Einführung in die Numerik (Potschka)

Robin Heinemann

30. April 2017

Inhaltsverzeichnis

0 Einführung

Beispiel 0.1 Simulation einer Pendelbewegung
Modellannahmen:

- Masse m an Stange
- keine Reibung
- Stange: Gewicht 0, starr, Länge l
- Auslenkung ϕ

Erste Fehlerquelle: Modellierungsfehler
Modellgleichungen:

$$F_T(\phi) = -m \cdot g \sin \phi$$

Konsistenzcheck:

$$F_T(0) = 0 \qquad \text{(Ruhelage)}$$
$$F_T\left(\frac{\pi}{2}\right) = F_G = -mg$$

Bewegungsgleichungen:

- Weg $s(t)$
- $\frac{\mathrm{d}s}{\mathrm{d}t} =: v(t)$ Geschwindigkeit
- $\frac{\mathrm{d}v}{\mathrm{d}t} =: a(t)$ Beschleunigung

Beziehungen:

- Bogenlänge $s(t) = l\phi(t)$
- 2. Newton'sches Gesetz ($F = ma$)

$$-mg\sin\phi(t) = m\frac{\mathrm{d}}{\mathrm{d}t}v(t) = m\frac{\mathrm{d}^2}{\mathrm{d}t^2}s(t) = ml\frac{\mathrm{d}^2}{\mathrm{d}t^2}\phi(t)$$

$\Longrightarrow$ DGL 2. Ordnung

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\phi(t) = -\frac{g}{l}\sin\phi(t) \quad t \geq 0$$

Für eindeutige Lösung braucht man zwei Anfangsbedingungen:

$$\phi(0) = \phi_0 \quad \frac{\mathrm{d}}{\mathrm{d}t}\phi(0) = u_0$$

Lösung bei kleiner Auslenkung: Linearisiere um $\phi = 0$

$$\sin\phi = \phi - \frac{1}{3!}\phi^3 + \cdots \approx \phi$$
$$\Longrightarrow \frac{\mathrm{d}^2}{\mathrm{d}t^2}\phi(t) = -\frac{g}{l}\phi(t)$$

Für $u_0 = 0$ findet man mit dem Ansatz $\phi(t) = A\cos(\omega t)$:

$$-\omega^2 A\cos(\omega t) = -\frac{g}{l}A\cos(\omega t)$$

die Lösung:

$$\phi(t) = \phi_0\cos\left(\sqrt{\frac{g}{l}}t\right)$$

Fehlerquelle: Abschneidefehler.
Numerische Lösung:
Setze $u(t) := \frac{\mathrm{d}}{\mathrm{d}t}\phi(t)$

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix}\phi\\u\end{pmatrix} = \begin{pmatrix}u\\-\frac{g}{l}\sin(\phi)\end{pmatrix}$$

Approximation mit Differenzenquotienten

$$\begin{pmatrix}u(t)\\-\frac{g}{l}\sin\phi(t)\end{pmatrix} = \frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix}\phi\\u\end{pmatrix} = \lim_{\Delta t\to 0}\frac{1}{\Delta t}\begin{pmatrix}\phi(t+\Delta t)-\phi(t)\\u(t+\Delta t)-u(t)\end{pmatrix} \approx \underbrace{\frac{1}{\Delta t}}_{\text{> 0, klein}}\begin{pmatrix}\phi(t+\Delta t)-\phi(t)\\u(t+\Delta t)-u(t)\end{pmatrix}$$

Fehlerquelle: Diskretisierungsfehler
Auf Gitter $t_n = n\Delta t$ mit Werten $\phi_n = \phi(n\Delta t), u_n = u(n\Delta t)$:

$$\phi_{n+1} = \phi_n + \Delta t u_n, u_{n+1} = u_n - \Delta t \frac{g}{l}\phi_n$$

Kleinerer Diskretisierungsfehler mit zentralen Differenzen:

$$-\frac{g}{l}\sin\phi(t) = \frac{\mathrm{d}^2}{\mathrm{d}t^2}\phi(t) \approx \frac{\phi(t+\Delta t) - 2\phi(t) + \phi(t-\Delta t)}{\Delta t^2}$$

Rekursionsformel:

$$\phi_{n+1} = 2\phi_n - \phi_{n-1} - \Delta t^2 \frac{g}{l}\sin\phi_n, n \geq 1$$

mit $\phi_1 = \phi_0 + \Delta t n_0$ (Expliziter Euler)
Letzte Fehlerquelle: Rundungsfehler

1 Fehleranalyse

1.1 Zahldarstellung und Rundungsfehler

Anforderung: Rechnen mit reellen Zahlen auf dem Computer.
Problem: Speicher endlich ($\Longrightarrow$ endliche Genauigkeit).
Lösung: Gleitkommazahlen, ein **Kompromiss** zwischen:

- Umfang darstellbarer Zahlen
- Genauigkeit
- Geschwindigkeit einfacher Rechenoperationen (+, -, ·, /)

Alternativen:

- Fixkommazahlen
- logarithmische Zahlen
- Rationalzahlen

Definition 1.1 Eine (normalisierte) Gleitkommazahl zur Basis $b \in \mathbb{N}, b \geq 2$, ist eine Zahl $x \in \mathbb{R}$ der Form

$$x = \pm m \cdot b^{\pm e}$$

mit der Mantisse $m = m_1 b^{-1} + m_2 b^{-2} + \ldots \in \mathbb{R}$ und dem Exponenten $e = e_{s-1}b^{s-1} + \cdots + e_0 b^0 \in \mathbb{N}$, wobei $m_i, e_i \in \{0, \ldots, b-1\}$. Für $x \neq 0$ ist die Darstellung durch die Normierungsvorschrift $m \neq 0$ eindeutig. Für $x = 0$ setzt man $m = 0$.

Beispiel 1.2 ($b = 10$) • m_i: i -te Nachkommastelle der Mantisse

- e: Verschiebt das Komma um e Stellen.

$$0.314 \times 10^1 = 3.14$$
$$0.123 \times 10^6 = 123\,000$$

Auf dem Rechner stehen nur endlich viele Stellen zur Verfügung:
r Ziffern + 1 Vorzeichen für Mantisse m
s Ziffern + 1 Vorzeichen für Exponenten.
Für $x = \pm[m_1 b^{-1} + \cdots + m_r b^{-r}] \cdot b^{\pm[e_{s-1} b^{s-1} + \cdots + e_0 b^0]}$ muss man also nur $(\pm)[m_1 \dots m_r](\pm)[e_{s-1} \dots e_0]$ abspeichern. Wählt man $b = 2$, so gilt $m_i, e_i \in \{0, 1\}$ und x kann mit $2 + r + s$ Bits gespeichert werden (Maschinenzahlen). Maschinenzahlen bilden das numerische Gleitkommagitter $A = A(b, r, s)$

Beispiel 1.3 ($b = 2, r = 3, s = 1$)

$$m = \frac{1}{2} + m_2 \frac{1}{4} + m_3 \frac{1}{8} \in \{\frac{4}{8}, \frac{5}{8}, \frac{6}{8}, \frac{7}{8}\}$$
$$e = e_0 \in \{0, 1\}$$

Da A endlich ist, gibt es eine größte/kleinste darstellbare Zahl:

$$\begin{aligned} x_{\{min/max\}} &= \pm(b-1)[b^{-1} + \cdots + b^{-r}] \cdot b^{(b-1)[b^{s-1} + \cdots + b^0]} \\ &= \pm\left(1 - b^{-r}\right) \cdot b^{(b^s - 1)} \end{aligned}$$

sowie eine kleinste positive/größte negative Zahl:

$$\begin{aligned} x_{posmin/negmax} &= \pm b^{-1} \cdot b^{-(b-1)[b^{s-1} + \cdots + b^0]} \\ &= b^{-b^s} \end{aligned}$$

Das gängigste Format ist das IEEE-Format, das auch hinter dem Python-Datentyp float steht:

$$x = \pm m \cdot 2^{c - 1022}$$

Dieser Datentyp ist 64 Bit (8 Byte) groß (doppelte Genauigkeit, double). Davon speichert 1 Bit das Vorzeichen, 52 Bits die Mantisse $m = 2^{-1} + m_2 2^{-2} + \cdots + m_{53} 2^{-53}$ und 11 Bits die Charakteristik $c = c_0 2^0 + \cdots + c_{10} 2^{10}$, mit $m_i, c_i \in \{0, 1\}$. Es gibt folgende spezielle Werte:

- Alle $c_i, m_i = 0 : x = \pm 0$
- Alle $m_i = 0, c_i = 1 : x = \pm\infty$
- Ein $m_i \neq 0$, alle $c_i = 1$: x = NaN (not a number)

Für c bleibt damit ein Bereich von $\{0, \dots, 2046\}$ beziehungsweise $c - 1022 \in \{-1022, \dots, 1024\}$. Damit gilt:

- $x_{max} \approx 2^{1024} \approx 1.8 \times 10^{308}, x_{min} = -x_{max}$

- $x_{posmin} = 2^{-1022} \approx 2.2 \times 10^{-308}, x_{negmax} = -x_{posmin}$

Ausgangsdaten $x \in \mathbb{R}$ einer numerischen Aufgabe und die Zwischenergebnisse einer Rechnung müssen durch Maschinenzahlen dargestellt werden. Für Zahlen des „zulässigen“ Bereichs $D = [x_{min}, x_{negmax}] \cup \{0\}[x_{posmin}, x_{max}]$ wird eine Rundungsoperation $\mathrm{rd} : D \to A$ verwendet, die

$$|x - \mathrm{rd}\, x| = \min_{y \in A} |x - y| \forall x \in D$$

erfüllt.

Beispiel 1.4 (Natürliche Rundung im IEEE-Format)

$$\mathrm{rd}(x) = \mathrm{sgn}(x) \cdot \begin{cases} 0, m_1, \ldots, m_{53} \cdot 2^e & m_{54} = 0 \\ (0, m_1, \ldots, m_{53} + 2^{-53}) \cdot 2^e & m_{54} = 1 \end{cases}$$

Rundungsfehler:

- absolut:
$$|x - \mathrm{rd}(x)| \leq \frac{1}{2} b^{-r} b^e$$
- relativ:
$$\left| \frac{x - \mathrm{rd}(x)}{x} \right| \leq \frac{1}{2} \frac{b^{-r} b^e}{|m| b^e} \leq \frac{1}{2} b^{-r+1}$$

Der relative Fehler ist für $x \in D \setminus \{0\}$ beschränkt durch die „Maschienengenauigkeit“

$$eps = \frac{1}{2} b^{-r+1}$$

Für $x \in D$ ist $\mathrm{rd}(x) = x(1 + \varepsilon), |(|\varepsilon) \leq eps$. Für das IEEE-Format (double)

$$eps = \frac{1}{2} 2^{-52} \approx 10^{-16}$$

Arithmetische Grundoperationen

$$* : \mathbb{R} \times \mathbb{R} \to \mathbb{R}, * \in \{x, -, +, /\}$$

werden auf dem Rechner ersetzt durch Maschinenoperationen:

$$\circledast : A \times A \to A$$

Dies ist normalerweise für $x, y \in A$ und $x * y \in D$ realisiert durch

$$x \circledast y := \mathrm{rd}(x * y) = (x * y)(1 + \varepsilon), |\varepsilon| \leq eps$$

Dazu werden die Operationen maschinenintern (unter Verwendung einer längeren Mantisse) ausgeführt, normalisiert und dann gerundet. Im Fall $x * y \notin D$ gibt es eine Fehlermeldung (overflow, underflow)

oder das Ergebnis NaN. Achtung: Das Assoziativ- und Distributivgesetz gilt dann nur näherungsweise. Im Allgemeinen ist für $x, y, z \in A$

$$(x \oplus y) \oplus z \neq x \oplus (y \oplus z)$$
$$(x \oplus y) \odot z \neq (x \odot z) \oplus (y \odot z)$$

Insbesondere gilt für $|y| \leq \frac{|x|}{b} eps$

$$x \oplus y = x$$

Damit ergibt sich eine alternative Charakterisierung der Maschienengenauigkeit: eps ist die kleinste positive Zahl in A, sodass $1 \oplus eps \neq 1$

1.2 Konditionierung numerischer Aufgaben

Eine numerische Aufgabe wird als **gut konditioniert** bezeichnet, wenn eine kleine Störung in den Eingangsdaten (Messfehler, Rundungsfehler) auch nur eine kleine Änderung der Ergebnisse zur Folge hat.

Beispiel 1.5 (Schnittpunkt von Geraden) Zwei Geraden, die sich (annähernd) rechtwinklig treffen sind gut konditioniert.
Zwei Geraden, die sich unter einem stumpfen, oder spitzen Winkel treffen sind schlecht konditioniert.

Beispiel 1.6 (Lineares Gleichungssystem)

$$\begin{pmatrix} 1 & 10^6 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \implies \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 & -10^6 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

$$b = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \implies x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$b = \begin{pmatrix} 1 \\ 10^{-3} \end{pmatrix} \implies x = \begin{pmatrix} -999 \\ 10^{-3} \end{pmatrix} \not\approx \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$\implies$ schlecht konditioniert.

Definition 1.7 Eine **numerische Aufgabe** berechnet aus Eingangsgrößen $x_j \in \mathbb{R}, j = 1, \ldots, m$ unter der funktionellen Vorschrift $f(x_1, \ldots, x_m), i = 1, \ldots, n$ Ausgangsgrößen $y_i = f_i(x_1, \ldots, x_m)$

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

Beispiel 1.8 (Lösung eines LGS) $Ay = x, f(x) = A^{-1}x$

Definition 1.9 Fehlerhafte Eingangsgrößen $x_i + \Delta x_i$ (Δx_i: Rundungsfehler, Maschienenfehler) ergeben fehlerhafte Resultate $y_i + \Delta y_i$. Wir bezeichnen $|\Delta y_i|$ als den absoluten Fehler und $\left|\frac{\Delta y_i}{y_i}\right|$ für $y_i \neq 0$ als den relativen Fehler.

1.2.1 Differentielle Fehleranalyse

Annahmen:

- kleine relative Datenfehler $|\Delta x_i| \ll |x_i|$
- f_i stetig partiell differenzierbar nach allen x_i

Dann gilt:

$$\begin{aligned} & y_i = f_i(x_i), y_i + \Delta y_i = f_i(x + \Delta x) \\ \implies & \Delta y_i = f_i(x + \Delta x) - f(x) \end{aligned}$$

Taylorentwicklung

$$= \sum_{j=1}^{m} \frac{\partial f_i}{\partial x_j} \Delta x_j + R_i^f(x, \Delta x)$$

mit einem Restglied R_i^f, das für $|\Delta x| = \max_{j=1,\dots,m} |\Delta x_j| \to 0$ schneller gegen 0 geht als $|\Delta x|$. Wenn f sogar zweimal stetig differenzierbar ist, gilt sogar, dass

$$\left| R_i^f(x, \Delta x) \right| \leq c |\Delta x|^2, c \in \mathbb{R}$$

Definition 1.10 (Landau-Notation) Seien $g, h : \mathbb{R}_+ \to \mathbb{R}, t \to 0^+$. Wir schreiben:

- $g(t) = \mathcal{O}(h(t)) : \iff \exists t_0, c \in \mathbb{R}_+ : \forall t \in (0, t_0] : |g(t)| \leq c|ht|$
- $gt = \sigma(ht) : \iff \exists t_0 \in \mathbb{R}_+, c : \mathbb{R}_+ \to \mathbb{R}, \lim_{t \to 0^+} c(t) = 0 : \forall t \in (0, t_0] : |g(t)| \leq c(t)|h(t)|$

Bemerkung 1.11

- Analoge Schreibweise für $t \to \infty$
- $\mathcal{O}$ und σ sind Symbole, keine Funktionen

$$\mathcal{O}\left(t^2\right) + \mathcal{O}\left(t^3\right) + \mathcal{O}\left(2t^2\right) = \mathcal{O}\left(t^2\right) \not\implies \mathcal{O}\left(t^3\right) + \in \sqcup^{\in} = 0$$

- $\sigma(t^n)$ ist stärker als $\mathcal{O}(t^n) : \sigma(t^n) + \mathcal{O}(t^n) = \mathcal{O}(t^n)$
- $\mathcal{O}\left(t^{n+1}\right)$ ist stärker als $\sigma(t^n)$: Wähle $c(t) = t$!

Beispiel 1.12 Ist $g(t)$ zweimal stetig differenzierbar, so gilt mit Taylor

$$g(t + \Delta t) = g(t) + \Delta t g'(t) + \frac{1}{2} \Delta t^2 g''(\tau), \tau \in [t, t + \Delta t]$$

$$\implies \frac{1}{\Delta t}(g(t + \Delta t) - g(t)) = g'(t) + \mathcal{O}(\Delta t)$$

Damit folgt dass Δy_i in erster Näherung, das heißt bis auf eine Größe der Ordnung $\mathcal{O}\left(|\Delta x|^2\right)$ gleich

$$\sum_{j=1}^{m} \frac{\partial f_i}{\partial x_j}(x)\Delta x_j$$

ist. Schreibweise

$$\Delta y_i \doteq \sum_{j=1}^{m} \frac{\partial f_i}{\partial x_j}(x)\Delta x_j$$

Für den komponentenweisen relativen Fehler gilt

$$\frac{\Delta y_i}{y_i} \doteq \sum_{j=1}^{m} \frac{\partial f_i}{\partial x_j}(x)\frac{\Delta x_j}{y_i} = \sum_{j=1}^{m} \underbrace{\frac{\partial f_i}{\partial x_j}(x)\frac{x_j}{f_i(x)}}_{=:k_{ij}(x)} \frac{\Delta x_j}{x_j}$$

Vernachlässigt haben wir dabei

$$\left|\frac{R_i^f(x_j, \Delta x)}{y_i}\right| = \mathcal{O}\left(\frac{|\Delta x|^2}{|y_i|}\right)$$

Diese Vernachlässigung ist nur zulässig falls

$$|\Delta x| = \sigma(|y_i|), i = 1, \ldots, n$$

damit

$$\mathcal{O}\left(\frac{|\Delta x|^2}{|y_i|}\right) = \sigma(|\Delta x|)$$

(stärker als $\mathcal{O}(|\Delta x|)$)

Definition 1.13 Die Größen $k_{ij}(x)$ heißen (relative) Konditionszahlen von f im Punkt x. Sie sind Maß dafür, wie sich kleine relative Fehler in den Ausgangsdaten x_j auf das Ergebnis y_i auswirken. Sprechweise:

- $|k_{ij}(x)| \gg 1$: Die Aufgabe $y = f(x)$ ist schlecht konditioniert
- sonst: Die Aufgabe $y = f(x)$ ist gut konditioniert
- $|k_{ij}(x)| < 1$: Fehlerdämpfung
- $|k_{ij}(x)| > 1$: Fehlerverstärkung.

Bemerkung 1.14 Man kann auch Störungen in f betrachten.

Beispiel 1.15 Implizit gegebene Aufgaben. Für $n = m$ sie y die gegebene Eingangsgröße und ein x mit $f(x) = y$ die Ausgabe (zum Beispiel: $f(x) = Ax + b$) Die differentielle Fehleranalyse auf der Umkehrfunktion $x = f^{-1}(y)$ liefert unter geeigneten Annahmen.

$$\frac{\Delta x_i}{x_i} \doteq \sum_{j=1}^{n} k_{ij}^{-1}(y)\frac{\Delta y_j}{y_j}, k_{ij}^{-1} = \frac{\partial f_i^{-1}}{\partial y_j}(y)\frac{y_j}{x_i}$$

Wir definieren die Matrizen

$$K^{-1}(y) = \left(k_{ij}^{-1}\right)_{i,j=1}^{n}, K(x) = (k_{ij}(x))_{i,j=1}^{n}$$

und betrachten deren Produkt:

$$\begin{aligned}(K^{-1}(y)K(x))_{ij} &= \sum_{l=1}^{n} k_{il}^{-1}(y)k_{lj}(x)\\ &= \sum_{l=1}^{n} \frac{\partial f_i^{-1}}{\partial y_l}(y)\frac{y_l}{x_i}\frac{\partial f_l}{\partial x_j}(x)\frac{x_j}{y_l}\\ &= \frac{x_j}{x_i}\sum_{l=1}^{n} \frac{\partial f_i^{-1}}{\partial y_l}\frac{\partial f_l}{\partial x_j} = \frac{x_j}{x_i}\frac{\mathrm{d}}{\mathrm{d}x_j}\left(f_i^{-1}(f(x))\right)\\ &= \frac{x_j}{x_i}\frac{\mathrm{d}x_i}{\mathrm{d}x_j} = \delta_{ij} = \begin{cases}1 & i = j\\ 0 & \text{sonst}\end{cases}\end{aligned}$$

K^{-1} ist gerade das Inverse von K.

Wiederholung: Numerische Aufgabe

$$f : x \in \mathbb{R}^m \mapsto y \in \mathbb{R}$$

Konditionszahlen:

$$\frac{\Delta y_i}{y_i} \doteq \sum_{j=1}^{m} k_{ij}(x)\frac{\Delta x_j}{x_j}$$

$$k_{ij}(x) = \frac{\partial f_i}{\partial x_j}(x)\frac{x_j}{f_i(x)}$$

1.2.2 Arithmetische Grundoperationen

Addition: $f(x_1, x_2) = x_1 + x_2, x_1, x_2 \in \mathbb{R} \setminus \{0\}$

$$k_{1j}(x) = \frac{\partial f}{\partial x_j}\frac{x_j}{f} = 1\frac{x_j}{x_1 + x_2} = \frac{1}{1 + \frac{x_{\bar{j}}}{x_j}}$$

$$\bar{j} = \begin{cases}2 & j = 1\\ 1 & j = 2\end{cases}$$

Die Addition ist schlecht konditioniert für $x_1 \approx -x_2$.

Definition 1.16 (Auslöschung) Unter Auslöschung versteht man den Verlust von Genauigkeit bei der Subtraktion von Zahlen gleichen Vorzeichens.

Beispiel 1.17 $b = 10, r = 4, s = 1$

$$\begin{aligned} x_1 &= 0.112\,587 \times 10^2 \quad \mathrm{rd}(x_1) &&= 0.1126 \times 10^2 \\ x_2 &= 0.112\,448 \times 10^2 \quad \mathrm{rd}(x_1) &&= 0.1124 \times 10^2 \\ x_1 + x_2 &= 0.225\,035 \times 10^2 \quad \mathrm{rd}(x_1) \oplus \mathrm{rd}(x_2) &&= 0.2250 \times 10^2 \\ x_1 - x_2 &= 0.129 \times 10^{-1} \quad \mathrm{rd}(x_1) \ominus \mathrm{rd}(x_2) &&= -0.2 \times 10^{-1} \end{aligned}$$ (Großer Fehler)

Multiplikation: $y = f(x_1, x_2) = x_1 x_2$

$$k_{1j}(x) = \frac{\partial f}{\partial x_j} \frac{x_j}{f} = x_j - \frac{x_j}{x_1 x_2} = 1$$

$\Longrightarrow$ gut konditioniert

Beispiel 1.18 (Lösungen quadratischer Gleichungen) Für $p, q \in \mathbb{R}$ betrachte:

$$\begin{aligned} 0 &= y^2 - py + q \\ y_{1,2} &= y_{1,2}(p, q) = \frac{p}{2} \pm \sqrt{\frac{p^2}{4} - q} \end{aligned}$$

nach Vieta $p = y_1 + y_2, q = y_1 \cdot y_2$

$$
\begin{aligned}
1 = \frac{\mathrm{d}p}{\mathrm{d}p} &= \frac{\partial y_1}{\partial p} + \frac{\partial y_2}{\partial p} \\
0 = \frac{\mathrm{d}q}{\mathrm{d}p} &= \frac{\partial y_1}{\partial p} y_2 + y_1 \frac{\partial y_2}{\partial p} \\
\implies (y_2 - y_1)\frac{\partial y_2}{\partial p} &= y_2 \\
\implies \frac{\partial y_2}{\partial p} &= \frac{y_2}{y_2 - y_1} \\
\implies \frac{\partial y_1}{\partial p} &= \frac{y_1}{y_1 - y_2} \\
0 = \frac{\mathrm{d}p}{\mathrm{d}q} &= \frac{\partial y_1}{\partial q} + \frac{\partial y_2}{\partial q} \\
1 = \frac{\mathrm{d}q}{\mathrm{d}q} &= \frac{\partial y_1}{\partial q} y_2 + y_1 \frac{\partial y_2}{\partial q} \\
\implies 1 &= (y_2 - y_1)\frac{\partial y_1}{\partial q} \\
\implies \frac{\partial y_1}{\partial q} &= \frac{1}{y_2 - y_1} \\
\implies \frac{\partial y_2}{\partial q} &= -\frac{1}{y_2 - y_1} \\
k_{11}(x) = \frac{\partial y_1}{\partial p}\frac{p}{y_1} &= \frac{y_1}{y_1 - y_2}\frac{y_1 + y_2}{y_1} = \frac{1 + y_2/y_1}{1 - y_2/y_1} \\
k_{12}(x) = \frac{\partial y_1}{\partial q}\frac{q}{y_1} &= \frac{1}{y_2 - y_1}\frac{y_1 y_2}{y_1} = \frac{1}{1 - y_1/y_2}
\end{aligned}
$$

Analog für k_{21}, k_{22}
Die Berechnung von y_1, y_2 ist schlecht konditioniert $y_1 \approx y_2$.
Konkretes Beispiel: $p = 4, q = 33.999, y_{1,2} = 2 \pm 10 \times 10^{-1}$

$$k_{12} = \frac{y_2}{y_2 - y_1} = \frac{2 - 10^{-2}}{-2 \times 10^{-2}} = -99.5$$

$\implies$ 100-fache Fehlerverstärkung.

1.3 Stabilität numerischer Algorithmen

Gegeben: Numerische Aufgabe $f : x \in \mathbb{R}^m \mapsto y \in \mathbb{R}^n$

Definition 1.19 (Verfahren / Algorithmus) Unter einem Verfahren / Algorithmus zur (gegebenenfalls näherungsweise) Berechnung von y aus x verstehen wir eine endliche Folge von elementaren Abbildungen $\varphi^{(k)}$, die durch sukzessiv Anwendung einen Näherungswert $\tilde{y}$ zu y liefern.

$$x = x^{(0)} \mapsto \varphi^{(1)}\left(x^{(0)}\right) = x^{(1)} \mapsto \ldots \mapsto \varphi^{(k)}\left(x^{(k-1)}\right) \mapsto \tilde{y} \to y$$

Im einfachsten Fall sind die $\varphi^{(i)}$ arithmetische Grundoperationen. Bei der Durchführung des Algorithmus auf dem Rechner treten in jedem Schritt Fehler auf (Rundungsfehler, Auswertungsfehler, …), die sich akkumulieren können.

Definition 1.20 (Algorithmus) Ein Algorithmus heißt stabil, wenn die im Verlauf der Rechnung akkumulierten Fehler den durch die Konditionierung der Aufgabe $y = f(x)$ bedingten unvermeidbaren Problemfehler nicht übersteigen.

Beispiel 1.21 (Lösung quadratischer Gleichungen) Annahme: $0 \neq q < p^2/4$
Für $\left|\frac{y_1}{y_2}\right| \gg 1$, das heißt $q \ll \frac{p^2}{4}$, ist die Aufgabe gut konditioniert. Algorithmus: $u = p^2/4, v = u - q, w = \sqrt{v}$.
Im Fall $p < 0$ wird zur Vermeidung von Auslöschung zunächst $\tilde{y}_2 = p/2 - w$ berechnet.
Fehlerfortpflanzung:

$$w = \sqrt{u-q} \begin{cases} \approx \frac{|p|}{2} & q > 0 \\ > \frac{|p|}{2} & q < 0 \end{cases}$$

$$\begin{aligned} \frac{\Delta y_2}{y_2} &\overset{\cdot}{\leq} \left|\frac{\frac{1}{2}p}{\frac{p}{2} - w}\right| \left|\frac{\Delta p}{p}\right| + \left|\frac{-w}{\frac{p}{2} - w}\right| \left|\frac{\Delta w}{w}\right| \\ &= \underbrace{\left|\frac{1}{1 - \frac{2w}{p}}\right|}_{\leq \frac{1}{2}} \left|\frac{\Delta p}{p}\right| + \underbrace{\left|\frac{1}{1 - \frac{p}{2w}}\right|}_{<1} \left|\frac{\Delta w}{w}\right| \end{aligned}$$

Die zweite Wurzel kann so bestimmt werden:

$$A : \tilde{y}_1 = \frac{p}{2} + w, \quad B : \tilde{y}_1 = \frac{q}{\tilde{y}_2}$$

Für $|q| \ll \frac{p^2}{4}$ ist $w \approx \frac{|p|}{2} \implies$ Auslöschung in Variante A

$$\left|\frac{\Delta y_1}{y_1}\right| \overset{\cdot}{=} \underbrace{\frac{1}{1 + \frac{2w}{p}}}_{\gg 1} \frac{\Delta p}{p} + \underbrace{\frac{1}{1 + \frac{p}{2w}}}_{\gg 1} \frac{\Delta w}{w}$$

$\implies$ Variante A ist instabil. Variante B ist stabil:

$$\left|\frac{\Delta y_1}{y_1}\right| \overset{\cdot}{\leq} \underbrace{\left|\frac{\Delta q}{q}\right|}_{\leq eps} + \underbrace{\left|\frac{\Delta y_2}{y_2}\right|}_{\approx eps}$$

Regel: Bei der Lösung quadratischer Gleichungen sollten nicht beide Wurzeln aus der Lösungsformel berechnet werden.
Konkretes Beispiel: $p = -4, q = 0.01$ (vierstellige Rechnung)

$$u = 4, v = 3.99, w = 1.9974948\ldots, \tilde{y}_2 = -3.997(4981\ldots)$$

$$\tilde{y}_1 = \begin{cases} \text{exakt:} & -0.9925915\ldots \\ A: & -0.003000 \quad \text{(rel. Fehler: 20\%)} \\ B: & -0.002502 \quad \text{(rel. Fehler: } 1.7 \times 10^{-4}\text{)} \end{cases}$$

Auswertung arithmetischer Ausdrücke
Vorwärtsrundungsfehleranalyse: Akkumulation des Rundungsfehlers ausgehend von Startwert.

Beispiel 1.22 $y = f(x_1, x_2) = x_1^2 - x_2^2 = (x_1 - x_2)(x_1 + x_2)$ Konditionierung:

$$\begin{aligned}\left|\frac{\Delta y}{y}\right| &\dot{\leq} \sum_{i=1}^{2}\left|\frac{\partial f}{\partial x_i}\frac{x_i}{f}\right|\left|\frac{\Delta x_i}{x_i}\right| \\ &= \left|2x_1\frac{x_1}{x_1^2 - x_2^2}\right|\left|\frac{\Delta x_1}{x_1}\right| + \left|-2x_2\frac{x_2}{x_1^2 - x_2^2}\right|\left|\frac{\Delta x_2}{x_2}\right| \\ &\leq 2\frac{x_1^2 + x_2^2}{|x_1^2 - x_2^2|}eps = 2\left|\frac{\left(\frac{x_1}{x_2}\right)^2 + 1}{\left(\frac{x_1}{x_2}\right)^2 - 1}\right|eps\end{aligned}$$

$\Longrightarrow$ schlecht konditioniert für $\left|\frac{x_1}{x_2}\right| \approx 1$

Algorithmus A	Algorithmus B
$u = x_1 \odot x_1$	$u = x_1 \oplus x_1$
$v = x_2 \odot x_2$	$v = x_1 \ominus x_2$
$\tilde{q} = u \ominus v$	$\tilde{q} = u \odot v$

Sei $x_1, x_2 \in A$. Für Maschinenoperationen $\circledast$ und $a, b \in A$ gilt

$$a \circledast b = (a * b)(1 + \varepsilon), |(|\varepsilon) \leq eps.$$

Algorithmus A:

$$\begin{aligned}u &= x_1^2(1 + \varepsilon_1), v = x_2^2(1 + \varepsilon_2) \\ \tilde{y} &= \left(x_1^2(1 + \varepsilon_1) - x_2^2(1 + \varepsilon_2)\right)(1 + \varepsilon_3) \\ &= \underbrace{x_1^2 - x_2^2}_{y} + x_1^2\varepsilon_1 - x_2^2\varepsilon_2 + \underbrace{\left(x_1^2 - x_2^2\right)}_{y}\varepsilon_3, |\varepsilon| \leq eps\end{aligned}$$

$$\Longrightarrow \left|\frac{\Delta y}{y}\right| \dot{\leq} eps\frac{x_1^2 + x_2^2 + \left|x_1^2 - x_2^2\right|}{\left|x_1^2 - x_2^2\right|} = eps\left(1 + \frac{\left(\frac{x_1}{x_2}\right)^2 + 1}{\left(\frac{x_1}{x_2}\right)^2 - 1}\right)$$