1 Rechnerarithmetik, Rundungsfehler und elementare Fehlerfortpflanzung

1.1 Zahlendarstellung

Rechenanlagen können nur endlich viele Zahlen exakt darstellen. Alle anderen Zahlen werden durch eine darstellbare Zahl approximiert. Auf den meisten Rechenanlagen gibt es zwei verschiedene Arten der Zahlendarstellung: Ganze Zahlen, Gleitpunktzahlen.

Ganze Zahlen (Integer-Zahlen)

Im Dezimalsystem haben ganze Zahlen die allgemeine Form

$$\pm\, z_n\, z_{n-1} \ldots z_2\, z_1\, z_0 = \pm\, (z_n\, 10^n + z_{n-1}\, 10^{n-1} + \ldots + z_2\, 10^2 + z_1\, 10 + z_0)$$

wobei z_i die üblichen Ziffern darstellen, d.h. $z_i \in \{0, 1 \ldots, 9\}$. Die meisten Rechner benutzen nicht die Darstellung im Dezimalsystem. Als Basis wird eine natürliche Zahl $\beta > 1$ gewählt. Im System bezüglich der Basis β hat eine ganze Zahl die allgemeine Form:

$$(\pm\, a_n\, a_{n-1} \ldots a_2\, a_1\, a_0)_\beta = \pm\, (a_n\, \beta^n + a_{n-1}\, \beta^{n-1} + \ldots + a_2\, \beta^2 + a_1\, \beta + a_0\, \beta^0)$$

wobei die a_i die sogenannten Ziffern darstellen,

$$a_i \in \{0, 1, 2, \ldots, \beta - 1\}\ .$$

Beispiele:

$$\begin{array}{llll}
\beta = 10, & n = 3: & 7662,\ -4520 & \\
\beta = 2, & n = 4: & (10011)_2 & = 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2 + 1 \\
 & & & = 16 + 2 + 1 = 19 \\
\beta = 8, & n = 3: & (7203)_8 & = 7 \cdot 8^3 + 2 \cdot 8^2 + 0 \cdot 8 + 3 \\
 & & & = 3584 + 128 + 3 = 3715\ .
\end{array}$$

Zum Speichern einer solchen Zahl stellt ein Rechner eine rechnerabhängige Anzahl von Stellen zur Verfügung, d.h. n ist im allgemeinen beschränkt. Somit gibt es eine Zahl Z, so dass alle Zahlen x mit $|x| > Z$ nicht mehr durch den Rechner darstellbar sind. Wird dies trotzdem versucht, meldet der Rechner im Idealfall einen **Überlauf (overflow)** und bricht die Rechnung ab.

Wir beschränken uns auf das Dezimalsystem. Andere, in der Praxis häufig anzutreffende Systeme sind Dual-, Oktal-, Hexadezimalsysteme.

Gleitpunktzahlen (Floating point numbers)
Jede reelle Zahl kann in einer Basis $\beta > 1$ wie folgt dargestellt werden:

$$\begin{aligned} & \pm\ (a_n\, a_{n-1} \ldots a_2\, a_1\, a_0\, .\, b_1\, b_2\, b_3 \ldots)_\beta \\ & \quad = \pm\ (a_n\, \beta^n + a_{n-1}\, \beta^{n-1} + \ldots + a_2\, \beta^2 + a_1\, \beta + a_0\, \beta^0 \\ & \quad + b_1\, \beta^{-1} + b_2\, \beta^{-2} + \ldots \quad . \end{aligned} \tag{1.1}$$

BEISPIELE:

$$\begin{aligned} \beta = 10, \quad -(8753.92)_{10} &= -(8 \cdot 10^3 + 7 \cdot 10^2 + 5 \cdot 10 + 3 \cdot 1 + 9 \cdot 10^{-1} \\ &\quad + 2 \cdot 10^{-2}) = -8753.92 \\ \beta = 2, \qquad (1011.01)_2 &= 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} \\ &= 11 + \tfrac{1}{4} = 11.25\ . \end{aligned}$$

Man möchte die Darstellung (1.1) standardisieren. Um dies zu tun beachtet man, dass jede reelle Zahl $x \neq 0$ geschrieben werden kann als

$$x = \pm\, d \cdot \beta^e \tag{1.2}$$

wobei e eine ganze Zahl ist und d die Ungleichung

$$\beta^{-1} \leq d < 1 \tag{1.3}$$

erfüllt. Ist z.B. in (1.1) die Ziffer $a_n \neq 0$, so würde diese Zahl dargestellt werden durch

$$\pm\ \underbrace{(0.a_n\, a_{n-1} \ldots a_2\, a_1\, a_0\, b_1\, b_2 \ldots)_\beta}_{d} \cdot \beta^{n+1}\ .$$

d heisst Mantisse und e heisst der Exponent. Zum Speichern einer solchen Zahl stellt ein Rechner eine rechnerabhängige Anzahl Stellen zur Verfügung und lässt Exponenten nur in einem festen Bereich zu. Diese Zahlen heissen **Maschinenzahlen**.

n-stellige, normalisierte Maschinenzahl zur Basis β
Diese Zahlen haben die Form

$$x = \pm\, (0.a_1\, a_2 \ldots a_n)_\beta \cdot \beta^e\ .$$

Ist $x \neq 0$, so ist die Mantisse $(0.a_1\, a_2 \ldots a_n)_\beta$ so normalisiert, dass $a_1 \neq 0$ gilt. Der Exponent ist eine ganze Zahl mit Vorzeichen im Intervall $\underline{m} \leq e \leq \overline{m}$, wobei $\underline{m}$ und $\overline{m}$ ganze Zahlen sind.

BEISPIEL:

Basis $\beta = 10$

Dies ist die allgemein übliche Dezimalzahlendarstellung, und man lässt deshalb $(\,)_{10}$ stets weg.

nicht normalisiert	normalisiert
-3.795	$-0.3795 \cdot 10^1$
$16.3 \cdot 10^{-4}$	$0.163 \cdot 10^{-2}$

Basis $\beta = 2$

Im sogenannten Dualsystem werden alle Zahlen nur mit Hilfe der Ziffern 0 und 1 dargestellt:

nicht normalisiert	normalisiert
$(1.01)_2 \left\{ = 1 + 0 \cdot \frac{1}{2} + 1 \cdot \frac{1}{4} = 1.25 \right\}$	$(0.101)_2 \cdot 2^1$

Die Zahlen β, n, $\underline{m}$, $\overline{m}$ sind maschinenabhängig.

BEISPIEL:

		β	n	$\underline{m}$	$\overline{m}$
IEEE P754 Norm$^{(*)}$	einfach genau	2	24	-125	128
	doppelt genau	2	53	-1021	1024
CRAY	einfach genau	2	48	-16384	16383
	doppelt genau	2	96	-16384	16383
TI		10	12	-98	100
IMS 800	einfach genau	2	24	-64	63
	doppelt genau	2	60	-64	64

$^{(*)}$ z.B. HP 9000 Serie, IBM PC/AT mit floating point hardware

Zahlen x mit $|x| \geq \beta^{\overline{m}}$ können nicht mehr dargestellt werden. Der Rechner meldet in diesem Fall einen Überfluss und bricht die Rechnung ab.

Ist $0 < |x| < \beta^{\underline{m}-1}$, so kann die Zahl nicht in normalisierter Form dargestellt werden. Die Maschine rechnet in diesem Fall (Unterfluss) meist mit der Zahl Null weiter.

Wenn auch die Zahlendarstellung in den meisten Maschinen als Basis 2 oder eine Potenz von 2 benutzt, stehen zur Aus- und Eingabe Programme zur Verfügung, die diese Darstellung in die uns geläufigere Dezimalzahldarstellung - also eine Darstellung zur Basis 10 - umrechnen. (Man mache sich klar, dass diese Umrechnung schon bei Ein- und Ausgabe zu Fehlern führt, vgl. Abschnitt 1.2).

Wir nehmen für die Vorlesung und Übung zur Vereinfachung an, dass die Basis der Zahlen tatsächlich stets $\beta = 10$ sei.

Eine n-stellige Gleitpunktzahl zur Basis 10 hat die Gestalt

$$x = \pm\, 0.a_1 \ldots a_n \cdot 10^e \quad \text{mit}$$
$$a_i \in \{0, 1, \ldots, 9) \quad \text{und} \quad a_1 \neq 0, \quad \text{falls} \quad x \neq 0\,.$$

Für die Handrechnung in Beispielen der Vorlesung und Übungen werden wir auch nicht normalisierte Schreibweisen verwenden.

Die meisten Grossrechner haben zwei Arten von Gleitpunktzahlen, solche mit einfacher und solche mit doppelter Genauigkeit.

1.2 Rundungsfehler

Die Menge der im Rechner darstellbaren Zahlen (**Maschinenzahlen**) ist endlich. Eine Zahl, die im Rechner dargestellt werden soll, selbst aber nicht Maschinenzahl ist, muss durch eine Maschinenzahl approximiert werden. Dies geschieht durch **Rundung**. Da Über- und Unterfluss relativ selten vorkommen, nehmen wir vereinfachend an, dass diese nicht auftreten, d.h. der Exponent kann jede beliebige ganze Zahl sein.

Definition der Runderegeln

Der reellen Zahl

$$x = \pm\, 0.a_1\, a_2 \ldots a_n\, a_{n+1}\, a_{n+2} \ldots \cdot 10^e,\ a_1 \neq 0$$

wird die auf n Stellen gerundete Maschinenzahl $f\ell(x)$ wie folgt zugeordnet:

$$\begin{aligned}&\text{Für} \quad 0 \le a_{n+1} \le 4 \ \text{ ist } \ f\ell(x) := \pm\, 0.a_1\, a_2 \ldots a_n \cdot 10^e \ , \\ &\text{für} \quad 5 \le a_{n+1} \le 9 \ \text{ ist } \ f\ell(x) := \pm\, [0.a_1\, a_2 \ldots a_n + 0.\underbrace{00\ldots0}_{n-1 \text{ Nullen}}1] \cdot 10^e \ .\end{aligned}$$

BEISPIELE: bei 3-stelliger Rechnung ($n = 3$)

$$\begin{aligned} f\ell\,(\ \ 0.3571) &= \ \ 0.357 \\ f\ell\,(-0.3576) &= -0.358 \\ f\ell\,(\ \ 0.4997 \cdot 10^{-1}) &= \ \ 0.500 \cdot 10^{-1} \\ f\ell\,(\ \ 0.9996 \cdot 10^{4}) &= \ \ 0.100 \cdot 10^{5} \ . \end{aligned}$$

Der durch Rundung entstehende **absolute Fehler** $x - f\ell(x)$ lässt sich abschätzen durch

$$|x - f\ell(x)| \le 0.\underbrace{00\ldots0}_{n \text{ Nullen}}5 \ \cdot \ 10^e = 0.5 \cdot 10^{-n+e} \ .$$

Ist $x \neq 0$, so erhält man für den **relativen Fehler** $\dfrac{x - f\ell(x)}{x}$

$$\frac{|x - f\ell(x)|}{|x|} \le \frac{0.5 \cdot 10^{-n+e}}{0.1 \cdot 10^e} = 0.5 \cdot 10^{-n+1} \ ,$$

da wegen $a_1 \neq 0$ gilt:

$$|x| \ge 0.1 \cdot 10^e \ .$$

Zur Abkürzung setzen wir

$$u := 5 \cdot 10^{-n} \ .$$

Die Zahl u heisst **Maschinengenauigkeit. Sie ist die beste Schranke für den relativen Fehler bei Rundung**.

Ist die Basis β eine gerade Zahl, so ist

$$u = \frac{\beta}{2} \cdot \beta^{-n}$$

die Maschinengenauigkeit.

u ist die kleinste positive Zahl ϵ mit

$$fl(1+u) \neq 1 \; .$$

BEISPIEL:

Für SUN workstations gilt $\beta = 2$ und $n = 52$. Somit ist $u = 1/2 \cdot 2^{-51} \approx 2.2 \cdot 10^{-16}$.

Für den TI 15C Taschenrechner gilt $\beta = 10$ und $n = 12$. Somit ist $u = 1/2 \cdot 10^{-11} = 5 \cdot 10^{-12}$.

Um Rechenprogramme, in die u wesentlich eingeht, maschinenunabhängig zu machen, kann u durch den folgenden Programmausschnitt approximiert werden:

```
E := 1
REPEAT E : = E/2
UNTIL  1.  = 1. + E
WRITE  E
```

Für u gilt dann $E < u \leq 2 \cdot E$.

Für die Analyse der Rundungsfehlerfortpflanzung (Abschnitt 1.3) ist die folgende Schreibweise nützlich:

$$fl(x) = x(1+\delta) \;\; \text{mit} \;\; |\delta| \leq u \; .$$

δ is hier der durch die Rundung entstehende relative Fehler.

Rundungsfehler bei arithmetischen Operationen

Führt man für zwei Maschinenzahlen x, y die Operationen $x+y$, $x-y$, $x \cdot y$, x/y durch, so braucht das Ergebnis nicht notwendigerweise eine Maschinenzahl zu sein.

BEISPIEL:

Sei $x = 0.345 \cdot 10^2$, $y = 0.784 \cdot 10^2$. Dann sind $x+y$, $x \cdot y$, x/y keine dreistelligen normalisierten Gleitpunktzahlen, wohl aber $x - y$.

Man hat also auf der Maschine nicht die üblichen, sondern neue Operationen: $\oplus$, $\ominus$, $\odot$, $\oslash$.

Konvention:

Unter n-stelliger Rechnung verstehen wir, dass jede Operation $+$, $-$, $\cdot$, $/$ zunächst auf mindestens $n+1$ Stellen genau berechnet und danach auf n Stellen gerundet wird. Erst dann werden weitere Operationen ausgeführt.

Die Gleitpunktoperationen sind also definiert durch:

$$\begin{array}{lcll} x \oplus y & := & f\ell(x+y) & = & (x+y)\,(1+\delta_1) \\ x \ominus y & := & f\ell(x-y) & = & (x-y)\,(1+\delta_2) \\ x \odot y & := & f\ell(x\cdot y) & = & (x\cdot y)\,(1+\delta_3) \\ x \oslash y & := & f\ell(x\,/\,y) & = & (x\,/\,y)\,(1+\delta_4) \end{array}$$

mit $|\delta_i| \leq u$.

Die arithmetischen Gesetze (Assoziativ-, Distributivgesetz) und bekannte Eigenschaften für reelle Zahlen (z.B. binomische Formeln) gelten für diese Maschinenoperationen nicht mehr allgemein.

BEISPIEL:

Es gilt für $x = 3490$, $y = 1$, $z = 4$:

$$(x+y)+z = (y+z)+x = 3495\ .$$

Bei dreistelliger Rechnung ergibt sich dagegen

$$x \oplus y = 3490, \quad \text{also} \quad (x \oplus y) \oplus z = 3490$$

und andererseits

$$y \oplus z = 5, \quad \text{also} \quad (y \oplus z) \oplus x = 3500\ .$$

Das Resultat hängt also von der Summationsreihenfolge ab. Man beachte, dass das zweite Resultat gleich der exakten Summe gerundet auf drei Stellen ist. Wie in diesem Beispiel ist ganz allgemein in der Regel der Fehler am kleinsten, wenn die Summanden in der Reihenfolge aufsteigender Beträge addiert werden.

Wie wesentlich die Summationsreihenfolge ist, sieht man auch bei der folgenden Aufgabe:

Man addiere tausendmal die 1 und einmal 1000. Exakt erhält man 2000. Hingegen ergeben sich bei 3-stelliger Rechnung die folgenden Resultate:

$$((\ldots(((1000 \oplus 1) \oplus 1) \oplus 1)\ldots) \oplus 1) = 1000$$

und

$$(((\ldots(((1 \oplus 1) \oplus 1)\ldots) \oplus 1) \oplus 1000) = 2000\ .$$

Je nach Reihenfolge der Additionen kann als Resultat jede Zahl zwischen 1000 und 2000 erhalten werden, deren letzte Ziffer eine 0 ist. Die Summationsreihenfolge kann jedoch nicht immer frei gewählt werden. Bildet man etwa

$$y_n = y_0 + \sum_{i=1}^{n} x_i\ ,\quad n \in \mathbb{N}\ ,$$

wobei x_i von y_{i-1} abhängt, so ist die Reihenfolge der Summation fest vorgeschrieben.

Abhilfe verschafft hier das Verfahren der **quasi-doppelt-genauen Summation**, bei dem in einem Hilfsspeicher R betragsmässig kleine Summanden aufaddiert werden, die bei n-stelliger Gleitpunktaddition keine Berücksichtigung finden. Ihre Summe wird erst dann zu grösseren Summanden addiert, wenn sie um weniger als n Grössenordnungen von den grösseren Zahlen abweicht.

```
R = 0 ;
Y = 0 ;
FOR       I = 1 TO N DO
          BEGIN
          READ X;
          U : = X + R ;
          V : = Y + U ;
          R : = U - (V - Y) ;
          Y = V;
          END ;
WRITE     Y
```

BEISPIEL:

Die binomischen Formeln haben keine Allgemeingültigkeit mehr. Für $a = 15.6$, $b = 15.7$ ist bei dreistelliger Rechnung

$$(a \ominus b) \odot (a \ominus b) = (a - b)(a - b) = 0.01\,,$$

aber:

$$(a \odot a) \ominus (2 \odot (a \odot b)) \odot (b \odot b) = -1\,.$$

Sogar das Vorzeichen ist falsch!

BEISPIEL:

Für die Matrix

$$A = \begin{pmatrix} 1.12 & 1.10 \\ 1.12 & 1.11 \\ 1.12 & 1.12 \\ 1.12 & 1.13 \end{pmatrix}$$

und den Vektor

$$x = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

ist bei exakter Rechnung $(xA)^T\ Ax = 0.0006 > 0$. Bei dreistelliger Rechnung ist dagegen

$$A^T \odot A = \begin{pmatrix} 5.00 & 4.99 \\ 4.99 & 4.97 \end{pmatrix}$$

und somit

$$x^T \odot (A^T \odot A) \odot x = -0.01 < 0\,.$$

Hier is wiederum sogar das Vorzeichen falsch. Das Vorzeichen wird auch falsch, wenn man zunächst $A^T A$ exakt berechnet und dann auf 3 Stellen rundet. Man erhält nämlich

$$f\ell(A^T A) = \begin{pmatrix} 5.02 & 5.00 \\ 5.00 & 4.97 \end{pmatrix}$$

und somit

$$x^T \odot f\ell(A^T A) \odot x = -0.01 < 0\,.$$

Man beachte, dass $A^T A$ positiv definit ist, während sowohl $A^T \odot A$ als auch $f\ell(A^T A)$ indefinit sind.[1]

Bei Subtraktion fast gleicher Zahlen tritt das Problem der **Auslöschung** auf. Mit 3-stelliger Rechnung erhält man

$$0.73563 - 0.73441 = 0.736 - 0.734 = 0.002 = 0.2 \cdot 10^{-2} ,$$

während die exakte Rechnung

$$0.73563 - 0.73441 = 0.00122$$

liefert. Man beachte, dass der absolute Fehler von derselben Grössenordnung ist wie jener, der beim Runden einer der Ausgangszahlen auf 3 Stellen entsteht. Hingegen ist der relative Fehler viel grösser als bei den Ausgangszahlen. Sogar die erste Stelle ist falsch. Dieses Phänomen tritt immer dann auf, wenn bei n-stelliger Rechnung zwei Zahlen subtrahiert werden, bei denen die ersten t Ziffern übereinstimmen mit $t \geq 1$. Wenn man z.B. die zwei Zahlen

$$\begin{aligned} x &= 0.b_1\, b_2 \ldots b_t\, b_{t+1}\, b_{t+2} \ldots b_n \cdot 10^e , \\ y &= 0.b_1\, b_2 \ldots b_t\, \widetilde{b}_{t+1}\, \widetilde{b}_{t+2} \ldots \widetilde{b}_n \cdot 10^e \end{aligned}$$

mit $b_{t+1} > \widetilde{b}_{t+1}$ und $b_{t+2} > \widetilde{b}_{t+2}$ subtrahiert, so erhält man

$$x - y = 0.\underbrace{0 \ldots 0}_{t \text{ Nullen}}\ \hat{b}_{t+1}\, \hat{b}_{t+2} \ldots \hat{b}_n \cdot 10^e .$$

Hier ist $\hat{b}_{t+1} = b_{t+1} - \widetilde{b}_{t+1}$ (siehe auch Abschnitt 1.3).

BEISPIEL: *Numerische Differentiation von* $\cos x$.

Bekanntlich gilt

$$\frac{d}{dx} \cos x \Big|_{x=\frac{\pi}{2}} = -\sin\frac{\pi}{2} = -1 = \lim_{h \to 0} \frac{\cos(\frac{\pi}{2} + h) - \cos(h)}{h} .$$

[1] Zur Definition von "positiv definit" vgl. gegebenenfalls den Abschnitt über lineare Gleichungssysteme.

Der Differenzenquotient $d(h) = \frac{\cos(\frac{\pi}{2}+h)-\cos(h)}{h}$ strebt somit für h nach 0 gegen -1. Auf einem Taschenrechner mit $u = 5 \cdot 10^{-12}$ erhält man die folgende Tabelle:

h	$d(h)$
0.1	-0.998334167
0.01	-0.999983333
0.001	-0.999999833
0.0001	-0.999999998
0.00001	-1.000000000
$\vdots$	$\vdots$
10^{-9}	-1.000000000
10^{-10}	-1.000000000
10^{-11}	0.
10^{-12}	0.

Man erkennt die vorausgesagte Konvergenz gegen -1 für $h \geq 10^{-5}$. Hingegen tritt für $h \leq 10^{-11}$ Auslöschung ein und das Resultat ist ganz falsch.

BEISPIEL:

Das Polynom $p(x) = \sum_{i=0}^{20} c_i\, x^i = \prod_{i=0}^{19} (x - i)$ hat die Nullstellen $0, 1, 2, \ldots, 19$ und ist somit symmetrisch bezüglich $x = 9.5$. Die MATLAB Routine

```
p = poly(0:19)
```

berechnet im Vektor p die Polynormkoeffizienten aus. Das MATLAB Programm

```
p = poly(0:19);
x = -0.00001 : 0.0000001 : 0.00001 ;
y = polyval (p,x) ;
plot (x,y)
print

xx = x + 19 ;
yy = polyval (p, xx) ;
print
exit
```

zeichnet zunächst das Polynom im x Intervall $[-10^{-5}, 10^{-5}]$, siehe Fig. 1.1, und danach im Interval $[19 - 10^{-5}, 19 + 10^{-5}]$, siehe Fig. 1.2. Da das

Polynom symmetrisch bezüglich $x = 9.5$ ist, erkennt man, dass offenbar bei der Polynomauswertung bei $x \approx 19$ grosse Rundungsfehler eingetreten sind. Das Resultat in Fig. 1.2 ist völlig falsch.

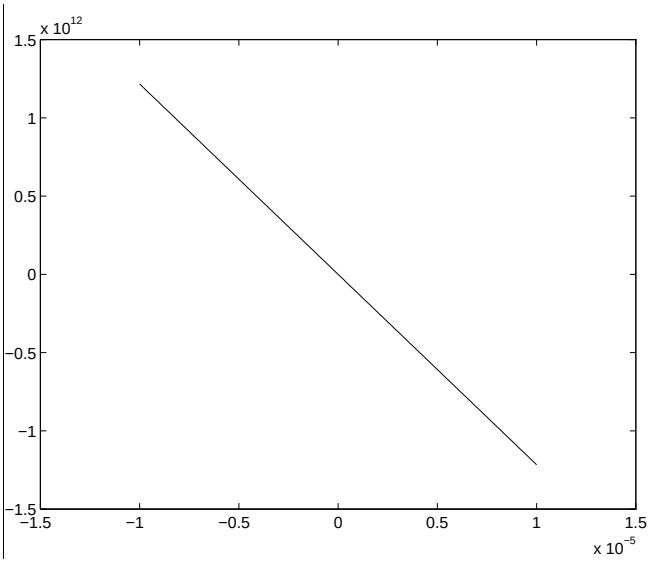

Fig. 1.1

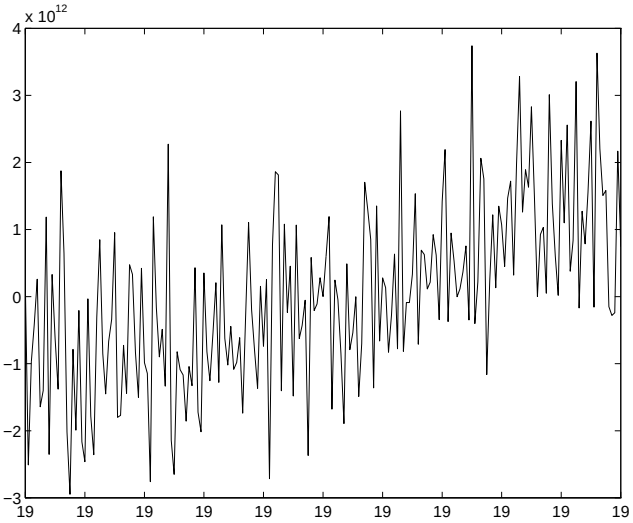

Fig. 1.2

1.3 Elementare Fehlerfortpflanzung

Wir wollen nun untersuchen, wie sich bei Auswertung einer Funktion f der Fehler im Argument x auf den Funktionswert $f(x)$ auswirkt. Infolge von Fehlern sei für die Zahl x nur eine Approximation $\tilde{x}$ bekannt. Unter der Annahme, dass $f(x)$ auch exakt berechnet werden kann, erhält man für $f \in C^2$ nach dem Satz von Taylor

$$f(\tilde{x}) = f(x) + f'(x)(\tilde{x} - x) + \frac{1}{2}\, f''(\xi)(\tilde{x} - x)^2 \,,$$

und daraus

$$f(\tilde{x}) - f(x) \approx f'(x)(\tilde{x} - x) \,.$$

Der absolute Fehler von $f(\tilde{x})$ ist also in erster Näherung proportional zum absoluten Fehler von $\tilde{x}$, wobei $f'(x)$ der Proportionalitätsfaktor ist.

Der relative Fehler[2] ist für $x \neq 0$ und $f(x) \neq 0$

$$\frac{f(x) - f(\tilde{x})}{f(x)} \approx \frac{f'(x)}{f(x)}\, x \;\cdot\; \frac{x - \tilde{x}}{x} \,.$$

Der Proportionalitätsfaktor ist hier

$$f'(x) \cdot \frac{x}{f(x)} \,.$$

Ganz allgemein möchte man das Verhältnis zwischen dem **relativen Fehler der Eingabe** und dem **relativen Fehler des Ergebnisses** messen:

$$\left|\frac{f(x) - f(\tilde{x})}{f(x)}\right| \quad \approx K \quad \left|\frac{x - \tilde{x}}{x}\right|$$

relativer Fehler des Ergebnisses — relativer Fehler der Eingabe

Der Proportionalitätsfaktor ist somit

$$K \approx \left| f'(x)\, \frac{x}{f(x)} \right| \,.$$

[2] Häufig werden relative Fehler in Prozent (%) angegeben. Ist $r := (x - \tilde{x})/x$ der relative Fehler, so ist $r_p := 100\, r$ der relative Fehler in Prozent. Oft wird nur $|r_p|$ angegeben.

Wir führen an dieser Stelle die **Konditionszahl** K_x ein, die man zum Beispiel wie folgt definieren könnte:

$$K_x = \max_{\widetilde{x}} \left\{ \left| \frac{f(x) - f(\widetilde{x})}{f(x)} \right| \Big/ \frac{|x - \widetilde{x}|}{|x|} , \text{ für } \widetilde{x} \text{ nahe bei } x \right\} .$$

Diese Definition soll nicht streng formalistisch gesehen werden, denn bei der Konditionszahl geht es nur um Grössenordnungen.

Man sagt, das Problem der Auswertung von f sei **schlecht konditioniert**, wenn $|K_x|$ sehr gross ist, z.B. $K_x = 10^6$ oder $K_x = 10^{40}$. Solche Fälle sind **keine** Seltenheit.

BEMERKUNG:

Selbst, wenn x exakt auf dem Rechner darstellbar ist, d.h. wenn gilt $x = f\ell(x)$, kann $f(x)$ nur durch eine Approximation $\hat{f}$ angenähert werden. Bestenfalls kann man erreichen, dass gilt

$$\hat{f} = f\ell(f(x)) = f(x)(1 + \delta) \text{ mit } |\delta| \leq u .$$

BEISPIEL: *(Subtraktion)*

Sei α eine Maschinenzahl, in der Rechenanlage also exakt darstellbar. Für die Subtraktion

$$f(x) = \alpha - x$$

einer beliebigen reellen Zahl x von α ergibt sich der relative Fehler ($f'(x) = -1$)

$$\delta_f := \frac{f(x) - f(\widetilde{x})}{f(x)} \approx \frac{x}{\alpha - x} \cdot \frac{x - \widetilde{x}}{x} ,$$

der umso grösser ist, je kleiner $|\alpha - x|$ ist (Auslöschung).

Sei $\alpha = 234$, und x werde jeweils auf 3 Stellen zu $\widetilde{x}$ gerundet.

x	233400	23340	2334	233.4
$\widetilde{x}$	233000	23300	2330	233
absoluter Fehler in $\widetilde{x}$	400	40	4	0.4
relativer Fehler in $\widetilde{x}$	0.00171138...	0.0017138...	0.0017138...	0.0017138...
$\alpha - x$ bei exakter Rechnung	-233166	-23106	-2100	0.6
absoluter Fehler von $\alpha - \widetilde{x}$ gegen $\alpha - x$	400	40	4	0.4
relativer Fehler von $\alpha - \widetilde{x}$	-0.001715...	-0.001731...	-0.001905...	0.6666...

BEISPIEL: *(Multiplikation)*

Sei α wieder eine Maschinenzahl,

$$f(x) = \alpha x, \quad f'(x) = \alpha \, .$$

Dann ergibt sich

$$K_x = \alpha \cdot \frac{x}{\alpha x} = 1 \, .$$

Bei Multiplikation mit einer Maschinenzahl sind also die relativen Fehler von Eingabe x und Ausgabe $f(x)$ gleich. Dasselbe gilt bei Division α/x und x/α.

In der Gleitpunktarithmetik sind also Multiplikation und Division harmlos, sofern sie nicht zum Über- oder Unterfluss führen.

BEISPIEL: *(Potenzieren)*

Findet man mit einer Maschinenzahl α den Wert

$$f(x) = x^\alpha \ ,$$

so erhält man mit

$$f'(x) = \alpha x^{\alpha-1} \quad (\alpha \neq 0)$$

den Faktor

$$K_x = \alpha \cdot x^{\alpha-1} \ \frac{x}{x^\alpha} = \alpha \ .$$

Dies bedeutet, dass z.B. für $\alpha = 2$ (Quadrieren) der relative Fehler verdoppelt und für $\alpha = 1/2$ (Wurzelziehen) halbiert wird.

In der nachfolgenden Tabelle ist $x = 100$, $\widetilde{x} = 99$, so dass der relative Fehler 0.01 ist. Der absolute Fehler von $\widetilde{x}^\alpha$ ist $x^\alpha - \widetilde{x}^\alpha$, und der relative Fehler ist $\delta_{x^\alpha} := (x^\alpha - \widetilde{x}^\alpha)/x^\alpha$.

α	x^α	$\widetilde{x}^\alpha$	$\lvert x^\alpha - \widetilde{x}^\alpha\rvert$	$\lvert\delta_{x^\alpha}\rvert$	$\lvert\frac{\delta_{x^\alpha}}{\delta_x}\rvert$
0.25	3.162278	3.154342	0.0079355	0.0025094	0.250943
0.5	10	9.9498744	0.0501256	0.00501256	0.501256
1	100	99	1	0.01	1
1.5	1000	985.03756	14.962438	0.0149624	1.49624
2	10000	9801	199	0.0199	1.99
10	10^{20}	$9.0438207 \cdot 10^{19}$	$9.56179 \cdot 10^{18}$	0.0956179	9.56179

BEISPIEL: *(Multiplikation $x - y$)*

Multipliziert man zwei reelle Zahlen x, y, die mit relativen Fehlern

$$\delta_x = \frac{\widetilde{x} - x}{x} \ , \quad \delta_y = \frac{\widetilde{y} - y}{y}$$

behaftet sind, so folgt aus

$$\widetilde{x} = x(1 + \delta_x) \ ,$$
$$\widetilde{y} = y(1 + \delta_y)$$

für die Approximation $\widetilde{x}\widetilde{y}$ von xy:

$$\begin{aligned} \widetilde{x}\widetilde{y} &= x(1+\delta_x)\, y(1+\delta_y)\ , \\ &= xy(1+\delta_x+\delta_y+\delta_x\delta_y)\ . \end{aligned}$$

Da $\delta_x\delta_y$ wesentlich kleiner als $\delta_x+\delta_y$ ist, addieren sich in erster Näherung die relativen Fehler.

Es ist also bei kleinen Werten von δ_x und δ_y der relative Fehler δ_{xy} von $\widetilde{x}\widetilde{y}$ ungefähr

$$\delta_{xy} \approx \delta_x + \delta_y\ .$$

Für den relativen Fehler $\delta_{y/x}$ der Division von $\widetilde{y}$ durch $\widetilde{x}$ findet man analog

$$\delta_{y/x} := \frac{\widetilde{y}/\widetilde{x} - y/x}{y/x} \approx \delta_y - \delta_x\ .$$

Bei geeigneten Grössen und Vorzeichen der relativen Fehler δ_x und δ_y können sich diese natürlich nahezu annullieren. Da jedoch normalerweise über die Vorzeichen der relativen Fehler keine Information vorliegt, nimmt man den schlimmsten Fall an, also

$$|\delta_{xy}| \lessapprox |\delta_x| + |\delta_y|$$

und

$$|\delta_{y/x}| \lessapprox |\delta_x| + |\delta_y|\ .$$