Numerische Mathematik für das Lehramt

Skriptum zur Vorlesung im SS 2009 †

PD Dr. Markus Neher
Universität Karlsruhe (TH)
Forschungsuniversität · gegründet 1825
Institut für Angewandte und Numerische Mathematik

10. Juli 2009

Inhaltsverzeichnis

Kapitel 8

Gleitpunktrechnung, Kondition, Stabilität

8.1 Gleitpunktzahlen

Numerische Berechnungen werden in der Regel nicht mit Variablen, sondern mit Zahlen durchgeführt. Die Darstellung von Zahlen ist dabei von entscheidender Bedeutung. Bereits die Konstruktion von Algorithmen kann vom gewählten Zahlenformat abhängen. Beispielsweise ist der heute gebräuchliche Algorithmus zur Division von Zahlen im römischen Zahlensystem praktisch nicht durchführbar. Noch stärker werden die Rechengenauigkeit und die Rechengeschwindigkeit durch das Zahlenformat beeinflusst.

Auch die Anschaulichkeit von Zahlen hängt wesentlich von der gewählten Darstellung ab. So wird $\frac{355}{113}$ durch 3.1416 nur approximiert, aber unter der Dezimalnäherung können sich die meisten Menschen die Zahl besser vorstellen. Auch im Fall großer Zahlen wie der Avogadro-Konstante

$$N_A = 602214175141592653589793 \approx 6.022 \cdot 10^{23}$$

ist die Angabe des exakten Werts wenig hilfreich. Die wesentlichen Informationen über den Wert der Zahl lassen sich aus der Dezimalnäherung einfacher und schneller ablesen.

In Anwendungen sind gemessene oder berechnete Größen üblicherweise nur ungefähr bekannt. Dies gilt sogar für mathematische Konstanten wie π oder mathematische Symbole wie $\sqrt{2}$, die lediglich Namen von Zahlen darstellen, aber nichts über den Wert der repräsentierten Zahl aussagen. Daher ist es üblich, bei praktischen Rechnungen sogenannte Gleitpunktzahlen der Bauart

$$\pm m \cdot b^e \tag{8.1}$$

mit der Mantisse m, der Basis b und dem Exponenten e zu verwenden. Dabei ist die Basis b des b-adischen Zahlensystems eine natürliche Zahl größer als 1, der Exponent e eine ganze Zahl und die Mantisse m eine rationale Zahl mit endlicher b-adischer Darstellung. Beispiele solcher Zahlen sind

$$3.1416 \cdot 10^0, \quad -44.123 \cdot 5^3, \quad 0.0001101 \cdot 2^{-1011}.$$

Die angegebene Gleitpunktdarstellung (8.1) einer Zahl ist nicht eindeutig. Beispielsweise gilt

$$3.1416 \cdot 10^0 = 3141600.00 \cdot 10^{-6} = 0.00000000000000000000031416 \cdot 10^{22}.$$

Beim Rechnen ist diese Mehrdeutigkeit unpraktisch. Deshalb fordern wir in (8.1) zusätzlich, dass – außer für die Zahl Null – die Mantisse im Intervall $[1, b)$ liegt. Durch eine Anpassung des

Exponenten (bei der im Gegenzug der Dezimalpunkt gleitet) existiert für jede Gleitpunktzahl $z \neq 0$ eine eindeutig bestimmte normalisierte Darstellung

$$z = \pm m_1.m_2 m_3 \dots m_l \cdot b^e$$

mit $m_1, m_2, \dots, m_l \in \{0, 1, \dots, b-1\}$, $m_1 \neq 0$, $e \in \mathbb{Z}$. Die Menge aller normalisierten Gleitpunktzahlen zur Basis b mit fester Mantissenlänge $l \geq 1$ und Exponentenbereich $e_{\min} \leq e \leq e_{\max}$, vereinigt mit der Zahl Null, bildet das Gleitpunktsystem $S_{\text{norm}}(b, l, e_{\min}, e_{\max}) \subset \mathbb{Q}$.

Beispiel 8.1 Das normalisierte dezimale Gleitpunktsystem $S_{\text{norm}}(10, 4, 0, 9)$ mit vier dezimalen Mantissenstellen und einer Dezimalstelle für den Exponenten umfasst die folgenden Zahlen:

$$\begin{aligned}
&0,\\
&+1.000 \cdot 10^0,\ -1.000 \cdot 10^0,\ +1.001 \cdot 10^0,\ -1.001 \cdot 10^0, \dots, +9.999 \cdot 10^0,\ -9.999 \cdot 10^0,\\
&+1.000 \cdot 10^1,\ -1.000 \cdot 10^1,\ +1.001 \cdot 10^1,\ -1.001 \cdot 10^1, \dots, +9.999 \cdot 10^1,\ -9.999 \cdot 10^1,\\
&\vdots\\
&+1.000 \cdot 10^9,\ -1.000 \cdot 10^9,\ +1.001 \cdot 10^9,\ -1.001 \cdot 10^9, \dots, +9.999 \cdot 10^9,\ -9.999 \cdot 10^9.
\end{aligned}$$

Die kleinste darstellbare positive normalisierte Gleitpunktzahl heißt mininorm, die größte heißt maxreal. In $S_{\text{norm}}(10, 4, 0, 9)$ gilt:

$$\text{mininorm} = 1.000 \cdot 10^0, \quad \text{maxreal} = 9.999 \cdot 10^9$$

Man beachte, dass die Gleitpunktzahlen in $S_{\text{norm}}(10, 4, 0, 9)$ nicht gleichabständig verteilt sind. Der Abstand der Nachbarzahlen $9.998 \cdot 10^9$ und $9.999 \cdot 10^9$ beträgt 10^6, der Abstand der Nachbarzahlen $1.000 \cdot 10^0$ und $1.001 \cdot 10^0$ nur 10^{-3}. Normalisierte Gleitpunktzahlen besitzen aber die wertvolle Eigenschaft, dass der relative Fehler zweier benachbarter Gleitpunktzahlen zueinander ungefähr konstant ist.

Wir diskutieren nun, wie Zahlen möglichst effizient auf einem Computer gespeichert werden können. Halten wir die Basis b und die Mantissenlänge l fest, müssen die folgenden Informationen zu einer Gleitpunktzahl gespeichert werden:

1. Das Vorzeichen der Zahl,
2. die Mantisse,
3. das Vorzeichen des Exponenten,
4. die Ziffern des Exponenten.

Mit einem kleinen Trick kann man die separate Speicherung des Vorzeichens des Exponenten umgehen. Man speichert auf dem Rechner in der Praxis nur nichtnegative Exponenten (mit l_e Stellen zur Basis b), interpretiert die gespeicherte Zahl aber so, dass vom Exponenten ein fester Basiswert subtrahiert wird.

Beispiel 8.2 Gegeben sei das dezimale normalisierte Gleitpunktsystem $S_{\text{norm}}(10, 4, -5, 4)$ mit vier dezimalen Mantissenstellen und einer Dezimalstelle für den Exponenten nach dem folgenden Schema:

$\pm$	e	m_1	m_2	m_3	m_4

$e, m_1, m_2, m_3, m_4 \in \{0, 1, \dots, 9\}$.

Vom gespeicherten Wert e wird bei der Umrechnung in eine Zahl der feste Basiswert 5 abgezogen. Normalisierte Gleitpunktzahlen besitzen also in $S_{\text{norm}}(10,4,-5,4)$ die Darstellung

$$\pm m_1.m_2m_3m_4 \cdot 10^{e-5} \quad (m_1 \neq 0,\ 0 \leq e \leq 9).$$

Die kleinste darstellbare positive normalisierte Gleitpunktzahl ist

$$\text{mininorm} = 1.000 \cdot 10^{-5}$$

(gespeichert wird $e = 0$), die größte ist

$$\text{maxreal} = 9.999 \cdot 10^{4}$$

(gespeichert wird $e = 9$).

Mit einem zweiten Trick kann man die Menge der darstellbaren Gleitpunktzahlen vergößern, ohne zusätzlichen Speicherplatz bereitstellen zu müssen. Besonders vorteilhaft ist es, dass dabei die Lücke zwischen der Null und der kleinsten positiven Gleitpunktzahl verkleinert wird. Dazu verzichtet man bei Zahlen, die mit dem kleinsten Exponentenwert gespeichert sind, auf die Normalisierungsbedingung $m_1 \geq 1$ und interpretiert die gespeicherten Ziffern der Mantisse als Nachkommastellen einer nicht normalisierten Gleitpunktzahl. Damit im Gleitpunktsystem keine Lücke entsteht, muss man dann auch die Exponentenanpassung geringfügig ändern.

Beispiel 8.3 Wir betrachten das dezimale Gleitpunktsystem $\mathcal{T} := S(10,4,-5,4)$ mit vier dezimalen Mantissenstellen und einer Dezimalstelle für den Exponenten. Für $1 \leq e \leq 9$ interpretieren wir die gespeicherte Zahl z wie in $S_{\text{norm}}(10,4,-5,4)$:

$$z = \pm m_1.m_2m_3m_4 \cdot 10^{e-5} \quad (m_1 \neq 0,\ 1 \leq e \leq 9).$$

Die kleinste darstellbare positive normalisierte Gleitpunktzahl ist nun

$$\text{mininorm} = 1.000 \cdot 10^{-4}$$

(gespeichert wird $e = 1$), die größte ist wie in $S_{\text{norm}}(10,4,-5,4)$ die Zahl

$$\text{maxreal} = 9.999 \cdot 10^{4}$$

(gespeichert wird $e = 9$).

Für $e = 0$ stellt der gespeicherte Wert die nicht normalisierte Gleitpunktzahl

$$z = \pm 0.m_1m_2m_3m_4 \cdot 10^{e-4} \quad (e = 0)$$

dar. Die Zahl Null wird durch $e = m_1 = m_2 = m_3 = m_4 = 0$ dargestellt. Die kleinste darstellbare positive (nicht normalisierte) Gleitpunktzahl ist

$$\text{minreal} := 0.0001 \cdot 10^{-4} = 1.0 \cdot 10^{-8},$$

die größte nicht normalisierte Gleitpunktzahl $0.9999 \cdot 10^{-4}$ schließt an mininorm an. Im Vergleich zu $S_{\text{norm}}(10,4,-5,4)$ ist die kleinste positive Gleitpunktzahl in $\mathcal{T}$ um drei Zehnerpotenzen kleiner.

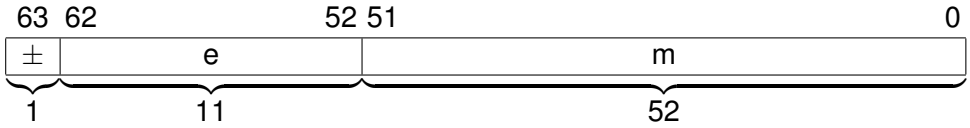
Man beachte insbesondere, dass bei der Neuinterpretation der Zahlen mit Exponentenwert $e = 0$ in $\mathcal{T}$ im Vergleich zu $S_{\text{norm}}(10,4,-5,4)$ keine Zahl verloren geht. Die gespeicherten Mantissenwerte $m_1,\ m_2,\ m_3,\ m_4$ repräsentieren in $S_{\text{norm}}(10,4,-5,4)$ für $e = 0$ die Zahl

$$m_1.m_2m_3m_4 \cdot 10^{-5},$$

in $\mathcal{T}$ die identische Zahl

$$0.m_1m_2m_3m_4 \cdot 10^{-4}.$$

Auf modernen Computern werden üblicherweise Gleitpunktzahlen zur Basis $b = 2$ (Dualzahlen, Binärzahlen) entsprechend dem IEEE 754 Double-Standard verwendet:



Eine solche Gleitpunktzahl besteht aus 64 Bits (1 Bit = 0 oder 1):

- dem Vorzeichenbit (Bit 63, “+” oder “-”)
- 11 Exponentenbits (Bits 52-62)
- 52 Mantissenbits (Bits 0-51)

Zur Umrechnung der gespeicherten Bitfolge in eine Zahl werden die folgenden Konventionen verwendet:

- Der Exponent $e = 2^{11} - 1 = 2047$ wird nur für Ausnahmefälle ($m = 0$: signed infinity; $m \neq 0$: NaN = not a number) benutzt.
- Für $1 \leq e \leq 2046$ werden vom Exponenten 1023 subtrahiert; für $e = 0$ werden 1022 subtrahiert.
- Für $1 \leq e \leq 2046$ werden die Mantissenbits als Nachkommastellen der Zahl
$$1.m \cdot 2^{(e-1023)}$$
interpretiert (normalisierte Gleitpunktzahl, bei der die führende 1 nicht gespeichert wird).
- $e = 0$, $m = 0$ stellt die Zahl 0 dar.
- $e = 0$, $m \neq 0$ stellt die Zahl
$$0.m \cdot 2^{-1022}$$
dar (nicht normalisierte Gleitpunktzahl im Unterlaufbereich).

Die auf dem Rechner darstellbaren Zahlen heißen Maschinenzahlen.

Somit ergibt sich der folgende Bereich darstellbarer Zahlen:

- Größte darstellbare positive normalisierte Gleitpunktzahl (maxreal):
$$2^{1024} - 2^{1023-52} \approx 1.798 \cdot 10^{308}.$$
- Kleinste darstellbare positive normalisierte Gleitpunktzahl (mininorm):
$$2^{-1022} \approx 2.225 \cdot 10^{-308}.$$
- Kleinste darstellbare positive nicht normalisierte Gleitpunktzahl (minreal):
$$2^{-1022-52} \approx 4.941 \cdot 10^{-324}.$$

Das IEEE 754 Double-Format ist also gegeben durch

$$\text{IEEE 754 Double} = S(2, 53, -1022, 1024).$$

Für die praktischen Beispiele benutzen wir im Folgenden das wesentlich übersichtlichere normalisierte Gleitpunktsystem $\mathcal{S} := S_{\text{norm}}(10, 4, -9, 9)$.

8.2 Gleitpunktarithmetik

Ein endliches Gleitpunktsystem enthält nur endlich viele Zahlen. Nicht exakt darstellbare Zahlen werden zur Speicherung auf dem Computer gerundet. Dies gilt auch für die Ergebnisse arithmetischer Verknüpfungen von Maschinenzahlen, sofern sie nicht exakt darstellbar sind.

Bezeichnet $\mathcal{R}$ die Menge der Maschinenzahlen, dann ist jede Rundung $\diamond$ eine Abbildung von $\mathbb{R}$ nach $\mathcal{R}$. Um eine Rundungsfehleranlyse zu ermöglichen, setzen wir für alle Rechnungen mit Gleitpunktzahlen die Existenz einer Konstante eps (der Maschinengenauigkeit) mit der folgenden Eigenschaft voraus: Für jedes x mit $|x| \in [\text{mininorm}, \text{maxreal}]$ gibt es ein ε mit

$$\diamond x = x(1+\varepsilon), \quad |\varepsilon| < \text{eps}. \tag{8.2}$$

Für den relativen Rundungsfehler, der bei der Speicherung von x auf dem Rechner entsteht, gilt unter diesen Annahmen

$$\frac{|\diamond x - x|}{|x|} \leq \text{eps}.$$

Wir verwenden im Folgenden ausschließlich die Rundung $\square : \mathbb{R} \to \mathcal{R}$, die $x \in \mathbb{R}$ auf die nächstgelegene Maschinenzahl $\square x \in \mathcal{R}$ abbildet. Gibt es zwei Maschinenzahlen, die den gleichen Abstand von x besitzen, wird auf die betragsgrößere Zahl gerundet. Für $\square$ ist (8.2) mit

$$\text{eps} := \frac{1}{2}\min\{x \geq \text{minreal} : 1 + x \in \mathcal{R}\}.$$

erfüllt. Im IEEE 754-Standard gilt beispielsweise $\text{eps} = 2^{-53}$, in $\mathcal{S}$ besitzt eps den Wert $\frac{1}{2} \cdot 10^{-3}$.

Der IEEE 754-Standard legt nicht nur ein Zahlenformat fest. Er verlangt auch, dass das Ergebnis jeder Elementaroperation auf dem Rechner mit Hilfe von $\square$ zu runden ist (man beachte, dass die Summe, das Produkt oder der Quotient zweier Maschinenzahlen nicht wieder eine Maschinenzahl sein müssen). Sind x und y Maschinenzahlen und bezeichnet $\boxdot$ die Realisierung der arithmetischen Grundoperation $\circ$ auf dem Rechner, muss für $\circ \in \{+, -, \cdot, /\}$ gelten:

$$\underbrace{x \boxdot y}_{\in \mathcal{R}} := \square(\underbrace{x \circ y}_{\in \mathbb{R}}).$$

Arithmetische Grundoperationen können auf dem Rechner dann folgendermaßen beschrieben werden:

$$x \boxdot y = (x \circ y)(1+\varepsilon) \quad \text{mit} \quad |\varepsilon| \leq \text{eps}. \tag{8.3}$$

Bei der Stabilitätsanalyse von Algorithmen nehmen wir im Folgenden an, dass eine ähnliche Eigenschaft auch für die üblichen auf dem Rechner verfügbaren Standardfunktionen (Wurzel, Exponentialfunktion, trigonometrische Funktionen, ...) gilt. Ist f eine Standardfunktion, $\tilde{f}$ die auf dem Rechner implementierte Prozedur zur Berechnung von f und x eine Maschinenzahl, für die $f(x)$ im Intervall $[\text{mininorm}, \text{maxreal}]$ liegt, soll die Genauigkeitsbedingung

$$\tilde{f}(x) = f(x)(1+\varepsilon) \quad \text{mit} \quad |\varepsilon| \leq \text{eps}$$

erfüllt sein.

8.2.1 Fehlerfortpflanzung bei den arithmetischen Grundoperationen

Wir haben bereits im Abschnitt 1.3 bemerkt, dass in Gleitpunktarithmetik sowohl das Assoziativgesetz bezüglich Addition und Multiplikation als auch das Distributivgesetz verletzt werden

können. In diesem Abschnitt untersuchen wir, wie sich vorhandene Fehler (in den Maschinenzahlen $\tilde{x}$ und $\tilde{y}$) in den arithmetischen Grundoperationen (zwischen $\tilde{x}$ und $\tilde{y}$) fortpflanzen. In der Diskussion der Genauigkeit von Gleitpunktberechnungen ist diese Frage von großer praktischer Bedeutung. Die Forderung (8.3) regelt nur, wie groß der bei einer einzelnen Gleitpunktoperation neu entstehende Fehler höchstens sein darf, macht aber keine Aussage darüber, ob und wie in den beteiligten Maschinenzahlen bereits enthaltene (z.B. durch Rundung bei der Abspeicherung oder in früheren Rechenoperationen entstandene) Fehler verstärkt werden.

Gegeben seien zwei Zahlen $x, y \in \mathbb{R}$ sowie ihre Gleitpunktnäherungen

$$\tilde{x} := \square x = x(1+\varepsilon_x), \quad \tilde{y} := \square y = y(1+\varepsilon_y), \quad |\varepsilon_x|, |\varepsilon_y| \leq \text{eps}.$$

Dann gilt für die arithmetischen Verknüpfungen von $\tilde{x}$ und $\tilde{y}$:

1. Multiplikation:

   $$\tilde{x}\tilde{y} = x(1+\varepsilon_x)y(1+\varepsilon_y) = xy(1+\varepsilon_x+\varepsilon_y+\varepsilon_x\varepsilon_y) \approx xy(1+\varepsilon_x+\varepsilon_y) =: xy(1+\varepsilon_{xy})$$

   mit

   $$\varepsilon_{xy} \approx \varepsilon_x + \varepsilon_y, \quad |\varepsilon_{xy}| \lesssim 2\,\text{eps}.$$

   Die übliche (und intuitive, aber mathematisch eigentlich unsinnige) Notation $\lesssim$ bedeutet hier, dass die Relation $\leq$ höchstens in der Größenordnung von eps^2 verletzt wird. Die relativen Fehler von x und y werden bei der Multiplikation also ungefähr addiert. Weiter gilt für das gerundete Ergebnis der Multiplikation

   $$\square(\tilde{x}\tilde{y}) = xy(1+\varepsilon_{xy})(1+\varepsilon) \approx xy(1+\varepsilon_x+\varepsilon_y+\varepsilon) =: xy(1+\varepsilon_{\square(xy)})$$

   mit $\left|\varepsilon_{\square(xy)}\right| \lesssim 3\,\text{eps}$.

2. Division:

   $$\frac{\tilde{x}}{\tilde{y}} = \frac{x(1+\varepsilon_x)}{y(1+\varepsilon_y)} = \frac{x}{y}(1+\varepsilon_x)(1-\varepsilon_y+\varepsilon_y^2-+\dots) \approx \frac{x}{y}(1+\varepsilon_x-\varepsilon_y) =: xy(1+\varepsilon_{\frac{x}{y}})$$

   mit $\left|\varepsilon_{\frac{x}{y}}\right| \lesssim 2\,\text{eps}$.

   Analog erhält man

   $$\square(\frac{\tilde{x}}{\tilde{y}}) = xy(1+\varepsilon_{\square(\frac{x}{y})}), \quad \left|\varepsilon_{\square(\frac{x}{y})}\right| \lesssim 3\,\text{eps}.$$

3. Addition:

   $$\begin{aligned}\tilde{x}+\tilde{y} &= x(1+\varepsilon_x)+y(1+\varepsilon_y) = x+y+x\varepsilon_x+y\varepsilon_y \\ &= (x+y)(1+\frac{x}{x+y}\varepsilon_x+\frac{y}{x+y}\varepsilon_y) =: (x+y)(1+\varepsilon_{x+y}).\end{aligned}$$

   Haben x und y gleiche Vorzeichen, dann gilt

   $$\left|\frac{x}{x+y}\right| \leq 1, \quad \left|\frac{y}{x+y}\right| \leq 1, \quad |\varepsilon_{x+y}| \leq 2\,\text{eps}$$

   (die relativen Fehler von x und y werden höchstens addiert). Eine problematische Situation tritt jedoch ein, wenn x und y verschiedene Vorzeichen besitzen. Im Fall $x \approx -y$ kann ε_{x+y} beliebig groß werden. Die Subtraktion ungefähr gleich großer Zahlen bezeichnet man als Auslöschung, weil sich die führenden Stellen in den Mantissen der Gleitpunktzahlen $\tilde{x}$ und $\tilde{y}$ gegenseitig auslöschen. Bei der Exponentenanpassung zur Normalisierung der Gleitpunktdifferenz wird die Mantisse von hinten mit Nullen ausgefüllt. Im gleichen Maß geht (relative) Genauigkeit verloren. Die Subtraktion annähernd gleich großer gerundeter Zahlen sollte daher möglichst vermieden werden.

Die beschriebene Fehlerfortpflanzung kann sich bereits in einfachsten Berechnungen negativ bemerkbar machen. Wir illustrieren dies im folgenden Beispiel.

Beispiel 8.4 Einfluss von Rundungsfehlern auf das Ergebnis einer Gleitpunktrechnung.

1. Nullstellen quadratischer Gleichungen: $x^2 - 2px + q = 0 \iff x_{1/2} = p \pm \sqrt{p^2 - q}$.

   Im Fall $p^2 \gg q$ gilt $\sqrt{p^2 - q} \approx |p|$, so dass bei der Berechnung der betragskleineren Nullstelle Auslöschung auftritt.

   Zahlenbeispiel: Berechnung der Nullstellen von $x^2 - 40x + 14 = 0$ in $\mathcal{S}$. Korrekte Ziffern werden schwarz, fehlerhafte Ziffern blau dargestellt. In $\mathcal{S}$ gilt:

   $$\square(p^2 - q) = \square(4.000 \cdot 10^2 \boxminus 1.400 \cdot 10^1) = 3.860 \cdot 10^2,$$
   $$\square\sqrt{3.860 \cdot 10^2} = \square 19.646\ldots = 1.965 \cdot 10^1,$$
   $$x_1 = 2.000 \cdot 10^1 \boxplus 1.965 \cdot 10^1 = 3.965 \cdot 10^1,$$
   $$x_2 = 2.000 \cdot 10^1 \boxminus 1.965 \cdot 10^1 = \square 0.35 = 3.500 \cdot 10^{-1}.$$

   Die zwei letzten Mantissenstellen in x_2 wurden bei der Normalisierung mit Nullen aufgefüllt. Die auf vier Dezimalstellen gerundeten Nullstellen des Polynoms sind

   $$x_1 = \square 39.646\ldots = 3.965 \cdot 10^1, \quad x_2 = \square 0.35311\ldots = 3.531 \cdot 10^{-1}.$$

   Obwohl sich die Größenordnungen von p^2 und q nicht einmal besonders stark unterscheiden, tritt bei der Berechnung von x_2 bereits ein relativer Fehler von ca. 0.9% auf. Dieser Fehler wächst umso stärker an, je größer p^2 im Vergleich zu q ist.

2. Numerische Differentiation: $f'(x) \approx \dfrac{f(x+h) - f(x)}{h}$.

   Bei der numerischen Differentiation treten zwei Fehler auf: der analytische Fehler in der Größenordnung von h (Satz von Taylor) und die Auslöschung durch Differenzbildung benachbarter Funktionswerte. Um den analytischen Fehler zu minimieren, sollte ein kleiner Wert für h gewählt werden. Für kleine Werte von h vergrößert sich jedoch der Effekt der Auslöschung.

   Zahlenbeispiel: $f(x) = \sqrt{x},\ x = 1\colon\ f'(1) = \dfrac{1}{2} = 5.000 \cdot 10^{-1}$. In $\mathcal{S}$ erhält man:

   $h = 1$: $\square(\sqrt{2} \boxminus 1) \boxslash 1 = (\square 1.4142\ldots \boxminus 1) = 1.414 - 1 = 4.140 \cdot 10^{-1}$
   (der analytische Fehler ist sehr groß),

   $h = 0.1$: $\square(\sqrt{1.1} \boxminus 1) \boxslash 0.1 = (\square 1.0488\ldots \boxminus 1) \boxslash 0.1 = (1.049 \boxminus 1) \boxslash 0.1$
   $= 4.900 \cdot 10^{-1}$ (der analytische Fehler ist groß),

   $h = 0.01$: $\square(\sqrt{1.01} \boxminus 1) \boxslash 0.01 = (\square 1.00498\ldots \boxminus 1) \boxslash 0.01$
   $= (1.005 \boxminus 1) \boxslash 0.01 = 5.000 \cdot 10^{-1}$
   (der analytische Fehler ist klein, die Auslöschung ist noch moderat),

   $h = 0.001$: $\square(\sqrt{1.001} \boxminus 1) \boxslash 0.001 = (\square 1.00049\ldots \boxminus 1) \boxslash 0.01$
   $= (1.000 \boxminus 1) \boxslash 0.001 = 0$
   (der analytische Fehler ist sehr klein, aber die Auslöschung wirkt katastrophal).

   Man kann zeigen, dass bei der numerischen Differentiation die Wahl $h = 2\sqrt{\text{eps}/f''(x)}$ optimal ist. Für größere Werte von h überwiegt der analytische Fehler, für kleinere Werte von h dominiert die Auslöschung.

Auslöschung kann man mit Mitteln der Rechnerarithmetik bekämpfen, indem man auf dem Computer Register bereitstellt, in denen Zwischenergebnisse von Berechnungen mit höherer Genauigkeit (längeren Mantissen) gespeichert werden. Derartige Register sind allerdings teuer und stehen dem Anwender nicht auf jedem Rechner zur Verfügung. Manchmal ist es aber möglich, Auslöschung dadurch zu vermeiden, dass man den auszuwertenden Ausdruck geeignet umformt.

Beispiel 8.5

1. Im Fall der quadratischen Gleichung gilt nach dem Satz von Vieta:

   $$x_1 x_2 = q.$$

   Man kann also x_1 wie in Beispiel 8.4 berechnen und anschließend $x_2 := \dfrac{q}{x_1}$ setzen. Im Vergleich zur obigen Rechnung erfordert dies eine zusätzliche Division. Dafür wird die Auslöschung vermieden. Im obigen Beispiel:

   $$x_2 = 1.400 \cdot 10^1 \boxslash 3.965 \cdot 10^1 = \square 3.5308\ldots \cdot 10^{-1} = 3.531 \cdot 10^{-1}.$$

2. Geht man von exakten Maschinenzahlen x und y aus, ist die Differenz $x - y$ im Fall $x \approx y$ ebenfalls eine Maschinenzahl. Die Differenz $x - y$ wird dann fehlerfrei berechnet, obwohl Auslöschung auftritt. Kritisch bleibt dagegen die Berechnung von Ausdrücken der Bauart $f(x) - f(y)$, wenn die Funktionswerte gerundet berechnet werden. Einige typische Umformungen zur Bekämpfung von Auslöschung sind:

   (a) $\sqrt{x} - \sqrt{y} = \dfrac{x - y}{\sqrt{x} + \sqrt{y}}$.

   Z.B. $x = 4.48$, $y = 4.47$:

   $$\square\sqrt{x} \boxminus \square\sqrt{y} = \square 2.1166\ldots \boxminus \square 2.1142\ldots = 2.117 \boxminus 2.114 = 3.000 \cdot 10^{-3},$$

   $$(x \boxminus y) \boxslash (\square\sqrt{x} \boxplus \square\sqrt{y}) = 0.01 \boxslash 4.231 = \square 0.0023635\ldots = 2.364 \cdot 10^{-3}.$$

   (b) $\ln x - \ln y = \ln \dfrac{x}{y} = \ln\left(1 + \dfrac{x - y}{y}\right) \approx \dfrac{x - y}{y}$ (Satz von Taylor, für $x \approx y$).

   Z.B. $x = 4.48$, $y = 4.47$:

   $$(\square \ln x) \boxminus (\square \ln y) = \square 1.4996\ldots \boxminus \square 1.4973\ldots$$
   $$= 1.500 \boxminus 1.497 = 3.000 \cdot 10^{-3},$$

   $$\square \ln(x \boxslash y) = \square \ln(\square 1.0022\ldots) = \square \ln(1.002) = \square 0.0019980\ldots = 1.998 \cdot 10^{-3},$$

   $$(x \boxminus y) \boxslash y = 0.01 \boxslash 4.47 = \square 0.0022371\ldots = 2.237 \cdot 10^{-3}.$$

   (Exakt: $\square(\ln 4.48 - \ln 4.47) = 2.235 \cdot 10^{-3}$.)

   (c) $\cos x - \cos y = 2 \sin \dfrac{x + y}{2} \sin \dfrac{y - x}{2}$.

   Z.B. $x = 4.48$, $y = 4.47$:

   $$(\square \cos x) \boxminus (\square \cos y) = \square(-0.23030\ldots) \boxminus \square(-0.24002\ldots)$$
   $$= -0.2303 \boxminus (-0.2400) = 9.700 \cdot 10^{-3},$$

   $$2 \boxdot \left(\square \sin \frac{x \boxplus y}{2}\right) \boxdot \left(\square \sin \frac{y \boxminus x}{2}\right) = 2 \boxdot \left(\square \sin(4.475)\right) \boxdot \left(\square \sin(-0.005)\right)$$
   $$= 2 \boxdot \left(\square(-0.97195\ldots)\right) \boxdot \left(\square(-0.0049999\ldots)\right)$$
   $$= 2 \boxdot 0.9720 \boxdot 0.005 = 9.720 \cdot 10^{-3}.$$

8.3 Die Kondition eines mathematischen Problems

Rundungsfehler und ihre Fortpflanzung sind nicht die einzige Fehlerquelle numerischer Berechnungen. Häufig enthalten bereits die Eingangsdaten einer mathematischen Aufgabenstellung Fehler, beispielsweise Messfehler oder Fehler, die auf Vereinfachungen in der Modellierung zurückgehen. Manchmal haben vereinfachende Annahmen keine praktische Auswirkung auf das Ergebnis (z.B. beim freien Fall eines Tennisballs aus 1,50 m Höhe darf man den Luftwiderstand, die Corioliskraft, die Abhängigkeit der Beschleunigung von der augenblicklichen Position sowie relativistische Effekte getrost vernachlässigen), ein anderes Mal führen sie zu unerklärbaren Widersprüchen zwischen Berechnungen und Beobachtungen.

Ein historisches Beispiel dafür liefert die Periheldrehung der Merkurbahn. Zu Beginn des 20. Jahrhunderts lagen präzise astronomische Messungen der Bahnkurve des Merkurs vor, die den Berechnungen auf Basis der Newton'schen Himmelsmechanik widersprachen. Zwischen der beobachteten jährlichen Präzession des Perihels von 5.75 Bogensekunden und dem theoretisch berechneten Wert von 5.32 Bogensekunden klaffte eine Lücke von 0.43 Bogensekunden, was immerhin 7.5% Abweichung entspricht. Um die Beobachtung in Einklang mit der Theorie zu bringen (!), suchte man lange Zeit vergeblich nach unbekannten Himmelskörpern, deren Gravitationskraft die Abweichung erklären sollte. Aufgelöst wurde der Widerspruch erst 1916 durch Einsteins Veröffentlichung seiner allgemeinen Relativitätstheorie, welche die vorliegenden Messungen bestätigte und die Unvollständigkeit der Newton'schen Himmelsmechanik offenbarte.

Sind gegenüber Beobachtungen abweichende Berechnungen auf fehlerhafte Modellierung zurückzuführen, ist dies mit den Mitteln der Numerischen Mathematik nicht zu klären. Manchmal führen aber trotz korrekter mathematischer Modellierung kleine Messfehler in Eingangsgrößen zu unverhältnismäßig großen Fehlern im berechneten Ergebnis. An dieser Stelle ist die Numerik gefordert, die Auswirkungen von Störungen in den Eingangsdaten eines Algorithmus abzuschätzen und gegebenenfalls den Algorithmus so abzuändern, dass Fehlerverstärkung in der Berechnung vermieden wird.

Gegeben ist die folgende Situation:

$$\text{Eingangsdaten} \quad \xrightarrow{f} \quad \text{Ergebnis.}$$

Gesucht ist die Abhängigkeit des Ergebnisses von kleinen Störungen in den Eingangsdaten.

8.3.1 Fehlerfortpflanzung

Wir untersuchen die Fehlerfortpflanzung nur für hinreichend glatte Funktionen. Ist f unstetig, können auch kleinste Fehler in den Eingangsdaten große Fehler im Ergebnis verursachen. Solche schlecht gestellten Probleme betrachten wir hier nicht.

Für den durch eine kleine Störung Δx verursachten relativen Fehler von $f(x)$ gilt:

1. Gegeben sei eine stetig differenzierbare, reellwertige Funktion $f : D \subseteq \mathbb{R} \to \mathbb{R}$. Für $y = f(x)$, $y + \Delta y = f(x + \Delta x)$ gilt dann:

   $$\Delta y = f(x + \Delta x) - f(x) = f'(\xi)\Delta x$$

   für ein ξ zwischen x und $x + \Delta x$. Für den relativen Fehler des Ergebnisses folgt:

   $$\frac{\Delta y}{y} = \frac{f'(\xi)\Delta x}{f(x)} = \frac{x f'(\xi)}{f(x)} \frac{\Delta x}{x} \approx \frac{x f'(x)}{f(x)} \frac{\Delta x}{x}.$$

2. Ist $f : D \subseteq \mathbb{R}^n \to \mathbb{R}^m$ stetig differenzierbar, gilt für $y_i = f_i(x_1, x_2, \ldots, x_n)$, $i = 1, 2, \ldots, m$, analog

$$\Delta y_i = f_i(x + \Delta x) - f_i(x) \approx \sum_{j=1}^{n} \Delta x_j \frac{\partial f_i}{\partial x_j}(x),$$

was auf die Abschätzung

$$|\Delta y_i| \lesssim \sum_{j=1}^{n} |\Delta x_j| \left| \frac{\partial f_i}{\partial x_j}(x) \right| \leq \left(\max_{k=1}^{n} |\Delta x_k| \right) \sum_{j=1}^{n} \left| \frac{\partial f_i}{\partial x_j}(x) \right| \leq \|\Delta x\|_\infty \left\| \left(\frac{\partial f_i}{\partial x_j}(x) \right) \right\|_\infty$$

führt, aus welcher

$$\frac{\|\Delta y\|_\infty}{\|y\|_\infty} \lesssim \frac{\|x\|_\infty \left\| \left(\frac{\partial f_i}{\partial x_j}(x) \right) \right\|_\infty}{\|f(x)\|_\infty} \frac{\|\Delta x\|_\infty}{\|x\|_\infty}$$

folgt. Dabei haben wir die Zeilensummennorm in naheliegender Weise für nicht quadratische Matrizen verallgemeinert.

Definition 8.6

1. Gegeben sei eine stetig differenzierbare, reellwertige Funktion $f : D \subseteq \mathbb{R} \to \mathbb{R}$. Dann heißt

$$\kappa_f(x) := (\text{cond}\, f)(x) := \left| \frac{x f'(x)}{f(x)} \right|$$

(relative) *Kondition* von f an der Stelle x.

2. Ist $f : D \subseteq \mathbb{R}^n \to \mathbb{R}^m$ stetig differenzierbar, dann heißt

$$\kappa_f(x) := (\text{cond}\, f)(x) := \frac{\|x\|_\infty \left\| \frac{\partial f_i}{\partial x_j}(x) \right\|_\infty}{\|f(x)\|_\infty}$$

(relative) *Kondition* von f an der Stelle x.

Die Kondition ist eine Eigenschaft des Problems (der Funktion), nicht eines Lösungsverfahrens (eines Algorithmus).

Beispiel 8.7

1. Addition zweier Zahlen: $f(x) := x + a,\ a \in \mathbb{R}\backslash\{0\}$ fest:

$$\kappa_f(x) = \left| \frac{x \cdot 1}{x + a} \right| = \left| \frac{x}{x + a} \right|.$$

Für $x = 0$ ist die Kondition Null (Null addiert man gewissermaßen fehlerfrei), für $x \approx -a$ ist die Kondition groß (Auslöschung).

2. Multiplikation zweier Zahlen: $f(x) := ax,\ a \in \mathbb{R}$ fest:

$$\kappa_f(x) = \left| \frac{ax}{ax} \right| = 1.$$

Die Multiplikation ist auf ganz $\mathbb{R}$ gut konditioniert.

8.3.2 Stabilität

Bei der Stabilitätsanalyse will man für einen gegebenen numerischen Algorithmus abschätzen, wie sich Fehler in den Eingangsdaten sowie Rundungsfehler im Verlauf der Rechnung auf das Ergebnis auswirken. Wir gehen von der folgenden Situation aus: Die Funktion $f : \mathbb{R} \to \mathbb{R}$ wird auf dem Rechner durch den Algorithmus $\tilde{f} : \mathbb{R} \to \mathcal{R}$ realisiert. Datenfehler (durch Messung oder Rundung der Eingangsdaten) und einzelne Rundungsfehler in Gleitpunktoperationen seien durch die relative Fehlerschranke eps wie in (8.3) beschränkt.

Definition 8.8 Der Algorithmus $\tilde{f}$ heißt auf D vorwärts stabil, falls es eine nicht zu große Konstante C_V gibt, so dass

$$\left| \frac{\tilde{f}(x) - f(x)}{f(x)} \right| \leq C_V \cdot \kappa_f(x) \cdot \text{eps}$$

für alle $x \in D$ gilt.

Der Algorithmus $\tilde{f}$ ist genau dann vorwärts stabil, wenn die Kondition des Problems f durch $\tilde{f}$ höchstens moderat vergrößert wird. Genaue Ergebnisse erhält man in einer numerischen Berechnung, wenn das Problem gut konditioniert und der verwendete Algorithmus stabil ist. Ein instabiler Algorithmus kann auch für ein gut konditioniertes Problem versagen. Schlecht konditionierte Probleme lassen sich auch mit stabilen Algorithmen nicht oder nicht zufriedenstellend lösen. Dann hilft nur die Umformulierung des Problems.

Beispiel 8.9 Es sei $f(x) := \sqrt{x+1} - \sqrt{x}$, $x \geq 0$. Aus

$$f'(x) = \frac{1}{2\sqrt{x+1}} - \frac{1}{2\sqrt{x}} = -\frac{\sqrt{x+1} - \sqrt{x}}{2\sqrt{x+1}\sqrt{x}}$$

folgt

$$\kappa_f(x) = \left| \frac{x f'(x)}{f(x)} \right| = \frac{1}{2} \frac{\sqrt{x}}{\sqrt{x+1}} < \frac{1}{2}.$$

Die Funktion ist gut konditioniert und sollte sich daher genau auswerten lassen.

Es sei nun $x \in \mathcal{R}$ und $\tilde{f}_1(x)$ sei der folgende Algorithmus:

$$\begin{aligned} z_1 &= \square(x+1), \\ z_2 &= \square\sqrt{z_1}, \\ z_3 &= \square\sqrt{x}, \\ z_4 &= z_2 \boxminus z_3, \\ \tilde{f}_1 &= z_4. \end{aligned}$$

Dann gilt:

$$\begin{aligned} z_1 &= (x+1)(1+\varepsilon_1), \\ z_2 &= \sqrt{z_1}(1+\varepsilon_2) = \sqrt{(x+1)(1+\varepsilon_1)}(1+\varepsilon_2) \approx \sqrt{(x+1)}(1+\tfrac{1}{2}\varepsilon_1+\varepsilon_2), \\ z_3 &= \sqrt{x}(1+\varepsilon_3), \\ z_4 &= (z_2 - z_3)(1+\varepsilon_4) = \big(\sqrt{x+1} - \sqrt{x} + \sqrt{x+1}(\tfrac{1}{2}\varepsilon_1+\varepsilon_2) - \varepsilon_3\sqrt{x}\big)(1+\varepsilon_4) \\ &\approx \sqrt{x+1} - \sqrt{x} + \sqrt{x+1}(\tfrac{1}{2}\varepsilon_1+\varepsilon_2+\varepsilon_4) - \sqrt{x}(\varepsilon_3+\varepsilon_4) = \tilde{f}_1(x). \end{aligned}$$

Die Fehlerterme ε_i können negativ sein. Der absolute Gesamtfehler ist durch

$$\frac{3}{2}\sqrt{x+1}\,\text{eps} + \sqrt{x}\,\text{eps} + (\sqrt{x+1}-\sqrt{x})\,\text{eps}$$

bestmöglich abgeschätzt. Also gilt

$$\left|\frac{\tilde{f}_1(x)-f(x)}{f(x)}\right| \leq \Big(\underbrace{\frac{\frac{3}{2}\sqrt{x+1}+\sqrt{x}}{\sqrt{x+1}-\sqrt{x}}}_{\approx 5x \text{ für } x\to\infty} +1\Big)\,\text{eps}.$$

Der Algorithmus $\tilde{f}_1$ ist nicht vorwärts stabil.

Alternativ betrachten wir den Algorithmus $\tilde{f}_2(x)$ gegeben durch

$$\begin{aligned} z_1 &= \square(x+1),\\ z_2 &= \square\sqrt{z_1},\\ z_3 &= \square\sqrt{x},\\ z_4 &= z_2 \boxplus z_3,\\ z_5 &= 1 \boxslash z_4,\\ \tilde{f}_1 &= z_5, \end{aligned}$$

welcher auf der Darstellung

$$f(x) = \frac{1}{\sqrt{x+1}+\sqrt{x}}$$

beruht. Analog zu oben gilt die Fehlerfortpflanzung

$$\begin{aligned} z_2 &\approx \sqrt{(x+1)}\,(1+\tfrac{1}{2}\varepsilon_1+\varepsilon_2)\\ z_3 &= \sqrt{x}\,(1+\varepsilon_3)\\ z_4 &= (z_2+z_3)(1+\varepsilon_4)\\ &\approx \sqrt{x+1}+\sqrt{x}+\sqrt{x+1}\,(\tfrac{1}{2}\varepsilon_1+\varepsilon_2+\varepsilon_4)+\sqrt{x}\,(\varepsilon_3+\varepsilon_4)\\ &= (\sqrt{x+1}+\sqrt{x})\left(1+\frac{\sqrt{x+1}}{\sqrt{x+1}+\sqrt{x}}\,(\frac{1}{2}\varepsilon_1+\varepsilon_2)+\frac{\sqrt{x}}{\sqrt{x+1}+\sqrt{x}}\,\varepsilon_3+\varepsilon_4\right) \end{aligned}$$

Der Satz von Taylor liefert die Entwicklung

$$\frac{1}{z_4} \approx \frac{1}{\sqrt{x+1}+\sqrt{x}}\left(1-\frac{\sqrt{x+1}}{\sqrt{x+1}+\sqrt{x}}\,(\frac{1}{2}\varepsilon_1+\varepsilon_2)-\frac{\sqrt{x}}{\sqrt{x+1}+\sqrt{x}}\,\varepsilon_3-\varepsilon_4\right),$$

welche für große Werte von x mit $\sqrt{x+1}\approx\sqrt{x}$ in

$$\frac{1}{z_4} \approx \frac{1}{\sqrt{x+1}+\sqrt{x}}\,(1-\frac{1}{4}\,\varepsilon_1-\frac{1}{2}\,\varepsilon_2-\frac{1}{2}\,\varepsilon_3-\varepsilon_4)$$

übergeht. Hieraus folgt zunächst

$$z_5 = \frac{1}{z_4}\,(1+\varepsilon_5) \approx f(x)(1-\frac{1}{4}\,\varepsilon_1-\frac{1}{2}\,\varepsilon_2-\frac{1}{2}\,\varepsilon_3-\varepsilon_4+\varepsilon_5)$$

und schließlich

$$\left|\frac{\tilde{f}_2(x)-f(x)}{f(x)}\right| \leq (\frac{1}{4}+\frac{1}{2}+\frac{1}{2}+1+1)\,\text{eps} = \frac{13}{4}\,\text{eps}.$$

Somit ist der Algorithmus $\tilde{f}_2$ vorwärts stabil (mit $C_V = \frac{13}{2}$).

Häufig ist die Vorwärtsanalyse eines Algorithmus zu aufwändig. Eventuell ist dann zumindest die von Wilkinson entwickelte Rückwärtsanalyse durchführbar. Bei dieser interpretiert man den vom Algorithmus $\tilde{f}$ berechneten Näherungswert $\tilde{f}(x)$ als exakten Funktionswert von f an der gestörten Stelle $\tilde{x}$:

$$\tilde{f}(x) = f(\tilde{x}).$$

Definition 8.10 Der Algorithmus $\tilde{f}$ heißt auf D rückwärts stabil, falls es eine nicht zu große Konstante C_R gibt, so dass zu jedem $x \in D$ ein $\tilde{x} \in D$ mit

$$\tilde{f}(x) = f(\tilde{x}), \quad \left|\frac{\tilde{x} - x}{x}\right| \leq C_R \cdot \text{eps}$$

existiert.

Bemerkung 8.11 Ist der Algorithmus $\tilde{f}$ rückwärts stabil, dann gilt

$$\left|\frac{\tilde{f}(x) - f(x)}{f(x)}\right| = \left|\frac{f(\tilde{x}) - f(x)}{f(x)}\right| \approx \kappa_f(x) \left|\frac{\tilde{x} - x}{x}\right| \leq C_R \cdot \kappa_f(x) \cdot \text{eps}.$$

Jeder rückwärts stabile Algorithmus ist also auch vorwärts stabil. Die Umkehrung gilt im Allgemeinen nicht.

Beispiel 8.12 (Rückwärtsanalyse der quadratischen Gleichung)
Wir betrachten wieder das in Beispiel 8.4 behandelte Nullstellenproblem:

$$x^2 - 2px + q = 0, \quad p^2 \gg q.$$

Zur Berechnung der betragskleineren Nullstelle

$$x_2 := f(p, q) := p - \sqrt{p^2 - q}$$

hatten wir in Beispiel 8.4 einen Algorithmus mit der Fehlerfortpflanzung

$$\begin{aligned} z_1 &= p^2(1 + \varepsilon_1), \\ z_2 &= (z_1 - q)(1 + \varepsilon_2), \\ z_3 &= \sqrt{z_2}\,(1 + \varepsilon_3), \\ z_4 &= (p - z_3)(1 + \varepsilon_4), \\ \tilde{f}_1 &= z_4 \end{aligned}$$

benutzt. Die Rückwärtsanalyse liefert für diesen Algorithmus

$$\begin{aligned} &\Big(p - \sqrt{\big(p^2(1 + \varepsilon_1) - q\big)(1 + \varepsilon_2)}(1 + \varepsilon_3)\Big)(1 + \varepsilon_4) \\ &= p(1 + \varepsilon_4) - \sqrt{\big(p(1 + \varepsilon_4)\big)^2 \underbrace{(1 + \varepsilon_1)(1 + \varepsilon_2)(1 + \varepsilon_3)^2}_{=:\,1 + \varepsilon_5} - q \underbrace{(1 + \varepsilon_2)(1 + \varepsilon_3)^2(1 + \varepsilon_4)^2}_{=:\,1 + \varepsilon_6}} \\ &= p(1 + \varepsilon_4) - \sqrt{\big(p(1 + \varepsilon_4)\big)^2 - q \underbrace{\Big(1 + \varepsilon_6 - \varepsilon_5(1 + \varepsilon_4)^2 \frac{p^2}{q}\Big)}_{=:\,1 + \varepsilon_7}} \\ &= p(1 + \varepsilon_4) - \sqrt{\big(p(1 + \varepsilon_4)\big)^2 - q(1 + \varepsilon_7)} = f\Big(p(1 + \varepsilon_4), q(1 + \varepsilon_7)\Big) \end{aligned}$$

mit

$$|\varepsilon_4| \leq \text{eps},$$

$$|\varepsilon_5| \approx |\varepsilon_1 + \varepsilon_2 + 2\varepsilon_3| \leq 4\,\text{eps},$$

$$|\varepsilon_6| \approx |\varepsilon_2 + 2\varepsilon_3 + 2\varepsilon_4| \leq 5\,\text{eps},$$

$$|\varepsilon_7| \approx \left|\varepsilon_6 - \varepsilon_5\,\frac{p^2}{q}\right| \lesssim (5 + 4\frac{p^2}{q})\,\text{eps}.$$

Nach Voraussetzung ist $4\dfrac{p^2}{q} \gg 1$, also ist der Algorithmus nicht rückwärts stabil.

8.4 Beispiele

Die Stabilitätsanalyse eines Algorithmus ist in der Regel schwierig. Zum Abschluss dieses Kapitels illustrieren wir das Vorgehen anhand von Beispielen.

8.4.1 Addition Es sei (vgl. Beispiel 8.7)

$$\begin{aligned} f(x) &:= x + a, \quad a \in \mathbb{R}\backslash\{0\} \text{ fest}, \\ \tilde{f}(x) &= \square(x + a) = (x + a)(1 + \varepsilon), \quad |\varepsilon| \leq \text{eps}. \end{aligned}$$

Die Vorwärtsanalyse liefert

$$\left|\frac{\tilde{f}(x) - f(x)}{f(x)}\right| = \left|\frac{(x + a)(1 + \varepsilon) - (x + a)}{x + a}\right| = |\varepsilon| = \left|\frac{x + a}{x}\right| \kappa_f(x)\ |\varepsilon| \leq \left|1 + \frac{a}{x}\right| \kappa_f(x)\,\text{eps}.$$

Die Addition ist vorwärts stabil, sofern $|x| \not\ll |a|$ gilt.

Im Fall $|x| \not\ll |a|$ ist die Addition auch rückwärts stabil, denn es gilt:

$$\tilde{f}(x) = (x + a)(1 + \varepsilon) = \underbrace{x\big(1 + \varepsilon + \frac{\varepsilon a}{x}\big)}_{\tilde{x}} + a.$$

Für $|x| \ll |a|$ ist $\dfrac{\varepsilon a}{x}$ groß, die Addition also nicht rückwärts stabil. Man kann dies auch durch die folgende Überlegung leicht einsehen: In $\mathcal{S}$ gilt für $a = 1$, $|x| \leq \frac{1}{2} \cdot 10^{-3} (= \text{eps})$

$$\square(1 + x) = 1 = 1 + 0 =: 1 + \tilde{x}.$$

$\tilde{x} = 0$ ist eindeutig bestimmt. Will man die Bedingung

$$\left|\frac{\tilde{x} - x}{x}\right| = 1 \overset{!}{\leq} C_R\,\text{eps}$$

erfüllen, muss $C_R \geq \dfrac{1}{\text{eps}}$ gelten.

8.4.2 Multiplikation Es sei

$$f(x) := ax, \quad a \in \mathbb{R} \text{ fest},$$
$$\tilde{f}(x) = \square(ax) = ax(1+\varepsilon), \quad |\varepsilon| \leq \text{eps}.$$

Vorwärtsanalyse: Mit der Kondition aus Beispiel 8.7 gilt:

$$\left|\frac{\tilde{f}(x) - f(x)}{f(x)}\right| = |\varepsilon| = \kappa_f(x)\ |\varepsilon| \leq \kappa_f(x)\,\text{eps}.$$

Die Multiplikation ist somit vorwärts stabil.

Rückwärtsanalyse:

$$\tilde{f}(x) = (ax)(1+\varepsilon) = a\big(\underbrace{x(1+\varepsilon)}_{\tilde{x}}\big).$$

Die Multiplikation ist auch rückwärts stabil.

8.4.3 Logarithmus Es sei

$$f(x) := \ln x, \quad \tilde{f}(x) = \square \ln x = \ln x\,(1+\varepsilon), \quad |\varepsilon| \leq \text{eps}.$$

Falls x nicht nahe bei 1 liegt, ist f gut konditioniert:

$$\kappa_f(x) = \left|\frac{xf'(x)}{f(x)}\right| = \left|\frac{1}{\ln x}\right|.$$

Vorwärtsanalyse:

$$\left|\frac{\tilde{f}(x) - f(x)}{f(x)}\right| = |\varepsilon| \leq |\ln x|\ \kappa_f(x)\,\text{eps}.$$

Der Algorithmus zur Berechnung der Logarithmusfunktion ist vorwärts stabil, wenn $|\ln x|$ nicht zu groß ist (z.B. für $x \in [e^{-10}, e^{10}]$).

Rückwärtsanalyse:

$$\tilde{f}(x) = (1+\varepsilon)\ln x = \ln\left(x^{1+\varepsilon}\right) =: \ln \tilde{x}$$

($\tilde{x}$ ist eindeutig bestimmt). Es gilt:

$$\left|\frac{\tilde{x} - x}{x}\right| = \left|\frac{xe^{\varepsilon \ln x} - x}{x}\right| \approx \left|\frac{x(1+\varepsilon\ln x) - x}{x}\right| = |\varepsilon|\,|\ln x|\,,$$

so dass der Algorithmus zur Berechnung der Logarithmusfunktion auch rückwärts stabil ist, wenn $|\ln x|$ nicht zu groß ist.

8.4.4 Produkte Es sei

$$f(x_1, x_2, \ldots, x_n) := \prod_{i=1}^{n} x_i.$$

Der Algorithmus $\tilde{f}(x_1, x_2, \ldots, x_n)$ zur Berechnung des Produkts sei definiert durch

$$z_1 = x_1 x_2(1+\varepsilon_1),$$
$$z_i = z_{i-1}x_{i+1}(1+\varepsilon_i), \quad i = 2, 3, \ldots, n-1,$$
$$\tilde{f} = z_{n-1}.$$

Rückwärtsanalyse:

$$\tilde{f}(x) = \prod_{i=1}^{n} x_i \prod_{i=1}^{n-1}(1+\varepsilon_i) = x_n \prod_{i=1}^{n-1}\big(x_i(1+\varepsilon_i)\big).$$

Die Berechnung von Produkten ist daher rückwärts stabil.

8.4.5 Summen Es sei

$$f(x_1, x_2, \ldots, x_n) := \sum_{i=1}^{n} x_i.$$

Der Algorithmus $\tilde{f}(x_1, x_2, \ldots, x_n)$ zur Berechnung der Summe sei definiert durch

$$\begin{aligned} z_1 &= (x_1 + x_2)(1+\varepsilon_2), \\ z_i &= (z_{i-1} + x_{i+1})(1+\varepsilon_{i+1}), \quad i = 2, 3, \ldots, n-1, \\ \tilde{f} &= z_{n-1}. \end{aligned}$$

Dann gilt:

$$\tilde{f}(x) = x_1 \prod_{i=2}^{n}(1+\varepsilon_i) + \sum_{j=2}^{n} x_j \prod_{i=j}^{n}(1+\varepsilon_i)$$

Wegen

$$\prod_{i=j}^{n}(1+\varepsilon_i) \approx 1 + \sum_{i=j}^{n} \varepsilon_i$$

gilt

$$\left|\tilde{f}(x) - f(x)\right| \lesssim (n-1)\,\text{eps}\,|x_1| + \sum_{j=2}^{n}(n+1-j)\,\text{eps}\,|x_j|\,.$$

Die Fehlerschranke für den absoluten Fehler wird am kleinsten, wenn die Zahlen nach aufsteigenden Beträgen summiert werden.

8.4.6 Lineare Gleichungssysteme Es sei $A \in \mathbb{R}^{n\times n}$ eine feste reguläre Matrix. Für variables $b \in \mathbb{R}^n$ betrachten wir die Funktion

$$f : \mathbb{R}^n \to \mathbb{R}^n, \quad x = f(b) := A^{-1}b.$$

Mit der Funktionalmatrix

$$\frac{\partial f}{\partial b} = A^{-1}$$

und $Ax = b$ erhalten wir die Kondition von f:

$$\kappa_f(b) = \frac{\|b\|\,\left\|A^{-1}\right\|}{\|A^{-1}b\|} = \frac{\|Ax\|\,\left\|A^{-1}\right\|}{\|x\|}.$$

Ihr maximaler Wert ist

$$\sup_{b\neq 0} \kappa_f(b) = \sup_{x\neq 0} \frac{\|Ax\|\,\left\|A^{-1}\right\|}{\|x\|} = \|A\|\,\left\|A^{-1}\right\| = \kappa(A).$$

8.4.7 Rekursionsformel für Integrale Für $n \in \mathbb{N}$ sei

$$I_n := \int_1^e (\ln x)^n dx.$$

Dieses Integral kann mit Hilfe einer Rekursionsformel geschlossen gelöst werden. Mit partieller Integration folgt:

$$\begin{aligned} n = 1: \quad & I_1 = \int_1^e \ln x \, dx = (x \ln x - x)\Big|_1^e = 1. \\ n > 1: \quad & I_n = x(\ln x)^n\Big|_1^e - \int_1^e xn(\ln x)^{n-1} \frac{1}{x} \, dx = e - nI_{n-1}. \end{aligned}$$

Aus der Rekursionsformel

$$I_n = e - nI_{n-1} \tag{8.4}$$

erhält man wegen $\ln x \geq 0$ für $x \geq 1$ die triviale Grobabschätzung

$$0 \leq I_n \leq e.$$

Benutzt man die genauere Abschätzung

$$\ln x \geq \frac{x-1}{e-1} \quad \text{für } 1 \leq x \leq e,$$

dann folgt

$$I_n \geq \int_1^e \left(\frac{x-1}{e-1}\right)^n dx = \frac{(x-1)^{n+1}}{(n+1)(e-1)^n}\Big|_1^e = \frac{e-1}{n+1}.$$

Mit Hilfe von (8.4) erhält man

$$I_n \leq e - n\frac{e-1}{n} = 1,$$

also insgesamt die Schranken

$$\frac{e-1}{n+1} \leq I_n \leq 1.$$

Berechnet man I_n für einige Werte von n, lässt sich vermuten, dass

$$I_n = e + (-1)^n \, n! \left\{ e \sum_{j=0}^{n-1} \frac{(-1)^j}{j!} - I_1 \right\} \tag{8.5}$$

gilt. Mit vollständiger Induktion weist man die angegebene Darstellung leicht nach.

Die Summe in (8.5) ist die n-te Partialsumme von e^{-1}. Der Wert in den geschweiften Klammern strebt also gegen Null, wodurch die Berechnungsformel für große Werte von n zum uneigentlichen Grenzwert "$0 \cdot \infty$" mutiert.

Außerdem gewinnt man aus der Rekursionsformel (8.4) für $n > k \geq 1$ die Darstellung

$$I_n = (-1)^{n-k} \frac{n!}{k!} I_k + c_{n,k} \tag{8.6}$$

mit gewissen Zahlen $c_{n,k} \in \mathbb{R}$. Wir fassen nun I_n in (8.6) als Funktion von I_k auf,

$$I_n = f(I_k), \quad n > k \geq 1,$$

und wollen untersuchen, wie sich Fehler in I_k auf I_n auswirken. Aus

$$|f'(I_k)| = \frac{n!}{k!}$$

erhalten wir

$$\kappa_f(I_k) = \left| \frac{I_k f'(I_k)}{I_n} \right| = \frac{n!\, I_k}{k!\, I_n} \geq \frac{(e-1)\, n!}{(k+1)!}.$$

Für $n \gg k$ ist f beliebig schlecht konditioniert. Die Rekursionsformel (8.4) eignet sich daher nicht zur Berechnung von I_n. Ein Zahlenbeispiel soll dies verdeutlichen. Maple 9.5 liefert (mit der Standardeinstellung für die Genauigkeit) ausgehend von $I_1 := 1.0$ die folgenden Näherungen $\tilde{I}_n$ (gültige Ziffern sind schwarz dargestellt, ungültige blau):

n	$\tilde{I}_n$
1	1.000000000
2	0.718281828
3	0.563436344
4	0.464536452
5	0.395599568
6	0.344684420
7	0.305490888
8	0.274354724
9	0.249089312
10	0.227388708
11	0.217006040
12	0.114209348
13	1.233560304
14	-14.55156243
15	220.9917182

Der Genauigkeitsverlust ist hier nicht auf mangelnde Stabilität des Berechnungsalgorithmus zurückzuführen, sondern auf die schlechte Kondition der Formel (8.4). Abhilfe ist nur durch eine grundlegende Änderung der Berechnungsweise zu erwarten. Es ist

$$\begin{aligned} I_n &= e + (-1)^n\, n! \left\{ e \left(e^{-1} - \sum_{j=n}^{\infty} \frac{(-1)^j}{j!} \right) - 1 \right\} \\ &= e \sum_{j=n+1}^{\infty} \frac{(-1)^{n+1+j}}{j!}\, n! \\ &\approx e \left(\frac{1}{n+1} - \frac{1}{(n+1)(n+2)} + - \dots \frac{1}{(n+1)(n+2)\cdots(n+N)} \right) =: J_{n,N} \end{aligned} \tag{8.7}$$

für geeignete Werte von N. Die zuletzt aufgetretene Reihe konvergiert sehr schnell. Mit N Summanden in (8.7) bekommt man bereits für $n = 10$ ungefähr N gültige Dezimalstellen von I_n. Außerdem ist (8.7) keine Rekursionsformel. Man kann $J_{n,N}$ für beliebige Werte von n und N bestimmen, ohne vorhergehende Integrale zu berechnen. Zum Vergleich mit der obigen Tabelle geben wir auch für $J_{n,N}$ einige Zahlenwerte an. Die Vorteile der geänderten Berechnungsformel sind deutlich zu sehen.

n	$J_{n,2}$	$J_{n,5}$	$J_{n,10}$	I_n
5	0.3883259755	0.3956070875	0.3955995478	0.3955995478
10	0.2265234857	0.2280019606	0.2280015154	0.2280015155
15	0.1598989310	0.1604263754	0.1604263089	0.1604263089
20	0.1235582649	0.1238038466	0.1238038307	0.1238038308
50	0.05227465055	0.05229363840	0.05229363829	0.05229363830
100	0.02664982184	0.02665235919	0.02665235919	0.02665235919

Index