_{1,n} \\ & \ddots & \vdots \\ 0 & & r_{n,n} \end{bmatrix}$$

Je nach Anzahl der Zeilen- und Spalten-Permutationen ist

$$\det A = \pm \det R = \pm r_{1,1} r_{2,2} \cdots r_{n,n}.$$

Aufwands-Betrachtung: $n^3/3$ Additionen/Multiplikationen für Gauß-Algorithmus und $n-1$ Multiplikationen für $\det R$. Dies ist das effizienteste Verfahren.

Kapitel 1

Fehleranalyse

Prolog

1.1 Zahldarstellung und Rundungsfehler

1.2 Konditionierung mathematischer Aufgaben

1.3 Stabilität numerischer Algorithmen

Prolog: Arten von Fehlern
Wir unterscheiden in der Numerischen Mathematik zwischen mehreren Fehlertypen.

- Eingangsfehler (Datenfehler) wie z.B. Messungenauigkeiten, Fehler bei der Datenerfassung
- Diskretisierungsfehler, z.B.
  - Riemannsche Summe als Approximation des Integrals
  - Differenzenquotient mit fester Schrittweite h als Approximation der Ableitung
  - $n-$te Partialsumme als Approximation des Grenzwertes einer Reihe
- Rundungsfehler, z.B. bei Dezimalzahlen mit 5 Nachkommastellen

$$\frac{1}{3} \approx 0.33333$$

- Abbruchfehler bei Iterationsverfahren, siehe Beispiel des Newton-Verfahrens in der Einleitung

Für diese Fehlertypen werden häufig quantitative Aussagen mit Hilfe von mathematischen Überlegungen bereitgestellt.

Nicht weiter betrachtet werden dagegen Fehler durch menschlichen Irrtum oder maschinelle Fehlfunktion.

Merke: Der Begriff des Fehlers in der Numerischen Mathematik weicht stark vom umgangssprachlichen Begriff ab.

1.1 Zahldarstellung und Rundungsfehler

Die Durchführung numerischer Berechnungen mit Computern oder Taschenrechnern erfordert es, die ganzen, rationalen, reellen oder komplexen Zahlen im Speicher zu verwalten. Dadurch bestehen von vornherein physikalische Grenzen: anstatt der unendlichen Mengen $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ steht nur ein endlicher Zahlenbereich zur Verfügung. Dieser wird von der vorhandenen Rechnerarchitektur und der verwendeten Programmierumgebung festgelegt.

Für die Numerische Mathematik sind die dargestellten reellen Zahlbereiche besonders wichtig.

1.1.1 Definition (Gleitkommazahlen):
Die Menge $fl = fl(b, r, s)$ von Gleitkommazahlen zur Basis $b \in \mathbb{N}$, $b \geq 2$, mit Mantissenlänge $r \in \mathbb{N}$ und Exponentenlänge s, besteht aus 0 sowie allen reellen Zahlen der Form

$$\pm \left(\sum_{j=1}^{r} m_j \cdot b^{-j} \right) \cdot b^E, \qquad E = \pm \sum_{k=0}^{s-1} e_k \cdot b^k,$$

mit $m_j, e_k \in \{0, \ldots, b-1\}$ und der Einschränkung $m_1 \neq 0$ (normalisierte Gleitkommazahl zur Eindeutigkeit der Darstellung).

1.1.2 Bemerkung:

(i) Moderne Rechner verwenden $b = 2, 10, 16$. Das IEEE Standard-Zahlenformat für "einfache Genauigkeit" (in Matlab `single(x)`) verwendet die Zahlenmenge $fl(2, 23, 8)$. Die "doppelte Genauigkeit" (in Matlab die Standardeinstellung) verwendet $fl(2, 52, 11)$. Die Mantissenlängen erhält man durch die Befehle `-log(eps('single'))/log(2)` bzw. `-log(eps)/log(2)`.

(ii) Die Verwendung der Gleitkommazahlen ist wesentlich, damit Zahlen unterschiedlicher Größe verarbeiten werden können: $1.78 * 10^{-33}$ g ist die Massenzunahme eines Körpers, dessen innere Energie um 1 eV steigt, ca. 10^{53} kg ist die Masse des beobachtbaren Universums.

(iii) Die Menge $fl = fl(b, r, s)$ ist eine endliche Teilmenge der rationalen Zahlen. Um einen Überblick zu erhalten, betrachtet man die größte darstellbare Zahl (und ihr Negatives)

$$a_{min,max} = \pm(1 - b^{-r})b^{b^s - 1}$$

und ebenso die kleinste positive darstellbare Zahl (und ihr Negatives)

$$a_{neqmax,posmin} = \pm b^{-b^s}.$$

Dann liegt die endliche Menge $fl(b, r, s)$ im "zulässigen Bereich"

$$D(fl) := [a_{min}, a_{negmax}] \cup \{0\} \cup [a_{posmin}, a_{max}].$$

Für Zahlen außerhalb dieses Bereichs wird in manchen Programmen eine spezielle Notation verwendet, z.B. `Inf` für Zahlen größer als a_{max}.

Die Mantissenlänge r entscheidet über die relative Genauigkeit der Zahldarstellung. Ist E ein zulässiger Exponent, so liegen im halboffenen Intervall $I_E := [b^{E-1}, b^E)$ genau $(b-1)b^{r-1}$ Gleitkommazahlen, nämlich

$$\begin{aligned} b^{E-1} &= 0.\underbrace{100\ldots0}_{r}\cdot b^E, \\ b^{E-1} + b^{E-r} &= 0.\underbrace{100\ldots1}_{r}\cdot b^E, \\ b^{E-1} + 2b^{E-r}&, \\ &\vdots \\ b^E - b^{E-r}&. \end{aligned}$$

Der Abstand zwischen zwei benachbarten Gleitkommazahlen in diesem Intervall ist konstant b^{E-r}. Im nachfolgenden Intervall I_{E+1} liegen genau so viele Gleitkommazahlen, ihr Abstand ist um den Faktor b größer. Daraus ergibt sich die folgende Darstellung für den Rundungsfehler.

1.1.3 Definition (Rundungsoperator):
Es seien $b \in \mathbb{N}$ gerade sowie $r, s \in \mathbb{N}$ gegeben. Der Rundungsoperator $\mathrm{rd} : D(fl) \to fl(b, r, s)$ ist definiert durch $\mathrm{rd}(0) = 0$ und für jede reelle Zahl

$$x = \pm\left(\sum_{j=1}^{\infty} m_j \cdot b^{-j}\right) \cdot b^E \neq 0$$

mit $m_1 \neq 0$ und $E \in \{0, \pm1, \ldots, \pm b^s - 1\}$ durch

$$\mathrm{rd}(x) = \mathrm{sgn}(x) \cdot \begin{cases} \left(\sum_{j=1}^{r} m_j \cdot b^{-j}\right) \cdot b^E & \text{, falls} \quad m_{r+1} < \frac{b}{2}, \\ \left(\sum_{j=1}^{r} m_j \cdot b^{-j} + b^{-r}\right) \cdot b^E & \text{, sonst.} \end{cases}$$

Hierdurch wird $|x - \mathrm{rd}(x)| = \min_{a \in fl} |x - a|$ erfüllt.

Bemerkung: Die Einschränkung auf gerades b hat praktische Gründe. Für ungerades b kann $\mathrm{rd}(x)$ nicht allein anhand der Mantissenstelle m_{r+1} bestimmt werden, damit die Minimalitätsbedingung erfüllt ist.

Anahnd des Rundungsoperators können wir schon die Unterscheidung zwischen absoluten und relativen Fehlern veranschaulichen.

1.1.4 Satz:
Es seien $b \in \mathbb{N}$ gerade, $r, s \in \mathbb{N}$. Weiter sei $x \in D(fl) \setminus \{0\}$ eine reelle Zahl im zulässigen Bereich. Dann gilt

(i) für den *absoluten Rundungsfehler* $|x - \mathrm{rd}(x)| \leq \frac{1}{2} b^{E-r}$

(ii) für den *relativen Rundungsfehler* $\left| \frac{x - \mathrm{rd}(x)}{x} \right| \leq \frac{1}{2} b^{-r+1}$.

Die Stabilität von Algorithmen hängt damit zusammen, wie sich Rundungsfehler auf die weiteren Rechenoperationen auswirken, man nennt dies die Fehler-Fortpflanzung. Zur Analyse ist eine Darstellung des Rundungsfehlers in der folgenden Form hilfreich.

1.1.5 Korollar:
Unter den Voraussetzungen von Satz 1.1.4 gilt

$$\mathrm{rd}(x) = x(1 + \delta_x) \quad \text{mit} \quad |\delta_x| \leq \frac{1}{2} b^{-r+1} =: \texttt{eps}.$$

Die Zahl `eps` wird die *Maschinengenauigkeit* oder *relative Rechengenaugkeit* der Gleitkommaarithmetik genannt.

Zur Bedeutung der Zahl `eps`: Das gesamte Intervall $[1 - \texttt{eps}, 1 + \texttt{eps})$ wird auf 1 gerundet; mit anderen Worten, die Zahl $1 + 2\texttt{eps}$ ist die kleinste darstellbare Zahl größer als 1.

Durch Rundung werden die üblichen Rechenoperationen eventuell verfälscht. Deshalb führen wir neue Symbole ein, die die mathematische Addition, Subtraktion etc. durch die numerische Operation ersetzen. Man beachte, dass bei den neuen Operationen die üblichen Rechengesetze (Assoziativgesetz, Distributivgesetze) verloren gehen!

1.1.6 Definition:
Bei dem Standardmodell der Gleitkommaarithmetik sind die Gleitkommaoperationen

$$\{\oplus, \ominus, \otimes, \oslash\} : fl \times fl \to fl$$

definiert durch

$$x \odot y := \mathrm{rd}(x \cdot y) = (x \cdot y)(1 + \delta), \quad |\delta| \leq \mathrm{eps}, \quad \odot \in \{\oplus, \ominus, \otimes, \oslash\}.$$

Den Einfluss der Zahlenbereiche erkennt man am Unterschied der Ergebnisse von

$$(1 \oplus 10^{-8}) \ominus 1$$

in einfacher bzw. doppelter Genauigkeit, in Matlab

$$(\texttt{single}(1) + \texttt{single}(1\texttt{e}-8)) - \texttt{single}(1), \qquad (1 + 1\texttt{e}-8) - 1.$$

Solche Effekte treten bei Addition von Zahlen mit gleichem Vorzeichen, aber stark unterschiedlichem Absolutbetrag auf (vgl. Übung).

1.2 Konditionierung eines mathematischen Problems

Wir betrachten die folgende mathematische Aufgabe: Gegeben sind Teilmengen X und Y von normierten Vektorräumen sowie eine Funktion $f : X \to Y$. Gesucht ist der Wert $y = f(x)$ für einen Punkt $x \in X$.

Fragestellung: Wie groß ist der Einfluss von Änderungen der Eingabedaten x bei **exakter** Rechnung, ohne die Berücksichtigung von Rundungsfehlern. Hierbei spielt der gewählte Algorithmus zur Berechnung von $y = f(x)$ und das gegebene Zahlenformat also keine Rolle:

$$\begin{array}{ccccc} \text{Eingabedaten} & & \text{Funktion} & & \text{Ausgabedaten} \\ \widetilde{x} = x + \Delta_x \in X & & & & \widetilde{y} = f(\widetilde{x}) \\ \text{mit absolutem Fehler} & \to & f & \to & \text{mit absolutem Fehler} \\ \Delta_x = \widetilde{x} - x & & & & \Delta_y = \widetilde{y} - y \end{array}$$

Wir betrachten den "Verstärkungsfaktor des relativen Fehlers", also $k(x)$ in der Ungleichung

$$\frac{\|\Delta_y\|}{\|y\|} \leq k(x) \frac{\|\Delta_x\|}{\|x\|},$$

wobei links die Norm im Wertebereich und rechts die Norm im Definitionsbereich steht. Spezieller im Fall $Y \subset \mathbb{R}^m$ und $X \subset \mathbb{R}^n$ betrachten wir die einzelnen Faktoren $k_{ij}(x)$ in

$$\frac{|\Delta_{y_i}|}{|y_i|} \leq \sum_{j=1}^{n} k_{ij}(x) \frac{|\Delta_{x_j}|}{|x_j|}$$

für die relativen Fehler in jeder Komponente der Vektoren y und x.

Bevor wir die genaue Betrachtung beginnen, sollen einige zentrale Begriffe und Notationen dargestellt werden. Diese wurden in der Analysis teilweise eingeführt.

1.2.1 Definition: Norm
Sei V ein Vektorraum über dem Körper $\mathbb{K} = \mathbb{R}$ oder $\mathbb{K} = \mathbb{C}$. Eine Abbildung $\|\cdot\| : V \to \mathbb{R}$ heißt eine Norm auf V, falls

(N1) $\|v\| \geq 0$ für alle $v \in V$, sowie $(\|v\| = 0 \Leftrightarrow v = 0)$;

(N2) $\|\alpha v\| = |\alpha| \|v\|$ für alle $\alpha \in \mathbb{K}$ und $v \in V$;

(N3) $\|v + w\| \leq \|v\| + \|w\|$ für alle $v, w \in V$.

Bemerkung:

(i) (N2) nennt man *positive Homogenität*, (N3) ist die *Dreiecksungleichung*.

(ii) Für jede Norm $\|\cdot\|$ gilt außerdem

$$\textbf{(N4)} \quad \|v \pm w\| \geq \Big| \|v\| - \|w\| \Big|, \quad v, w \in V.$$

Hieran erkennt man sofort, dass die Normabbildung selbst gleichmäßig stetig auf V ist.

(iii) Die Norm induziert eine Metrik $d : V \times V \to \mathbb{R}$ mit $d(v, w) = \|v - w\|$. Damit sind die topologischen Begriffe der Analysis im normierten Raum V verfügbar: Konvergenz, offene Umgebungen, etc.

1.2.2 Beispiele von Normen auf $\mathbb{K}^n$:
Die ℓ_p-Normen mit $1 \leq p < \infty$

$$\|x\|_p = \left(\sum_{k=1}^{n} |x_k|^p \right)^{1/p}$$

und die Maximumnorm

$$\|x\|_\infty = \max_{1 \leq k \leq n} |x_k|.$$

Die oben erwähnten Normen sind alle äquivalent in dem folgenden Sinne. Insbesondere ist Konvergenz in $\mathbb{K}^n$ bezüglich aller Normen das gleiche.

1.2.3 Satz:
Auf dem endlich-dimensionalen Vektorraum $\mathbb{K}^n$ sind alle Normen äquivalent, d.h.: Zu je zwei Normen $\|\cdot\|$ und $\|\cdot\|'$ gibt es positive Konstanten m, M, so dass

$$m\|x\|' \leq \|x\| \leq M\|x\|' \quad \text{für alle} \quad x \in \mathbb{K}^n.$$

Beweis: Es genügt zu zeigen, dass jede beliebige Norm $\|.\|$ äquivalent zur ℓ_1-Norm $\|.\|_1$ ist. Dazu betrachten wir die Funktion

$$f : \mathbb{K}^n \to \mathbb{R}, \quad f(x) = \|x\|,$$

wobei der Definitionsbereich $\mathbb{K}^n$ mit der ℓ_1-Norm versehen ist. Deshalb müssen wir die Stetigkeit von f erst noch beweisen, sie folgt nicht direkt aus Bemerkung (ii) zu Definition 1.2.1. Wir schreiben $x \in \mathbb{K}^n$ mit den kanonischen Einheitsvektoren $e_k = (\delta_{k,j})_{j=1,\ldots,n}$, also

$$x = \sum_{k=1}^{n} x_k e_k.$$

Die Dreiecksungleichungen (N4) und (N3) ergeben

$$|f(x) - f(y)| = |\, \|x\| - \|y\| \,| \leq \|x - y\| \leq \sum_{k=1}^{n} |x_k - y_k| \; \|e_k\|,$$

wobei hier jeweils die "neue" Norm $\|.\|$ steht. Setzen wir

$$M := \max_{1 \leq k \leq n} \|e_k\|,$$

so erhalten wir die Lipschitz-Stetigkeit von f, nämlich

$$|f(x) - f(y)| \leq M \sum_{k=1}^{n} |x_k - y_k| = M\|x - y\|_1.$$

Nun schränken wir f ein auf die Einheitsspäre bzgl. der ℓ_1-Norm,

$$S = \{x \in \mathbb{K}^n : \|x\|_1 = 1\}.$$

Axiom (N1) und die obige Überlegung zur Stetigkeit (mit $y = 0$) liefert $f(S) \subset (0, M]$. Hier ist die 0 ausgenommen, weil der Nullpunkt nicht in S liegt. S ist kompakt (wieder als Teilmenge von $\mathbb{K}^n$ mit der Topologie der ℓ_1-Norm) und f ist stetig, also ist auch das Bild $f(S)$ kompakt in $\mathbb{R}$. Damit existiert das Minimum $0 < m = \min_{x \in S} f(x)$ und mit der Homogenität (N2) folgt

$$m\|x\|_1 \leq \|x\| \leq M\|x\|_1 \quad \text{für alle} \quad x \in \mathbb{K}^n.$$

In den Übungen wird die Äquivalenz der ℓ_p-Normen mit $p = 1, 2, \infty$ mit expliziten Konstanten angegeben.

1.2.4 Bemerkung:
$\|\cdot\|$ sei eine Norm auf $\mathbb{K}^n$. Wir betrachten häufig Folgen $\left(x^{(k)}\right)_{k \in \mathbb{N}}$ von Vektoren $x^{(k)} \in \mathbb{K}^n$. Die Konvergenz gegen einen Vektor $x \in \mathbb{K}^n$ ist erklärt durch

$$x^{(k)} \to x \quad \Longleftrightarrow \quad \lim_{k \to \infty} \|x^{(k)} - x\| = 0.$$

Dies ist nach Satz 1.2.3 gleichbedeutend mit der *komponentenweisen* Konvergenz

$$\lim_{k \to \infty} x_j^{(k)} = x_j, \qquad j = 1, \ldots, n.$$

Mit dem Normbegriff lassen sich die folgenden beiden Begriffe nochmals genau angeben.

1.2.5 Definition (absolute und relative Fehler):
Sei V ein normierter Vektorraum mit zugehöriger Norm $\|\cdot\|$. Weiter seien $x, \widetilde{x} \in V$ sowie $\Delta_x := \widetilde{x} - x$. Dann heißt

$$\|\Delta_x\| = \|\widetilde{x} - x\|$$

der *absolute Fehler* von $\widetilde{x}$ an x. Im Fall $x \neq 0$ heißt

$$\frac{\|\Delta_x\|}{\|x\|}$$

der *relative Fehler* von $\widetilde{x}$ an x.

Häufig verwendet man auch Prozent-Angaben für den relativen Fehler. Für $\frac{\|\Delta_x\|}{\|x\|} = 1$ beträgt der relative Fehler 100 %.

In vielen mathematischen Aussagen zur Fehler-Analyse spielt die Größenordnung eine wichtige Rolle, etwa: Wie groß ist der Fehler einer Riemann-Summe zu einem bestimmten Integral bezogen auf die Schrittweite h? Dabei interessiert man sich für kleine Schrittweiten $h > 0$ und gibt die Größenordnung als $\mathcal{O}(h^p)$ an. Dies bedeutet, dass der Fehler für Schrittweiten $h \to 0$ mindestens so schnell gegen Null konvergiert wie die Potenz h^p. Genauer ist die folgende Definition, in der die "Groß-Oh" und die "Klein-oh" Notationen von Landau eingeführt werden. Das hier aufgeführte Beispiel erhält man, wenn man in den folgenden Bezeichnungen $V = \mathbb{R}_+$ als Menge der Schrittweiten h, $W = \mathbb{R}$ als Wertebereich für den Integrationsfehler $f(h)$ und $g(h) = h^p$ setzt und die Grenzwerte für $h_0 = 0$ betrachtet.

1.2.6 Bezeichnungen (Landau-Symbole):
Es seien V, W normierte Vektorräume, $f : V \to W$ und $g : V \to \mathbb{R}_+$ seien Funktionen. Wir sagen:

- f ist von der Ordnung $\mathcal{O}(g)$ bei $x = x_0$, geschrieben $f(x) = \mathcal{O}(g(x))$ für $x \to x_0$, wenn $C, r > 0$ existieren mit

  $$\|f(x)\|_W \leq Cg(x) \quad \text{für alle} \quad \|x - x_0\|_V < r,$$

  oder anders ausgedrückt,

  $$\limsup_{x \to x_0} \frac{1}{g(x)} \|f(x)\|_W < \infty.$$

- f ist von der Ordnung $o(g)$ bei $x = x_0$, geschrieben $f(x) = o(g(x))$ für $x \to x_0$, wenn

  $$\lim_{x \to x_0} \frac{1}{g(x)} \|f(x)\|_W = 0.$$

Die entsprechenden Versionen für die Fälle $x \to \infty$ (anstatt $x \to x_0$) oder $x \to -\infty$ im Reellen erhält man durch leichte Modifikation.

Beispiel: Die Funktion $f(x) = x^3 - 2x^5 + x^7$ erfüllt

$$f(x) = \mathcal{O}(x^3) \text{ bei } x = 0, \qquad f(x) = \mathcal{O}(x^7) \text{ bei } x = \pm\infty.$$

1.2.7 Definition (Konditionszahl):
Seien V, W normierte Vektorräume mit zugehörigen Normen $\|\cdot\|_V$ bzw. $\|\cdot\|_W$. Weiter sei $f : V \to W$ eine Funktion und $x \in V \setminus \{0\}$ sowie $f(x) \neq 0$.

Die Zahl $k(x) \geq 0$ heißt *Konditionszahl* des relativen Fehlers von f an der Stelle x, wenn für *kleine* Fehler $\|\widetilde{x} - x\|_V$ gilt

$$\frac{\|f(\widetilde{x}) - f(x)\|_W}{\|f(x)\|_W} \leq k(x) \cdot \frac{\|\widetilde{x} - x\|_V}{\|x\|_V} + o\left(\frac{\|\widetilde{x} - x\|_V}{\|x\|_V}\right).$$

Der erste Term auf der rechten Seite gibt also den wesentlichen Anteil des relativen Fehlers auf der linken Seite an, der zweite Summand ist vernachlässigbar. Man beachte, dass **die** Konditionszahl hier gar nicht definiert wird, sondern eine obere Schranke $k(x)$ auf der rechten Seite genügt. Natürlich ist man an möglichst guten oberen Schranken interessiert.

Wichtigstes Mittel zur Bestimmung von Konditionszahlen ist die mehrdimensionale Differentialrechnung. Dabei gehen wir davon aus, dass die betrachtete Funktion f einen Definitionsbereich in $\mathbb{K}^n$ und den Wertebereich $\mathbb{K}^m$ hat. Anstatt der Gesamt-Kondition $k(x)$ werden im folgenden Satz einzelne Konditionszahlen $k_{ij}(x)$ bezüglich aller Komponenten gebildet.

1.2.8 Satz (Konditionszahl differenzierbarer Funktionen):
Es sei $\Omega \subset \mathbb{K}^n$ offen und $f = (f_1, \ldots, f_m) : \Omega \to \mathbb{K}^m$ stetig differenzierbar. Dann gilt für x mit lauter von 0 verschiedenen Komponenten und $f_i(x) \neq 0$

$$\left|\frac{f_i(\widetilde{x}) - f_i(x)}{f_i(x)}\right| \leq \sum_{j=1}^{n} k_{i,j}(x) \cdot \left|\frac{\widetilde{x}_j - x_j}{x_j}\right| + o\left(\left|\frac{\widetilde{x}_j - x_j}{x_j}\right|\right)$$

mit den *Konditionszahlen* bzgl. der relativen Eingangsfehler

$$k_{i,j}(x) := \left|\frac{\partial f_i}{\partial x_j}(x) \cdot \frac{x_j}{f_i(x)}\right|.$$

Der Beweis ist ganz einfach: der Satz ist nur eine Umformulierung der "totalen Differenzierbarkeit" von f im Punkt x.

Erweiterung der Aussage: Die Kondition $k_i(x)$ der Komponentenfunktion f_i lässt sich durch Verwendung der Richtungsableitung

$$\partial_v f_i(x) = \lim_{h \to 0} \frac{f_i(x + hv) - f_i(x)}{h}$$

in Richtung $v = \frac{1}{\|\widetilde{x} - x\|_2}(\widetilde{x} - x)$ bestimmen. Man erhält

$$\left|\frac{f_i(\widetilde{x}) - f_i(x)}{f_i(x)}\right| \leq k_i(x) \cdot \frac{\|\widetilde{x} - x\|_2}{\|x\|_2} + o\left(\frac{\|\widetilde{x} - x\|_2}{\|x\|_2}\right),$$

wobei

$$k_i(x) = \max_{\|w\|_2=1} |\partial_w f_i(x)| \, \frac{\|x\|_2}{|f_i(x)|} = \|\nabla f_i(x)\|_2 \, \frac{\|x\|_2}{|f_i(x)|}$$

ist und ∇f_i der Gradient von f_i, also die Richtung des steilsten Anstiegs von f_i ist.

Eine recht vage Formulierung versucht nun zusammenzufassen, ob ein Problem natürlich stabil oder instabil hinsichtlich der relativen Fehler ist. Wir verwenden das Zeichen $\gg$ für "sehr viel größer als", ohne genau zu präzisieren, was damit gemeint ist.

1.2.9 Bezeichnung (Konditionierung math. Probleme):
Gegeben sei die Funktion f in 1.2.8. Man nennt die Berechnung von $y = f(x)$ an der Stelle $x \in \Omega$ *schlecht konditioniert*, wenn mindestens ein $k_{i,j} \gg 1$ ist; andernfalls nennt man es *gut konditioniert*. Im Fall $k_{i,j} < 1$ spricht man von *Fehlerdämpfung* und im Fall $k_{i,j} > 1$ von *Fehlerverstärkung*.

1.2.10 Beispiel (Konditionierung arithmetischer Grundoperationen):

- Die Addition $y = f(x_1, x_2) = x_1 + x_2$ mit $x_1, x_2 \in \mathbb{R} \setminus \{0\}$ hat die Konditionszahlen

$$\begin{aligned} k_{1,1}(x_1, x_2) &= \left| \frac{\partial f}{\partial x_1}(x) \cdot \frac{x_1}{f(x)} \right| = \left| \frac{x_1}{x_1 + x_2} \right| \\ k_{1,2}(x_1, x_2) &= \left| \frac{\partial f}{\partial x_2}(x) \cdot \frac{x_2}{f(x)} \right| = \left| \frac{x_2}{x_1 + x_2} \right|. \end{aligned}$$

  Sie besitzt Fehlerdämpfung an der Stelle (x_1, x_2), wenn beide Komponenten gleiches Vorzeichen haben, ist aber schlecht konditioniert für $x_1 \approx -x_2$. Dies entspricht der Intuition: Stellenauslöschung bei der Subtraktion fast gleicher Zahlen verursacht große Instabilität.

- Die Multiplikation $y = f(x_1, x_2) = x_1 \cdot x_2$ mit $x_1, x_2 \in \mathbb{R} \setminus \{0\}$ hat die Konditionszahlen

$$\begin{aligned} k_{1,1}(x_1, x_2) &= \left| \frac{\partial f}{\partial x_1}(x) \cdot \frac{x_1}{f(x)} \right| = \left| x_2 \cdot \frac{x_1}{x_1 \cdot x_2} \right| = 1 \\ k_{1,2}(x_1, x_2) &= \left| \frac{\partial f}{\partial x_2}(x) \cdot \frac{x_2}{f(x)} \right| = \left| x_1 \cdot \frac{x_2}{x_1 \cdot x_2} \right| = 1. \end{aligned}$$

  Sie ist überall (d.h. für alle $(x_1 \neq 0, x_2 \neq 0)$) gut konditioniert. Dasselbe gilt für die Division.

1.2.11 Beispiel: Konditionierung von $y = f(x_1, x_2) = x_1^2 - x_2^2$.

Für $x_1, x_2 \in \mathbb{R} \setminus \{0\}$ erhalten wir die Konditionszahlen

$$\begin{aligned} k_{1,1}(x_1, x_2) &= \left| \frac{\partial f}{\partial x_1}(x) \cdot \frac{x_1}{f(x)} \right| = \left| \frac{2}{1 - \left(\frac{x_2}{x_1}\right)^2} \right| \\ k_{1,2}(x_1, x_2) &= \left| \frac{\partial f}{\partial x_2}(x) \cdot \frac{x_2}{f(x)} \right| = \left| \frac{2}{1 - \left(\frac{x_1}{x_2}\right)^2} \right| \end{aligned}$$

Also ist die Berechnung von $f(x_1, x_2)$ schlecht konditioniert für $x_1 \approx \pm x_2$. Dies sieht man auch intuitiv: Ursache ist wieder die Stellenauslöschung bei der Subtraktion.

1.2.12 Beispiel: Lösung der quadratischen Gleichung $y^2 - py + q = 0$

Die Variablen des gestellten Problems in Satz 1.2.8 sind die Koeffizienten $p, q \in \mathbb{R}$, die Funktionen sind die Ausdrücke für die Nullstellen

$$y_1(p,q) = \frac{p}{2} + \sqrt{\frac{p^2}{4} - q}, \qquad y_2(p,q) = \frac{p}{2} - \sqrt{\frac{p^2}{4} - q}.$$

Wir setzen wie üblich $p, q \neq 0$ voraus. Der Vietasche Satz erlaubt die Substitutionen $p = y_1 + y_2$ und $q = y_1 \cdot y_2$. Außerdem ist $y_1 - y_2 = 2\sqrt{\frac{p^2}{4} - q}$, also

$$\frac{\partial y_1}{\partial p} = \frac{1}{2} + \frac{p}{2(y_1 - y_2)} = \frac{y_1}{y_1 - y_2}, \qquad \frac{\partial y_2}{\partial p} = -\frac{y_2}{y_1 - y_2},$$

$$\frac{\partial y_1}{\partial q} = \frac{-1}{y_1 - y_2}, \qquad \frac{\partial y_2}{\partial q} = \frac{1}{y_1 - y_2}.$$

Einsetzen liefert die Konditionszahlen

$$k_{1,1}(p,q) = \left|\frac{\partial y_1}{\partial p} \cdot \frac{p}{y_1}\right| = \left|\frac{y_1 + y_2}{y_1 - y_2}\right|$$

$$k_{1,2}(p,q) = \left|\frac{\partial y_1}{\partial q} \cdot \frac{q}{y_1}\right| = \left|\frac{y_2}{y_1 - y_2}\right|$$

und analog für $k_{2,1}$ und $k_{2,2}$. Also ist die Berechnung von y_1, y_2 schlecht konditioniert für $y_1 \approx y_2$. Dies ist wieder intuitiv erklärbar: In diesem Fall findet Auslöschung unterhalb der Wurzel statt.

1.3 Numerische Stabilität von Algorithmen

Nun untersuchen wir nicht nur das gestellte Problem, sondern auch den zur Lösung verwendeten Algorithmus auf seine Stabilität. Für das gleiche Problem können verschiedene Algorithmen verwendet werden, die ganz unterschiedliche Eigenschaften haben können. Das Standardproblem wird wieder als Funktionsauswertung $y = f(x)$ formuliert. Dies ist ja gerade das Prinzip eines Computerprogramms: zu einer Liste von Eingabe-Parametern (hier ein Punkt $x \in \mathbb{R}^n$) ist eine weitere Liste von Ausgabe-Parametern (hier das Ergebnis $f(x) \in \mathbb{R}^m$) gesucht.

1.3.1 Definition:
Ein *Algorithmus* zur Auswertung der Funktion

$$f : \Omega \to \mathbb{R}^m, \quad \Omega \subset \mathbb{R}^n \text{ offen}$$

ist eine Zerlegung

$$f = \phi^{(r)} \circ \phi^{(r-1)} \circ \cdots \circ \phi^{(0)}$$

in elementare Abbildungen (arithmetische Grundoperationen oder Standardfunktionen wie Wurzel, Sinus etc.)

$$\phi^{(i)} : \Omega_i \to \Omega_{i+1}, \qquad \Omega_i \subset \mathbb{R}^{n_i}, \quad i = 0, \ldots, r.$$

Er führt die Eingabedaten $x \in \Omega = \Omega_0$ über eine Kette von Zwischenergebnissen

$$x =: x^{(0)} \mapsto \phi^{(0)}(x^{(0)}) =: x^{(1)} \mapsto \ldots \mapsto \phi^{(r)}(x^{(r)}) =: x^{(r+1)} = y$$

zu $y = f(x) \in \mathbb{R}^m = \Omega_{r+1}$.

Fehlerfortpflanzung: Ein Rundungsfehler, der bei Berechnung von $x^{i+1} = \phi^{(i)}(x^{(i)})$ auftritt, geht in die sog. *Restabbildung*

$$\phi^{(r)} \circ \cdots \circ \phi^{(i+1)}$$

als Eingangsfehler ein und wird hierdurch an das Endresultat weitergegeben.

Die numerische Stabilität kann offensichtlich nur dann vorliegen, wenn das gestellte Problem selbst gut konditioniert ist. In diesem Sinne wird der Begriff wie folgt formuliert.

1.3.2 Definition und Satz (numerische Stabilität)
Ein Algorithmus $\phi^{(r)} \circ \cdots \circ \phi^{(0)}$ heißt *numerisch stabil*, wenn der im Verlaufe der Ausführung akkumulierte Fehler den durch die Konditionierung des Problems $y = f(x)$ bedingten Fehler nicht wesentlich übersteigt.

Der Algorithmus ist numerisch stabil, wenn alle Restabbildungen

$$\phi^{(r)} \circ \cdots \circ \phi^{(i+1)}, \qquad i = 0, \ldots, r-1,$$

gut konditioniert sind.

Wir schauen uns Algorithmen zu zwei Beispielen aus dem vorherigen Abschnitt an. Die elementaren Operationen ϕ_k sind die arithmetischen Operationen und die Wurzel. Im ersten Beispiel führen wir die Analyse der Fehlerfortpflanzung durch, indem wir z.B.

$$x_1 \oplus x_2 = (x_1 + x_2)(1 + \delta)$$

schreiben, wobei auf der rechten Seite die exakte Addition und Multiplikation steht und der Rundungsfehler durch δ ausgedrückt wird. Hierbei ist $|\delta| \leq$ `eps` kleiner oder gleich der Maschinengenauigkeit, siehe 1.1.5. Im zweiten Beispiel werden die Konditionszahlen der Restabbildungen bestimmt. Beide Wege sind bei der Untersuchung der numerischen Stabilität zulässig.

1.3.3 Beispiel: Numerische Stabilität von

$$y = f(x_1, x_2) = \underbrace{x_1^2 - x_2^2}_{\text{Alg. A}} = \underbrace{(x_1 + x_2)(x_1 - x_2)}_{\text{Alg. B}}$$

.

A Mit exakten Eingabedaten x_1, x_2 und relativen Rundungsfehlern $|\delta_k| \leq \texttt{eps}$ ist

$$\begin{aligned} y_A &= (x_1 \odot x_1) \ominus (x_2 \odot x_2) \\ &= (x_1^2(1+\delta_1) - x_2^2(1+\delta_2))(1+\delta_3) \\ &= x_1^2 - x_2^2 + \delta_1 x_1^2 - \delta_2 x_2^2 + \delta_3(x_1^2 - x_2^2) + \mathcal{O}(\texttt{eps}^2). \end{aligned}$$

Der relative Fehler bei Weglassen des $\mathcal{O}(\texttt{eps}^2)$-Terms ist

$$\frac{|y_A - y|}{|y|} \preceq \texttt{eps} \cdot \frac{x_1^2 + x_2^2 + |x_1^2 - x_2^2|}{|x_1^2 - x_2^2|} = \texttt{eps} \cdot \left(1 + \left|\frac{1 + (x_2/x_1)^2}{1 - (x_2/x_1)^2}\right|\right);$$

der Rundungsfehler wirkt sich stark aus bei $x_1 \approx \pm x_2$. Weil auch die Konditionszahl in Beispiel 1.2.10 diese Größenordnung aufweist, ist der Alg. numerisch stabil.

B Mit exakten Eingabedaten x_1, x_2 und relativen Rundungsfehlern $|\delta_k| \leq \texttt{eps}$ ist

$$\begin{aligned} y_B &= (x_1 \oplus x_2) \odot (x_1 \ominus x_2) \\ &= (x_1 + x_2)(1+\delta_1)(x_1 - x_2)(1+\delta_2)(1+\delta_3) \\ &= x_1^2 - x_2^2 + (\delta_1 + \delta_2 + \delta_3)(x_1^2 - x_2^2) + \mathcal{O}(\texttt{eps}^2). \end{aligned}$$

Der relative Fehler bei Weglassen des $\mathcal{O}(\texttt{eps}^2)$-Terms ist

$$\frac{|y_B - y|}{|y|} \preceq 3\texttt{eps};$$

der Rundungsfehler wirkt sich also nicht stark aus. Algorithmus B ist besser!

Fazit: Faktorisierung liefert oft numerisch stabilere Algorithmen.

1.3.4 Beispiel: Lösung der quadratischen Gleichung $y^2 - py + q = 0$

Zusätzlich zu den Voraussetzungen in 1.2.12 sei nun $p > 0$ und $q \ll p^2/4$. Dann ist das Problem zu diesen Eingabe-Parametern gut konditioniert: keine Auslöschung unter der Wurzel.

Die Berechnung beider Lösungen erfolgt in folgenden Schritten:

$$u = (p \odot p) \oslash 4, \quad v = u \ominus q, \quad w = SQRT(v),$$

dann mit $p > 0$

$$\begin{aligned} &\text{Alg. A:} \quad y_1 = (p \oslash 2) \oplus w, \quad y_2 = (p \oslash 2) \ominus w \\ &\text{Alg. B:} \quad y_1 = (p \oslash 2) \oplus w, \quad y_2 = q \oslash y_1 \end{aligned}$$

Wir betrachten hier nur einige Restabbildungen in Satz 1.3.2. Die Konditionszahlen $k_{i,j}$ für den relativen Fehler werden wie üblich berechnet.

- Die Rundungsfehler an p, w übertragen sich auf y_1 gemäß

$$\frac{|\Delta_{y_1}|}{|y_1|} \preceq \underbrace{\left|\frac{p/2}{p/2+w}\right|}_{\leq 1} \left|\frac{\Delta_p}{p}\right| + \underbrace{\left|\frac{w}{p/2+w}\right|}_{\leq 1} \left|\frac{\Delta_w}{w}\right| \leq 2\texttt{eps}.$$

  Die Restabbildung für y_1 ist also gut konditioniert.

- Alg. A ergibt für y_2

$$\frac{|\Delta_{y_2}|}{|y_2|} \preceq \underbrace{\left|\frac{p/2}{p/2-w}\right|}_{\gg 1} \left|\frac{\Delta_p}{p}\right| + \underbrace{\left|\frac{w}{p/2-w}\right|}_{\gg 1} \left|\frac{\Delta_w}{w}\right|$$

  mit großen Verstärkungsfaktoren im vorliegenden Fall $|q| \ll p^2/4$, also $w \sim p/2$. Die Restabbildung von Algorithmus A für y_2 ist schlecht konditioniert.

- Alg. B ergibt für y_2

$$\frac{|\Delta_{y_2}|}{|y_2|} \preceq \left|\frac{\Delta_q}{q}\right| + \left|\frac{\Delta_{y_1}}{y_1}\right|$$

  mit Konditionszahlen 1. Die Restabbildung von Algorithmus B für y_2 ist gut konditioniert.

Daher ist Alg. B numerisch stabil für diese Eingabeparameter, Alg. A ist es nicht.

Der Nachweis der numerischen Stabilität von größeren Algorithmen (Gauß-Elimination etc.) erfordert einen sehr hohen Aufwand. Wir werden im Verlauf der Vorlesung häufig auf Resultate in der Literatur verweisen und vereinzelt numerische Vergleiche von Algorithmen anstellen ($\rightarrow$ Pivotstrategien bei der Gauß-Elimination), aber keine exakte Analyse vornehmen.