Numerik für LAK WS 2000/2001

D. Lind BUGH Wuppertal

Inhaltsverzeichnis

1 Zahldarstellungen und Fehleranalyse

1.1 Zahldarstellung

In der (reinen) Mathematik sind Zahlen Objekte, die als ganze Zahlen, rationale Zahlen, relle Zahlen (endliche und „unendliche“ Dezimalbrüche) und komplexe Zahlen (Zahlen des Typs $a + b\,\mathrm{i}$) angegeben und geschrieben werden.

Werden Rechner eingesetzt, so ist eine Darstellungen von Zahlen nur in eingeschränkter Form möglich:

Analogrechner: Zahlen werden durch physikalische Größen (meistens *Spannung* oder Stromstärke) repräsentiert.

Digitalrechner: Zahlen werden durch Ziffernfolgen *endlicher* Länge dargestellt. Eventuell ist die Länge im Rahmen der durch den Speicher des Rechners gegebenen Grenzen variabel (beim *kontrollierten Rechnen* manchmal nötig), meistens jedoch fest.

Beim Taschnerechner werden Dezimalziffern verwendet (oft bis zu 12, von denen 10 angezeigt werden), bei der Programmierung von Computern wird intern meistens auf Darstellungen im *Binärsystem* zurückgegriffen. Solche Darstellungen sind *Bitfolgen*, in denen die Ziffern 0 und 1 verwendet werden.

Wir gehen hier nur auf Digitalrechner ein, da sie heute bei fast allen Anwendungen dominieren und in Form von Taschenrechnern und PCs Allgemeingut geworden sind.

Vorab einige Bemerkungen zur Binärdarstellung: Interpretiert man eine binäre Ziffernfolge wie üblich von rechts nach links im Dualsystem, so ist unter $\boxed{a_n a_{n-1} \dots a_1 a_0}$

mit $a_0, \dots, a_n \in \{0, 1\}$ die Zahl $\sum_{k=0}^{n} a_k 2^k$ zu verstehen.

Beispiel: $\boxed{0001\,0000\,0001\,0000}$ ist als natürliche Zahl aufgefasst eine Darstellung für 4112.

Sollen auf diese Weise ganze Zahlen dargestellt werden, so muss zusätzlich eine Stelle ($=1$ Bit) für das Vorzeichen reserviert werden.

Man nennt bei Binärdarstellungen eine Ziffernfolge der Länge

4 Bit ein halbes ***Byte*** (auch Halfbyte oder Nibble genannt),
8 Bit 1 ***Byte***,
16 Bit 1 ***Wort***,
32 Bit 1 ***Langwort***.

Für die Darstellung ganzer Zahlen (Datentyp ***integer***) verwendet man oft 1 Wort und kann so Zahlen z von $-32\,767$ bis $+32\,767$ darstellen. Verwendet man 1 Langwort (Datentyp ***long integer***), so kommt man bis $-2\,147\,483\,647 \leq z \leq +2\,147\,483\,647$.

Es sei angemerkt, dass unter einem ***Wort*** auch das kleinste einzeln ansprechbare Speicherfeld eines Digitalrechners verstanden werden kann.

Bei binärer Darstellung von Dezimalstellen braucht man im Prinzip nur ein halbes Byte je Dezimalziffer. Damit lassen sich sogar die Ziffern von 0 bis 9 und darüber hinaus die Zahlen 10, 11, 12, 13, 14 und 15 darstellen. Bei der Darstellung von Binärdaten auf dem Bildschirm werden daher oft Binärfolgen als Folgen von ***Bytes*** angezeigt. Dabei werden die Werte 0 – 9 dezimal angegeben und A steht für 10, B für 11, C für12, D für 13, E für 14 und F für 15.

Da sich in einem Byte nicht noch für jede der Ziffern 0 bis 9 zusätzlich ein Vorzeichen unterbringen lässt, muss bei der Dezimaldarstellung ganzer Zahlen durch Bytes neben der maximalen Ziffernzahl zusätzlich eine Stelle (= 1 Byte) für das Vorzeichen berücksichtigt werden. Man verschwendet also sowohl bei den Ziffern als auch beim Vorzeichen Speicherplatz, wenn man einzelne Dezimalziffern speichern will.

Wegen der bequemeren Lesbarkeit von Beispielen werden wir trotzdem ab jetzt so tun, als ob die Zahldarstellungen in dezimaler Form erfolgten. Wir zählen aber das Vorzeichen bei der Länge der Dezimaldarstellung nicht mit, da es in der tatsächlich üblichen binären Darstellung kaum ins Gewicht fällt. Ein negatives Vorzeichen wird im Bedarfsfall durch einen Querstrich über der ersten Ziffer angegeben.

Dann lassen sich bei der Nutzung n-stelliger Speicherfelder im Wesentlichen zwei Darstellungsvarianten für Zahlen unterscheiden:

(I) Festkommadarstellung (auch Festpunktdarstellung genannt).

$$n = \underbrace{n_1}_{\substack{\text{Vorkomma-}\\ \text{stellen}}} + \underbrace{n_2}_{\substack{\text{Nachkomma-}\\ \text{stellen}}}$$

Beispiel ($n = 8, n_1 = 6, n_2 = 2$, kaufmännisches Rechnen):

$+340,21$ wird gespeichert als | 0 0 0 3 4 0 \| 2 1 |,

$-1432,50$ wird gespeichert als | $\overline{0}$ 0 1 4 3 2 \| 5 0 |.

Das Komma ist durch die feste Stellenzahl vorgegeben. Der Trennstrich in den Beispielen wurde nur wegen der besseren Lesbarkeit eingefügt.

Es sind zwar Rechner realisierbar, die mit variablen Darstellungslängen arbeiten (vor Jahrzehnten bot die IBM einen solchen „Mathematikrechner“ unter dem Namen 1620 an), in den meisten Programmiersprachen ist dieser Datentyp jedoch unter den numerischen Datentypen nicht vorgesehen. In der Numerik üblich ist die

(II) Gleitkommadarstellung (auch Gleitpunktdarstellung genannt)

$$\begin{array}{ccccc} n & = & t & + & e \\ & & \text{Mantissen-}\\ & & \text{stellen} & & \text{Exponenten-}\\ & & & & \text{stellen} \end{array}$$

Ist E die Basis der Darstellung, so verstehen wir unter $\boxed{a_1a_2\dots a_tb_1b_2\dots b_e}$ die Zahl $a\cdot E^{b-t}$ und nennen die durch $(a_1\dots a_t)$ angegebene Zahl $a\cdot E^{-t}$ die ***Mantisse*** der Darstellung. Die durch $(b_1\dots b_e)$ angegebene Zahl b heißt ***Exponent*** der Darstellung.

Wir bezeichnen im Folgenden den größten darstellbaren Exponenten mit b_{max}, den kleinsten mit b_{min}. Wenn der Exponent mit Vorzeichen gespeichert wird, wäre z.B. bei 8 Binärstellen $b_{min}=-128$ und $b_{max}=+127$. Man vereinbart nun, dass bei der Speicherung eines Gleitkommawertes an Stelle des tatsächlichen Exponenten b die Zahl $b^{(+)}:=b-b_{min}$ gespeichert wird. Dann ist $b^{(+)}$ eine nichtnegative Zahl. Beim Rechnen und der Ausgabe wird der ***Offset*** $-b_{min}$ (er ist **positiv**!) wieder subtrahiert.

Eine Mantisse im Stellenwertsystem zur Basis E ist nach unserer Vereinbarung die Zahl $0.a_1a_2\dots a_t$ (wir schreiben das Komma als Punkt, da dies später bei Intervallangaben günstiger ist). Abweichend von dieser Konvention wird bei manchen PASCAL-Dialekten die Mantisse solcher Darstellungen in der Form $a_1.a_2\dots a_t$ ausgegeben und dafür der Exponent um 1 vermindert.

Dies zeigt, dass solche Darstellungen nicht eindeutig sind. Für $E=10, t=6$ und $e=3$ lässt sich z.B. $10,03$ in der Form $0.1003\cdot 10^2$, $1.003\cdot 10^1$, $10.03\cdot 10^0$, $100.3\cdot 10^{-1}$, $1003\cdot 10^{-2}$, … schreiben.

Zur Herstellung der Eindeutigkeit verlangen wir, dass die Mantisse von der Form $0.a_1a_2\dots a_t$ mit einer von 0 verschiedenen Ziffer a_1 ist:

Definition 1.1 *Eine Gleitpunktdarstellung mit t-stelliger Mantisse*

$$a:=0.a_1a_2\dots a_t \quad \textit{bzw.} \quad a:=-0.a_1a_2\dots a_t$$

heißt normalisiert, wenn $a_1\neq 0$ gilt.

Die Zahl 0 lässt sich nicht normalisieren und wird daher mit lauter Nullen in der Mantisse und dem gespeicherten Exponenten $b^{(+)}$ angegeben.

Bei binärer Zahldarstellung kann man nun das Vorzeichen der Mantisse bei normalisierten Darstellungen ohne Zusatzstellen angeben und so ein Bit für die Mantissenlänge gewinnen (so war es beim PASCAL für den ATARI): da a_1 in diesem Fall stets den Wert 1 hat, kann man dies beim Rechnen voraussetzen und gibt an Stelle der 1 das Vorzeichen mittels a_1 an (z.B. 0 für + und 1 für −):

Beispiel ($t = 16, e = 8, E = 2$):

4112,0 würde gespeichert als

$0000000010000000 b_1 b_1 \ldots b_8$

da die Zahl positiv ist (in der Mantisse hat man sich an der ersten Stelle eine 1 zu denken!). Der Exponent wäre nach unserer Konvention 00001101 (=13), würde jedoch um den Offset 128 erhöht als 10001101 gespeichert.

Bevor wir die „Niederungen“ der Darstellung in realen Rechner verlassen, sollen noch kurz die gbräuchlichsten Typen für Gleitpunktzahlen angegeben werden:

Single: 4 Byte (=32 Bit), die sich auf eine Mantisse von 3 Byte (=24 Bit) und einen Exponenten von 1 Byte Länge verteilen.

Real: 6 Byte (=48 Bit), die sich auf eine Mantisse von 5 Byte (=40 Bit) und einen Exponenten von 1 Byte Länge verteilen.

Double: 8 Byte (=64 Bit), die sich auf eine Mantisse von 53 Bit und einen Exponenten von 11 Bit Länge verteilen.

Extended: 10 Byte (=80 Bit), die sich auf eine Mantisse von 8 Byte (=64 Bit) und einen Exponenten von 2 Byte Länge verteilen.

Es sei nur noch angemerkt, dass die binäre Darstellung eine Quelle für kleine Umwandlungsfehler ist. Dies konnte man früher sehr leicht demonstrieren, indem man bei einem BASIC-Rechner folggenden Befehl eintippte: PRINT(2000.1-2000). Nach Betätigung der Taste RETURN wurde dann eine Zahl ausgegeben, die mit 0.09999 begann!

Ab jetzt werden wir die Mantissenstellen nur für Ziffern verwenden und das Vorzeichen separat vor die Darstellung einer Zahl z setzen. Beim Exponenten b halten wir uns an die Vereinbarung $b_{min} \leq b \leq b_{max}$, trennen ihn von der Mantisse durch ein nichtkursives E und versehen ihn mit einem Vorzeichen. So ist z.B. -0.2045679907 E +05 eine normalisierte Darstellung der Zahl 20 456,79907.

Definition 1.2 *Wenn die Mantissenlänge t, die Exponentenlänge e und die Basis E bei einem Rechner fest gewählt sind, so heißen alle normalisiert exakt darstellbaren Zahlen einschließlich der Zahl 0 seine* **Maschinenzahlen**.

Die Menge dieser Maschinenzahlen soll mit $\mathcal{A}$ bezeichnet werden. Mit $\mathcal{A}^*$ sei die Menge $\mathcal{A} \setminus \{0\}$ der normalisierten Maschinenzahlen bezeichnet.

1.2 Rundung

Wir setzen ab jetzt zusätzlich voraus, dass die Basis E von Maschinenzahlen von der Form $2H$ ist, da dies die Definition von ***Rundungsregeln*** erleichert. Positive Maschinenzahlen sollen also von der Form $0.a_1a_2\ldots a_t \cdot E^b$ mit $0 < a-1 \leq E-1$, $0 \leq a_2 \ldots, a_t \leq E-1$ und $b_{min} \leq b \leq b_{max}$ mit $E = 2H$ sein.

Ist nun eine solche Menge $\mathcal{A}$ von Maschinenzahlen fest gewählt, so soll die Umwandlung einer gegebenen Zahl x in eine Gleitpunktdarstellung durch eine Funktion rd erfolgen, die $\mathbb{R}$ so nach $\mathcal{A}$ so abbildet, dass für alle $x \in \mathbb{R}$ gilt:

$$|\mathrm{rd}(x) - x| \leq |y - x| \quad \text{für alle} \quad y \in \mathcal{A} \quad \text{(Minimalitätsbedingung für Rundungsfehler)}$$

Offensichtlich muss $\mathrm{rd}(x) = x$ für alle $x \in \mathcal{A}$ gelten. Eine Funktion mit der verlangten Eigenschaft erhält man durch folgende Vereinbarung, in der $M := H - 1$ gesetzt ist:

Schritt 1: Schreibe $|x|$ in der Form $0.x_1x_2\ldots x_tx_{t+1}\ldots \cdot E^y$ mit geeignetem Exponenten y.

Schritt 2: Setze $x^* := \begin{cases} (0.\,x_1\ldots x_{t-1}x_t + 0.0\ldots 01)\cdot E^y, & \text{falls } x_{t+1} \geq H\,, \\ 0.\,x_1\ldots x_{t-1}x_t \cdot E^y, & \text{falls } x_{t+1} < H\,. \end{cases}$

und normalisiere die Darstellung in der Form $0.a_1\ldots a_t \cdot E^b$.

Schritt 3: Setze $\mathrm{rd}(x) := \begin{cases} 0.0\ \ldots\ 0 \cdot E^0, & \text{falls } b < b_{min}, \\ \mathrm{sgn}(x)0.M\ldots M \cdot E^{b_{max}}, & \text{falls } b > b_{max}, \\ \mathrm{sgn}(x)0.\,a_1\ \ldots\ a_t \cdot E^b, & \text{falls } b_{min} \leq b \leq b_{max}. \end{cases}$

Hier ist sgn die durch $\mathrm{sgn}(x) := \begin{cases} -1, & \text{falls } x < 0, \\ 0, & \text{falls } x = 0, \\ +1, & \text{falls } x > 0. \end{cases}$ definierte ***Signumfunktion***.

1.2.1 Fehlertypen

Wir gehen jetzt erst einmal von exakten Werten und beliebigen Näherungen aus und definieren:

Definition 1.3 *Sei $x \in \mathbb{R}$ und $\tilde{x}$ eine Näherung für x. Dann heißt*

$\Delta x := \tilde{x} - x$ *der (***absolute***)* **Fehler** *der Näherung.*
Im Falle $x \neq 0$ *heißt*
$\varepsilon_x := \frac{\Delta x}{x}$ *der* **relative Fehler** *der Näherung.*

Bei vorgegebenen Maschinenzahlen zur Basis $E = 2H$ gilt (M sei wieder gleich $E-1$):

Wenn x eine Zahl ist, die in der Form

$$\operatorname{sgn}(x)\, 0.a_1a_2a_3 \ldots \cdot E^b \quad \text{mit} \quad b_{min} \leq b \leq b_{max}$$

normalisiert darstellbar ist, so beträgt der relative Fehler $\varepsilon_{\mathrm{rd}(x)} := \frac{\mathrm{rd}(x)-x}{x}$
wegen $|x| \geq 0.1 \cdot E^b$ dem Betrag nach höchstens

$$\frac{H \cdot E^{-(t+1)} \cdot E^b}{0.1 \cdot E^b} = H \cdot E^{-t}\,.$$

Man nennt diese Größe die **Maschinengenauigkeit**. Wir bezeichnen sie mit eps.

Wenn in der exakten normalisierten Darstellung von x nach dem Runden nicht neu normalisiert werden muss, gilt $|\varepsilon_{\mathrm{rd}(x)}| \leq \mathrm{eps}$ sogar noch im Falle $b = b_{max}$.

Gilt $b > b_{max}$, so kann $|\varepsilon_{\mathrm{rd}(x)}|$ beliebig nahe an 1 herankommen. Gilt $x \neq 0$ und $\mathrm{rd}(x) = 0$, so ergibt sich $|\varepsilon_{\mathrm{rd}(x)}| = 1$. Es gilt also für eine Zahl $x \in \mathbb{R}$ nur dann $|\varepsilon_{\mathrm{rd}(x)}| \leq \mathrm{eps}$, wenn die Bedingung

$$\mathrm{rd}(x) \in \mathcal{A} \setminus \{0.M \ldots M \cdot E^{b_{max}}, -0.M \ldots M \cdot E^{b_{max}}, 0\}$$

erfüllt ist. Man bezeichnet den Fall $b > b_{max}$ als ***(Exponenten-)Überlauf***, den Fall $b < b_{min}$ als ***(Exponenten-)Unterlauf***. Während ein aus Berechnungen resultierender Überlauf bei den meisten Programmiersprachen mit einem anschließenden Programmabbruch (Laufzeitfehler) gemeldet wird, erfolgt die Ersetzung des Ergebnisses durch 0 im Falle des Unterlaufs meistens stillschweigend. Manchmal kann man eine Compileroption setzen, nach der auch Unterläufe gemeldet werden.

1.2.2 Handhabung von gerundeten Näherungswerten

Ist $|\tilde{x}|$ ein Näherungswert für $|x|$, der in der normalisierten Form $0.a_1 \ldots a_s \cdot E^b$ ohne Exponentenüberlauf oder -unterlauf durch Runden auf die s-te Mantissenstelle bzw. in der Form $v_1v_2 \ldots v_{s-k}.n_1 \ldots n_k$ durch Runden auf die k-te Nachkommastelle bestimmt wurde, so nennt man alle seine s Ziffern **zuverlässig**.

Eine griffigere (leicht abweichende!) Definition, in der beide Fälle zusammen erfasst werden, ist die Forderung, dass der **relative Fehler** von $|\tilde{x}|$ kleiner oder gleich $H \cdot E^{-s}$ sein muss.

Alle Ziffern von $|\tilde{x}|$, die auch bei Rundung von x auf spätere Stellen erhalten bleiben, heißen **gültige** Ziffern.

Wir geben eine Näherung $\tilde{z}$ für z mit zuverlässigen Ziffern in der Form $z \doteq \tilde{z}$ an.

Beispiel ($\pi = 3.14159265\ldots$):

$\pi \doteq 3.1416$	Alle Ziffern sind zuverlässig, da auf 5 Stellen gerundet wurde. Die Ziffern 3,1,4,1 sind sogar gültig
$\pi \approx 3.14160$	Die Ziffern 6 und 0 sind nicht zuverlässig, da sie nicht aus einer Rundung resultieren (hier müssten 5 und danach 9 stehen).

Bei der Angabe von Fehlerschranken in der Form $z = u \pm \Delta u$ wird oft vereinbart, dass die letzte Ziffer der Fehlerschranke im Stellenwertschema an derselben Stelle stehen muss, wie die letzte Ziffer von u.

Also heißt es z.B. *nicht* $u = 8.141 \pm 0.593 \cdot 10^{-2}$, sondern $u = 8.141 \pm 0.6 \cdot 10^{-2}$ bzw. $u = 8.141 \pm 0.006$.

Bei Rechnungen von Hand normalisiert man nur selten und setzt lieber den Punkt (also das Komma) an die passende Stelle. Dann heißt Runden auf t Stellen bei einer Darstellung des Typs

$$a_1 a_2 \ldots a_k \,.\, a_{k+1} \ldots a_t a_{t+1} \ldots \quad (a_1 \neq 0)\,,$$

dass auf die t-te Stelle gerundet wird (nach Inspektion der Ziffer a_{t+1}).

Bei Darstellungen der Form

$$0\,.\,\underbrace{0\ldots0}_{v \text{ Nullen}}\, a_1 \ldots a_t a_{t+1} \quad (a_1 \neq 0)$$

rundet man auf die $(t+v)$-te Stelle, zählt also die führenden Nullen nicht mit.

Beispiele ($t = 4$)*:*

$0.12378 \;\to\; 0.1238\,,$ $\qquad 0.025773 \;\to\; 0.02577\,,$ $\qquad 0.0071552 \;\to\; 0.007155\,.$

1.3 Gleitpunktarithmetik

Wir kalkulieren ab jetzt Exponentenüberläufe nicht mehr ein (wer programmiert, muss darauf selber achten!) und nehmen an, dass mit Darstellungen aus der Menge $\mathcal{A}$ wie folgt gerechnet wird:

Das Symbol $*$ vertrete eine der Operationen $+$, $-$, $\cdot$ und $:$ in $\mathbb{R}$.
Dann soll für die entsprechende Rechnerarithmetik $\circledast$ gelten:

$$\mathrm{rd}(x) \;\underset{\text{Operation der Maschine mit Maschinenzahlen}}{\circledast}\; \mathrm{rd}(y) \;=\; \mathrm{rd}(\,\mathrm{rd}(x) \;\underset{\text{exaktes Rechnen mit Maschinenzahlen}}{*}\; \mathrm{rd}(y)\,) \quad \text{für alle } \mathrm{rd}(x), \mathrm{rd}(y) \in \mathcal{A}$$

Bei den Operationen $\oplus$, $\ominus$, $\odot$ und $\oslash$ soll also der relative Fehler des Ergebnisses dem Betrag nach höchtens eps betragen, wenn kein Überlauf oder Unterlauf auftritt.

Man realisiert diese Forderung, indem man während der Operation $*$ mit mehr Mantissenstellen arbeitet und am Schluss auf die vorgeschriebene Stellenzahl rundet (so wird das auch bei korrekten Arithmetikbibliotheken von Programmiersprachen realisiert).

Wir betrachten dazu einige ***Beispiele*** und vereinbaren zur Kennzeichnung der Basis folgende Kurznotation: an Stelle von $a \cdot E^b$ wird $a_E b$ geschrieben. Als Mantissenlänge wählen wir $t = 4$.

Multiplikation: $0.1235_{10}3 \odot 0.7430_{10}2$**.**

$$\begin{aligned} 0.1235_{10}3 \cdot 0.7430_{10}2 &= 0.09176050_{10}5 = 0.9176050_{10}4 \\ \mathrm{rd}(0.9176050_{10}4) &= 0.9176_{10}4 \\ \text{Also } 0.1235_{10}3 \odot 0.7430_{10}2 &= 0.9176_{10}4\,. \end{aligned}$$

> Bei der Multiplikation wird bei der Ausführung temporär mit der doppelten Mantissenlänge gearbeitet (im Falle $E = 2$ wird allerdings eleganter und damit sparsamer gerechnet!)

Addition: $0.1285_{10}(-1) \oplus 0.7488_{10}1$**.**

Schritt 1: Mantisse des Operanden mit dem kleineren Exponenten vorne passend mit Nullen auffüllen, um die Exponenten gleich zu machen. Von der Mantisse nur $t+1$ Stellen behalten, den Rest abschneiden: $0.1285_{10}(-1) = 0.001285_{10}1 \rightarrow 0.00128_{10}1$

Schritt 2: Diese Operanden addieren. Danach Mantisse auf t Stellen runden, gegebenenfalls Darstellung normalisieren:

$$\begin{array}{rl} & 0.00128_{10}1 \\ + & 0.7488\ _{10}1 \\ \hline & 0.75008_{10}1 \end{array}$$

$\mathrm{rd}(0.75008_{10}1) = 0.7501_{10}1\,.$

Also $0.1285_{10}(-1) \oplus 0.7488_{10}1 = 0.7501_{10}1\,.$

> Bei der Addition wird die Mantisse des Operanden mit dem kleineren Exponenten soweit nach rechts verschoben, dass beide Exponenten gleich sind. Dabei werden (mit führenden Nullen) nur $t+1$ Mantissenstellen beibehalten. Beide Darstellungen werden addiert, das Ergebnis auf t Stellen gerundet und gegebenenfalls neu normalisiert.

Damit lässt sich die Maschinengenauigkeit eps wie folgt charakterisieren:

Satz 1.1 *Bei der Rechnung mit Gleitpunktarithmetik (t und $E = 2H$ fest gewählt) gilt:*

$$\mathit{eps} \;=\; \mathit{min}\,\{g \in \mathcal{A}^* \mid (1 \oplus g) \ominus 1 > 0 \;\mathit{und}\; g > 0\}$$

Beweis:

(1) Es gilt $1 = 0.\underbrace{10\ldots0}_{t \text{ Stellen}}{}_E 1$ und $\text{eps} = H \cdot E^{-t} = 0.H\,0\ldots0_E(1-t)$.
Also gilt $1 \oplus \text{eps} = 0.\underbrace{10\ldots01}_{t \text{ Stellen}}{}_E 1$, da nach dem Schema

$$(*) \quad \left\{ \begin{array}{rl} & 0.10\ldots0 \\ + & 0.00\ldots0H \\ \hline & 0.10\ldots0H \end{array} \right.$$

addiert und dann aufgerundet wird.
Also gilt $(1 \oplus \text{eps}) \ominus 1 > 0$.

(2) Für jede Zahl $g \in \mathcal{A}^*$ mit $0 < g < \text{eps}$ führt die Addition nach dem Schema $(*)$ zur Abrundung des Ergebnisses und man erhält $1 \oplus g = 1$. Also resultiert in diesem Fall $(1 \oplus g) \ominus 1 = 0$.

□

Damit ließe sich eps auch über die Beziehung in Satz 1.1 definieren. Auf einem Rechner lässt sich eps bestimmen, indem mit immer kleineren Summanden g getestet wird, ob $(1 \oplus g) \ominus 1 > 0$ gilt. Man kann dazu von $g := 1$ ausgehen und halbiert g fortlaufend.

Die Aussage in Satz 1.1 zeigt bereits, dass für die Operationen $\oplus$, $\ominus$, $\odot$ und $\oslash$ nicht die üblichen algebraischen Gesetze gelten (wir wählen zur Demonstration $t = 2$ und $E = 10$ als Beispiel):

$\oplus$ ist zwar *kommutativ*, **nicht** aber *assoziativ*:
Zwar gilt $x \oplus y = y \oplus x$ für alle $x, y \in \mathcal{A}^*$, es gilt jedoch z.B.

$$(0.14_{10}2 \oplus -0.13_{10}2) \oplus 0.21_{10}1 = 0.10_{10}1 \oplus 0.21_{10}1 = 0.31_{10}1$$

und

$$0.14_{10}2 \oplus (-0.13_{10}2 \oplus 0.21_{10}1) = 0.14_{10}2 \oplus -0.11_{10}2 = 0.30_{10}1\,.$$

Hier sind die Ergebnisse verschieden. Die erste Additionsvariante liefert dabei das genauere Ergebnis.

Auch $\odot$ ist *kommutativ*, **nicht** aber *assoziativ*:
Es gilt z.B.

$$(0.12_{10}0 \odot 0.18_{10}0) \odot 0.37_{10}0 = 0.22_{10}(-1) \odot 0.37_{10}0 = 0.81_{10}(-2)$$

und

$$0.12_{10}0 \odot (0.18_{10}0 \odot 0.37_{10}0) = 0.12_{10}0 \odot 0.67_{10}(-1) = 0.80_{10}(-2)\,.$$

Wieder sind beide Ergebnisse verschieden. Jetzt liefert jedoch die zweite Rechnung das genauere Resultat.

Was bei solchen Rechnungen genauer ist, hängt von den Größenordnungen der Operanden (auch ihren Vorzeichen bei der Addition) und der gewählten Zusammenfassungsreihenfolge ab. Bei längeren Rechnungen kann daher überlegt werden, welche von mehreren algebraisch gleichwertigen Termen das genauere numerische Ergebnis liefern.

Dazu muss man jedoch etwas mehr über die Fortpflanzung von Fehlern aus Vorrechnungen in daran anschließende Folgerechnungen wissen.

1.4 Fehlerfortpflanzung

1.4.1 Fehlerabschätzungen bei arithmetischen Operationen

Ein Eingangsbeispiel: Zu berechnen sei auf einer Maschine $x^2 - y^2$ für $x = 0.231$ und $y = 0.228$. Die Mantissenlänge t betrage 2.

Wir lassen den Exponentenfaktor weg und setzen dafür jeweils den Punkt an die passende Stelle. Dann sind die Schritte bei der Berechnung:

$x = 0.231$	$\rightarrow$	$\tilde{x} = 0.23$	(Rundung)
$x = 0.224$	$\rightarrow$	$\tilde{y} = 0.22$	(Rundung)
Variante 1:			
$\tilde{x}^2 = 0.053$	und	$\tilde{y}^2 = 0.048$	(Quadrieren)
$\tilde{x}^2 \ominus \tilde{y}^2 = 0.0050$			(Ergebnis)
Variante 2:			
$\tilde{x} \oplus \tilde{y} = 0.45$	und	$\tilde{x} \ominus \tilde{y} = 0.01$	(Summe und Differenz bilden)
$0.45 \odot 0.01 = 0.0045$			(Ergebnis)

Bei der ersten Variante ist der absolute Fehler gegenüber dem exakten Wert $x^2 - y^2 = 0.003185$ gleich 0.001815 und der relative Fehler etwa gleich 47 %, bei der zweiten beträgt der absolute Fehler nur 0.001315 und der relative Fehler nur rund 41 %.

Dass hier Unterschiede bei der Berechnung der Quadratdifferenz auftreten, lässt sich mit der Empfindlichkeit der Subtraktion gegenüber Eingangsfehlern bei fast gleich großen Operanden erklären. Zur Untersuchung der Operationen $\oplus$, $\ominus$, $\odot$, $\oslash$ gehen wir nun von Näherungen $\tilde{x} = x + \Delta x$ und $\tilde{y} = y + \Delta y$ reller Zahlen $x \neq 0$ und $y \neq 0$ aus und stellen uns der Einfachheit halber vor, dass mit diesen Näherungswerten *exakt* gerechnet wird. Da die Subtraktion $a - b$ durch die Addition $a + (-b)$ ersetzt werden kann und sich die Division $a : b$ in der Form $a \cdot \frac{1}{b}$ deuten läßt, werden nur die Addition, die Multiplikation und die Kehrwertbildung untersucht.

Addition/Subtraktion:

Bei exakter Rechnung gilt:

$$\tilde{\mathbf{x}} \pm \tilde{\mathbf{y}} \;=\; (x + \Delta x) \pm (y + \Delta y) \;=\; (\mathbf{x} + \mathbf{y}) \pm (\mathbf{\Delta x} + \mathbf{\Delta y})$$

Also addieren sich die **absoluten Fehler**.

Für den relativen Fehler erhält man im Falle $x + y \neq 0$:

$$\varepsilon_{\tilde{\mathbf{x}} \pm \tilde{\mathbf{y}}} \;=\; \frac{\Delta x + \Delta y}{x \pm y} \;=\; \frac{x \cdot \varepsilon_{\tilde{x}} \pm y \cdot \varepsilon_{\tilde{y}}}{x \pm y} \;=\; \frac{\mathbf{x}}{\mathbf{x} \pm \mathbf{y}} \cdot \varepsilon_{\tilde{\mathbf{x}}} \pm \frac{\mathbf{y}}{\mathbf{x} \pm \mathbf{y}} \cdot \varepsilon_{\tilde{\mathbf{y}}}$$

Falls also $x - y$ nahe bei 0 liegt und x und y relativ groß gegenüber der Differenz sind, werden Eingangsfehler enorm verstärkt (siehe Eingangsbeispiel!). Bei der Addition ist damit zu rechnen, wenn x und y entgegengesetztes Vorzeichen haben und betragsmäßig nahezu gleich sind.

Der Vorgang heißt **Subtraktionsauslöschung**. Er kann als harmlos gelten, wenn nicht mit fehlerbehafteten Eingamgswerten gerechnet wird.

Multiplikation:

Setzt man $x \cdot y \neq 0$ voraus, so gilt:

$$\tilde{\mathbf{x}} \cdot \tilde{\mathbf{y}} \;=\; (x + \Delta x) \cdot (y + \Delta y) \;=\; x \cdot y + x \cdot \Delta x + y \cdot \Delta y + \Delta x \cdot \Delta y$$

Wenn Δx und Δy klein gegen x und y sind, kann man das Produkt $\Delta x \cdot \Delta y$ vernachlässigen und erhält für den relativen Fehler von $\tilde{x} \cdot \tilde{y}$:

$$\varepsilon_{\tilde{\mathbf{x}} \cdot \tilde{\mathbf{y}}} \;=\; \frac{\tilde{x} \cdot \tilde{y} - x \cdot y}{x \cdot y} \underset{\uparrow}{\doteq} \frac{y \cdot \Delta x + x \cdot \Delta y}{x \cdot y} \;=\; \frac{\Delta x}{x} + \frac{\Delta y}{y} \;=\; \varepsilon_{\tilde{\mathbf{x}}} + \varepsilon_{\tilde{\mathbf{y}}}$$

Vernachlässigung von Gliedern höherer Ordnung

Also addieren sich (in etwa) bei der Multiplikation die **relativen** Fehler der Faktoren.

Kehrwertbildung:

Setzt man $x \neq 0$ und $\tilde{x} \neq 0$ voraus, so gilt im Falle $|\frac{\Delta x}{x}| \ll 1$:

$$\tilde{x}^{-1} \;=\; \frac{1}{x + \Delta x} \;=\; \frac{1}{x} \cdot \frac{1}{1 + \frac{\Delta x}{x}} \;=\; \frac{1}{x} \cdot \Big(1 - \frac{\Delta x}{x} + (\frac{\Delta x}{x})^2 - \; + \ldots\Big) \;\doteq\; \frac{1}{x} \cdot \Big(1 - \frac{\Delta x}{x}\Big)$$

Daraus ergibt sich für den relativen Fehler von $\tilde{x}^{-1}$:

$$\varepsilon_{\tilde{\mathbf{x}}^{-\mathbf{1}}} \;=\; \frac{\frac{1}{\tilde{x}} - \frac{1}{x}}{\frac{1}{x}} \;\doteq\; -\frac{\Delta x}{x} \;=\; -\varepsilon_{\tilde{\mathbf{x}}}$$

Der relative Fehler des Kehrwerts hat also in etwa den gleichen Betrag wie der relative Fehler des Eingangswerts und besitzt das entgegengesetze Vorzeichen.

Damit ergibt sich für die **Division** die Fehlerregel $\quad \varepsilon_{\tilde{\mathbf{x}} : \tilde{\mathbf{y}}} \doteq \varepsilon_{\tilde{\mathbf{x}}} - \varepsilon_{\tilde{\mathbf{y}}}$.

Wir fassen die Regeln zusammen und lassen dabei die Näherungskennzeichnung ˜ weg. Zusätzlich wird angenommen, dass jede Operation $\circledast$ auf dem Rechner mit einem relativen Rundungsfehler α_* behaftet ist:

$$\begin{aligned}
\varepsilon_{\mathbf{x}\pm\mathbf{y}} &\doteq \tfrac{\mathbf{x}}{\mathbf{x}\pm\mathbf{y}}\varepsilon_{\mathbf{x}} + \tfrac{\mathbf{y}}{\mathbf{y}\pm\mathbf{x}}\varepsilon_{\mathbf{y}} + \alpha_{\pm} && (x, y, x \pm y \neq 0)\\
\varepsilon_{\mathbf{x}\cdot\mathbf{y}} &\doteq \varepsilon_{\mathbf{x}} + \varepsilon_{\mathbf{y}} + \alpha_{\cdot} && (x, y \neq 0)\\
\varepsilon_{\mathbf{x}:\mathbf{y}} &\doteq \varepsilon_{\mathbf{x}} - \varepsilon_{\mathbf{y}} + \alpha_{:} && (x, y \neq 0)
\end{aligned}$$

Für das Quadratwurzelziehen kann bei sorgfältiger Realisierung gelten:

$$\varepsilon_{\sqrt{\mathbf{x}}} \doteq \tfrac{\mathbf{1}}{\mathbf{2}}\varepsilon_{\mathbf{x}} + \alpha_{\sqrt{}} \qquad (x > 0)$$

Begründung für $x > 0$:

$$\sqrt{x + \Delta x} = \sqrt{x}\sqrt{1 + \tfrac{\Delta x}{x}} \doteq \sqrt{x}(1 + \tfrac{\Delta x}{2x}) = \sqrt{x}(1 + \tfrac{1}{2}\varepsilon_x)$$

Gehören bei solchen Operationen $x, y, x * y, \mathrm{rd}x \circledast \mathrm{rd}(y)$ zu den Zahlen, die ohne Überlauf oder Unterlauf in $\mathcal{A}$ abbildbar bzw. darstellbar sind, so haben alle relativen Fehler und der arithmetische Maschinenfehler α maximal den Betrag eps.

1.4.2 Fehlerfortpflanzung bei Algorithmen

Grundlegend für die Abschätzung relativer Fehler bei „Kettenrechnungen“ ist die multiplikative Beziehung zwischen einem exakten Wert x, einer Näherung $\tilde{x}$ für x und dem (von jetzt ab ohne das Zeichen ˜ geschriebenen!) relativen Fehler ε_x dieser Näherung in der Form $\tilde{x} = (1 + \varepsilon_x) \cdot x$.

Beispiel: Aus den Kathetenlängen a und b eines rechtwinkligen Dreiecks soll die Hypotenusenlänge c berechnet werden.

Wir fassen die Berechnung als Verkettung von 4 Funktionen f_1, f_2, f_3, f_4 auf, deren Eingangswerte und Ausgangswerte im Bedarfsfall als Vektoren notiert werden. Die Menge der möglichen Eingabepaare sei mit D_0 bezeichnet, also

$$D_0 := \left\{ \begin{pmatrix} a \\ b \end{pmatrix} \,\middle|\, a, b \in \mathbb{R}^+ \right\} .$$

Dann erfolgt die maschinelle Bestimmung einer möglichst genauen Näherung c^* von c folgendermaßen ($\mathcal{A}^+$ sei die Menge postiver Maschinenzahlen):

$$\begin{aligned}
D_0 \xrightarrow{f_1} D_1 &:= \left\{ \begin{pmatrix} a^* \\ b^* \end{pmatrix} \,\middle|\, a^*, b^* \in \mathcal{A}^+ \right\}\\
&\qquad \text{mit} \quad f_1 : \begin{pmatrix} a \\ b \end{pmatrix} \mapsto \begin{pmatrix} \mathrm{rd}(a) \\ \mathrm{rd}(b) \end{pmatrix},\\
D_1 \xrightarrow{f_2} D_2 &:= \left\{ \begin{pmatrix} u^* \\ v^* \end{pmatrix} \,\middle|\, u^*, v^* \in \mathcal{A}^+, u^* \leq v^* \right\}\\
&\qquad \text{mit} \quad f_2 : \begin{pmatrix} a^* \\ b^* \end{pmatrix} \mapsto \begin{pmatrix} a^* \odot a^* \\ b^* \odot b^* \end{pmatrix},
\end{aligned}$$

$$D_2 \xrightarrow{f_3} D_3 := \mathcal{A} \qquad \text{mit} \quad f_3 : \begin{pmatrix} u^* \\ v^* \end{pmatrix} \mapsto v^* \oplus u^* ,$$

$$D_3 \xrightarrow{f_4} D_4 := \mathcal{A} \qquad \text{mit} \quad f_4 : x^* \mapsto \sqrt[②]{x^*} ,$$

Ausgabe: $c^* := f_4(x^*)$.

Also erfolgt die Berechnung in der Form

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f_1}{\mapsto} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \overset{f_2}{\mapsto} \begin{pmatrix} {y_1}^② \\ {y_2}^② \end{pmatrix} \overset{f_3}{\mapsto} {y_1}^② \oplus {y_2}^② \overset{f_4}{\mapsto} \sqrt[②]{{y_1}^② \oplus {y_2}^②}$$

Jede dieser Funktionen ist mit Rundungsfehlern behaftet. Notiert man in den Eingabe- und Ergebniskomponenten nur die maximalen Fehlerbeträge 1. Ordnung, so erkennt man, dass der relative Fehler des Ergebnisses bei als genau vorausgesetzten Werten x_1 und x_2 im Bereich 3eps liegt:

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} \hookrightarrow \begin{pmatrix} \text{eps} \\ \text{eps} \end{pmatrix} \hookrightarrow \begin{pmatrix} 2\text{eps} + \text{eps} \\ 2\text{eps} + \text{eps} \end{pmatrix} \hookrightarrow 3\text{eps} + \text{eps} \hookrightarrow 3\text{eps}$$

Dass sich bei der Addition der Summanden y_1 und y_2 der relative Fehlerbetrag nur um maximal eps erhöht, liegt daran, dass beide Summanden positiv sind:

$$\max|\varepsilon_{y_1 \oplus y_2}| \doteq \frac{y_1}{y_1 + y_2} 3\text{eps} + \frac{y_2}{y_1 + y_2} 3\text{eps} + \text{eps} = 3\text{eps} + \text{eps} \leq 4\text{eps}$$

Beim Quadratwurzelziehen halbiert sich dann 4eps und es kommt maximal eps als relativer Rundungsfehler der Operation hinzu.

Hätten wir als Beispiel die Berechnung der Kathetenlänge b aus der Hypotenusenlänge c und der Kathetenlänge a gewählt, so hätte sich für sehr nahe beieinander liegende Werte von a und c eine Verstärkung der Rundungsfehler wegen Subtraktionsauslöschung ergeben.

Ein solcher Effekt tritt auch bei der iterativen Berechnung von π nach der Methode von ARCHIMEDES auf:

Beispiel: Bezeichnet man im Einheitskreis die Seitenlänge eines regelmäßigen n-Ecks mit s_n, so gilt bei Verdoppelung der Eckenzahl:

$$s_{2n} = \sqrt{2 - \sqrt{4 - s_n^2}}$$

Würde man mit dieser Formel auf einem Rechner iterativ (unter Multiplikation von s_n mit der jeweiligen Eckenzahl n) den Kreisumfang berechnen, so würden sich die Näherungswerte erst verbessern und dann recht schnell wegen der Substraktionsauslöschung verschlechtern.

Durch Erweiterung von s_{2n} mit $\sqrt{2+\sqrt{4-s_n^2}}$ erhält man hier mit

$$s_{2n} = \frac{s_n}{\sqrt{2+\sqrt{4-s_n^2}}}$$

eine Iterationsformel, bei der dieses Phänomen nicht auftritt.

Fasst man jede Iteration als Anwendung einer Funktion $f_n : \mathcal{A}^+ \to \mathcal{A}^+$ mit $f_n : u_n \mapsto u_{2n}$ auf (Berechnung von $u_{2n} := 2n \cdot s_{2n}$ aus u_n mit Hilfe von $s_n := \frac{u_n}{n}$), so handelt es sich in diesem Beispiel um die Verkettung von Funktionen, die eindimensionalen Eingabewerten eindimensionale Ausgabewerte zuordnen.

Wir verallgemeinern diese Sichtweise:

Definition 1.4 *Eine endliche Kette von Funktionen*

$$\begin{array}{rcll} f_1 & : & \mathbb{R}^n & \to \mathbb{R}^{n_1} \\ f_2 & : & \mathbb{R}^{n_1} & \to \mathbb{R}^{n_2} \\ & & \vdots & \\ f_r & : & \mathbb{R}^{n_{r-1}} & \to \mathbb{R}^m \end{array}$$

heißt ein **Algorithmus** *zur Bestimmung von*

$$(y_1, \ldots, y_m) := f_r(f_{r-1}(\ldots(f_1(x_1, \ldots, x_n))\ldots))$$

aus $(x_1, \ldots, x_n)$.

Wird nun eine solcher Algorithmus auf dem Rechner realisiert, so arbeitet jede der zugehörigen *Maschinenfunktionen* $f_{\textcircled{1}}, \ldots, f_{\textcircled{r}}$ mit fehlerbehafteten Eingaben und muss ihre Ausgabe runden. Bezeichnet man die relativen Fehler von Eingabewerten mit ε und die relativen Fehler bei der Ausgaberundung mit α, so sieht die Fehlerfortpflanzung im rein eindimensionalen Fall so aus:

Auf dem Rechner:

$$x \xrightarrow{\text{rd}} x_0 \xrightarrow{f_1} f_1(x_0) \xrightarrow{\cdot(1+\alpha_1)} x_1 \xrightarrow{f_2} f_2(x_1) \xrightarrow{\cdot(1+\alpha_2)} x_2 \longrightarrow \cdots \xrightarrow{f_r} f_r(x_{r-1}) \xrightarrow{\cdot(1+\alpha_r)} \tilde{y}$$

exakt in $\mathbb{R}$:

$$x \xrightarrow{f_1} f_1(x) \xrightarrow{f_2} f_2(f_1(x)) \longrightarrow \cdots \xrightarrow{f_r} f_r(x_{r-1}) = y$$

Setzt man $x_0 = x(1+\varepsilon_0)$ und sieht die Verkettung aller Funktionen als eine bis auf die Endrundung genau realisierte Funktion f an, so erhält man unter der Annahme der *zweimaligen stetigen Differenzierbarkeit* von f mit dem Satz von TAYLOR:

$$\tilde{y} = f(x+\varepsilon_0 x) = \left[f(x) + \frac{\varepsilon_0 x}{1!} f'(x) + \frac{\varepsilon_0^2 x^2}{2!} f''(x+\Theta\varepsilon_0 x) \right] \cdot (1+\alpha) \quad \text{mit } 0 < \Theta < 1$$

Also gilt

$$\tilde{y} \doteq (1+\alpha)y + \varepsilon_0 x f'(x)\,.$$

Man nennt daher $\Delta^* y := \alpha \cdot y + \varepsilon_0 x f'(x)$ den **unvermeidbaren** Fehler der Berechnung von y.

Falls f eine zweimal stetig partiell nach allen Argumenten ableitbare Funktion von $\mathbb{R}^n$ nach $\mathbb{R}$ ist, liefert der Satz von TAYLOR für Funktionen von n Variablen *vor* der Ausgaberundung (mit $\mathrm{rd}(\vec{\mathrm{x}}) := (\mathrm{rd}(x_1), \ldots, \mathrm{rd}(x_n))$ als Rundungsfunktion):

$$f(\mathrm{rd}(\vec{\mathrm{x}})) \doteq f(\vec{\mathrm{x}}) + \sum_{k=1}^{n} \frac{x_k \cdot \varepsilon_k}{1!} \frac{\partial f(\vec{\mathrm{x}})}{\partial x_k}$$

Nach der Ausgaberundung erhält man also

$$\tilde{y} \doteq (1+\alpha)\left[y + \sum_{k=1}^{n} \frac{x_k \cdot \varepsilon_k}{1!} \frac{\partial f(\vec{\mathrm{x}})}{\partial x_k}\right]$$

und damit den **unvermeidbaren Fehler**[1]

$$\Delta^* y := y \cdot \alpha + \sum_{k=1}^{n} \frac{x_k \cdot \varepsilon_k}{1!} \frac{\partial f(\vec{\mathrm{x}})}{\partial x_k}\,.$$

Wenn die relativen Eingangsfehler viel größer als die Maschinengenauigkeit sind (z.B. beim Rechnen mit fehlerbehafteten Messwerten in der Physik), sieht man den Term $\alpha \cdot y$ als vernachlässigbar an und erhält die in der Physik übliche Fehlerabschätzung

$$\Delta y \approx \sum_{k=1}^{n} \Delta x_k \cdot \frac{\partial f(\vec{\mathrm{x}})}{\partial x_k}\,.$$

Kennt man die Vorzeichen der Fehler von x_1 bis x_k nicht, so bildet man die Beträge aller Summanden und erhält so eine Abschätzung des Fehlers nach oben.

---

[1] Wenn in einem Algorithmus eine Funktion $\varphi : \mathbb{R}^n \to \mathbb{R}^m$ mit der Vorschrift

$$\vec{\mathrm{y}} := \begin{pmatrix} \varphi_1(\vec{\mathrm{x}}) \\ \vdots \\ \varphi_m(\vec{\mathrm{x}}) \end{pmatrix}$$

auf einem Rechner realisiert wird, erhält man durch Betrachtung der einzelnen Komponenten den in der Form

$$\Delta^* \vec{\mathrm{y}} := \begin{pmatrix} \alpha_1 & & 0 \\ & \ddots & \\ 0 & & \alpha_m \end{pmatrix} \vec{\mathrm{y}} + \begin{pmatrix} \frac{\partial \varphi_1}{\partial x_1} & \cdots & \frac{\partial \varphi_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial \varphi_m}{\partial x_1} & \cdots & \frac{\partial \varphi_m}{\partial x_n} \end{pmatrix} \begin{pmatrix} \varepsilon_1 & & 0 \\ & \ddots & \\ 0 & & \varepsilon_n \end{pmatrix} \vec{\mathrm{x}}$$

definierten unvermeidbaren Fehler des Algorithmus.

Wir beschränken uns auf den Fall, dass eine Funktion $f : \mathbb{R}^n \to \mathbb{R}$ gegeben ist. Dann gilt im Falle $y := f(\vec{x}) \neq 0$ für den *unvermeidbaren relativen* Fehler von $\tilde{y}$:

$$\varepsilon_y^* := \frac{\Delta^* y}{y} = \alpha + \sum_{k=1}^{n} \varepsilon_k \cdot \underbrace{\frac{x_k}{f(\vec{x})} \cdot \frac{\partial f(\vec{x})}{\partial x_k}}_{\textbf{Konditionszahlen}}$$

Sind die Konditionszahlen (eigentlich müssten sie *Konditionsfunktionen* heißen) groß, so verstärken sich Eingabefehler sehr stark und man nennt das Berechnungsproblem **schlecht konditioniert**.

Ein Beispiel für schlecht konditionierte Berechnungen ist die Subtraktion von Werten, die nahezu gleich groß sind und das gleiche Vorzeichen haben. Manchmal kann man bei der Verkettung von Operationen durch den Übergang von einem Term zu einem algebraisch gleichwertigen Term wenigstens den Einfluss von Rundungsfehlern auf der Maschine verkleinern:

Beispiel: Es sei $y := \sqrt{x_1} - \sqrt{x_2}$ für $x_1, x_2 > 0$ auf einer Maschine zu berechnen. Wird dies in dieser Form realisiert, so gilt:

$$\varepsilon_y^* = \alpha + \left(\varepsilon_{x_1} \frac{x_1}{2\sqrt{x_1}} - \varepsilon_{x_2} \frac{x_2}{2\sqrt{x_2}} \right) \frac{1}{\sqrt{x_1} - \sqrt{x_2}}$$

Die Konditionszahlen $\dfrac{\sqrt{x_1}}{2(\sqrt{x_1} - \sqrt{x_2})}$ und $-\dfrac{\sqrt{x_2}}{2(\sqrt{x_1} - \sqrt{x_2})}$ werden dem Betrag nach groß, wenn $x_1 \approx x_2$ gilt und der Betrag von x_1 groß ist. Für $x_1 = 0.5041$ und $x_2 = 0.4900$ erhält man z.B. die Konditionszahlen 35.5 und -35. Werden also x_1 und x_2 mit dem relativen Fehler 10^{-12} im Rechner gespeichert, so kann das Ergebnis mit dem relativen Fehler $7 \cdot 10^{-11}$ behaftet sein.

Noch größer wird der Fehler, wenn sich zwei Maschinenzahlen x_1 und x_2 nur noch um wenige Vielfache von eps unterscheiden (wir wählen $E = 10$), z.B.:

$$y := \sqrt[②]{1 \oplus (k \cdot \text{eps})} \ominus \sqrt[②]{1} \quad \text{mit} \quad 1 \leq k \leq 10$$

Hier ergibt sich für $k = 1$ in $\mathcal{A}$ wegen $1 \oplus \text{eps} = 1 + 2\,\text{eps}$ (Aufrundung!):

$$\sqrt[②]{1 \oplus \text{eps}} = \sqrt[②]{1 + (2\,\text{eps})} \doteq (1 + \frac{2\,\text{eps}}{2}) \ominus 1 \oplus \text{eps} = 1 + 2\,\text{eps}$$

Damit ist $\tilde{y}$ gleich $2\,\text{eps}$, d.h. der relative Fehler des Resultats beträgt etwa 300 %, da $\sqrt{1 + \text{eps}} \doteq 1 + \frac{\text{eps}}{2}$ und damit $y \doteq \frac{\text{eps}}{2}$ gilt.

Erhöht man k, so lässt der Effekt nach.

Erweitert man den Term für y mit $(\sqrt{x_1} + \sqrt{x_2})$ und berechnet y auf der Maschine in der Form

$$\begin{aligned} z &:= x_1 \ominus x_2, \\ t &:= \textcircled{2}\sqrt{x_1} \oplus \textcircled{2}\sqrt{x_2}, \\ y &:= z \textcircled{:} t, \end{aligned}$$

so ergibt sich $\tilde{y}$ im Falle $x_1 = 1 \oplus \text{eps}, x_2 = 1$ zu $\frac{2\,\text{eps}}{(1+2\,\text{eps})+1} \doteq \text{eps}$. Hier beträgt der relative Fehler nur noch etwa 100 %.

Die Verfeinerung der bisherigen Betrachtungen zu einer Theorie der Fehlerfortpflanzung als „Lehre des Umgangs mit dem Zeichen $\doteq$“ ist zwar möglich, wird aber auch von Kritikern als Missbrauch der Analysis angesehen. Dies liegt daran, dass zu oft von Fehlergliedern höherer Ordnung abgesehen werden muss. Damit besteht die Gefahr, dass Scheinprobleme gelöst werden. Trotzdem eignet sich die Theorie, um sich einen globalen Überblick darüber zu verschaffen, ob ein Berechnungsproblem „kritisch“ von seinen Eingabewerten ab hängt.

Eine bessere Fehlerkontrolle hat man beim **kontrollierten Rechnen** (z.B. PSCAL SC), da dort an Stelle von Werten mit **Intervallen** gearbeitet wird. Durch den großen Abschätzungsaufwand laufen allerdings die mit der Intervallrechnungsoption compilierten Programme deutlich langsamer!

2 Numerische Methoden in der linearen Algebra

2.1 Lineare Gleichungssysteme

2.1.1 Der Gaußalgorithmus für LGS mit genau einer Lösung

Beispiel: Ein lineares Gleichungssystem in *Stufenform* lässt sich leicht durch *Rückwärtseinsetzen* lösen:

$$\begin{array}{rcrcrcrl}
2x_1 & - & 3x_2 & + & x_3 & = & -8 & \\
 & & 2x_2 & + & 5x_3 & = & -6 & \\
 & & & & -2x_3 & = & 4 & \\
 & & & & & & & \\
 & & & & x_3 & = & -2, & \\
 & & 2x_2 & - & 10 & = & -6; & \text{also } x_2 = 2, \\
2x_1 & - & 6 & - & 2 & = & -8; & \text{also } x_1 = 0.
\end{array}$$

Ist nun ein LGS mit n Variablen und n Gleichungen in der Form

$$\begin{array}{l}
a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + = \quad b_1 \\
a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n + = \quad b_2 \\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n + = \quad b_n
\end{array}$$

gegeben, so versucht man es durch Anwendung der folgenden **Äquivalenzumformungen** auf Stufenform zu bringen:

(1) Gleichungen miteinander vertauschen,
(2) eine Gleichung mit einer Zahl $c \neq 0$ multiplizieren,
(3) eine Gleichung durch die Summe oder Differenz eines Vielfachen von ihr und einem Vielfachen einer **anderen** Gleichung ersetzen.

Gelingt dies, so kann man anschließend schrittweise nach den Variablen $x_n, \ldots, x_2, x_1$ auflösen. Dieses Verfahren heißt das *Eliminationsverfahren* von GAUSS.

Wendet man dieses Verfahren in der Numerik an, so bringt man es in eine Form, die möglichst kleine relative Rundungsfehlerbeträge garantiert. Der einfacheren Notation wegen schreibt man ein lineares Gleichungssystem (ab jetzt kurz LGS genannt) in ***Matrixschreibweise*** und gibt strengere Regeln für das Ausführen von Äquivalenzumformungen vor. Dabei wird die Regel (2) allenfalls zur anfänglichen (!) Normierung von Gleichungszeilen eingesetzt. Danach wird nur noch mit den Regeln (1) und (2) gearbeitet.

Zur Demonstration dieses Verfahrens nehmen wir an, dass ein LGS in der Form

$$(*)\quad \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \vdots & & & \vdots \\ a_{n1} & a_{12} & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$$

gegeben ist und die Matrix A keine „Nullzeile“ enthält. In solchen Fällen ist es üblich, am Anfang jede Gleichung durch die Quadratwurzel aus der Summe der Koeffizientenquadrate oder den betragsgrößten Koeffizienten zu dividieren. Es wird also für $k = 1, \dots n$ bei jeder Gleichung

$$\sum_{t=1}^{n} a_{kt} x_t \;=\; b_k$$

zunächst $\mu_k := \max\limits_{t\,=\,1,\,\dots\,,\,n} a_{kt}$ oder $\mu_k := \sqrt{\sum\limits_{t=1}^{n} a_{kt}^2}$ berechnet. Danach wird jeweils die k-te Gleichung durch

$$\sum_{t=1}^{n} a'_{kt} x_t \;=\; b'_k \quad \text{mit} \quad a'_{kt} := \frac{a_{kt}}{\mu_k} \quad \text{für } t = 1 \dots, n \quad \text{und} \quad b'_k := \frac{b_k}{\mu_k}$$

ersetzt. Eine Matrix A, bei der man diese Umformung bereits durchgeführt hat, nennt man **equilibriert**. Wir setzen nun voraus, dass die Matrix A im LGS $(*)$ bereits equilibriert ist.

Dann kann man das LGS $A\vec{x} = \vec{b}$ folgendermaßen lösen:

Gauss-Algorithmus (für ein LGS mit höchstens einer Lösung):

A muss auf Stufenform gebracht werden:

(1) Setze den Teilmatrixzähler i auf 1.

(2) Bestimme $\max\limits_{j=i,\ldots,n} |a_{ji}|$ und merke dir den Index m der Zeile, in der das Maximum beobachtet wurde.
Falls $i \neq m$ gilt: Vertausche die i-te Matrixzeile mit der m-ten Zeile und vertausche b_i mit b_m.
Überschreibe dabei die Werte in A und in $\vec{b}$ und sehe das LGS danach wieder als in der Form $A\vec{x} = \vec{b}$ gegeben an.

(3) Falls $a_{ii} \neq 0$ und $i < n$ gilt, ist für die Zeilen mit den Nummern $j = i+1 \ldots, n$ folgende Operation durchzuführen:
Berechne $\mu_{ji} := \frac{a_{ji}}{a_{ii}}$ und bestimme für $k = i+1, \ldots, n$ jeweils die Differenz $a'_{jk} := a_{jk} - a_{ik} \cdot \mu_{ji}$.
Bestimme $b'_j := b_j - b_i \cdot \mu_j ji$. Überschreibe danach die j-te Zeile von A und die j-te Komponente von $\vec{b}$ mit den neuen Werten.
Setze den Teilmatrixzähler auf $i+1$ und gehe nach (2).
Falls $a_{ii} = 0$ gilt: STOP, LGS nicht eindeutig lösbar!
Falls $i = n$ und $a_{ii} \neq 0$ gilt: gehe zum Verfahren ***Rückwärtseinsetzen***

Rückwärtseinsetzen

(4) Berechne $x_n := \frac{b_n}{a_{nn}}$.

(6) Für $j = n-1$ bis 1 (also abwärts!) berechne man: $x_j := (b_j - \sum\limits_{k=j+1}^{n} a_{jk}x_k) : a_{jj}$.

Definition 2.1 *Die Maximumbestimmung mit anschließender Zeilenvertauschung im* GAUSS*-Algorithmus nennt man* ***Teilpivotisierung.***
Der resultierende Koeffizient a_{ii} in der i-ten Teilmatrix heißt das ***i-te Pivotelement.***

Dass sich die Teilpivotisierung günstig auswirkt, soll nun am Fall einer 2×2-Matrix mit einstelliger Gleitpunktrechnung demonstriert werden. Wir notieren dabei nur die Matrix und die zugehörige rechte Seite in einem Rechenschema:

Beispiel:

Zu lösen ist:

$$\begin{pmatrix} 0.005 & 1.0 \\ 1.0 & 1.0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 1.0 \end{pmatrix}$$

Wir vergleichen zwei Lösungswege:

Variante 1 (ohne Teilpivotsuche):

exakte Rechnung:

$$\begin{array}{cc|c} 0.005 & 1 & 0.5 \\ 1 & 1 & 1 \end{array}$$

$$\Downarrow$$

$$\begin{array}{cc|c} 0.005 & 1 & 0.5 \\ 0 & -199 & -99 \end{array}$$

$$\Downarrow$$

$$x_2 = 0.4974\ldots$$
$$x_1 = 0.5025\ldots$$

1-stellige Gleitpunktrechnung:

$$\begin{array}{cc|c} 0.005 & 1.0 & 0.5 \\ 1.0 & 1.0 & 1.0 \end{array}$$

$$\Downarrow$$

$$\begin{array}{cc|c} 0.005 & 1.0 & 0.5 \\ 0 & -200 & -100 \end{array}$$

$$\Downarrow$$

$$\tilde{x}_2 = 0.5$$
$$\tilde{x}_1 = 0.0$$

Hier beträgt der relative Fehler von $\tilde{x}_1$ offensichtlich -100 %, für $\tilde{x}_2$ beträgt er rund 0.5 %.

Variante 2 (mit Teilpivotsuche):

exakte Rechnung:

$$\begin{array}{cc|c} 0.005 & 1 & 0.5 \\ 1 & 1 & 1 \end{array}$$

$$\Downarrow$$

$$\begin{array}{cc|c} 1 & 1 & 1 \\ 0.005 & 1 & 0.5 \end{array}$$

$$\Downarrow$$

$$\begin{array}{cc|c} 1 & 1 & 1 \\ 0 & 0.995 & 0.495 \end{array}$$

$$\Downarrow$$

$$x_2 = 0.4974\ldots$$
$$x_1 = 0.5025\ldots$$

1-stellige Gleitpunktrechnung:

$$\begin{array}{cc|c} 0.005 & 1.0 & 0.5 \\ 1.0 & 1.0 & 1.0 \end{array}$$

$$\Downarrow$$

$$\begin{array}{cc|c} 1.0 & 1.0 & 1.0 \\ 0.005 & 1.0 & 0.5 \end{array}$$

$$\Downarrow$$

$$\begin{array}{cc|c} 1.0 & 1.0 & 1.0 \\ 0 & 1.0 & 0.5 \end{array}$$

$$\Downarrow$$

$$\tilde{x}_2 = 0.5$$
$$\tilde{x}_1 = 0.5$$

Hier beträgt der relative Fehler von $\tilde{x_1}$ etwa $0,5$ % und der relative Fehler von $\tilde{x}_2$ etwa $-0,5$ %.

Wie das obige Rechenschema zeigt, kann man eine LGS $A\vec{x} = \vec{\mathrm{b}}$ durch Notieren der **erweiterten Matrix**

$$(\mathrm{A}|\mathrm{b}) := \left(\begin{array}{ccc|c} a_{11} & \dots & a_{1n} & b_1 \\ \vdots & & \vdots & \vdots \\ a_{n1} & \dots & a_{nn} & b_n \end{array}\right)$$

angeben. Zeilenumformungen in einem LGS entsprechen dann Zeilenumformungen in der Matrix (A|b).

Wenn im GAUSS-Algorithmus bei der Pivotisierung durch Umbenennung der Variablen sogar Spalten vertauscht werden, spricht man von **Totalpivotisierung**. In diesem Fall sucht man bei der Umformung der i-ten Teilmatrix nicht nur das Betragsmaximum der Koeffizienten in der i-ten Spalte, sondern in der ganzen i-ten Teilmatrix von A.

Wenn a_{rs} ein betragsmäßig maximaler Koeffizient unter den Koeffizienten a_{jk} mit $i \le j \le n$ und $i \le k \le n$ ist, werden die i-te Zeile und die r-te Zeile von (A|b) vertauscht. Danach vertauscht man die i-te und s-te Spalte von A in (A|b) und merkt sich die zugehörige Umbenennung der Variablen in einem „Nummernfeld“:

Sind vor der ersten solchen Vertauschung noch alle Variablen fortlaufend nummeriert, so sieht dieses Feld so aus:

1	2	$\cdots$	i	$\cdots$	s	$\cdots$	n
1.	2.	$\cdots$	$i.$	$\cdots$	$s.$	$\cdots$	$n.$

Nach der Vertauschung von Spalte i mit Spalte s erhält man als Einträge im Feld:

1	2	$\cdots$	s	$\cdots$	i	$\cdots$	n
1.	2.	$\cdots$	$i.$	$\cdots$	$s.$	$\cdots$	$n.$

Aktualisiert man das Feld bei jeder Spaltenvertauschung, so kann man am Schluss ablesen, welche Variablennummern den Variablen $x'_1, \dots, x'_n$ zugewiesen werden müssen, die beim Rückwärtseinsetzen verwendet werden.

So wäre z.B. eine Lösung des Typs

$$x'_1 = 1.0,\ x'_2 = -2.4,\ x'_3 = 0.1,\ x'_4 = 0$$

bei dem Feld

4	2	1	3

in der Form $x_4 = 1.0,\ x_2 = -2.4,\ x_1 = 0.1,\ x_3 = 0$ auszugeben.

2.1.2 Der GAUSS-Algorithmus und die L-R-Zerlegung

Die Umformungsschritte des GAUSS-Algorithmus lassen sich als Multiplikation beider Seiten der Matrizengleichung $A\vec{x} = \vec{b}$ mit *Umformungsmatrizen* auffassen. Wir nummerieren für solche Betrachtungen kurzzeitig die Matrix A und die rechte Seite $\vec{b}$ eines LGS nach dem Teilmatrixzähler i aus der Beschreibung des GAUSS-Algorithmus und bezeichnen die n-zeilige *Einheitsmatrix* mit I.

1. Die *Zeilenvertauschung*, z.B. der i-ten mit der m-ten Zeile für das System $A^{(i)}\vec{x'} = \vec{b}^{(i)}$ lässt sich auffassen als Übergang

$$A^{(i)}\vec{x} = \vec{b}^{(i)} \;\mapsto\; P_{im}A^{(i)}\vec{x} = P_{im}\vec{b}^{(i)} \qquad (i > m \geq n)$$

mit

$$P_{im} := \begin{array}{c} \\ (i) \\ \\ (m) \\ \\ \end{array} \begin{array}{c} \begin{array}{ccc} i & & m \end{array} \\ \left(\begin{array}{ccc} \begin{matrix}1&&0\\&\ddots&\\0&&1\end{matrix} & 0 & 0 \\ 0 & 0 & 1 \\ 0 & \begin{matrix}1&&0\\&\ddots&\\0&&1\end{matrix} & 0 \\ 1 & 0 & 0 \\ 0 & 0 & \begin{matrix}1&&0\\&\ddots&\\0&&1\end{matrix} \end{array}\right) \end{array}$$

(Frobeniusmatrix)

Es gilt $P_{im}^{-1} = P_{im}$.

2. Eine *Spaltenvertauschung*, z.B. der i-ten mit der s-ten Zeile für das System $A^{(i)}\vec{x} = \vec{b}^{(i)}$ lässt sich auffassen als Übergang

$$A^{(i)}\vec{x} = \vec{b}^{(i)} \;\mapsto\; (A^{(i)}P_{is})P_{is}\vec{x} = \vec{b}^{(i)} \qquad (i > s \geq n)$$

mit der Frobeniusmatrix P_{is}.

3. Alle *Subtraktionsschritte* in der i-ten Teilmatrix sind interpretierbar als der Übergang

$$A^{(i)}\vec{x} = \vec{b}^{(i)} \;\mapsto\; G_iA^{(i)}\vec{x} = G_i\vec{b}^{(i)} \qquad (i > s \geq n)$$

mit

$$G_i := \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & -\mu_{i+1,i} & 1 \\ & & \vdots & & \ddots \\ & & -\mu_{n,i} & & & 1 \end{pmatrix}.$$

$$\text{Hier gilt } \mathrm{G}_i^{-1} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & \mu_{i+1,i} & 1 & \\ & & \vdots & & \ddots \\ & & \mu_{n,i} & & 1 \end{pmatrix} .$$

Bezeichnet man die Zeilenvertauschungsmatrix bei der Aufstellung der i-ten Teilmatrix mit P_i, die eventuell dabei verwendete Spaltenvertauschungsmatrix mit P_i^*, und die Subtraktionsmatrix mit G_i, so entspricht der GAUSS-ALGORITHMUS dem Übergang von $\mathrm{A}\vec{\mathrm{x}} = \vec{\mathrm{b}}$ zu

$$\underbrace{\mathrm{G}_{n-1}\mathrm{P}_{n-1}\cdots\mathrm{G}_1\mathrm{P}_1}_{=:\mathrm{B}\ \text{(regulär!)}} \cdot A \cdot \underbrace{\mathrm{P}_1^*\cdots\mathrm{P}_{n-1}^*}_{=:\mathrm{P}^*\ \text{(regulär!)}} \cdot \vec{\mathrm{x}}' = \mathrm{G}_{n-1}\mathrm{P}_{n-1}\cdots\mathrm{G}_1\mathrm{P}_1 \cdot \vec{\mathrm{b}}$$

mit

$$\vec{\mathrm{x}}' := \mathrm{P}_{n-1}^*\cdots\mathrm{P}_1^* \cdot \vec{\mathrm{x}} \quad \text{(Variablenpermutation)} .$$

Da alle Umformungsmatrizen regulär sind (d.h. es existiert jeweils die *inverse Matrix*), hat das LGS $\mathrm{A}\vec{\mathrm{x}} = \vec{\mathrm{b}}$ die gleiche Lösungsmenge wie das resultierende System $\mathrm{R}\vec{\mathrm{x}}' = \vec{\mathrm{c}}$ mit einer oberen *Dreiecksmatrix*

$$R := \mathrm{G}_{n-1}\mathrm{P}_{n-1}\cdots\mathrm{G}_1\mathrm{P}_1\, A\, \mathrm{P}_1^*\cdots\mathrm{P}_1^* = \begin{pmatrix} r_{11} & r_{12} & \dots & r_{1n} \\ 0 & r_{22} & & r_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & \dots & 0 & r_{nn} \end{pmatrix} .$$

Wenn man $\mathrm{R}\vec{\mathrm{x}}' = \vec{\mathrm{c}}$ gelöst hat, muss nur noch die Nummerierung der Variablen $x'_1, \dots, x'_n$ geändert werden, falls $\mathrm{P}^* \neq \mathrm{I}$ gilt. Ist $\vec{\mathrm{y}}$ die Lösung von $\mathrm{R}\vec{\mathrm{x}}' = \vec{\mathrm{c}}$, so ist $\vec{\mathrm{x}}$ die Lösung des linearen Gleichungssystems $\mathrm{P}^*\vec{\mathrm{x}} = \vec{\mathrm{y}}$.

Sieht man von den Vertauschungen und Vergleichen ab, so fallen beim GAUSS-Algorithmus im Falle eines LGS mit n Gleichungen und n Variablen rund $\frac{1}{3}n^3$ multiplikative und rund $\frac{1}{3}n^3$ additive Operationen an.

Wir betrachten ein Beispiel mit 3 Gleichungen und 3 Variablen, in dem die Teilpivotisierung mit der Totalpivotisierung verglichen wird (bei zweistelliger Rundung). Dabei werden die Matrix und die rechte Seite jeweils durch ein Rechteck umrandet (dies soll ab jetzt der besseren Übersicht halber stets die „Standardansicht“ sein).

Beispiel:

Zu lösen ist mit 2-stelliger Gleitpunktrechnung:

$$\begin{pmatrix} 0.0 & 2.0 & 1.0 \\ 1.0 & 10 & 1.0 \\ 1.0 & 1.0 & 1.0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1.0 \\ 1.0 \\ 0.0 \end{pmatrix}$$

Bei exakter Rechnung würde sich hier $x_3 = \frac{7}{9}, x_2 = \frac{1}{9}, x_1 = -\frac{8}{9}$ ergeben. Mit *nachträglicher* Rundung auf zwei Stellen resultieren also die Werte $x_3 = 0.78, x_2 = 0.11, x_1 = -0.89$ als „Sollwerte“ für die Lösung des LGS.

Variante 1 (Teilpivotisierung):

0.0	2.0	1.0	1.0
1.0	10	1.0	1.0
1.0	1.0	1.0	0.0

$\Downarrow$ P_{12}

1.0	10	1.0	1.0
0.0	2.0	1.0	1.0
1.0	1.0	1.0	0.0

$\Downarrow$ $\mu_{2,1} = 0, \mu_{3,1} = -1.0$

1.0	10	1.0	1.0
×	2.0	1.0	1.0
×	−9.0	0.0	−1.0

$\Downarrow$ P_{23}

1.0	10	1.0	1.0
×	−9.0	0.0	−1.0
×	2.0	1.0	1.0

$\Downarrow$ $\mu_{3,2} = -0.22$

Variante 2 (Totalpivotisierung):

0.0	2.0	1.0	1.0
1.0	10	1.0	1.0
1.0	1.0	1.0	0.0

$P_{12}\cdot \Downarrow$ $x'_1 = x_2, x'_2 = x_1$

10	1.0	1.0	1.0
2.0	0.0	1.0	1.0
1.0	1.0	1.0	0.0

$\Downarrow$ $\mu_{2,1} = 0.2, \mu_{3,1} = 0.1$

10	1.0	1.0	1.0
×	−0.20	0.80	0.80
×	0.90	0.90	−0.10

$P_{23}\cdot \Downarrow$

1.0	10	1.0	1.0
×	0.90	0.90	−0.10
×	−0.20	0.80	0.80

$\Downarrow$ $\mu_{3,2} = -0.22$

(Teilpivotisierung):

1.0	10	1.0	1.0
×	2.0	1.0	1.0
×	×	1.0	0.78

$\Downarrow$

$x_3 = 0.78$
$x_2 = 0.11$
$x_1 = -0.90$

(Totalpivotisierung):

10	1.0	1.0	1.0
×	0.90	0.90	−0.10
×	×	1.0	0.78

$\Downarrow$

$x'_3 = 0.78 \quad \Rightarrow \; x_3 = 0.78$
$x'_2 = -0.89 \quad \Rightarrow \; x_2 = 0.11$
$x'_1 = 0.11 \quad \Rightarrow \; x_1 = -0.89$

Das mit Totalpivotisierung bestimmte Ergebnis ist etwas genauer. Dass die Matrizenfolge nicht länger wirkt, liegt daran, dass am Anfang zwei Vertauschungen kombiniert wurden

Wenn mehrere Systeme mit der gleichen Matrix A und verschiedenen rechten Seiten zu lösen sind, kann man sich beim Lösen des ersten LGS neben der resultierenden Dreiecksmatrix R die nötigen Umnummerierungen, Zeilenvertauschungen und Faktoren μ_{rs} bei den Subtraktionsschritten merken.

Da das Endsystem $\mathrm{R}\vec{x}' = \vec{c}$ von der Form

$$(\mathrm{G}_{n-1}\mathrm{P}_{n-1}\cdots\mathrm{G}_1\mathrm{P}_1 \cdot A \cdot \mathrm{P}_1^*\cdots\mathrm{P}_{n-1}^*) \;\cdot\; \vec{\mathrm{x}}' = \mathrm{G}_{n-1}\mathrm{P}_{n-1}\cdots\mathrm{G}_1\mathrm{P}_1 \;\cdot\; \vec{\mathrm{b}}$$

ist, kann man es auch der Form

$$\underbrace{\mathrm{G}'_{n-1}\cdots\mathrm{G}'_1}_{=:\mathrm{S}}(\underbrace{\mathrm{P}_{n-1}\cdots\mathrm{P}_1}_{=:\mathrm{P}}\cdot A \cdot \underbrace{\mathrm{P}_1^*\cdots\mathrm{P}_{n-1}^*}_{=:\mathrm{P}^*}) \;\cdot\; \vec{\mathrm{x}}' = \mathrm{G}'_{n-1}\cdots\mathrm{G}'_1(\mathrm{P}_{n-1}\cdots\mathrm{P}_1 \;\cdot\; \vec{\mathrm{b}})$$

schreiben.

Dabei sind die Subtraktionsmatrizen G'_k so zu wählen, dass sie nach der ***Vorabpermutierung*** aller Zeilen von A durch P und der Spalten von A durch P^* die schrittweise Überführung von $\mathrm{A}' := \mathrm{PAP}^*$ in R bewirken. Dass dies möglich ist, zeigt folgende Betrachtung:

Vertauscht P_k die k-te Zeile von A mit der s-ten Zeile (es muss $k < s \leq n$ gelten!), so ist das Matrizenprodukt $\mathrm{P}_k\mathrm{G}_r$ für alle r mit $1 \leq r < k$ gleich $\mathrm{G}'_r\mathrm{P}_k$ mit einer Matrix G'_r, die durch Vertauschen von $\mu_{r,k}$ mit $\mu_{r,s}$ in G_r entsteht:

$$\mathrm{G_r} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & 1 & & & \\ & -\mu_{r+1,r} & 1 & & \\ 0 & \vdots & \ddots & \ddots & \\ & -\mu_{k,r} & & & 0 \\ & \vdots & & \ddots & \\ & -\mu_{s,r} & & & \\ & \vdots & & & \\ & -\mu_{n,r} & & & 1 \end{pmatrix} \quad \mapsto \quad \mathrm{G'_r} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & 1 & & & \\ & -\mu_{r+1,r} & 1 & & \\ 0 & \vdots & \ddots & \ddots & \\ & -\mu_{s,r} & & & 0 \\ & \vdots & & \ddots & \\ & -\mu_{k,r} & & & \\ & \vdots & & & \\ & -\mu_{n,r} & & & 1 \end{pmatrix}$$

Damit erhält man:

$$\begin{aligned} \mathrm{G}_{n-1}\mathrm{P}_{n-1}\mathrm{G}_{n-2}\mathrm{P}_{n-2}\cdots\mathrm{G}_1\mathrm{P}_1 &= \mathrm{G}_{n-1}\mathrm{G'}_{n-2}\mathrm{P}_{n-1}\mathrm{P}_{n-2}\cdots\mathrm{G}_1\mathrm{P}_1 = \ldots \\ &= \mathrm{G}_{n-1}\mathrm{G'}_{n-2}\ldots\mathrm{G'}_1\mathrm{P}_{n-1}\ldots\mathrm{P}_1 \end{aligned}$$

Setzt man $\mathrm{G'}_{n-1} := \mathrm{G}_{n-1}$, so ergibt sich die behauptete Darstellung.

Damit ist folgender „Trick“ möglich, um die Koeffizienten von $\mathrm{G'}_1, \mathrm{G'}_2, \ldots, \mathrm{G'}_{n-1}$ unter der Diagonale zu erhalten: Sobald $\mathrm{A}^{(i)}$ bestimmt ist, speichert man für $i < k \leq n$ die Faktoren μ_{ik} auf den Plätzen der durch Subtraktion in 0 verwandelten Koeffizienten a_{ik} der Matrix $\mathrm{A}^{(i)}$. So ergeben sich bei nachfolgenden Zeilenvertauschungen automatisch die Koeffizienten der Matrizen $\mathrm{G'}_j$ für $i < j \leq n$. Nachfolgende Spaltenvertauschungen wirken sich dabei **nicht** auf die berechneten μ_{ik} aus, da sie erst ab einer späteren Spalte wirksam werden.

Mit Hilfe der so gespeicherten Matrix

$$\mathrm{K} := \left(\begin{array}{ccccc|c} r_{11} & r_{12} & \ldots & r_{1,n-1} & r_{1n} & c_1 \\ t_{21} & r_{22} & & r_{2,n-1} & r_{2n} & c_2 \\ \vdots & & \ddots & & \vdots & \vdots \\ t_{n1} & t_{n2} & \ldots & t_{n-1,n} & r_{nn} & c_n \end{array}\right)$$

kann dann zunächst die Lösung $\vec{\mathrm{x}}'$ des ersten LGS berechnet werden, indem das LGS $\mathrm{R}\vec{\mathrm{x}}' = \vec{\mathrm{c}}$ durch Rückwärtseinsetzen gelöst wird.

Da $S^{-1} = (\mathrm{G'}_{n-1}\cdots\mathrm{G'}_1)^{-1}$ gleich

$$\mathrm{L} := \begin{pmatrix} 1 & 0 & \ldots & 0 \\ t_{21} & 1 & & 0 \\ \vdots & & \ddots & \vdots \\ t_{n-1,1} & & & 0 \\ t_{n1} & \ldots & t_{n,n-1} & 1 \end{pmatrix}$$

ist, kann nach Vorgabe einer neuen rechten Seite $\vec{\mathrm{v}}$ das LGS $\mathrm{A}\vec{\mathrm{x}} = \vec{\mathrm{v}}$ folgendermaßen gelöst werden:

1. Bringe die Koeffizienten $v_1, \ldots, v_n$ in die Reihenfolge, die beim ersten Lösen aus den Zeilenvertauschungen resultierte (d.h. bestimme $\mathrm{P}\vec{\mathrm{v}}$)).
2. Löse nacheinander die Systeme
   (1) $\mathrm{L}\vec{\mathrm{u}} = \mathrm{P}\vec{\mathrm{v}}$ (durch *Vorwärtseinsetzen*)
   (2) $\mathrm{R}\vec{\mathrm{y}} = \vec{\mathrm{u}}$ (durch *Rückwärtseinsetzen*)
   (3) $\mathrm{P}^*\vec{\mathrm{x}} = \vec{\mathrm{y}}$ (reines Umnummerieren der Variablen!)

Aus der Bildung der Matrizen G, P und P^* folgt, dass für die Matrix $A' := PAP^*$ gilt:

$$P\,A\,P^* \;=\; L \cdot R\,.$$

Definition 2.2 *Die Darstellung einer regulären Matrix A′ in der Form* $P\,A\,P^* \;=\; L \cdot R$ *mit einer unteren Dreiecksmatrix L und einer oberen Dreiecksmatrix R heißt* ***L-R-Zerlegung*** *der Matrix A′.*

Wir betrachten ein Beispiel mit exakter Rechnung, in dem nur die Matrix A umgeformt wird.

Beispiel:

$$\begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \underset{P_1^* = P_{13}}{\mapsto} \begin{pmatrix} 3 & 2 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \mapsto \left(\begin{array}{ccc} 3 & 2 & 1 \\ \hline 1/3 & 1/3 & -1/3 \\ 1/3 & 1/3 & 2/3 \end{array}\right)$$

$$\underset{\substack{P_2^* = P_{23} \\ P_2 = P_{23}}}{\mapsto} \left(\begin{array}{ccc} 3 & 1 & 2 \\ \hline 1/3 & 2/3 & 1/3 \\ 1/3 & -1/3 & 1/3 \end{array}\right) \mapsto \left(\begin{array}{ccc} 3 & 1 & 2 \\ \hline 1/3 & 2/3 & 1/3 \\ 1/3 & -1/2 & 1/2 \end{array}\right)$$

Bei der Multiplikation von L und R ergibt sich A′:

$$\begin{pmatrix} 1 & 0 & 0 \\ 1/3 & 1 & 0 \\ 1/3 & -1/2 & 1 \end{pmatrix} \cdot \begin{pmatrix} 3 & 1 & 2 \\ 0 & 2/3 & 1/3 \\ 0 & -1/2 & 1/2 \end{pmatrix} = \begin{pmatrix} 3 & 1 & 2 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} = PAP^*\,.$$

Würde jetzt die rechte Seite $\vec{v} = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}$ vorgegeben, so könnte man das System $A\vec{x} = \vec{v}$ nach Vertauschung von v_2 mit v_3 so lösen:

(1) Löse $\begin{pmatrix} 1 & 0 & 0 \\ 1/3 & 1 & 0 \\ 1/3 & -1/2 & 1 \end{pmatrix} \vec{u} = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}$.
Es ergibt sich $u_1 = 1, u_2 = 2/3, u_3 = 2$.

(2) Löse $\begin{pmatrix} 3 & 1 & 2 \\ 0 & 2/3 & 1/3 \\ 0 & 0 & 1/2 \end{pmatrix} \vec{y} = \begin{pmatrix} 1 \\ 2/3 \\ 2 \end{pmatrix}$.
Es ergibt sich $y_3 = 4, y_2 = -1, y_1 = -2\,.$

(3) Also ergibt sich die Lösung zu $x_1 = -1, x_2 = 4, x_3 = -2$.

Für Fehlerbetrachtungen bei Matrizen ist folgende Vereinbarung üblich:

Definition 2.3 *Sind A und B quadratische Matrizen mit derselben Zeilenzahl n, so nennt man A genau dann* **kleiner oder gleich** *B (Kurznotation: $A \leq B$), wenn $a_{ik} \leq b_{ik}$ für alle i und k von 1 bis n gilt.*

Mit Hilfe von Fehlerananlysen wurde gezeigt, dass für eine mit Gleitpunktrechnung bestimmte L-R-Zerlegung einer $n \times n$-Matrix A stets gilt:

Bezeichnet man die mit Gleitpunktarithmetik berechneten Dreiecksmatrizen mit $\bar{\mathrm{L}}$ und $\bar{\mathrm{R}}$, so gilt für die durch

$$f_{ik} := \left| a_{ik} - \sum_{s=1}^{i} \bar{l}_{is}\bar{\mathrm{r}}_{sk} \right| \quad (i = 1, \ldots, n;\ k = 1, \ldots, n)$$

definierte *Fehlermatrix* F im Falle der Teil- oder Totalpivotsuche:

$$\mathrm{F} \leq 2a\frac{\mathrm{eps}}{1-\mathrm{eps}} \begin{pmatrix} 0 & \cdots & 0 \\ 1 & \cdots & 1 \\ 1 & 2 & \cdots & 2 \\ 1 & 2 & 3 & \cdots & 3 \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & n-2 & n-2 & n-2 \\ 1 & 2 & 3 & \ldots & n-2 & n-1 & n-1 \end{pmatrix} \quad \textit{(Formel F)}$$

Daei ist a das Betragsmaximum aller Pivotelemente, die in den Matrizen $\mathrm{A}^{(1)}, \ldots, \mathrm{A}^{(n-1)}$ bei einer Totalpivotisierung auftreten würden.

Für a gibt es Abschätzungen, die vom Matrizentyp und Pivotisierungsvariante abhängen:

Bezeichnet man das Betragsmaximum aller Koeffizienten von A mit a_0, so gilt

(1) $a \leq 2^{n-1} \cdot a_0$ bei Teilpivotsuche,

(2) $a \leq 2a_0$ bei *Tridiagonalmatrizen*

$$\begin{pmatrix} a_1 & b_2 & & & 0 \\ c_2 & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & b_n \\ 0 & & & c_n & a_n \end{pmatrix}$$

und Teilpivotsuche,

(3) $a \leq (n-1) \cdot a_0$ bei *Hessenbergmatrizen*, d. h. Matrizen des Typs

$$\begin{pmatrix} a_{11} & \cdots\cdots & a_{1n} \\ b_{21} & a_{22} & \cdots & a_{2n} \\ & \ddots & \ddots & \vdots \\ 0 & & b_{n-1,n} & a_{nn} \end{pmatrix},$$

und Teilpivotsuche,

(4) $a \leq f(n)$ mit

$$f(n) := \sqrt{2 \cdot \sqrt{3} \cdot \sqrt[3]{4} \cdots \sqrt[n-1]{n} \cdot n}$$

bei Totalpivotsuche unabhängig vom Matrizentyp.
Einige Werte von $f(n)$ sind:

n	10	20	50	100
$f(n)$ gerundet	19	67	530	3300

In der Praxis werden meistens nur Matrizen beobachtet, bei denen a höchstens $n \cdot a_0$ beträgt.

2.1.3 Das Gauß-Jordan-Verfahren

Hat man ein eindeutig lösbares LGS $A\vec{x} = \vec{b}$ in die Form $R\vec{x}' = \vec{c}$ mit einer oberen Dreiecksmatrix R gebracht, so kann man die erweiterte Matrix

$$\left(\begin{array}{cccc|c} r_{11} & r_{12} & \cdots & r_{1n} & c_1 \\ & r_{22} & & r_{2n} & c_2 \\ & & \ddots & \vdots & \vdots \\ 0 & & & r_{nn} & c_n \end{array}\right)$$

mit Zeilenumformungen auf eine Form bringen, bei der sich die Lösung ablesen lässt. Dazu dividiert man die letzte Zeile durch r_{nn} und subtrahiert dann geeignete Vielfache dieser Zeile von allen darüberstehenden Zeilen:

$$\left(\begin{array}{cccc|c} r_{11} & r_{12} & \cdots & r_{1n} & c_1 \\ & r_{22} & & r_{2n} & c_2 \\ & & \ddots & \vdots & \vdots \\ 0 & & & 1 & c'_n \end{array}\right) \Rightarrow \left(\begin{array}{cccc|c} r_{11} & r_{12} & \cdots & 0 & c'_1 \\ & r_{22} & & 0 & c'_2 \\ & & \ddots & 0 & \\ 0 & & & 1 & c'_n \end{array}\right)$$

mit $c'_j := c_j - r_{n-1,n} \cdot c'_n$

Setzt man dieses Verfahren mit der vorletzten Zeile fort, so erhält man schließlich an Stelle von R die Einheitsmatrix I und die rechte Seite gibt die Lösung $\vec{x}'$ an.

Wenn das LGS mehr als eine Lösung hat, stellt man bei der Durchführung des GAUSS-Algorithmus mit Totalpivotisierung fest, dass man an Stelle einer Dreiecksmatrix eine Matrix erhält, in der es ab einem Zeilenindex $s < n$ nur noch ***Nullzeilen*** gibt und $c_n = c_{n-1} = \ldots = c_s = 0$ gilt. In diesem Fall lassen sich die beschrieben Umformungen erst ab der s-ten Zeile durchführen. Wir formulieren daher das Verfahren gleich so, dass der Fall mehrdeutig lösbarer LGS mit erfasst wird:

Gauss-Jordan-Algorithmus:

(0) (A|b) muss mit dem Gauß-Algorithmus (ggfalls mit Totalpivotisierung) auf die Stufenform

$$\left(\begin{array}{cccc|c} r_{11} & r_{12} & \ldots & r_{1n} & c_1 \\ & r_{22} & r_{2n} & & c_2 \\ & & \ddots & \vdots & \vdots \\ 0 & & & 1 & c'_n \end{array}\right) \quad \text{bzw.} \quad \left(\begin{array}{ccccc|c} r_{11} & r_{12} & \ldots & \ldots & r_{1n} & c_1 \\ & r_{22} & & & r_{2n} & c_2 \\ 0 & & \ddots & & \vdots & \vdots \\ & & & r_{ss} \ldots & r_{sn} & c_s \\ 0 & \ldots & \ldots & \ldots & 0 & c_{s+1} \\ \vdots & & & & \vdots & \vdots \\ 0 & \ldots & \ldots & \ldots & 0 & c_n \end{array}\right)$$

gebracht werden.

Im ersten Fall gilt $r_{11}, r_{22}, \ldots, r_{nn} \neq 0$ und das Jordanverfahren (1) ist mit dem Teilmatrixzählerwert $k := n$ zu starten.

Ist im zweiten Fall mindestens einer der Werte $c_{s+1}, \ldots, c_n$ von Null verschieden, so ist das LGS unlösbar (STOP).

Gilt im zweiten Fall $c_{s+1} = \ldots = c_n = 0$, so ist das Jordanverfahren (1) mit Teilmatrixzähler $k := s$ zu starten.

(1) Setze r_{kk} gleich 1, nachdem $r_{k,k+1}, \ldots, r_{k,n}, c_k$ durch r_{kk} dividiert wurden.

(2) Subtrahiere für $j = k-1$ bis hinunter zu $j = 1$ von der j-ten Zeile der erweiterten Matrix das $r_{j,k}$-fache der k-ten Zeile (in der k-ten Spalte resultieren Nullen!). Die Werte in der erweiterten Matrix sind dabei zu überschreiben.

(3) Falls $k > 1$ gilt, vermindere k um 1 und gehe nach (1).
Falls $k = 1$ gilt: dividiere $r_{1,s+1}, \ldots, r_{1,n}, c_1$ durch r_{11} und setze danach $r_{11} := 1$.

Da es sich bei den beschriebenen Umformungen um Äquivalenzumformungen handelt, ändert sich Lösungsmenge des LGS nichts. Bei lösbaren LGS erhält man als Endform

entweder ein LGS des Typs

$$\begin{pmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{pmatrix} \begin{pmatrix} x'_1 \\ \vdots \\ x'_n \end{pmatrix} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}$$

oder

$$\begin{pmatrix} 1 & & 0 & r_{1,s+1} & \dots & r_{1,n} \\ & \ddots & & \vdots & & \vdots \\ 0 & & 1 & r_{s,s+1} & \dots & r_{s,n} \\ 0 & & \dots & \dots & & 0 \\ \vdots & & & & & \vdots \\ 0 & & \dots & \dots & & 0 \end{pmatrix} \begin{pmatrix} x'_1 \\ \\ \vdots \\ \\ x'_n \end{pmatrix} = \begin{pmatrix} c_1 \\ \\ \vdots \\ \\ c_n \end{pmatrix}.$$

Im ersten Fall lässt sich sofort $\vec{\mathrm{x}}' = \vec{c}$ ablesen. Beim zweiten Fall kann man offensichtlich den Variablen $x'_{s+1}, \dots, x'_n$ beliebige Werte $\lambda_1, \dots, \lambda_{n-s}$ zuschreiben und man erhält dann eine Lösung in der Form

$$\begin{aligned} x'_i &= c_i - \sum_{k=1}^{n-s} r_{i,s+k}\lambda_k \quad \text{für } i = 1, \dots, s\,, \\ x'_j &= \lambda_{j-s} \qquad\qquad\qquad\quad \text{für } j = s+1, \dots, n\,. \end{aligned}$$

Dies lässt sich auch vektoriell schreiben:

$$\begin{pmatrix} x'_1 \\ \vdots \\ x'_s \\ x'_{s+1} \\ \\ \vdots \\ \\ x'_n \end{pmatrix} = \begin{pmatrix} c'_1 \\ \vdots \\ c'_s \\ 0 \\ \\ \vdots \\ \\ 0 \end{pmatrix} + \lambda_1 \begin{pmatrix} -r_{1,s+1} \\ \vdots \\ -r_{s,s+1} \\ 1 \\ 0 \\ \\ \vdots \\ \\ 0 \end{pmatrix} + \lambda_2 \begin{pmatrix} -r_{1,s+2} \\ \vdots \\ -r_{s,s+2} \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \dots + \lambda_{n-s} \begin{pmatrix} -r_{1,n} \\ \vdots \\ -r_{s,n} \\ 0 \\ \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

mit $\lambda_1, \dots, \lambda_{n-s} \in \mathbb{R}$.

Dies ist die Darstellung einer $(n-s)$-dimensionalen linearen Mannigfaltigkeit des $\mathbb{R}^n$ im Sinne der linearen Algebra. Falls $n = 3$ und $s = 2$ gilt, lässt sich eine solche Menge z.B. als Beschreibung einer Ebene auffassen.

Vom Standpunkt der Numerik aus handelt es sich bei der Lösung mehrdeutiger Systeme auf dem Rechner um ein fragwürdiges Verfahren, da sich die „Nullzeilen“ der erweiterten Matrix so gut wie nie ganz exakt ergeben. Damit hat man das Problem, berechnete Koeffizienten beim Unterschreiten einer geeigneten Betragsschranke gleich 0 setzen zu müssen. Das Programmlisting einer PASCAL-Funktion, in der alle bisher behandelten Umformungsroutinen für (A|b) umgesetzt sind, zeigt dies:

```
function GAUSS( var AM      :  hmatrix;
                n           :  integer;
                var X       :  vektor;
                var Index   :  gvektor;
                var Zeilen  :  gvektor;
                ug          :  extended): integer;
```

(∗

Diese Funktion bestimmt eine Lösung x zu Ax = b, wenn es mindestens eine Lösung gibt. Dabei ist die rechte Seite b in der letzten Spalte der erweiterten Matrix AM = (A,b) gespeichert.

Die Datentypen sind:

```
 hmatrix  =  ARRAY[1..Nmax1,1..Nmax] OF EXTENDED;
 vektor   =  ARRAY[1..Nmax] OF EXTENDED;
 gvektor  =  ARRAY[1..Nmax] OF INTEGER;
```

Es wird 0 als Funktionswert zurückgegeben, wenn A regulär ist. In X steht dann die Lösung des LGS.

Wenn das System mehr als eine Lösung hat, wird die Dimension der Lösungsmannigfaltigkeit zurückgegeben.

In X steht dann eine spezielle Lösung des LGS und eine Basis des homogenen Systems ist in AM gespeichert.

Der Rückgabewert -1 bedeutet, dass das System unlösbar ist.

Bei eindeutiger Lösbarkeit ist aus der Matrix AM am Schluss die L-R-Zerlegung der Koeffizientenmatrix (bei vorheriger Zeilen- und Spaltenumordnung!) abzulesen.

Nmax ist global festzulegen, es muss Nmax1 = Nmax + 1 gelten.

ug ist als geeignetes Vielfaches von eps zu wählen.
∗)

```
var
 i,j,k,np,zzi          :  integer;
 verg,max,nug,sum      :  extended;
 f                     :  boolean;
 Y                     :  array[1..Nmax] of extended;
```

(∗ Es folgt eine lokale Prozedur für die Totalpivotisierung. ∗)

```
function pivot(i: integer): extended;

var
 j,k,z,zi,si    :   integer;
 max,verg   :   extended;

begin
    max:= abs(AM[i][i]);
    zi:= i;
    si:= i;
    for j:= i to n do
    for k:= i to n do
    begin
      verg:= abs(AM[j][k]);
      if(verg > max) then
      begin
        max:= verg;
        si:= k;
        zi:= j;
      end;
    end;
    if max > nug then
    begin
      if(zi <> i) then
      begin
        for k:= 1 to np do
        begin
          verg:= AM[i][k];
          AM[i][k]:= AM[zi][k];
          AM[zi][k]:= verg;
        end;
      z:= Zeilen[i];
      Zeilen[i]:= Zeilen[zi];
      Zeilen[zi]:= z;
    end;
    if(i <> si) then
    begin
      z:= Index[i];
      Index[i]:= Index[si];
      Index[si]:= z;
```

```
          for j:= 1 to n do
          begin
            verg:= AM[j][i];
            AM[j][i]:= AM[j][si];
            AM[j][si]:= verg;
          end;else max:= 0;
      pivot:= max;
end;

(* - - - - Gauss bzw. Gauss-Jordan-Algoritmus - - - - *)

begin
      np:= n + 1;
      f:= true;
      nug:= n*n*ug;
      for i:= 1 to n do
      begin
        Index[i]:= i;
        Zeilen[i]:= i;
      end;
      i:= 0;
      zzi:= 0;
      repeat
        i:= i+1;
        if pivot(i) >= nug then
        begin
          max:= AM[i][i];
          for j:= i+1 to n do
          begin
            verg:= AM[j][i]/max;
            for k:= i+1 to np do AM[j][k]:= AM[j][k] - verg*AM[i][k];
            AM[j][i]:= verg;
          end;
        end else
        begin
          zzi:= i;
          for j:= zzi to n do
          begin
            if abs(AM[j][np]) > nug then f:= false
            else for k:= i to np do AM[j][k]:= 0;
            AM[j,j]:= 1;
          end;
        end;
      until (i = n) or (zzi > 0) or not f;
```

```
    if f then
    begin
      if zzi > 0 then
      begin
        for i:= zzi-1 downto 1 do
        begin
          for k:= zzi to np do AM[i,k]:= AM[i,k]/AM[i,i];
          for j:= i-1 downto 1 do
          for k:= zzi to np do AM[j,k]:= AM[j,k] - AM[j,i]*AM[i,k];
        end;
        GAUSS:= n-zzi+1;
        for j:= 1 to n do X[Index[j]]:= AM[j,np];
        for i:= zzi-1 downto 1 do
        for k:= zzi to n do AM[i,k]:= -AM[i,k];
      end else
      begin
        for j:= n downto 1 do
        begin
          sum:= AM[j][np];
          for k:= j+1 to n do sum:= sum - AM[j][k]*Y[k];
          if abs(sum) < abs(AM[j][j])*1.0E20 then
          Y[j]:= sum/AM[j][j]
          else f:= false;
        end;
        if f then
        begin
          GAUSS:= 0;
          for j:= 1 to n do X[Index[j]]:= Y[j];
        end else GAUSS:= -1;
      end;
    end else GAUSS:= -1;
end;
```

Hier wird nach Vorgabe einer Schranke „ug“ dafür gesorgt, dass Koeffizienten a_{ik} bzw. c_i mit $|a_{ik}| < n^2 \cdot \mathrm{ug}$ bzw. $|c_i| < n^2 \cdot \mathrm{ug}$ gleich 0 gesetzt werden. Der quadratische Faktor hat sich bei der Wahl $\mathrm{ug} = \mathrm{eps}$ gut bewährt.

Abschließend sollen Beispiele betrachtet werden, in denen mit exakter Rechnung umgeformt wird.

Beispiel a)

Zu lösen ist das LGS $A\vec{x} = \vec{b}$ mit

$$\begin{pmatrix} 2 & 3 & -1 & 1 \\ 1 & 0 & 2 & 1 \\ -1 & 2 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \vec{x} = \begin{pmatrix} 2 \\ 4 \\ 0 \\ 2 \end{pmatrix}.$$

Formt man die erweiterte Matrix (A|b) mit dem GAUSS-Algorithmus um, so erhält man bei Teilpivotisierung

$$\left(\begin{array}{cccc|c} 2 & 3 & -1 & 1 & 2 \\ 0 & \frac{7}{2} & \frac{1}{2} & \frac{1}{2} & 1 \\ 0 & 0 & \frac{19}{7} & \frac{5}{7} & \frac{24}{7} \\ 0 & 0 & 0 & \frac{84}{133} & \frac{84}{133} \end{array}\right).$$

Das JORDAN-Verfahren liefert dann folgende Matrizen:

$$\left(\begin{array}{cccc|c} 2 & 3 & -1 & 1 & 2 \\ 0 & \frac{7}{2} & \frac{1}{2} & \frac{1}{2} & 1 \\ 0 & 0 & \frac{19}{7} & \frac{5}{7} & \frac{24}{7} \\ 0 & 0 & 0 & 1 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{cccc|c} 2 & 3 & -1 & 0 & 1 \\ 0 & \frac{7}{2} & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & \frac{19}{7} & 0 & \frac{19}{7} \\ 0 & 0 & 0 & 1 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{cccc|c} 2 & 3 & -1 & 0 & 1 \\ 0 & \frac{7}{2} & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right)$$

$$\Rightarrow \left(\begin{array}{cccc|c} 2 & 3 & 0 & 0 & 2 \\ 0 & \frac{7}{2} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{cccc|c} 2 & 3 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{cccc|c} 2 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{array}\right)$$

Nach der Division durch 2 in der ersten Zeile lässt sich der Lösungsvektor $\vec{x} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$ ablesen.

Beispiel b)

Zu lösen ist das LGS $A\vec{x} = \vec{b}$ mit

$$\begin{pmatrix} 2 & 3 & -1 & 1 \\ 1 & 0 & 2 & 1 \end{pmatrix} \vec{x} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

Formt man die erweiterte Matrix (A|b) (anfänglich wurde die Matrix A durch Hinzufügen von „Nullzeilen“ zu einer quadratischen Matrix ergänzt) mit dem GAUSS-Algorithmus um, so erhält man bei Teilpivotisierung

$$\left(\begin{array}{cccc|c} 2 & 3 & -1 & 1 & 1 \\ 0 & -\frac{3}{2} & \frac{5}{2} & \frac{1}{2} & -\frac{3}{2} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right).$$

Das JORDAN-Verfahren liefert dann in zwei Schritten folgende Matrizen:

$$\left(\begin{array}{cccc|c} 2 & 3 & -1 & 1 & 1 \\ 0 & 1 & -\frac{5}{3} & -\frac{1}{3} & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right) \Rightarrow \left(\begin{array}{cccc|c} 2 & 0 & 4 & 2 & -2 \\ 0 & 1 & -\frac{5}{3} & -\frac{1}{3} & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right) \Rightarrow \left(\begin{array}{cccc|c} 1 & 0 & 2 & 1 & -1 \\ 0 & 1 & -\frac{5}{3} & -\frac{1}{3} & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right).$$

Wählt man nun x_3 und x_4 als Parameter, d.h. setzt $x_3 := \lambda_1$, $x_4 := \lambda_2$ mit $\lambda_1, \lambda_2 \in RZ$, so lässt sich dies in der letzten Matrix folgendermaßen schreiben:

$$\left(\begin{array}{cccc|c} 1 & 0 & 2 & 1 & -1 \\ 0 & 1 & -\frac{5}{3} & -\frac{1}{3} & 1 \\ 0 & 0 & 1 & 0 & \lambda_1 \\ 0 & 0 & 0 & 1 & \lambda_2 \end{array}\right)$$

Das JORDAN-Verfahren kann nun noch einmal auf die dritte und vierte Spalte angewendet werden und liefert:

$$\left(\begin{array}{cccc|c} 1 & 0 & 2 & 0 & -1-\lambda_2 \\ 0 & 1 & -\frac{5}{3} & 0 & 1+\frac{1}{3}\lambda_2 \\ 0 & 0 & 1 & 0 & \lambda_1 \\ 0 & 0 & 0 & 1 & \lambda_2 \end{array}\right) \Rightarrow \left(\begin{array}{cccc|c} 1 & 0 & 0 & 0 & -1-\lambda_2-2\lambda_1 \\ 0 & 1 & 0 & 0 & 1+\frac{5}{3}\lambda_1+\frac{1}{3}\lambda_2 \\ 0 & 0 & 1 & 0 & \lambda_1 \\ 0 & 0 & 0 & 1 & \lambda_2 \end{array}\right)$$

Also lässen sich alle Lösungsvektoren in der Form

$$\vec{\mathrm{x}} = \begin{pmatrix} -1 \\ 1 \\ 0 \\ 0 \end{pmatrix} + \lambda_1 \begin{pmatrix} -2 \\ \frac{5}{3} \\ 1 \\ 0 \end{pmatrix} + \lambda_2 \begin{pmatrix} -1 \\ \frac{1}{3} \\ 0 \\ 1 \end{pmatrix} \qquad (\lambda_1, \lambda_2 \in \mathbb{R})$$

darstellen.

Eine weitere Anwendung des Verfahrens ist die Invertierung regulärer Matrizen. Fasst man die Inverse einer regulären $n \times n$-Matrix A als Matrix von n unbekannten Spaltenvektoren auf, so entspricht die Bestimmung von A dem simultanen Lösen der n linearen Gleichungssysteme

$$\mathrm{A}\vec{\mathrm{x}}_1 = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}, \ \mathrm{A}\vec{\mathrm{x}}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ \ldots \ , \mathrm{A}\vec{\mathrm{x}}_n = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

Die Matrix X mit den Koeffizienten x_{ik} ist dann gleich A^{-1}. Das zugehörige Rechenverfahren sieht dann so aus, dass die um rechts mit der Einheitsmatrix I erweiterte Matrix $(\mathrm{A}|\,\mathrm{I})$ mit dem GAUSS-JORDAN-Verfahren mit Teilpivotisierung so umgeformt wird, dass in der linken Hälfte der erweiterten Matrix die Einheitsmatrix I entsteht. Dann hat man jedoch die obigen LGS gelöst und die rechte Hälfte der umgeformten erweiterten Matrix ist A^{-1}:

Beispiel: Gesucht ist die Inverse A^{-1} der Matrix $A = \begin{pmatrix} 2 & 0 & 1 & 2 \\ 1 & 2 & 1 & 1 \\ 0 & 1 & 2 & -1 \\ -1 & -1 & -3 & 1 \end{pmatrix}$.

Rechenschema:

$$\left(\begin{array}{cccc|cccc} 2 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 1 & 2 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & -1 & 0 & 0 & 1 & 0 \\ -1 & -1 & -3 & 1 & 0 & 0 & 0 & 1 \end{array}\right)$$

$\Downarrow$ Zeile 2 $-\frac{1}{2}$ Zeile 1
Zeile 4 $+\frac{1}{2}$ Zeile 1

$$\left(\begin{array}{cccc|cccc} 2 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 2 & \frac{1}{2} & 0 & -\frac{1}{2} & 1 & 0 & 0 \\ 0 & 1 & 2 & -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & -\frac{5}{2} & 2 & \frac{1}{2} & 0 & 0 & 1 \end{array}\right)$$

$\Downarrow$ Zeile 3 $-\frac{1}{2}$ Zeile 2
Zeile 4 $+\frac{1}{2}$ Zeile 2

$$\left(\begin{array}{cccc|cccc} 2 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 2 & \frac{1}{2} & 0 & -\frac{1}{2} & 1 & 0 & 0 \\ 0 & 0 & \frac{7}{4} & -1 & \frac{1}{4} & -\frac{1}{2} & 1 & 0 \\ 0 & 0 & -\frac{9}{4} & 2 & \frac{1}{4} & \frac{1}{2} & 0 & 1 \end{array}\right)$$

$\Downarrow$ Zeile 4 $+\frac{9}{7}$ Zeile 3

$$\left(\begin{array}{cccc|cccc} 2 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 2 & \frac{1}{2} & 0 & -\frac{1}{2} & 1 & 0 & 0 \\ 0 & 0 & \frac{7}{4} & -1 & \frac{1}{4} & -\frac{1}{2} & 1 & 0 \\ 0 & 0 & 0 & \frac{5}{7} & \frac{4}{7} & -\frac{1}{7} & \frac{9}{7} & 1 \end{array}\right)$$

$\Downarrow$ Zeile 4 $\cdot \frac{7}{5}$

$$\left(\begin{array}{cccc|cccc} 2 & 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 2 & \frac{1}{2} & 0 & -\frac{1}{2} & 1 & 0 & 0 \\ 0 & 0 & \frac{7}{4} & -1 & \frac{1}{4} & -\frac{1}{2} & 1 & 0 \\ 0 & 0 & 0 & 1 & \frac{4}{5} & -\frac{1}{5} & \frac{9}{5} & \frac{7}{5} \end{array}\right)$$

$\Downarrow$ Zeile 3 + Zeile 4
Zeile 1 - 2 Zeile 4

$$\left(\begin{array}{cccc|cccc} 2 & 0 & 1 & 0 & -\frac{3}{5} & \frac{2}{5} & -\frac{18}{5} & -\frac{14}{5} \\ 0 & 2 & \frac{1}{2} & 0 & -\frac{1}{2} & 1 & 0 & 0 \\ 0 & 0 & \frac{7}{4} & 0 & \frac{21}{20} & -\frac{7}{10} & \frac{14}{5} & \frac{7}{5} \\ 0 & 0 & 0 & 1 & \frac{4}{5} & -\frac{1}{5} & \frac{9}{5} & \frac{7}{5} \end{array}\right)$$

$$\Downarrow \text{ Zeile } 3 \cdot \tfrac{4}{7}$$

$$\left(\begin{array}{cccc|cccc} 2 & 0 & 1 & 0 & -\frac{3}{5} & \frac{2}{5} & -\frac{18}{5} & -\frac{14}{5} \\ 0 & 2 & \frac{1}{2} & 0 & -\frac{1}{2} & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & \frac{3}{5} & -\frac{2}{5} & \frac{8}{5} & \frac{4}{5} \\ 0 & 0 & 0 & 1 & \frac{4}{5} & -\frac{1}{5} & \frac{9}{5} & \frac{7}{5} \end{array}\right)$$

$$\Downarrow \begin{array}{l} \text{Zeile } 2 - \frac{1}{2}\,\text{Zeile } 3 \\ \text{Zeile } 1 - \text{Zeile } 3 \end{array}$$

$$\left(\begin{array}{cccc|cccc} 2 & 0 & 0 & 0 & -\frac{6}{5} & \frac{4}{5} & -\frac{26}{5} & -\frac{18}{5} \\ 0 & 2 & 0 & 0 & -\frac{4}{5} & \frac{6}{5} & -\frac{4}{5} & -\frac{2}{5} \\ 0 & 0 & 1 & 0 & \frac{3}{5} & -\frac{2}{5} & \frac{8}{5} & \frac{4}{5} \\ 0 & 0 & 0 & 1 & \frac{4}{5} & -\frac{1}{5} & \frac{9}{5} & \frac{7}{5} \end{array}\right)$$

$$\Downarrow \begin{array}{l} \text{Zeile } 2 \cdot \frac{1}{2} \\ \text{Zeile } 1 \cdot \frac{1}{2} \end{array}$$

$$\left(\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & -\frac{3}{5} & \frac{2}{5} & -\frac{13}{5} & -\frac{9}{5} \\ 0 & 1 & 0 & 0 & -\frac{2}{5} & \frac{3}{5} & -\frac{2}{5} & -\frac{1}{5} \\ 0 & 0 & 1 & 0 & \frac{3}{5} & -\frac{2}{5} & \frac{8}{5} & \frac{4}{5} \\ 0 & 0 & 0 & 1 & \frac{4}{5} & -\frac{1}{5} & \frac{9}{5} & \frac{7}{5} \end{array}\right)$$

Also ist $A^{-1} = \begin{pmatrix} -\frac{3}{5} & \frac{2}{5} & -\frac{13}{5} & -\frac{9}{5} \\ -\frac{2}{5} & \frac{3}{5} & -\frac{2}{5} & -\frac{1}{5} \\ \frac{3}{5} & -\frac{2}{5} & \frac{8}{5} & \frac{4}{5} \\ \frac{4}{5} & -\frac{1}{5} & \frac{9}{5} & \frac{7}{5} \end{pmatrix}$ die inverse Matrix von A.

Wäre hier eine Teilpivotisierung durchgeführt worden, so hätte sich trotzdem die gleiche Endmatrix in der rechten Hälfte ergeben (man vertausche in der dritten Matrix des Rechenschemas die beiden letzten Zeilen miteinander und prüfe dies nach!).

2.1.4 Nachiteration

Wir beschränken uns jetzt auf den Fall eines LGS des Typs $\mathrm{A}\vec{y} = \vec{\mathrm{b}}$ mit regulärer $n \times n$-Matrix A. Dann ist die Lösung gegeben durch $\vec{\mathrm{x}} = \mathrm{A}^{-1}\vec{\mathrm{b}}$. Sieht man $\vec{\mathrm{b}}$ als Eingangsvektor an, so handelt es sich bei dem Berechnungsproblem um die Anwendung der Vektorfunktion f mit der Vorschrift $\vec{\mathrm{x}} := \mathrm{A}^{-1}\vec{\mathrm{b}}$.

Bezeichnet man die Koeffizienten von A^{-1} mit c_{ik}, so gilt für den unvermeidbaren relativen Fehler einer solchen Berechnung:

$$\varepsilon^*_{x_k} = \alpha \; + \; \sum_{j=1}^{n} \frac{b_j}{x_k}\, c_{kj}\, \varepsilon_{b_j} \qquad (k = 1, \ldots, n) \; .$$

Dies liegt daran, dass $x_k = \sum_{j=1}^{n} c_{kj}\, b_j$ gilt. Also ist $\frac{\partial x_k}{\partial b_j} = c_{kj}$ und die Formel folgt aus der ersten Formel auf Seite 16.

Falls eine Komponente x_k gleich 0 ist, macht diese Aussage keinen Sinn und man betrachtet den absoluten unvermeidbaren Fehler

$$\Delta^* x_k = \sum_{j=1}^{n} \varepsilon_{b_j} b_j\, c_{kj} \; .$$

Wenn also die Matrix A nahezu singulär ist, muss mit einer starken Empfindlichkeit der Lösungsbestimmung gegenüber kleinen Änderungen der rechten Seite des LGS gerechnet werden.

In solchen Fällen setzt man eine Näherungslösung $\vec{\mathrm{x}}^{(1)}$ in das LGS ein und bestimmt $\vec{\mathrm{d}}^{(1)} := \vec{\mathrm{b}} - \mathrm{A}\vec{\mathrm{x}}^{(1)}$. Löst man nun das LGS $\mathrm{A}\vec{z} = \vec{\mathrm{d}}^{(1)}$ und setzt $\vec{\mathrm{x}}^{(2)} := \vec{\mathrm{x}}^{(1)} + \vec{\mathrm{z}}$, so würde bei exakter Rechnung und Lösungsbestimmung gelten

$$\mathrm{A}\vec{\mathrm{x}}^{(2)} = \mathrm{A}(\vec{\mathrm{x}}^{(1)} + \vec{\mathrm{z}}) = \mathrm{A}\vec{\mathrm{x}}^{(1)} + \mathrm{A}\vec{z} = \vec{\mathrm{b}} - \vec{\mathrm{d}}^{(1)} + \vec{\mathrm{d}}^{(1)} = \vec{\mathrm{b}} \; .$$

Wegen der Rundungsfehler ist allerdings auch $\vec{\mathrm{x}}^{(2)}$ im Allgemeinen nur eine Näherungslösung. Sie ist jedoch üblicherweise genauer als $\vec{\mathrm{x}}^{(1)}$.

Setzt man dieses Verfahren so lange fort, bis $\|\vec{\mathrm{d}}^{(s+1)}\|_1^2 := \max\limits_{1 \le k \le n} (d_k^{(s+1)})^2$ oder $\|\vec{\mathrm{d}}^{(s+1)}\|_2^2 := \sum\limits_{k=1}^{n} (d_k^{(s+1)})^2$ nicht mehr kleiner als der entsprechende Vorgängerwert ist, so nennt man dies das Lösen des LGS durch ***Nachiteration***.

Für dieses Verfahren bietet sich die im Abschnitt 2.1.2 angegebene L-R-Zerlegung oder die Bestimmung von A^{-1} aus 2.1.3 an.

Bevor weitere Iterationsverfahren besprochen werden können, muss die Rolle der ***Eigenwerte*** von Matrizen geklärt werden. Dabei geht es auch um die Nullstellen von ***Polynomen***. Daher werden erst einmal Näherungsverfahren für Nullstellenbestimmungen betrachtet.

2.2 Polynome und ihre Nullstellen

2.2.1 Polynome und Hornerschema

Hat eine Funktion $f : \mathbb{R} \to \mathbb{R}$ eine Vorschrift des Typs

$$f(x) := \sum_{k=0}^{n} a_k\, x^k \quad \text{mit} \quad a_0, \ldots, a_n \in \mathbb{R}; a_n \neq 0\ ,$$

so heißt f *ganzrational* vom Grad n. Den Term in der Vorschrift nennt man ein *Polynom* n-ten Grades über $\mathbb{R}$.

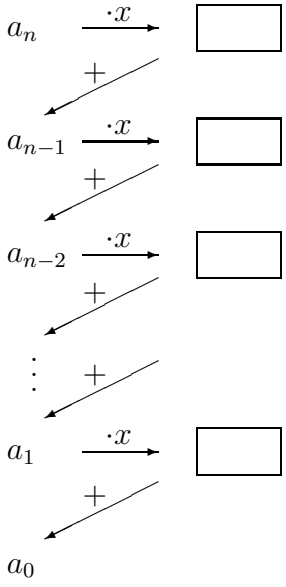
Bei der Berechnung von Funktionswerten berechnet man die Potenzen von x nicht explizit, sondern geht bei gegebenem x so vor:

$$a_n \overset{\cdot x}{\longrightarrow} a_n x \overset{+a_{n-1}}{\longrightarrow} a_n x + a_{n-1} \overset{\cdot x}{\longrightarrow} a_n x^2 + a_{n-1}x \overset{+a_{n-2}}{\longrightarrow} a_n x^2 + a_{n-1}x + a_{n-2} \overset{\cdot x}{\longrightarrow} \ldots$$

$$\ldots \overset{+a_1}{\longrightarrow} a_n x^{n-1} + \ldots + a_2 x + a_1 \overset{\cdot x}{\longrightarrow} a_n x^n + \ldots + a_2 x^2 + a_1 x \overset{+a_0}{\longrightarrow} a_n x^n + \ldots + a_2 x^2 + a_1 x + a_0$$

Diese Methode heißt HORNERschema.

Schema:



Vorteil des Verfahrens:

Man hat nur n Multiplikationen mit x durchzuführen und spart das vorherige Potenzieren. Würde man vorab x^2, x^3, ... , x^n berechnen, so brauchte man dafür $n-1$ Multiplikationen und anschließend noch einmal n Multiplikationen zur Berechnung der Produkte $a_k\, x^k$.

Beispiel:

$f(x) = 1 - 2x + x^2 - 1.5x^3$

$f(2) = (((-1.5) \cdot 2 + 1) \cdot 2 - 2) \cdot 2 + 1 = -11$

Man weiß, dass ein reelles Polynom ungeraden Grades mindestens eine reelle Nullstelle hat. Geht man von den reellen zu den komplexen Zahlen über, so liefert der Fundamentalsatz der Algebra die Aussage, dass ein Polynom n-ten Grades des Typs $\sum_{k=0}^{n-1} a_k x^k + x^n$

mit reellen (oder komplexen) Koeffizienten sich mit geeigneten Zahlen $z_1, z_2, \ldots, z_n \in \mathbb{C}$ in der Form $(x - z_1)(x - z_2) \cdots (x - z_n)$ als Produkt schreiben lässt. Man nennt dieses Produkt die *Linearfaktorzerlegung* von $p(x)$.

Interessiert man sich nur für reelle Nullstellen, so kann man für jedes relle Polynom ein Intervall angeben, **außerhalb** dessen keine Nullstellen liegen:

Satz 2.1
Voraussetzung: *Es sei n eine positive natürliche Zahl. $p(x)$ sei ein Polynom n-ten Grades des Typs* $p(x) := \sum_{k=0}^{n-1} a_k x^k + a_n x^n$ *mit reellen Koeffizienten und* $a_n > 0$.

Behauptung: *Ist n gerade, so hat p außerhalb des Intervalls $[u; v]$ mit*

$$u := \min\{-1, -\frac{1}{a_n} \sum_{k=0}^{n-1} |a_k|\} \quad \text{und} \quad v := \max\{1, \frac{1}{a_n} \sum_{k=0}^{n-1} |a_k|\}$$

nur positive Werte.

Ist n ungerade, so hat p für

$x > \max\{1, \frac{1}{a_n} \sum_{k=0}^{n-1} |a_k|\}$ *nur positive*

und für

$x < \min\{-1, -\frac{1}{a_n} \sum_{k=0}^{n-1} |a_k|\}$ *nur negative Werte.*

Beweis:
Es gilt

$$p(x) = a_n x^{n-1} \Big(x + \sum_{k=0}^{n-1} \frac{a_k}{a_n} \underbrace{x^{(k-n)+1}}_{= \frac{1}{x^{n-k-1}}} \Big) .$$

Für $x > 1$ gilt also:

$$\begin{aligned} p(x) &= a_n x^{n-1} \Big(x - \sum_{k=0}^{n-1} \frac{|a_k|}{a_n} \Big) \\ &> 0 \quad \text{für} \quad x > \sum_{k=0}^{n-1} \frac{|a_k|}{a_n} = \frac{1}{a_n} \sum_{k=0}^{n-1} |a_k| \end{aligned}$$

Damit sind beide Behauptungen des Satzes für positive x gezeigt. Für $x < 0$ hängen die Behauptungen vom Grad n des Polynoms ab.

Für **gerades** n und $x < -1$ gilt:

$$p(x) = \underbrace{a_n x^{n-1}}_{negativ!} \Big(x + \sum_{k=0}^{n-1} \frac{a_k}{a_n} x^{1+k-n} \Big)$$

Es gilt außerdem $x + \sum_{k=0}^{n-1} \frac{a_k}{a_n} x^{1+k-n} < x + \sum_{k=0}^{n-1} \frac{|a_k|}{a_n}$ für $|x| > 1$.

Also gilt $p(x) > 0$ unter der Annahme $x < -1$ und $x < -\sum_{k=0}^{n-1} \frac{|a_k|}{a_n}$.

$p(x)$ ist also bei geradem n für alle x mit $x < \min\{-1, -\frac{1}{a_n} \sum_{k=0}^{n-1} |a_k|\}$ positiv.

Für $x < -1$ und **ungerades** n gilt:

$$\begin{aligned} p(x) &< \underbrace{a_n x^{n-1}}_{positiv!} \Big(x + \sum_{k=0}^{n-1} \frac{a_k}{a_n}\Big) \\ &< 0 \text{ für } x < -\sum_{k=0}^{n-1} \frac{|a_k|}{a_n} = -\frac{1}{a_n} \sum_{k=0}^{n-1} |a_k| \end{aligned}$$

$p(x)$ ist also bei ungeradem n für alle x mit $x < \min\{-1, -\frac{1}{a_n} \sum_{k=0}^{n-1} |a_k|\}$ negativ.

□

Ist ein Polynom $p(x) := \sum_{k=0}^{n} a_k x^k$ über $\mathbb{R}$ mit $a_n > 0$ gegeben, so kann man sich also bei der Suche nach reellen Nullstellen auf das Innere des in Satz 2.1 angegebenen Intervalls $U_0 := [u; v]$ beschränken.

Wie findet man nun alle Nullstellen von p in U_0 (wenn es überhaupt welche gibt!)?

In der Numerik bestimmt man diese Nullstellen **einzeln**, d.h. nach der Bestimmung einer Nullstelle z_s wird nicht das von der Schule her bekannte Verfahren der Polynomdivision (dort wird $p_s(x) := p(x) : (x - z_s)$ bestimmt) zur Abtrennung von Nullstellen angewendet. Stattdessen hält man sich an folgendes Verfahren:

Regel 1: Man „isoliere“ alle Nullstellen, d.h. finde maximal n Teilintervalle $U_i := [u_i; v_i]$ mit $[u_i; v_i] \cap [u_j; v_j] = \emptyset$ für alle i, j mit $i \neq j$, von denen jedes genau eine reelle Nullstelle enthält.

Regel 2: Man wende in jedem der Teilintervalle U_s ein numerische Verfahren an, das die zugehörige Nullstelle z_s mit der gewünschten Genauigkeit liefert.

Zur Isolation von Nullstellen wird oft ein Verfahren herangezogen, bei dem man den größten gemeinsamen Teiler eines Polynoms und seiner ersten Ableitung bestimmt. Ist dieser ggT ein konstantes Polynom, so weiß man, dass alle Nullstellen z_s von $p(x)$ *einfach* sind, d.h. in der komplexen Linearfaktorzerlegung gibt es genau n verschiedene Faktoren $(x - z_1), \ldots, (x - z_n)$.

Die bei der fortlaufenden Division mit Rest auftretenden Polynome werden jeweils mit -1 multipliziert:

Berechnung der STURMschen Kette eines Polynoms $\mathrm{p}(\mathrm{x})$:

① Setze zuerst $f_0(x)$:= $f(x)$, dann $f_1(x)$:= $\frac{\mathrm{d}}{\mathrm{d}x} f_0(x)$. Setze den Polynomzähler n auf 1.

② Wenn f_n nicht das *Nullpolynom* $o(x)$ ist (d.h. es gilt **nicht** $f_n(x) \equiv 0$), so bestimme man das Polynom
$f_{n+1}(x)$:= $(-1)\cdot$ Restpolynom bei Division von f_{n-1} durch f_n.

③ Gilt $f_{n+1} \neq o(x)$ so erhöhe man n um 1 und gehe nach ②.
Ansonsten beende man das Verfahren und notiere die Polynome $f_0, f_1, \ldots, f_n$ in der Bestimmungsreihenfolge.

Beispiel:

$$\begin{aligned} f_0(x) &:= x^3 - 3x^2 + 6x + 4 \\ f_1(x) &:= 3x^2 - 6x + 6 \\ f_2(x) &:= -2x - 2 \\ f_3(x) &:= -15 \end{aligned}$$

Das letzte Polynom in der STURMschen Kette eines Polynoms $p(x)$ ist der ggT von $p(x)$ und $p'(x)$.

Für eine so gebildete Kette definiert man:

$w(u)$:= Anzahl der Vorzeichenwechsel in der Folge $f_0(u), f_1(u), \ldots, f_n(u)$.
(Nullen werden dabei nicht berücksichtigt!)

Dann gilt:

Satz 2.2 (Satz von Sturm)
Voraussetzung: *Es sei f eine Polynom über $\mathbb{R}$ und $f_0, \ldots, f_n$ seine* STURM*sche Kette.*
Behauptung: *In jedem Intervall $[a; b]$ mit $f(a) \cdot f(b) \neq 0$ ist $w(a) - a(b)$ gleich der Anzahl reeller Nullstellen von f in $[a; b]$.*

Wir beweisen diesen Satz nicht und wenden ihn nur auf das obige Beispiel an:

Es sei $f(x)$:= $x^3 - 3x^2 + 6x + 4$. Außerhalb von $[-13; 13]$ kann nach Satz 2.1 keine reelle Nullstelle liegen. Für die Vorzeichen in der STURMschen Kette ergibt sich:

x	f_0	f_1	f_2	f_3	$w(x)$
-13	$-$	$+$	$-$	$-$	2
13	$+$	$+$	$-$	$-$	1

Also liegt in $[-13; 13]$ genau eine reelle Nullstelle. Diese Nullstelle ist *einfach*, da der ggT von f_0 und f_1 das konstante Polynom f_3 ist.

Die Nullstelle von f in $[-13; 13]$ ist nicht rational, da alle rationalen Nullstellen dieses Polynoms nach einem Satz von GAUSS[1] ganzzahlige Teiler von 4 wären. Es gilt jedoch $f(-4) \neq 0, f(-2) \neq 0, f(-1) \neq 0, f(1) \neq 0, f(2) \neq 0$ und $f(4) \neq 0$.

2.2.2 Numerische Nullstellenbestimmung

Wir beschränken uns nicht auf den Fall von Polynomfunktionen, sondern setzen jetzt eine ***stetige*** Funktion $f : \mathbb{R} \to \mathbb{R}$ voraus, deren Werte in einem Intervall $[a; b]$ das Vorzeichen wechseln. Es soll also ein Intervall $[a; b]$ mit $f(a) \cdot f(b) < 0$ existieren.

Dann hat f nach dem ***Zwischenwertsatz*** mindestens eine reelle Nullstelle z_0 in $[a; b]$, d.h. es gibt $z_0 \in [a; b]$ mit $f(z_0) = 0$. Für die numerische Bestimmung von z_0 eigenen sich unter Anderem folgende Verfahren:

Intervallhalbierung:

⓪ Man wähle $[a_0; b_0]$ mit $a_0 := a$ und $b_0 := b$ als ***Startintervall***.
Der Intervallzähler k wird auf 0 gesetzt.

① Man bestimme $m_k := \dfrac{a_k + b_k}{2}$ und berechne $f(m_k)$.
Gilt $f(m_k) = 0$, so beende man das Verfahren und setze $z_0 := m_k$. Ansonsten folgt Schritt ②.

② Gilt $f(a_k) \cdot f(m_k) < 0$, so setze man $a_{k+1} := a_k$ und $b_{k+1} := m_k$, ansonsten (dann gilt $f(m_k) \cdot f(b_k) < 0$!) setze man $a_{k+1} := m_k$ und $b_{k+1} := b_k$. Danach wird der Intervallzähler k um 1 erhöht und man geht zu ①.

Dieses Verfahren liefert entweder die Nullstelle z_0 selbst oder wenigstens eine ***Intervallschachtelung***

$$[a_0; b_0] \supset [a_1; b_1] \supset [a_2; b_2] \supset \dots$$

für z_0, d.h. z_0 liegt in allen diesen Intervallen und die Länge von $[a_n; b_n]$ geht mit $n \to \infty$ gegen 0.

Man weiß, dass für die Länge $|I_k|$ des Intervalls $I_k := [a_k; b_k]$ gilt:

$$|I_0| = b - a\,,\ |I_1| = b_1 - a_1 = \frac{b-a}{2}\,,\ \dots\,,\ |I_k| = \frac{1}{2^k}(b-a)$$

[1] **Satz:** Ist $p(x)$ ein Polynom mit ganzzahligen Koeffizienten der Form

$$p(x) = x^n + a_{n-1}x^{n-1} + \dots + a_1 x + a_0\,,$$

so sind alle rationalen Nullstellen von p ganzzahlig und Teiler des absoluten Gliedes a_0.

Bricht man nach der Bestimmung von $m_n := \dfrac{a_n + b_n}{2}$ ab, so kann man z_0 in der Form

$$z_0 = \underbrace{m_n}_{\text{Mitte von } I_n} \pm d_n \quad \text{mit} \quad d_n := \underbrace{\frac{1}{2}(b_n - a_n)}_{\text{halbe Länge von } I_n}$$

angeben.

Auf einem Rechner mit Gleitpunktarithmetik „konvergiert“ das Verfahren in folgendem Sinn:

Wenn sich die Intervallenden a_n und b_n nur noch so wenig unterscheiden, dass $(1 \oplus \text{eps}) \odot a_n \oslash b_n$ gilt, ergibt sich m_n auf dem Rechner zu a_n. Also sind alle Intervalle ab I_n gleich und man kann die Iteration stoppen.

Beispiel: Die Nullstelle von $f(x) := x^3 - 3x^2 + 6x + 4$ liegt im Intervall $[-1; 0]$, da $f(-1) \cdot f(0) = (-6) \cdot 4 < 0$. Also sei $I_0 := [-1; 0]$ das Startintervall.

Tabelle:

k	a_k	b_k	m_k	$f(m_k) = 0$?	$f(a_k) \cdot f(m_k) < 0$?
0	-1	0	-0.5	nein	ja
1	-1	-0.5	-0.75	nein	nein
2	-0.5	-0.75	-0.625	nein	nein
3	-0.5	-0.625	-0.5625	nein	nein
4	-0.5	-0.5625	-0.53125	nein	nein
5	-0.5	-0.53125	-0.515625	nein	nein
6	-0.5	-0.515625	-0.5078125	nein	ja
...					

Also gilt für die Nullstelle $z_0 = -0.5078125 \pm 0.0078125$ und man würde gerundet angeben $z_0 = -0.508 \pm 0.008$.

Für das nächste Verfahren schicken wir eine Skizze vorweg:

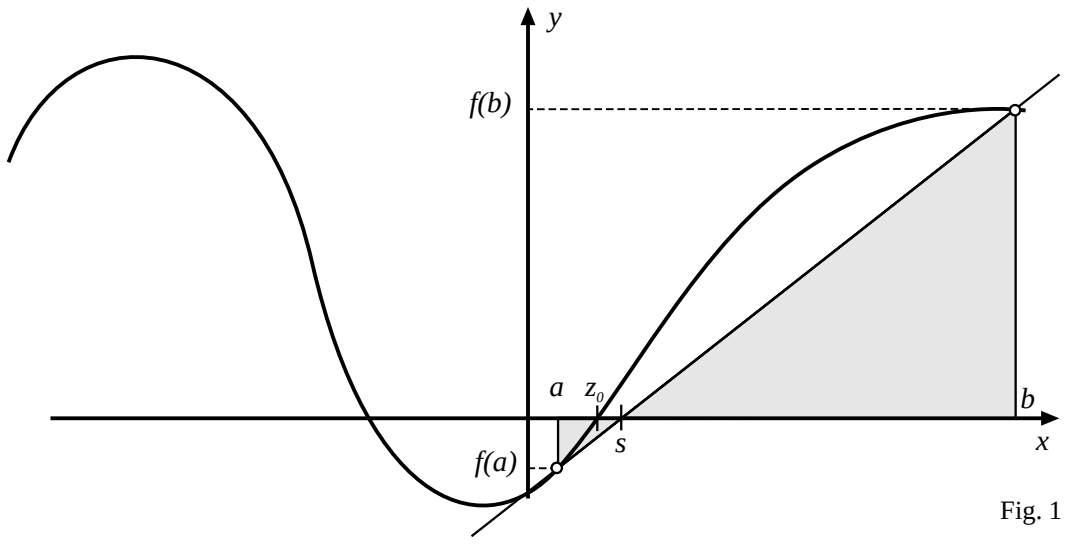


Fig. 1

Ersetzt man bei einer stetigen Funktion f mit Vorzeichenwechsel in $[a;b]$ den Graph durch eine ***Sehne***, die durch die Punkte $(a;f(a))$ und $(b;f(b))$ geht, so kann man den Schnittpunkt s der Sehne als Näherung für eine Nullstelle von f im Intervall $[a;b]$ verwenden. Man nennt dieses Verfahren die ***Sehnenmethode*** oder auch ***Regula falsi***.

In Fig. 1 gilt nach dem zweiten Strahlensatz

$$\frac{|f(b)|}{|f(a)|} = \frac{b-s}{s-a}\,.$$

Daher ergibt sich $s = \dfrac{|f(b)|\cdot a + |f(a)|\cdot b}{|f(b)| + |f(a)|}$ und man hat folgendes Verfahren.

Regula falsi für eine Funktion *f* mit Startintervall $[\mathbf{a};\mathbf{b}]$ und $\mathbf{f(a)\cdot f(b) < 0}$:

⓪ Man wähle $[a_0;b_0]$ mit $a_0{:=}\, a$ und $b_0{:=}\, b$ als Startintervall.
Der Intervallzähler k wird auf 0 gesetzt.

① Man bestimme $s_k{:=}\, \dfrac{|f(b_k)|\cdot a_k + |f(a_k)|\cdot b_k}{|f(b_k)| + |f(a_k)|}$. Gilt $f(s_k)=0$, so beende man das Verfahren und setze $z_0{:=}\, s_k$. Ansonsten folgt Schritt ②.

② Gilt $f(a_k)\cdot f(s_k) < 0$, so setze man $a_{k+1}{:=}\, a_k$ und $b_{k+1}{:=}\, s_k$, ansonsten (dann gilt $f(s_k)\cdot f_{b_k} < 0$!) setze man $a_{k+1}{:=}\, s_k$ und $b_{k+1}{:=}\, b_k$. Danach wird der Intervallzähler k um 1 erhöht und man geht zu ①.

Die Problematik des Verfahrens besteht darin, dass im Fall des Nichtabbrechens zwar eine Intervallfolge $I_0, I_1, I_2, \ldots$ mit $I_0 \supset I_1 \supset I_2 \supset \ldots$ unsd $z_0 \in I_k$ für $k = 1, 2, \ldots$ definiert wird, jedoch im Allgemeinen nur ***eines*** der Intervallenden gegen z_0 konvergiert:

Beispiel: Gegeben sei die Funtionsvorschrift $f(x){:=}\, x^2 - 2$. Dann hat f nur die Nullstelle $z_0 = \sqrt{2}$. Als Startintervall für ihre Bestimmung sei $[1;2]$ gewählt.

Tabelle:

k	a_k	b_k	s_k	$f(s_k)=0$?	$f(a_k)\cdot f(s_k)<0$?
0	1.000000000	2.000000000	1.333333333	nein	nein
1	1.333333333	2.000000000	1.400000000	nein	nein
2	1.400000000	2.000000000	1.411764706	nein	nein
3	1.411764706	2.000000000	1.413793103	nein	nein
4	1.413793103	2.000000000	1.414141414	nein	nein
5	1.414141414	2.000000000	1.414201183	nein	nein
6	1.414201183	2.000000000	...	...	...

Will man z_0 nur auf 4 Nachkommastellen berechnen, so kann man testen, ob z.B. $f(1.4142)\cdot f(1.4143) < 0$ gilt. Da dies der Fall ist, kann man in diesem Fall $z_0 := 1.41425 \pm 0.00005$ angeben.

Wir kommen später noch einmal auf die ***Regula falsi*** zurück und gehen dann auf eine „schnellere“ Variante ein.

Wenn f nicht nur stetig sondern auch überall stetig diffenzierbar ist, kann man ein auf ISAAC NEWTON zurückgehendes Verfahren anwenden. Ist x_k eine Näherung für eine Nullstelle z_0 von f, so wird der Schnittpunkt der Tangente im Punkt $(x_k; f(x_k))$ an den Funktionsgraphen als neue Näherung für z_0 angesehen:

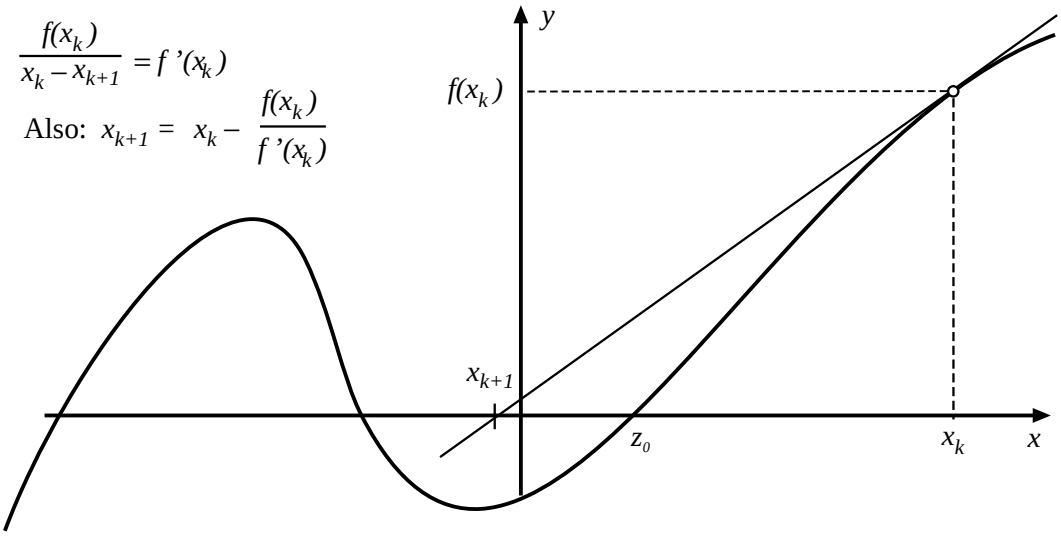


Fig. 1

Iterationsverfahren von Newton zur Nullstellenbestimmung:
Es sei $f : \mathbb{R} \to \mathbb{R}$ eine überall stetig differenzierbare Funktion, die an einer Stelle z_0 den Wert 0 annimmt. In einer Umgebung $\mathcal{U}$ von z_0 sei $f'(x) \neq 0$ für alle $x \in \mathcal{U}$.

⓪ Man wähle einen hinreichend nahe bei z_0 liegenden Startwert x_0 mit $f'(x_0) \neq 0$ und setze den Schrittzähler k auf 0.

① Man bestimme $x_{k+1} := x_k - \dfrac{f(x_k)}{f'(x_k)}$.

② Falls $f(x_k) = 0$ gilt, beende man das Verfahren und setze $z_0 := x_{k+1}$.
Ansonsten wird der Schrittzähler um 1 erhöht und man geht nach ①.

Falls der Startwert x_0 nahe genug bei der interessierenden Nullstelle liegt und $f'(z_0) \neq 0$, konvergiert das Verfahren sehr rasch (im Falle $f'(z_0) = 0$ konervergiert es immer noch, dann aber langsamer!). Mit Hilfe des Satzes von TAYLOR lässt sich zeigen, dass bei jeder zweimal stetig differenzierbaren Funktion in erster Näherung gilt:

Ist $x_k = z_0 + \delta$ mit kleinem δ, so gilt:

$$x_{k+1} \doteq \begin{cases} z_0 + \frac{1}{2}\delta & \text{falls } f'(z_0) = 0, \\ z_0 - \frac{1}{2}\frac{f''(z_0)}{f'(z_0)}\delta^2 & \text{falls } f'(z_0) \neq 0 \end{cases}$$

Den unteren Fall nennt man *quadratische Konvergenz*.

Beispiel: Quadratwurzelberechnung nach dem NEWTON-HERON-Verfahren:

Gegeben ist eine Funktion f mit der Vorschrift $f(x) = x^2 - a$. Wir setzen a als positiv voraus. Wegen $f'(x) = 2x$ ergibt sich mit $x_0 := 1$ als Startwert die Iterationsfolge:

$$\begin{aligned} x_0 &:= 1 \\ x_{k+1} &:= x_k - \frac{x_k^2 - a}{2x_k} = x_k - \frac{1}{2}x_k + \frac{1}{2}\frac{a}{x_k} = \frac{1}{2}(x_k + \frac{a}{x_k}) \quad \text{für } k = 0, 1, 2, \ldots \end{aligned}$$

Hier hat man mit jeder Näherung x_k auch ein Intervall, in dem $\sqrt{a}$ liegt. Gilt nämlich $x_k < \sqrt{a}$, so ist $\frac{a}{x_k} > \sqrt{a}$ (und umgekehrt). Also enthält das Intervall mit den Enden x_k und $\frac{a}{x_k}$ stets $\sqrt{a}$.

Im allgemeinen Fall iteriert man auf dem Rechner so lange, bis sich die Mantisse von x_{n+1} gegenüber der von x_n nur noch in der(den) letzten Stelle(n) ändert. Anschließend kann man nach Wahl eines Faktors $r > 0$ prüfen, wo zwischen

$$(1 - r \cdot \text{eps})x_n, \ x_n, \ (1 + r \cdot \text{eps})x_n$$

eine Nullstelle von f liegt.

Dass eine konvergente Iterationsfolge $x_0, x_1, x_2, \ldots$ des NEWTON-Verfahrens bei einer überall stetig differenzierbaren Funktion f eine Nullstelle von f als Grenzwert besitzt, sieht man so ein:

Es gelte $\lim_{k\to\infty} x_k = z$. Da $x_{k+1} = x_k - \dfrac{f(x_k)}{f'(x_k)}$ für $k = 0, 1, 2, \ldots$ gilt, folgt für den Grenzübergang $k \to \infty$ auf beiden Seiten der Gleichung:

$$\lim_{k\to\infty} x_{k+1} = \lim_{k\to\infty} x_k - \lim_{k\to\infty} \frac{f(x_k)}{f'(x_k)}$$

Aus der Stetigkeit von f und f' folgt also

$$z = z - \frac{f(z)}{f'(z)} \quad (*)$$

und damit $f(z) = 0$.

Sieht man von der speziellen Wahl des Funktionenquotienten in Gleichung $(*)$ ab, so lässt sich das NEWTON-Verfahren als iteratives Lösungsverfahren einer Gleichung des Typs $x = g(x)$ durch den Ansatz $x_{k+1} = g(x_k)$ auffassen (beim NEWTON-Verfahren ist g die Funktion mit der Vorschrift $g(x) := x - \frac{f(x)}{f'(x)}$).

Wir wenden diese Idee kurz auf den mehrdimensionalen Fall an, um ein Lösungsverfahren für LGS herzuleiten:

2.3 Gesamtschrittverfahren zur Lösung von LGS

Bei einem linearen Gleichungssystem
$(*)$ $A\vec{x} = \vec{b}$
mit regulärer $n \times n$-Matrix A kann man bei geeigneter Nummerierung der Variablen und Anordnung der Zeilen annehmen, dass $a_{ii} \neq 0$ für alle $i = 1, \ldots, n$ gilt. Dann lässt es sich durch „Auflösen“ der 1. Gleichung nach x_1, der zweiten nach x_2, ... und der n-ten Gleichung nach x_n in die Form

$$\begin{array}{lllll}
x_1 = & -\frac{a_{12}}{a_{11}}x_2 & -\frac{a_{13}}{a_{11}}x_3 & \ldots\ldots\ldots\ldots & -\frac{a_{1n}}{a_{11}}x_n + \frac{b_1}{a_{11}} \\
x_2 = -\frac{a_{21}}{a_{22}}x_1 & & -\frac{a_{23}}{a_{22}}x_3 & \ldots\ldots\ldots\ldots & -\frac{a_{2n}}{a_{22}}x_n + \frac{b_2}{a_{22}} \\
\vdots & & & & \\
x_n = -\frac{a_{n1}}{a_{nn}}x_1 & -\frac{a_{n2}}{a_{nn}}x_2 & -\frac{a_{n3}}{a_{nn}}x_3 & \ldots -\frac{a_{n-1,n}}{a_{nn}}x_{n-1} & + \frac{b_n}{a_{nn}}
\end{array}$$

bringen. In Matrizenschreibweise lautet dieses System:

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} 0 & -q_{12} & \ldots\ldots\ldots\ldots & -q_{1n} \\ -q_{21} & 0 & -q_{23} \quad \ldots\ldots & -q_{2n} \\ \vdots & & & \vdots \\ -q_{n-1,1} & \ldots. & -q_{n-1,n-2} \quad 0 & -q_{n-1,n} \\ -q_{n1} & \ldots\ldots\ldots\ldots & -q_{n,n-1} \quad 0 & \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} + \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_{n-1} \\ s_n \end{pmatrix}$$

mit $q_{ik} := -\frac{a_{ik}}{a_{ii}}$ für $i = 1, \ldots, n;\ k = 1, \ldots, n; k \neq i$ und $q_{ii} := 0; s_i := \frac{b_i}{a_{ii}}$ für $i = 1, \ldots, n$.

Es liegt nahe, das System $\vec{x} = Q\vec{x} + \vec{s}$ iterativ nach Wahl eines *Startvektors* $\vec{v}$ mit der Iterationsvorschrift

Gesamtschrittverfahren:
Bilde die Folge $\vec{x}^{(0)}, \vec{x}^{(1)}, \ldots$ mit:
$\vec{x}^{(0)} := \vec{u}$
$\vec{x}^{(k+1)} := Q\vec{x}^{(k)} + \vec{s}$ für $k = 0, 1, \ldots$

zu lösen. Dass dieser Ansatz je nach Typ der ursprünglichen Matrix A erfolgreich ist oder scheitert, sollen zwei kleine Beispiele zeigen:

Beispiel: Das LGS $A\vec{x} = \vec{b}$ ist zu lösen.

a) $A = \begin{pmatrix} 1/2 & 1/3 \\ 1/4 & 1/2 \end{pmatrix}$ und $\vec{b} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

b) $A = \begin{pmatrix} 1/3 & 1/2 \\ 1/2 & 1/4 \end{pmatrix}$ und $\vec{b} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

Offensichtlich sind beide Systeme lösbar. Die Lösung von a) ist $\begin{pmatrix} 3 \\ -3/2 \end{pmatrix}$, die von b) ist $\begin{pmatrix} -3/2 \\ 3 \end{pmatrix}$.

In a) erhält man die Matrix $Q = \begin{pmatrix} 0 & -2/3 \\ -1/2 & 0 \end{pmatrix}$ und den Vektor $\vec{s} := \begin{pmatrix} 2 \\ 0 \end{pmatrix}$ für die Iterationsvorschrift. Mit dem Startvektor $\vec{x}^{(0)} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ ergibt sich bei dreistelligem Runden die Vektorfolge

$$\vec{x}^{(0)} = \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ \vec{x}^{(1)} = \begin{pmatrix} 2.00 \\ -0.500 \end{pmatrix},\ \vec{x}^{(2)} = \begin{pmatrix} 2.33 \\ -1.00 \end{pmatrix},\ \vec{x}^{(3)} = \begin{pmatrix} 2.67 \\ -1.17 \end{pmatrix}, \ldots$$

Die Folge konvergiert anscheinend, so ist man z.B. mit $\vec{x}^{(6)} = \begin{pmatrix} 2.93 \\ -1.45 \end{pmatrix}$ schon recht nahe bei der Lösung.

In b) erhält man die Matrix $Q = \begin{pmatrix} 0 & -3/2 \\ -2 & 0 \end{pmatrix}$ und den Vektor $\vec{s} := \begin{pmatrix} 3 \\ 0 \end{pmatrix}$ für die Iterationsvorschrift.

Wenn man hier nicht die Lösung $\begin{pmatrix} -1.5 \\ 3 \end{pmatrix}$ als Startvektor wählt, divergiert die Iterationsfolge sehr schnell, z.B.:

$$\vec{x}^{(0)} = \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ \vec{x}^{(1)} = \begin{pmatrix} 3.00 \\ -2.00 \end{pmatrix},\ \vec{x}^{(2)} = \begin{pmatrix} 6.00 \\ -6.00 \end{pmatrix},\ \vec{x}^{(3)} = \begin{pmatrix} 12.0 \\ -12.0 \end{pmatrix}, \ldots$$

Konvergenzbedingungen für die Iterationsfolge folgen aus einem Satz, den wir erst nach der Einführung von *Matrixnormen* formulieren und beweisen können.
Wir formulieren diese Kriterien für Iterationsmatrizen Q, in denen nicht notwendig wie bisher $q_{ii} = 0$ für $i = 1, \ldots, n$ gelten muss. Das hat den Vorteil, dass sich auch Varianten des Gesamtschrittverfahrens damit beurteilen lassen, bei denen man **nicht** die Auflösung nach den Hauptdiagonalsummanden des LGS anwendet.

Gegeben sei eine $n \times n$-Matrix Q und ein Vektor $\vec{s}$ mit n Koeffizienten.
Dann konvergiert die mit dem Gesamtschrittverfahren gebildete Vektorfolge gegen einen nicht vom Startvektor $\vec{x}^{(0)}$ abhängenden Grenzvektor $\vec{z}$, wenn ***mindestens eines*** der folgenden Kriterien erfüllt ist:

(I) *Starkes Zeilensummenkriterium:*

Das Gesamtschrittverfahren konvergiert, wenn $\|Q\|_\infty := \max_i \sum_{k=1}^{n} |q_{ik}|$ kleiner als 1 ist.

(II) *Starkes Spaltensummenkriterium:*

Das Gesamtschrittverfahren konvergiert, wenn $\|Q\|_1 := \max_k \sum_{i=1}^{n} |q_{ik}|$ kleiner als 1 ist.

(III) *Quadratsummenkriterium:*

Das Gesamtschrittverfahren konvergiert, wenn $\|Q\|_\infty^2 := \sum_{i=1}^{n}\sum_{k=1}^{n} q_{ik}^2$ kleiner als 1 ist.

Diese Kriterien sind ***hinreichend***, d.h. jedes von ihnen garantiert die Konvergenz der Iterationsfolge. Notwendige Kriterien lassen sich nur mit Hilfe der Eigenwerttheorie von Matrizen formulieren.

Bevor wir auf solche Fragen eingehen, soll jedoch gezeigt werden, wie man im Spezialfall eines LGS mit ***symmetrischer*** Koeffizientenmatrix A eine Gesamtschrittiteration ansetzen kann:

Beispiel: Gegeben ist das LGS $(*)$ $\begin{pmatrix} 1 & -1 & 1 \\ -1 & 2 & -2 \\ 1 & -2 & 3 \end{pmatrix} \vec{x} = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}$.

Schreibt man es in die Form $\vec{x} = \vec{x} - \frac{2}{7}\begin{pmatrix} 1 & -1 & 1 \\ -1 & 2 & -2 \\ 1 & -2 & 3 \end{pmatrix}\vec{x} + \frac{2}{7}\begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}$ um,

so resultiert die Gleichung

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 5/7 & 2/7 & -2/7 \\ 2/7 & 3/7 & 4/7 \\ -2/7 & 4/7 & 1/7 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ -2/7 \end{pmatrix}.$$

Obwohl hier Q keines der drei Konvergenzkriterien erfüllt, konvergieren die Iterationsfolgen, z.B. bei Rundung auf 10 Stellen:

$$\vec{x}^{(0)} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \ldots, \vec{x}^{(12)} = \begin{pmatrix} 0.4614\ldots \\ -0.3044\ldots \\ -0.6475\ldots \end{pmatrix}, \ldots, \vec{x}^{(283)} = \begin{pmatrix} 0.000000000 \\ -1.000000000 \\ -1.000000000 \end{pmatrix}$$

Die Regel, nach der wir den Faktor $\frac{2}{7}$ gebildet haben, folgt aus der Eigenwerttheorie symmetrischer Matrizen. Um ihre Wirkung später klären zu können, wenden wir uns daher Matrixnormen und Eigenwerten zu.

2.4 Matrixnormen

2.4.1 Vektorraumnormen

Definition 2.4 *Eine Abbildung* $\| \ \|$ *des Vektorraums* $\mathcal{V} := K^n$ *mit* $K = \mathbb{R}$ *oder* $K = \mathbb{C}$ *nach* $\mathbb{R}$ *heißt genau dann eine* ***Norm****, wenn sie folgenden drei Bedingungen genügt:*

(1) $\|\vec{o}\| = 0$ *und* $\|\vec{x}\| > 0$ *für alle* $\vec{x} \in \mathcal{V}$ *mit* $\vec{x} \neq 0$.

(2) $\|\lambda\vec{x}\| = |\lambda| \cdot \|\vec{x}\|$ *für alle* $\vec{x} \in \mathcal{V}$ *und alle* $\lambda \in K$.

(3) $\|\vec{x} + \vec{y}\| \leq \|\vec{x}\| + \|\vec{y}\|$ *für alle* $\vec{x}, pfeily \in \mathcal{V}$

Man nennt die Eigenschaft (3) die ***Dreiecksungleichung***. Aus (2) folgt, dass $\|\vec{o}\| = 0$ gilt.

Beispiele für Normen im $\mathbb{R}^n$ sind durch folgende Vorschriften gegeben:

$$\begin{aligned} \|\vec{x}\|_1 &:= \sum_{k=1}^{n} |x_k| \\ \|\vec{x}\|_2 &:= \sqrt{\sum_{k=1}^{n} x_k^2} \\ \|\vec{x}\|_\infty &:= \max_{1 \leq k \leq n} |x_k| \end{aligned}$$

Man nennt $\|\vec{x}\|_2$ die ***euklidische*** Norm und $\|\vec{x}\|_\infty$ die ***Maximumnorm*** von $\vec{x}$.

$\|\vec{x}\|_1$ und $\|\vec{x}\|_2$ sind Sonderfälle der durch

$$\|\vec{x}\|_p := (\sum_{k=1}^{n} |x_k|)^{1/p} \quad (p \in \mathbb{R}, p \geq 1)$$

definierten L_p-Normen.

Für $p \to \infty$ erhält man für alle $\vec{x} \in \mathbb{R}^n$: $\lim_{p\to\infty} \|\vec{x}\|_p = \|\vec{x}\|_\infty$.

Wenn man einen Vektor $\vec{x}$ als Zeile $(x_1\, x_2\, \ldots\, x_n)$ schreibt, verwendet man für diese Zeile das Symbol $\vec{x}^{\mathrm{T}}$ lies: „$\vec{x}$ transponiert“. Spiegelt man eine Matrix A an der Hauptdiagonalen, so schreibt man dafür A^{T}.

Offensichtlich lässt sich das Quadrat der euklidischen Norm von $\vec{x}$ in der Form $\|\vec{x}\|^2 = \vec{x}^{\mathrm{T}}\vec{x}$ angeben, wenn man Matrizen mit nur einem Koeffizienten a mit diesem Koeffizienten *identifiziert*.

Man kann diese Normbildung verallgemeinern, indem man eine $n \times n$-Matrix S mit folgender Eigenschaft heranzieht:

Definition 2.5 *Eine $n \times n$-Matrix S mit reellen Koeffizienten s_{ik} heißt genau dann* ***positiv definit****, wenn sie symmetrisch ist und für alle $\vec{x} \in \mathbb{R}^n$ gilt:*

$$\vec{x}^{\mathrm{T}}\, S\, \vec{x} = \sum_{i=1}^{n} s_{ik}\, x_k^2 \;+\; 2 \sum_{\substack{i,\,k=1 \\ i<k}}^{n} s_{ik}\, x_i\, x_k \;>\; 0 \;\; \textit{falls} \;\; \vec{x} \neq \vec{o}\,.$$

Für jede positiv definite Matrix $n \times n$-Matrix S wird durch $\|\vec{x}\|_{\mathrm{S}} := \sqrt{\vec{x}^{\mathrm{T}}\mathrm{S}\vec{x}}$ (für alle $\vec{x} \in \mathbb{R}^n$) eine Norm definiert:

zu (1): Dies gilt offensichtlich wegen der positiven Definitheit von S.

zu (2): $\|\lambda\vec{x}\|_{\mathrm{S}} = \sqrt{\lambda\vec{x}^{\mathrm{T}}\mathrm{S}(\lambda\vec{x})} = \sqrt{\lambda^2\vec{x}^{\mathrm{T}}\mathrm{S}\vec{x}} = |\lambda|\sqrt{\vec{x}^{\mathrm{T}}\mathrm{S}\vec{x}} = |\lambda| \cdot \|\lambda\vec{x}\|_{\mathrm{S}}$

zu (3): Es gilt[2]:

$$\begin{aligned}
\|\vec{x}+\vec{y}\|_{\mathrm{S}} &= \sqrt{(\vec{x}+\vec{y})^{\mathrm{T}}\mathrm{S}(\vec{x}+\vec{y})} \;=\; \sqrt{\vec{x}^{\mathrm{T}}\mathrm{S}\vec{x} + \vec{x}^{\mathrm{T}}\mathrm{S}\vec{y} + \vec{y}^{\mathrm{T}}\mathrm{S}\vec{x} + \vec{y}^{\mathrm{T}}\mathrm{S}\vec{y}} \\
&= \sqrt{\|\vec{x}\|_{\mathrm{S}}^2 + 2\vec{x}^{\mathrm{T}}\mathrm{S}\vec{y} + \|\vec{y}\|_{\mathrm{S}}^2} \;\leq\; \sqrt{\|\vec{x}\|_{\mathrm{S}}^2 + 2|\vec{x}^{\mathrm{T}}\mathrm{S}\vec{y}| + \|\vec{y}\|_{\mathrm{S}}^2} \\
&\leq \sqrt{\|\vec{x}\|_{\mathrm{S}}^2 + 2\|\vec{x}\| \cdot \|\vec{y}\| + \|\vec{y}\|_{\mathrm{S}}^2} \;=\; \|\vec{x}\|_{\mathrm{S}} \;+\; \|\vec{y}\|_{\mathrm{S}}
\end{aligned}$$

Mit Vektornormen lässt sich z.B. der Fehler bei der Lösung eines linearen Gleichungssystems $\mathrm{A}\vec{x} = \vec{b}$ global für alle Koeffizienten einer Näherungslösung $\vec{z}$ angeben. So macht $\|\underbrace{\mathrm{A}\vec{z} - \vec{b}}_{=:\vec{f}}\|$ eine Aussage über den Fehler, mit dem die Gleichungen erfüllt sind.

Die Maximumnorm gibt den größten absoluten Fehlerbetrag für alle Gleichungen an, die anderen Normen definieren „mittlere“ Fehler.

Für Konvergenzbetrachtungen ist folgender Satz wichtig, auf dessen Beweis wir aus Platz- und Zeitgründen verzichten:

Satz 2.3 *Der $\mathbb{R}^n$ und der $\mathbb{C}^n$ sind bezüglich jeder Norm $\|\ \|$* ***vollständig****, d.h. jede* CAUCHY-*Folge $(\vec{x}_k)_{k\in\mathbb{N}}$ bezüglich $\|\ \|$ hat einen Grenzvektor $\vec{g}$ in $\mathbb{R}^n$ bzw. $\mathbb{C}^n$.*

[2] S ist positiv definit und durch $\vec{x} * \vec{y} := \vec{x}^{\mathrm{T}}\mathrm{S}\vec{y}$ wird eine in beiden Argumenten lineare Vorschrift definiert. Außerdem gilt stets $\vec{x}^{\mathrm{T}}\mathrm{S}\vec{y} = \vec{y}^{\mathrm{T}}\mathrm{S}\vec{x}$. Daraus folgt für alle $\vec{x}, \vec{y} \in \mathbb{R}^n$ die CAUCHY-SCHWARZsche Ungleichung: $|\vec{x}^{\mathrm{T}}\mathrm{S}\vec{y}| \leq \|\vec{x}\|_{\mathrm{S}} \cdot \|\vec{y}\|_{\mathrm{S}}$.

Man nennt einen mit einer Norm $\| \ \|$ versehenen Vektorraum $\mathcal{V}$ *normiert*. Ist $\mathcal{V}$ bezüglich dieser Norm auch vollständig, so heißt $\mathcal{V}$ ein BANACH-Raum.

Die Begriffe CAUCHY-Folge und *Grenzvektor* sind dabei wie folgt definiert:

Definition 2.6
a) Eine Folge $(\vec{x}_k)_{k\in\mathbb{N}}$ eines normierten Vektorraums $\mathcal{V}$ heißt genau dann eine CAUCHY*-Folge, wenn bezüglich seiner Norm $\| \ \|$ gilt:*

Zu jedem $\varepsilon \in \mathbb{R}^+$ gibt es $N_\varepsilon \in \mathbb{N}$ so, dass gilt: $\|\vec{x}_s - \vec{x}_t\| < \varepsilon$ für alle $s, t > N_\varepsilon$.

*b) Ein Vektor $\vec{g}$ eines normierten Vektorraums $\mathcal{V}$ heißt genau dann **Grenzvektor** einer Folge $(\vec{x}_k)_{k\in\mathbb{N}}$ von Vektoren aus $\mathcal{V}$, wenn gilt:*

Zu jedem $\varepsilon \in \mathbb{R}^+$ gibt es $N_\varepsilon \in \mathbb{N}$ so, dass gilt: $\|\vec{g} - \vec{x}_k\| < \varepsilon$ für alle $k > N_\varepsilon$.

Da man im $\mathbb{R}^n$ und $\mathbb{C}^n$ für zwei verschiedene Normen $\| \ \|$ und $\| \ \|_*$ stets zwei positive Konstanten c_1, c_2 mit

$$c_1\|\vec{x}\|_* \le \|\vec{x}\| \le c_2\|\vec{x}\|_* \quad \text{für alle } \vec{x} \in \mathbb{R}^n \text{ (bzw.)}\mathbb{C}^n$$

angeben kann, sind alle Normen bei Konvergenzuntersuchungen gleichberechtigt!

Um vom Fehlervektor $\vec{f} := A\vec{z} - \vec{b}$ (er wird der *Defekt* von $\vec{z}$ genannt) auf den Fehler von $\vec{z}$ rückschließen zu können, führen wir mit Vektornormen verträgliche *Matrixnormen* ein:

2.4.2 Normen für Matrizen

Definition 2.7 *Es sei $K = \mathbb{R}$ oder $K = \mathbb{C}$.*
*a) Eine Abbildung der Menge $\mathcal{M}^{(m,n)}$ aller $m \times n$-Matrizen über K heißt eine **Matrixnorm** genau dann, wenn gilt:*

(1) $\|A\| > 0$ *für alle* $A \in \mathcal{M}^{(m,n)}$ *mit* $A \neq \mathcal{O}$ (Nullmatrix).

(2) $\|\lambda A\| = |\lambda| A$ *für alle* $A \in \mathcal{M}^{(m,n)}, \lambda \in K$.

(3) $\|A + B\| \le \|A\| + \|B\|$ *für alle* $A, B \in \mathcal{M}^{(m,n)}$.

*b) Ist $\| \ \|_*$ eine Norm in K^n und $\| \ \|_{**}$ eine Norm in K^m, so heißt eine Matrixnorm $\| \ \|$ in $\mathcal{K}^{(m,n)}$ genau dann mit diesen Normen **verträglich**, wenn $\|A\vec{x}\|_{**} \le \|A\| \cdot \|\vec{x}\|_*$ für alle $A \in \mathcal{M}^{(m,n)}$ und alle $\vec{x} \in K^n$ gilt.*

Im Falle quadratischer reeller $n \times n$-Matrizen mit $\| \ \|_* = \| \ \|_{**}$ hat man folgende Beispiele für verträgliche Normen:

$\| \|_\infty$ und die durch $\|\mathrm{A}\|_\infty := \max\limits_i \sum\limits_{k=1}^{n} |a_{ik}|$ definierte Matrixnorm,

$\| \|_1$ und die durch $\|\mathrm{A}\|_1 := \max\limits_k \sum\limits_{i=1}^{n} |a_{ik}|$ definierte Matrixnorm,

$\| \|_2$ und die durch $\|\mathrm{A}\|_2 := \sqrt{\sum\limits_{i=1}^{n}\sum\limits_{k=1}^{n} |a_{ik}|^2}$ definierte Matrixnorm.

Die Matrixnorm $\|\mathrm{A}\|_\infty$ ist die optimale (=kleinstmögliche) Wahl für den Faktor c in einer Abschätzung des Typs $\|\mathrm{A}\vec{\mathrm{x}}\|_\infty \le c\|\vec{\mathrm{x}}\|_\infty$:

Satz 2.4 *Für die Norm $\|A\|_\infty$ quadratischer $n \times n$-Matrizen A über $K = \mathbb{R}$ oder $K = \mathbb{C}$ gilt*

$$\sup_{\substack{\vec{\mathrm{x}} \in K^n \\ \vec{\mathrm{x}} \neq \vec{o}}} \frac{\|A\vec{\mathrm{x}}\|_\infty}{\|\vec{\mathrm{x}}\|_\infty} = \|A\|_\infty$$

Beweis:
Es sei t der Zeilenindex von A, für den die Summe $\sum\limits_{k=1}^{n} |a_{ik}|$ maximal wird. Dann gilt für jeden Vektor $\vec{\mathrm{x}} \in K^n$ und sein Bild $\vec{\mathrm{y}} := \mathrm{A}\vec{\mathrm{x}}$:

$$\begin{aligned} |y_i| &= |\sum_{k=1}^{n} a_{ik}x_k| \le \sum_{k=1}^{n} |a_{ik}| \cdot |x_k| \\ &\le \max_k |x_k| \sum_{k=1}^{n} |a_{ik}| \le \|\vec{\mathrm{x}}\|_\infty \sum_{k=1}^{n} |a_{ik}| \\ &\le \|\vec{\mathrm{x}}\|_\infty \cdot \|\mathrm{A}\|_\infty \\ &\qquad\qquad \text{für } i = 1, \ldots, n \end{aligned}$$

Daraus folgt $\|\vec{\mathrm{y}}\|_\infty \le \|\vec{\mathrm{x}}\|_\infty \cdot \|\mathrm{A}\|_\infty$ und damit $\dfrac{\|\mathrm{A}\vec{\mathrm{x}}\|_\infty}{\|\vec{\mathrm{x}}\|_\infty} \le \|\mathrm{A}\|_\infty$ für $\vec{\mathrm{x}} \neq \vec{\mathrm{o}}$.

Für den Vektor $\vec{\mathrm{s}}$ mit $s_k := \mathrm{sgn}(a_{ts})$ (bzw. $s_k := \dfrac{\overline{a_{ts}}}{|a_{ts}|}$ im Falle $K = \mathbb{C}$)[3] erhält man wegen $\|\vec{\mathrm{s}}\|_\infty = 1$ in dieser Abschätzung durch Betrachtung von $|y_t|$ das Gleichheitszeichen. Das Supremum des Normquotienten über alle Vektoren ist also gleich der Matrixnorm $\|\mathrm{A}\|_\infty$.

□

[3] Bei einer komplexen Zahl $z = u + \mathrm{i}\,v$ bezeichnet man mit $\overline{z}$ die Zahl $u - \mathrm{i}\,v$ und nennt sie die *konjugiert Komplexe* von z.

Für andere Vektorraumnormen $\| \ \|_*$ des K^n mit $K = \mathbb{R}$ bzw. $K = \mathbb{C}$ kann man durch die Definition

$$\text{lub}(A) := \sup_{\substack{\vec{x} \in K^n \\ \vec{x} \neq \vec{o}}} \frac{\|A\vec{x}\|_*}{\|\vec{x}\|_*}$$

stets eine solche optimale Matrixnorm definieren. Im Falle der Norm $\| \ \|_2$ ist allerdings die Eigenwerttheorie nötig, da bezüglich dieser Vektorraumnorm lub(A) gleich $\sqrt{\lambda_{\max}(A)}$ ist. Dabei ist $\lambda_{\max}(A)$ der größte ***Eigenwert*** der Matrix $A^T A$ (Näheres dazu im nächsten Abschnitt).

Wir setzen jetzt stets die Matrixnorm als optimal gewählt voraus und lassen den Normindex weg. Eine Konsequenz dieser Wahl ist, dass für die Einheitsmatrix I stets $\|I\| = 1$ gilt. Aus der Verträglichkeit der Matrixnorm mit der jeweiligen Vektorraumnorm folgt außerdem, dass $\|A \cdot B\| \leq \|A\| \cdot \|B\|$ für alle Matrizen A und B gilt[4].

Nun zur Anwendung der Theorie auf eindeutig lösbare LGS:

Es sei $A\vec{x} = \vec{b}$ eindeutig lösbar. Die exakte Lösung sei mit $\vec{u}$ bezeichnet. Wir schließen den hier uninteressanten Fall $\vec{b} = \vec{u} = \vec{o}$ aus. Sei $\vec{z}$ eine Näherungslösung mit $\vec{z} = \vec{u} + \vec{d}$ ($\vec{d}$ ist also ein ***Fehlervektor***.)

Dann gilt für den Defekt $\vec{f}$

$$\vec{f} := A\vec{z} - \vec{b} = A(\vec{u} + \vec{d}) - \vec{b} = \underbrace{A\vec{u} - \vec{b}}_{=\vec{o}} + A\vec{d} = A\vec{d} \quad \text{und damit} \quad \vec{d} = A^{-1}\vec{f}\,.$$

Mit verträglichen Normen ergibt sich daraus $\|d\| \leq \|A^{-1}\| \cdot \|f\|$. Da auch $\|b\| \leq \|A\| \cdot \|u\|$ gilt, folgt:

> Fehlerabschätzung mit Hilfe **verträglicher** Normen für ein eindeutig lösbares LGS $A\vec{x} = \vec{b}$ mit quadratischer Matrix A und nichttrivialer Lösung $\vec{u}$:
> Ist $\vec{z}$ eine Näherungslösung und $\vec{f}$ der Defekt $A\vec{z} - \vec{b}$, so gilt für die Norm des Fehlervektors $\vec{d} := \vec{z} - \vec{u}$ die Abschätzung
>
> $$\frac{\|\vec{d}\|}{\|\vec{u}\|} \ \leq \ \|A^{-1}\| \cdot \|A\| \cdot \frac{\|\vec{f}\|}{\|\vec{b}\|}\,.$$

Man nennt den in der Formel auftretenden Faktor $\|A^{-1}\| \cdot \|A\|$ die ***Kondition*** der Matrix A und bezeichnet sie mit cond(A). Wenn sich cond(A) abschätzen lässt, kann man aus dem beobachteten Defekt $\vec{f}$ leicht auf den relativen Fehler der Näherungslösung $\vec{z}$ zurückschließen.

[4] Man nennt diese Eigenschaft die *Submultiplikativität.*

Beispiel: Zu lösen sei $A\vec{x} = \vec{b}$ mit $A = \begin{pmatrix} 1/2 & 1/3 \\ 1/4 & 1/5 \end{pmatrix}$ und $\vec{b} = \begin{pmatrix} 1 \\ 3/5 \end{pmatrix}$.
Angenommen, es wirdmit zweistelliger Gleitpunktarithmetik gerechnet und als Normen sind $\| \|_\infty$ für Vektoren und $\| \|_\infty$ für Matrizen gewählt.
Auf dem Rechner wird das System

$$\begin{pmatrix} 0.5 & 0.33 \\ 0.25 & 0.20 \end{pmatrix} \vec{x} = \begin{pmatrix} 1.0 \\ 0.60 \end{pmatrix}$$

gelöst, bei dem die Matrix A in einem Koeffizienten rundungsfehlerbehaftet ist. Mit der auf dem Rechner bestimmten Näherung $B^{-1} = \begin{pmatrix} 13 & -22 \\ -17 & 33 \end{pmatrix}$ für A^{-1} ergibt sich mit $\vec{z} = \begin{pmatrix} 0.0 \\ 3.0 \end{pmatrix}$ sogar die Lösung des exakten Systems. Auf dem Rechner wird allerdings der Defekt $\vec{f} = \begin{pmatrix} -0.010 \\ 0.0 \end{pmatrix}$ berechnet. Weil cond(A) auf dem Rechner gleich $0.83 \cdot 50 = 42$ ist, würde man schließen, dass für für den Betrag der maximalen Koeffizientenabweichung des Lösungsvektors gilt:

$$\frac{\|\vec{d}\|_\infty}{\|\vec{u}\|_\infty} \leq 42 \frac{0.010}{1.0} = 0.42$$

Eine so unsichere Lösung versucht man durch Nachiteration zu verbessern. Rechnet man dabei jedoch mit B, so ergibt sich mit $\begin{pmatrix} 0.066 \\ 2.8 \end{pmatrix}$ eine Näherung, deren Defekt der Norm nach noch größer als bei der Ausgangsnäherung ist.
Fazit: Bei sehr großen Konditionswerten cond(A) sollte man nicht mit der inversen Matrix rechnen. Dabei ist allerdings zu berücksichtigen, dass solche Konditionswerte mit wachsender Zeilenzahl von A tendenziell wachsen.

Um den Einfluss einer fehlerbehafteten Koeffizientenmatrix auf das Ergebnis eines LGS abschätzen zu können, sehen wir nun von den arithmetischen Effekten ab, die zu einem fehlerbehafteten Lösungsvektor $\vec{z}$ selbst bei exakt dargestellter Koeffizientenmatrix A führen. Wir setzen eine Vektorraumnorm $\| \|$ und als zugehörige Matrixnorm die durch lub(A) definierte Matrixnorm voraus, um den Normindex einzusparen. Unter cond(A) ist daher stets $\text{lub}(A) \cdot \text{lub}(A^{-1})$ zu verstehen.

Gegeben sei ein eindeutig lösbares LGS $A\vec{x} = \vec{b}$ mit $\vec{b} \neq \vec{o}$, dessen Lösung mit $\vec{u}$ bezeichnet sei.

Wird an Stelle dieses Systems das System $\tilde{A}\vec{x} = \vec{b}$ mit einer Näherung $\tilde{A}$ ($\tilde{A}$ unterscheide sich so wenig von A, dass $\tilde{A}^{-1}$ existiert!) für A gelöst, so hat dieses eine Lösung der Form $\vec{u} + \vec{d}$.

Unter der Annahme, dass $\tilde{A} = A \cdot (I + F)$ mit $\|F\| < 1$ gilt, lässt sich folgender Satz anwenden, den wir lediglich zitieren:

Satz 2.5
Voraussetzung: *Es sei F eine $n \times n$-Matrix und I die Einheitsmatrix gleichen Typs. Als Matrixnorm sei eine lub-Norm $\| \ \|$ gewählt und es gelte bezüglich dieser Norm $\|\mathrm{F}\| < 1$.*
Behauptung: *Die inverse Matrix $(\mathrm{I}+\mathrm{F})^{-1}$ zur Matrix $\mathrm{I}+\mathrm{F}$ existiert und es gilt:*

$$\|(\mathrm{I}+\mathrm{F})^{-1}\| \ \leq \ \frac{1}{1-\|\mathrm{F}\|}\,.$$

Das System $\mathrm{A}(\mathrm{I}+\mathrm{F})\vec{x} = \vec{b}$ ist nach den gemachten Annahmen also eindeutig lösbar und hat die Lösung $\vec{u}+\vec{d} = (\mathrm{I}+\mathrm{F})^{-1}\mathrm{A}^{-1}\vec{b} = (\mathrm{I}+\mathrm{F})^{-1}\vec{u}$. Also gilt:

$$\begin{aligned}\vec{d} \ &= \ (\mathrm{I}+\mathrm{F})^{-1}\vec{u} - \vec{u} \ = \ (\,(\mathrm{I}+\mathrm{F})^{-1} \ - \ \mathrm{I}\,)\vec{u} \\ &= \ (\mathrm{I}+\mathrm{F})^{-1}(\mathrm{I}-(\mathrm{I}+\mathrm{F}))\vec{u} \ = \ -(\mathrm{I}+\mathrm{F})^{-1}\mathrm{F}\vec{u}\end{aligned}$$

Daraus ergibt sich mit Satz 2.5 die Abschätzung:

$$\frac{\|\vec{d}\|}{\|\vec{u}\|} \ = \ \frac{\|(\mathrm{I}+\mathrm{F})^{-1}\mathrm{F}\vec{u}\|}{\|\vec{u}\|} \ \leq \ \|(\mathrm{I}+\mathrm{F})^{-1}\| \cdot \|\mathrm{F}\| \ \leq \ \frac{\|\mathrm{F}\|}{1-\|\mathrm{F}\|}\,.$$

Im letzten Beispiel ist $\mathrm{F} = \begin{pmatrix} 0 & -0.04 \\ 0 & 0.05 \end{pmatrix}$ bekannt. Also kann dort auch ohne Kenntnis der Lösung des exakten Systems geschlossen werden, dass für die Abweichung $\vec{d}$ der aus $\tilde{\mathrm{A}}$ bestimmbaren Lösung von der wahren Lösung $\vec{u}$ gilt:

$$\frac{\vec{d}}{\vec{u}} \ \leq \ \frac{0.05}{1-0.05} \approx 0.053\,.$$

Der Einfluss des falschen Matrixkoeffizienten auf die Lösung ist also nicht sehr groß.

Wenn man weder F noch A^{-1} genau kennt und im Rechner wie im Beispiel eine Näherung B^{-1} für A^{-1} sowie eine Näherungslösung $\vec{z}$ berechnet hat, kann man eine von COLLATZ hergeleitete Abschätzung verwenden:

Ist B^{-1} eine Näherung für A^{-1} mit $\|\mathrm{I}-\mathrm{B}^{-1}\mathrm{A}\| < 1$ und $\vec{z}$ eine Näherungslösung des LGS $\mathrm{A}\vec{x} = \vec{b}$ mit dem Defekt $\vec{f} := \mathrm{A}\vec{z} - \vec{b}$, so gilt für die wahre Lösung $\vec{u}$:

$$\|\vec{z}-\vec{u}\| \ \leq \ \frac{\|\mathrm{B}^{-1}\| \cdot \|\vec{f}\|}{1-\|\mathrm{I}-\mathrm{B}^{-1}\mathrm{A}\|} \qquad \text{(Abschätzung von COLLATZ)}$$

Will man den ***relativen Fehler*** abschätzen, so kann man ausnutzen, dass wegen $\vec{z} = \vec{u}+\vec{d}$ gilt:

$$\|\vec{u}\| \ = \ \|\vec{z}-\vec{d}\| \ \geq \ \underbrace{\|\vec{z}\| \ - \ \frac{\|\mathrm{B}^{-1}\| \cdot \|\vec{f}\|}{1-\|\mathrm{I}-\mathrm{B}^{-1}\mathrm{A}\|}}_{\text{bekannt}\,!}$$

Der Vorteil der Abschätzung ist, dass man nicht immer die Matrix A invertieren muss. Wenn z.B. A selbst schon von der Form $A = I + F$ mit $\|F\| < 1$ ist, kann $B^{-1} := I - F$ gewählt werden und man erhält $\|\vec{d}\| \leq \frac{\|I - F\| \cdot \|\vec{f}\|}{1 - \|F\|^2}$.

Beispiel: Es sei

$$\begin{pmatrix} 1.5 & 0.20 & 0.10 \\ 0.10 & 1.2 & -0.10 \\ 0.20 & -0.10 & 1.3 \end{pmatrix} \vec{x} = \begin{pmatrix} 1.4 \\ 0.20 \\ -1.1 \end{pmatrix}$$

mit zweistelliger Gleitpunktarithmetik zu lösen.

Die Matrix erlaubt das *Gesamtschrittverfahren*. Als Iterationsvorschrift ergibt sich:

(i) Sei $\vec{x}^{(0)} := \vec{o}$ der Startvektor.

(ii) $$\vec{x}^{(k+1)} := \begin{pmatrix} 0.0 & -0.13 & -0.067 \\ -0.083 & 0.0 & 0.083 \\ -0.15 & 0.077 & 0.0 \end{pmatrix} \vec{x}^{(k)} + \begin{pmatrix} 0.93 \\ 0.17 \\ -0.85 \end{pmatrix}$$

Schon der Vektor $\vec{x}^{(2)}$ scheint in der Nähe der Lösung $\vec{u}$ zu liegen:
$\vec{x}^{(2)} = \begin{pmatrix} 0.99 \\ 0.010 \\ -1.0 \end{pmatrix}$ mit $\vec{f} = \begin{pmatrix} 0.0 \\ 0.01 \\ 0.0 \end{pmatrix}$.
Also kann man wegen

$$A = I + F \quad \text{mit} \quad F = \begin{pmatrix} 0.50 & 0.20 & 0.10 \\ 0.10 & 0.20 & -0.10 \\ 0.20 & -0.10 & 0.30 \end{pmatrix} \quad (\text{also}: \ \|F\| = 0.8)$$

schließen:

$$I - F = \begin{pmatrix} 0.5 & -0.20 & -0.10 \\ -0.10 & 0.80 & 0.10 \\ -0.20 & 0.10 & 0.70 \end{pmatrix} \quad (\text{also}: \ \|I - F\| = 1.0) \ .$$

Daraus folgt:

$$\max_k \ |x_k^{(2)} - u_k| \ \leq \ \frac{1 \cdot 0.01}{1 - 0.64} = 0.028 \ .$$

$\vec{x}^{(2)}$ liegt allerdings deutlich näher an der wahren Lösung.

2.5 Eigenwerte von Matrizen

2.5.1 Eigenwerte und euklidisches Skalarprodukt

Unter einer *linearen* Abbildung $f : \mathbb{R}^n \to \mathbb{R}^n$ versteht man eine Funktion mit der Eigenschaft

$$f(\mu\vec{x} + \lambda\vec{y}) = \mu\vec{x} + \lambda\vec{y} \quad \text{für alle } \vec{x}, \vec{y} \in \mathbb{R}^n \text{ und alle } \mu, \lambda \in \mathbb{R}\,.$$

Die Abbildungsvorschrift einer solchen Abbildung f ist von der Form $\vec{x} \mapsto A\vec{x}$ mit einer $n \times n$-Matrix A, deren Spalten $\vec{a}_1, \ldots, \vec{a}_n$ die Bilder $f(\vec{e}_1), \ldots, f(\vec{e}_n)$ der Vektoren $\vec{e}_1, \ldots, \vec{e}_n$ der *Standardbasis* sind. Diese Basis ist gegeben durch:

$$\vec{e}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix},\ \vec{e}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix},\ \ldots,\ \vec{e}_{n-1} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \end{pmatrix},\ \vec{e}_n = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

Man interessiert sich nun besonders dafür, ob es Vektoren $\vec{v}$ gibt, die durch f auf ein λ-faches von $\vec{v}$ abgebildet werden. Dazu definiert man:

Definition 2.8 *Eine Zahl λ heißt genau dann **Eigenwert** einer $n \times n$-Matrix A mit reellen Koeffizienten, wenn es einen Vektor $\vec{v} \in \mathbb{R}^n \setminus \{\vec{0}\}$ mit $A\vec{v} = \lambda\vec{v}$ gibt.*
*Man nennt $\vec{v}$ in diesem Fall einen **Eigenvektor** zum Eigenwert λ von A und bezeichnet ihn mit $\vec{v}_\lambda$.*

In der Theorie lassen sich alle Eigenwerte über folgende Überlegung bestimmen:

$$\begin{aligned} A\vec{x} &= \lambda\vec{x} \text{ muss nichttrivial lösbar sein, d.h.} \\ \det(A - \lambda I) &= 0 \text{ muss gelten.} \end{aligned}$$

Diese Determinante ist ein Polynom maximal n-ten Grades und heißt das ***charakteristische*** Polynom p_A der Matrix A. Seine reellen Nullstellen sind die reellen Eigenwerte von A. Die nichtreellen, komplexen Nullstellen von p_A sind ebenfalls Eigenwerte, wenn die Vorschrift $\vec{x} \mapsto A\vec{x}$ als Abbildung des $\mathbb{C}^n$ in den $\mathbb{C}^n$ auffasst. Dann gibt es nämlich zu jedem solchen Eigenwert einen Eigenvektor mit komplexen Koeffizienten.

Alle Eigenvektoren des $\mathbb{C}^n$ zum selben Eigenwert λ einer Matrix A bilden einen Untervektorraum des $\mathbb{C}^n$. Man nennt ihn den zu λ gehörenden *Eigenraum* der Abbildung $f : \vec{z} \mapsto A\vec{z}$.

Eine reelle Matrix muss nicht immer reelle Eigenwerte besitzen, so haben z.B.

$$A := \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \quad \text{und} \quad B := \begin{pmatrix} 1 & 1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -1 & 1 \end{pmatrix}$$

keine rellen Eigenwerte, da A das charakteristische Polynom $p_A = (1-\lambda)^2 + 1$ besitzt und B das charakteristische Polynom $\mathrm{p_B}{:=}\,((1-\lambda)^2+1)^2$ besitzt.

Ein Satz aus der linearen Algebra besagt allerdings, dass eine ***symmetrische*** relle Matrix nur relle Eigenwerte hat. Beim Beweis dieses Satzes fasst man reelle Matrizen als Sonderfälle von Matrizen mit komplexen Koeffizienten auf, betrachtet die Abbildung mit der Vorschrift $\vec{x} \mapsto \mathrm{A}\vec{x}$ vom $\mathbb{C}^n$ in den $\mathbb{C}^n$ und folgert aus der Symmetrie von A, dass alle Eigenwerte reell sind.

Um diesen Satz in der üblichen Form formulieren zu können, schicken wir eine Definition vorweg:

Definition 2.9 *Unter dem* ***(euklidischen) Skalarprodukt*** $\langle\, \vec{\mathrm{x}}, \vec{\mathrm{y}}\,\rangle$ *von Vektoren des* $\mathbb{R}^n$ *versteht man die durch* $\langle\, \vec{\mathrm{x}}, \vec{\mathrm{y}}\,\rangle{:=}\,\vec{\mathrm{x}}^{\mathrm{T}}\vec{\mathrm{y}} = \sum_{\mathrm{k}=1}^{\mathrm{n}} \mathrm{x_k y_k}$ *definierte Zahl.*

Ein Vektor $\vec{\mathrm{x}} \in \mathbb{R}^n$ *und ein Vektor* $\vec{\mathrm{y}} \in \mathbb{R}^n$ *heißen genau dann zueinander* ***orthogonal****, wenn* $\langle\, \vec{\mathrm{x}}, \vec{\mathrm{y}}\,\rangle = 0$ *gilt.*

Damit kann der angesprochene Satz wie folgt formuliert werden:

Satz 2.6 (Eigenvektorbasen)
Voraussetzung: *Es sei* A *eine reelle symmetrische* $n \times n$*-Matrix.*
Behauptung:

(1) Alle Eigenwerte von A *sind reell.*

(2) Eigenvektoren $\vec{\mathrm{v}}_\lambda, \vec{\mathrm{v}}_\mu$ *zu Eigenwerten* $\lambda \neq \mu$ *sind zueinander* ***orthogonal.***

(3) Ist μ *ein mehrfacher Eigenwert mit der Vielfachkeit* s*, so existieren* s *Eigenvektoren* $\vec{\mathrm{v}}_\mu^{(1)}, \ldots, \vec{\mathrm{v}}_\mu^{(s)}$ *mit* $\langle\, \vec{\mathrm{v}}_\mu^{(\mathrm{i})}, \vec{\mathrm{v}}_\mu^{(\mathrm{j})}\,\rangle = 0$ *für* $i \neq j$*, die eine Basis des Eigenraums zu* μ *bilden.*

Aus dem Satz folgt, dass es zu jeder symmetrischen Matrix A eine **Orthonormalbasis** des $\mathbb{R}^{\mathrm{n}}$ gibt, die nur aus Eigenvektoren von A besteht. Dabei sind alle verwendeten Eigenvektoren $\vec{\mathrm{v}}$ so normiert, dass $\|\vec{\mathrm{v}}\|_2 = 1$ gilt.

Dass die Betrachtung komplexer Eigenvektoren auch bei reellen Matrizen nützlich ist, zeigt ein 1931 von dem Russen GERSCHGORIN bewiesener Satz:

Satz 2.7 (Satz von GERSCHGORIN**)** *Ist* A *eine* $n \times n$*-Matrix mit komplexen Koeffizienten* a_{ik}*, so liegen alle Eigenwerte von* A *in folgender Teilmenge* $\mathcal{G}$ *von* $\mathbb{C}$*:*

$$\mathcal{G}{:=}\,\bigcup_{i\,=\,1}^{n} \left\{\, a_{ii} \,+\, z \,\middle|\, z \in \mathbb{C},\ |z| \leq \sum_{\substack{k=1 \\ k\neq i}}^{n} |a_{ik}| \,\right\}$$

Also: Außerhalb der Kreise mit den Mittelpunkten $a_{11}, a_{22}, \ldots, a_{nn}$ und den Radien

$$\sum_{k=2}^{n} |a_{1k}|, \ldots, \sum_{\substack{k=1 \\ k\neq i}}^{n} |a_{ik}|, \ldots, \sum_{k=1}^{n-1} |a_{nk}|$$

liegen keine Eigenwerte. Wenn A nur reelle Koeffizienten hat, liegen alle Kreismittelpunkte auf der reellen Achse.

Beweis:

① Es sei A eine $n \times n$-Matrix mit komplexen Koeffizienten. Ist λ ein Eigenwert von A, so gibt es einen zugehörigen Eigenvektor $\vec{v}$, der so gewählt werden kann, dass mindestens einer seiner Koeffizienten den Wert 1 hat und alle Koeffizienten maximal den Betrag 1 haben. Der Index dieses Koeffizienten sei j, d.h. es gelte $v_j = 1$.

② Aus $A\vec{v} = \lambda\vec{v}$ folgt

$$\lambda = \lambda \cdot v_j = \sum_{k=1}^{n} a_{jk} v_k = a_{jj} + \sum_{\substack{k=1 \\ k\neq j}}^{n} a_{jk} v_k \,.$$

Daraus folgt:

$$|\lambda - a_{jj}| \leq \sum_{\substack{k=1 \\ k\neq j}}^{n} |a_{jk}| \cdot |v_k| \leq \sum_{\substack{k=1 \\ k\neq j}}^{n} |a_{jk}| \,.$$

③ Das heißt aber $\lambda \in \left\{ a_{jj} + z \,\middle|\, z \in \mathbb{C},\ |z| \leq \sum_{\substack{k=1 \\ k\neq j}}^{n} |a_{jk}| \right\}$.

Damit gilt wie behauptet $\lambda \in \mathcal{G}$.

□

Beispiel: Gegeben ist die Matrix $A = \begin{pmatrix} 3 & 0 & 1 \\ 2 & 1 & 0 \\ -1 & 1 & -1 \end{pmatrix}$.

a) Man bestimme mit dem Satz von GERSCHGORIN eine Menge $\mathcal{G}$, in der alle Eigenwerte von A liegen.
b) Man bestimme alle Eigenwerte aus dem charakteristischen Polynom und gebe für jeden der zugehörigen Eigenräume eine Basis an.

Zu a): Es sind folgende Kreisscheiben zu bilden:

$$\begin{aligned} \mathcal{K}_1 &: \text{Mittelpunkt 3, Radius } 0+1=1; \\ \mathcal{K}_2 &: \text{Mittelpunkt 1, Radius } 2+0=2; \\ \mathcal{K}_3 &: \text{Mittelpunkt -1, Radius } 1+1=2\,. \end{aligned}$$

Die Menge $\mathcal{G}$ ist $\mathcal{K}_1 \cup \mathcal{K}_2 \cup \mathcal{K}_3$:

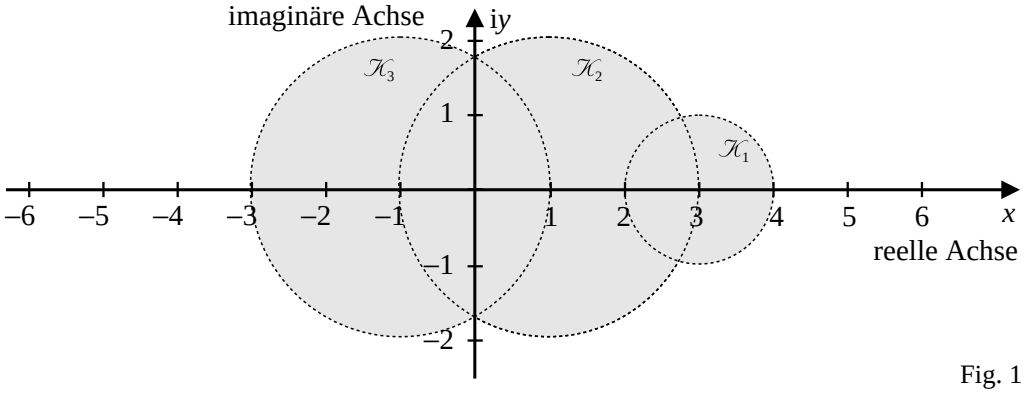


Fig. 1

Zu b): Das charakteristische Polynom ist:

$$p_A = \begin{vmatrix} 3-\lambda & 0 & 1 \\ 2 & 1-\lambda & 0 \\ -1 & 1 & -1-\lambda \end{vmatrix} = \lambda^2 \cdot (3-\lambda)\,.$$

Damit hat A den zweifachen Eigenwert $\lambda_1 = \lambda_2 = 0$ und den einfachen Eigenwert $\lambda_3 = 3$. Dass mindestens ein Eigenwert gleich 0 sein muss, hätte man der Matrix sofort ansehen können: A ist singulär, da die erste Zeile gleich der Differenz aus der zweiten und der dritten Zeile ist. Da bei einer singulären quadratischen Matrix A die Gleichung $A\vec{x} = \vec{o}$ nichttrivial lösbar ist, besitzt A den Eigenwert 0.

Der Eigenraum zum Eigenwert 0 ist die Lösungsmenge des linearen Gleichungssystems $A\vec{x} = \vec{o}$. Das GAUSS-Verfahren liefert:

$$\left(\begin{array}{ccc|c} 3 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 \\ -1 & 1 & -1 & 0 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 3 & 0 & 1 & 0 \\ 0 & 1 & -2/3 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right).$$

Also ist der Eigenraum gleich der Menge

$$\mathcal{E}_0 := \left\{ t \begin{pmatrix} -1/3 \\ 2/3 \\ 1 \end{pmatrix} \;\middle|\; t \in \mathbb{R} \right\}.$$

Obwohl der Eigenwert 0 doppelt ist, enthält hier die Basis des Eigenraums nur einen einzigen Vektor!

Der Eigenraum zum Eigenwert 3 ist die Lösungsmenge des linearen Gleichungssystems $(\mathrm{A} - 3\mathrm{I})\vec{x} = \vec{o}$. Das GAUSS-Verfahren liefert:

$$\left(\begin{array}{ccc|c} 0 & 0 & 1 & 0 \\ 2 & -2 & 0 & 0 \\ -1 & 1 & -4 & 0 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 2 & -2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -1 & 1 & -4 & 0 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 2 & -2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right)$$

Also ist der Eigenraum gleich der Menge

$$\mathcal{E}_3 := \left\{ t \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \;\middle|\; t \in \mathbb{R} \right\}.$$

Die Matrix A besitzt zwar ausschließlich reelle Eigenwerte, ist aber nicht symmetrisch. Daher gibt es hier keine Basis des ganzen $\mathbb{R}^n$ aus Eigenvektoren.

Wenn eine reelle Matrix A (auch) nichtreelle Eigenwerte hat, kann man ausnutzen, dass mit jeder komplexen Nullstelle $z := u + \mathrm{i}\,v$ eines Polynoms p mit rellen Koeffizienten auch deren *Konjugierte* $\overline{z} := u - \mathrm{i}\,v$ eine Nullstelle von p ist. Wir gehen hier nicht auf Näherungsverfahren für diesen Fall ein, da ihre Behandlung zuviel Platz und Zeit kosten würde[5].

Im Folgenden werden daher nur noch symmetrische Matrizen behandelt, da es zu jeder solchen $n \times n$-Matrix eine Eigenvektorbasis des $\mathbb{R}^n$ gibt.

2.5.2 Eigenwertbestimmung bei symmetrischen Matrizen

Bei einer symmetrischen $n \times n$-Matrix A kommen neben direkten Verfahren[6] auch iterative Verfahren in Frage, bei denen man das charakteristische Polynom zunächst noch nicht kennen muss. Ein solches Verfahren geht auf MISES zurück. Vor seiner Anwendung muss man allerdings sicher sein, dass die untersuchte Matrix nur nichtnegative Eigenwerte hat. Dies ist immer durch den Übergang von A zu einer Matrix $\mathrm{A} + \mathrm{a} \cdot \mathrm{I}$ erreichbar, da mit einem Eigenwert λ von A die Matrix $\mathrm{B} := \mathrm{A} + \mathrm{a} \cdot \mathrm{I}$ den Eigenwert $a + \lambda$ hat[7]. Die Matrix B hat dabei dieselben Eigenvektoren zum Eigenwert $a + \lambda$ wie die Matrix A zum Eigenwert λ. Wenn $\lambda_{\min}$ der kleinste Eigenwert von A ist, so gilt nach dem Satz von GERSCHGORIN

$$\lambda_{\min} \geq u := \min_i \left(a_{ii} - \sum_{\substack{k=1 \\ k \neq i}}^{n} |a_{ik}| \right).$$

Die Matrix $\mathrm{B} := \mathrm{A} - u \cdot \mathrm{I}$ besitzt also keine negativen Eigenwerte und mit jedem Eigenwert μ von B ist $\lambda := \mu + u$ ein Eigenwert von A. Auf B ist folgender Satz anwendbar:

[5] Wer daran interessiert ist, findet ein solches Verfahren unter dem Namen GRAEFFE-Verfahren in der Literatur.

[6] Damit sind Verfahren gemeint, in denen zuerst das charakteristische Polynom p_A oder ein Teilerpolynom von p_A bestimmt wird.

[7] Man nennt den Übergang von A zu $\mathrm{A} + \mathrm{a} \cdot \mathrm{I}$ eine *Spektralverschiebung,* da die Menge der Eigenwerte von A das *Spektrum* der Abbildung $f : \vec{x} \mapsto \mathrm{A}\vec{x}$ genannt wird.

Satz 2.8 (Iterationsverfahren von MISES)

Voraussetzung: *Es sei* B *eine symmetrische* $n \times n$*-Matrix mit nichtnegativen Eigenwerten.* $\vec{s} \in \mathbb{R}^n$ *sei ein Vektor, in dessen Darstellung* $\vec{s} = \sum_{k=1}^{n} s_k \vec{e}_k$ *bezüglich einer Eigenvektorbasis von* B *der Koeffizient* s_j *eines Eigenvektors zum größten Eigenwert* $\lambda_{\max}$ *von 0 verschieden ist.*

Behauptung: *Die durch*

$$\begin{aligned} \vec{x}^{(0)} &:= \frac{1}{\|\vec{s}\|}\vec{s} \\ \vec{x}^{(k+1)} &:= \frac{B\vec{x}^{(k)}}{\|B\vec{x}^{(k)}\|} \quad \textit{für } k = 0, 1, \ldots \end{aligned}$$

definierte Vektorfolge konvergiert für $k \to \infty$ *gegen einen Eigenvektor* $\vec{u}$ *von* $\lambda_{\max}$ *und es gilt*

$$\lim_{k\to\infty} \frac{\langle x^{(k)}, B\vec{x}^{(k)} \rangle}{\langle x^{(k)}, x^{(k)} \rangle} = \lambda_{\max}.$$

Anmerkungen: Die in der Behauptung auftretende Norm kann beliebig gewählt werden, wir verwenden im Beweis die Norm $\| \ \|_2$.
Der Quotient $\frac{\langle x^{(k)}, B\vec{x}^{(k)} \rangle}{\langle x^{(k)}, x^{(k)} \rangle}$ wird als RAYLEIGH-Quotient von $\vec{x}^{(k)}$ bezeichnet.

Beweis:

① o.B.d.A. kann angenommen werden, dass $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ für die Eigenwerte von B gilt.

② Sei $\lambda' := \lambda_i$ der erste Eigenwert in dieser Kette, für den $\lambda_i < \lambda_1$ gilt. Falls $\lambda_1 = \lambda_2 = \ldots = \lambda_n$ gilt, setzen wir $\lambda' := 0$ und $i := n + 1$.

③ Sei $q := \frac{\lambda'}{\lambda_1}$. Dann gilt $0 \leq q < 1$.

④ Für $\vec{x}^{(0)}$ gilt $\vec{x}^{(0)} = \frac{1}{\sqrt{\sum_{k=1}^{n} s_k^2}} \sum_{k=1}^{n} s_k \vec{e_k} = .$

⑤ Betrachtet man die Vektorfolge $\vec{y}^{(0)}, \vec{y}^{(1)}, \ldots$ mit

$$\vec{y}^{(0)} := \vec{s}, \ \vec{y}^{(1)} := B\vec{y}^{(0)}, \ \vec{y}^{(2)} := B\vec{y}^{(1)} = B^2\vec{y}^{(0)}, \ \vec{y}^{(3)} = B\vec{y}^{(2)} = B^3\vec{y}^{(0)}, \ \ldots,$$

so gilt offensichtlich $\vec{x}^{(k)} = \frac{1}{\|\vec{y}^{(k)}\|}\vec{y}^{(k)}$ für alle $k \in \mathbb{N}_0$.

⑥ Da die Matrix B^k die Eigenwerte $\lambda_1^k, \ldots, \lambda_n^k$ besitzt, gilt für $k = 1, 2, \ldots$:

$$\vec{\mathrm{y}}^{(k)} = \mathrm{B}^k \vec{\mathrm{y}}^{(0)} = \mathrm{B}^k \Big(\sum_{r=1}^{n} s_r \vec{\mathrm{e}}_r \Big) = \sum_{r=1}^{n} s_r \mathrm{B}^k \vec{\mathrm{e}}_r = \sum_{r=1}^{n} \lambda_r^k s_r \vec{\mathrm{e}}_r = \lambda_1^k \sum_{r=1}^{i-1} s_r \vec{\mathrm{e}}_r \; + \; \sum_{r=i}^{n} \lambda_r^k s_r \vec{\mathrm{e}}_r$$

⑦ Also gilt für $k = 1, 2, \ldots$:

$$\vec{\mathrm{x}}^{(k)} = \sum_{r=1}^{i-1} \frac{\lambda_1^k s_r}{\sqrt{\sum_{r=1}^{i-1} \lambda_1^{2k} s_r^2 \; + \; \sum_{r=i}^{n} \lambda_r^{2k} s_r^2}} \, \vec{\mathrm{e}}_r + \sum_{r=i}^{n} \frac{\lambda_r^k s_r}{\sqrt{\sum_{r=1}^{i-1} \lambda_1^{2k} s_r^2 \; + \; \sum_{r=i}^{n} \lambda_r^{2k} s_r^2}} \, \vec{\mathrm{e}}_r \; .$$

Dividiert man in beiden Summen die Zähler und Nenner der Koeffizienten durch λ_1^k, so erkennt man, dass die erste Summe für $k \to \infty$ gegen den Vektor

$$\vec{\mathrm{u}} := \sum_{r=1}^{i-1} \frac{s_r}{\sqrt{\sum_{t=1}^{i-1} s_t^2}} \vec{\mathrm{e}}_r$$

konvergiert. Dieser Vektor ist ein Eigenvektor zum größten Eigenwert λ_1 von B.

In der zweiten Summe sind alle Koeffizienten dem Betrag nach kleiner als q^k. Daher konvergiert die zweite Summe für $k \to \infty$ gegen den Nullvektor. Also konvergiert $\vec{\mathrm{x}}^{(k)}$ wie behauptet gegen einen Eigenvektor $\vec{\mathrm{u}}$ von λ_1.

⑧ Aus der Konvergenz von $\vec{\mathrm{x}}^{(k)}$ gegen $\vec{\mathrm{u}}$ folgt

$$\lim_{k \to \infty} \frac{\langle \, \vec{\mathrm{x}}^{(\mathrm{k})}, \mathrm{B}\vec{\mathrm{x}}^{(\mathrm{k})} \, \rangle}{\langle \, \vec{\mathrm{x}}^{(\mathrm{k})}, \vec{\mathrm{x}}^{(\mathrm{k})} \, \rangle} = \frac{\langle \, \vec{\mathrm{u}}, \mathrm{B}\vec{\mathrm{u}} \, \rangle}{\langle \, \vec{\mathrm{u}}, \vec{\mathrm{u}} \, \rangle} = \frac{\langle \, \vec{\mathrm{u}}, \lambda_1 \vec{\mathrm{u}} \, \rangle}{\langle \, \vec{\mathrm{u}}, \vec{\mathrm{u}} \, \rangle} = \lambda_1 \frac{\langle \, \vec{\mathrm{u}}, \vec{\mathrm{u}} \, \rangle}{\langle \, \vec{\mathrm{u}}, \vec{\mathrm{u}} \, \rangle} = \lambda_1 \, .$$

Damit ist auch der zweite Teil der Behauptung bewiesen.

□

Bei der Umsetzung dieses Satzes in ein Computerprogramm kann man an Stelle der L_2-Norm bei der Erzeugung der Vektorfolge $(\vec{\mathrm{x}}_k)_{k \in \mathbb{N}_0}$ auch die Maximumnorm verwenden, um Rechenzeit zu sparen. Zur Bestimmung des größten Eigenwertes von A verwendet man dabei die bereits angesprochene Matrix $\mathrm{B} := \mathrm{A} - u \cdot \mathrm{I}$. Den kleinsten Eigenwert $\lambda_{\min}$ von A bestimmt man aus der Matrix $\mathrm{B}^* := v \cdot \mathrm{I} - \mathrm{A}$ mit $v := \max\limits_{j} \Big(a_{jj} + \sum\limits_{\substack{k=1 \\ k \neq j}}^{n} |a_{jk}| \Big)$.

Diese Matrix hat ebenfalls nur nichtnegative Eigenwerte und ihr maximaler Eigenwert ist $v - \lambda_{\min}$. In beiden Fällen wird so lange nach Wahl eines geeigneten Startvektors iteriert, bis sich die Vektoren der Vektorfolge „stationär“ verhalten. Danach wird der zuletzt bestimmte Vektor als normierter Eigenvektor $\vec{\mathrm{e}}_1$ bzw. $\vec{\mathrm{e}}_n$ angesehen und man bestimmt den zugehörigen Eigenwert als RAYLEYGH-Quotient.

Das Verfahren kann allerdings instabil werden, wenn die im Rechner gespeicherte symmetrische Matrix dicht unterhalb des größten Eigenwertes einen weiteren Eigenwert besitzt.

2.5.3 Abspaltung von Eigenwerten bei symmetrischen Matrizen

Nach der Bestimmung des größten Eigenwertes λ_1 und eines zugehörigen, normierten Eigenvektors $\vec{e}_1$ einer symmetrischen $n \times n$-Matrix A mit nichtnegativen Eigenwerten kann man den nächsten Eigenwert λ_2 wie folgt bestimmen:

Wird $\vec{e}_1$ als erster Vektor einer Basis des $\mathbb{R}^n$ gewählt, so gibt es normierte Eigenvektoren $\vec{u}_2, \ldots, \vec{u}_n$ zu den Eigenwerten $\lambda_2, \ldots, \lambda_n$, die zusammen mit $\vec{e}_1$ eine Orthonormalbasis des $\mathbb{R}^n$ bilden. Daher gilt für jeden Vektor $\vec{x} \in \mathbb{R}^n$:

$$\vec{x} \;=\; \langle\, \vec{x}, \vec{e}_1 \,\rangle\, \vec{e}_1 \;+\; \sum_{k=2}^{n} \langle\, \vec{x}, \vec{u}_k \,\rangle\, \vec{u}_k$$

bzw.

$$\vec{x} - \langle\, \vec{x}, \vec{e}_1 \,\rangle\, \vec{e}_1 \;=\; \sum_{k=2}^{n} \langle\, \vec{x}, \vec{u}_k \,\rangle\, \vec{u}_k \,.$$

Daraus folgt durch Anwendung von A:

$$A\vec{x} - \lambda_1 \langle\, \vec{x}, \vec{e}_1 \,\rangle\, \vec{e}_1 \;=\; \sum_{k=2}^{n} \lambda_k \langle\, \vec{x}, \vec{u}_k \,\rangle\, \vec{u}_k \,.$$

Die linke Seite lässt sich deuten als Anwendung der Matrix

$$C := A \;-\; \lambda_1 \begin{pmatrix} e_{11} \cdot e_{11} & e_{11} \cdot e_{12} & \ldots & e_{11} \cdot e_{1n} \\ e_{12} \cdot e_{11} & e_{12} \cdot e_{12} & \ldots & e_{12} \cdot e_{1n} \\ \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots \\ e_{1n} \cdot e_{11} & e_{1n} \cdot e_{12} & \ldots & e_{1n} \cdot e_{1n} \end{pmatrix}$$

auf den Vektor $\vec{x}$. Da $C\vec{e_1} = \vec{o}$ und $C\vec{u}_k = \lambda_k \vec{u}_k$ für $k = 2, \ldots, n$ gilt, hat C die Eigenwerte $0, \lambda_2, \ldots, \lambda_n$. Offensichtlich ist die Matrix C gleich dem Produkt $\vec{e}_1 \cdot \vec{e}_1^{\,T}$. Dieses Produkt liefert im Gegensatz zum euklidischen Skalarprodukt $\vec{e}_1^{\,T} \cdot \vec{e}_1$ eine $n \times n$-Matrix!.

Auf C kann wieder das Verfahren nach MISES angewendet werden, da C symmetrisch ist und keine negativen Eigenwerte besitzt. Nach der Bestimmung von λ_2 und eines normierten Eigenvektors $\vec{e}_2$ aus C bildet man die nächste Differenzmatrix $C_2 := C - \lambda_2 \vec{e}_2 \cdot \vec{e}_2^{\,T}$ und verfährt mit dieser wie mit C. So erhält man eine Folge von Matrizen, die „immer singulärer“ werden, d.h. den Eigenwert 0 mit wachsender Mehrfachheit besitzen. Gleichzeitig ergeben sich alle Eigenwerte und eine Folge von Eigenvektoren, die eine *Orthonormalbasis* des $\mathbb{R}^n$ bilden.

Die wiederholte Anwendung dieser Abspaltung liegt auch der sogenannten *Hauptkomponentenmethode* der Faktorenanalyse in der Psychologie zu Grunde.

Eine weitere Anwendung ist die Extremwertbestimmung bei zweimal stetig partiell nach allen Argumenten differenzierbaren Funktionen $f : \mathbb{R}^n \to \mathbb{R}$. Hier ist eine hinreichende Bedingung für das Vorliegen eines lokalen Maximums oder Minimums an einer Stelle $\vec{x}_0$, dass gilt:

(1) $\frac{\partial f}{\partial x_k}(\vec{x}_0) = 0$ für $k = 0 \ldots, n$

(2) Alle Eigenwerte der Matrix

$$\mathrm{D}^2 f := \begin{pmatrix} \frac{\partial^2 f}{\partial {x_1}^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial {x_2}^2} & & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & & & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_{n-1}} & \frac{\partial^2 f}{\partial {x_n}^2} \end{pmatrix}$$

(die Matrix ist an der Stelle $\vec{x}_0$ zu bilden) sind ***positiv***.

Schließlich kann noch bei einem eindeutig lösbaren LGS $(*)$ $\mathrm{A}\vec{x} = \vec{b}$ ein verallgemeinertes Gesamtschrittverfahren mit der Eigenwerttheorie begründet werden. Dazu multipliziert man beide Seiten mit A^{T} und erhält so ein System $\mathrm{S}\vec{x} = \vec{r}$, bei dem die Matrix S symmetrisch ist und nur positive Eigenwerte besitzt. Kennt man den größten und kleinsten Eigenwert $\lambda_{\max}$ und $\lambda_{\min}$ von S, so liefert die Iterationsvorschrift

$$\vec{x}^{k+1} := (I - \mu S)\vec{x}^k + \mu\vec{r}$$

mit $\mu := \frac{2}{\lambda_{\max}+\lambda_{\min}}$ bei jedem Startvektor eine gegen die Lösung von $(*)$ konvergente Vektorfolge. Kennt man die Eigenwerte nicht, so kann man an Stelle von μ den Faktor $\frac{2}{\|\mathrm{A}\|+\varepsilon}$ mit einem beliebig kleinen $\varepsilon > 0$ und einer beliebigen der üblichen Matrixnormen wählen.

2.5.4 Hinweis auf ein direktes Verfahren zur Eigenwertbestimmung

Will man für beliebige quadratische Matrizen mit reellen Koeffitzienten das charakteristische Polynom bestimmen, so kann man die Methode von KRYLOW anwenden. Bei dieser wird von einem Startvektor $\vec{x}_0$ ausgehend eine Vektorfolge gemäß der Vorschrift $\vec{x}_{k+1} := \mathrm{A}\vec{x}_k$ bis maximal zum Vektor $\vec{x}_n$ gebildet. Dann sind maximal die Vektoren $\vec{x}_0, \ldots, \vec{x}_{n-1}$ linear unabhängig und man erhält in diesem Fall durch Lösen der Vektorgleichung[8]

$$a_0\vec{x}_0 + a_1\vec{x}_1 + \ldots + a_{n-1}\vec{x}_{n-1} = (-1)^{n+1}\vec{x}_n$$

die Koeffizienten des charakteristischen Polynoms.

Falls nur $\vec{x}_0, \ldots, \vec{x}_{k-1}$ mit $k < n$ linear unabhängig sind, erhält man aus der Gleichung

$$b_0\vec{x}_0 + b_1\vec{x}_1 + \ldots + b_{k-1}\vec{x}_{k-1} = \vec{x}_k$$

einen Teiler des charakteristischen Polynoms.

Auf ein so bestimmtes Polynom lassen sich dann die Verfahren zur Nullstellenbestimmung anwenden, um Eigenwerte von A zu bestimmen.

[8] Sie entspricht einem LGS mit n Variablen.

3 Interpolation

3.1 Problemstellung

Bei der *Interpolation* ist eine Funktionenmenge Φ gegeben, bei der alle Funktionen den gleichen Definitionsbereich $\mathcal{D}$ (z.B. $\mathbb{R}$ oder $\mathbb{R}^n$) besitzen und diesen nach $\mathbb{R}$ abbilden.

Dann lautet das *Interpolationsproblem* für eine Funktion $f : \mathcal{D} \to \mathbb{R}$:

Stützstellen: Gegeben seien $n+1$ Stellen $x_0, \ldots, x_n$ aus $\mathcal{D}$.

Interpolationsfunktion: Gesucht ist eine minimale Anzahl von Koeffizienten $\alpha_1, \ldots, \alpha_m$ und eine Teilmenge $\{\varphi_1, \ldots, \varphi_m\}$ von Φ so, dass für $i = 0, \ldots, n$ gilt:

$$\sum_{k=1}^{m} \alpha_k \varphi_k(x_i) = f(x_i)\,.$$

Wir werden uns zwar auf den Fall $\mathcal{D} = \mathbb{R}$ beschränken, weisen jedoch darauf hin, dass in der Praxis auch mehrdimensionale Interpolationsprobleme auftreten.

Die wichtigsten eindimensionalen Fälle lassen sich wie folgt durch Angabe der Funktionsterme für φ_k kennzeichnen:

Stützstellen:
Gegeben sind ein Intervall $[a; b]$ mit $a < b$ mit einer Zerlegung $x_0, \ldots, x_n$ des Typs

$$a = x_0 \ < \ x_1 \ < \ \ldots \ < \ x_n = b$$

und für $k = 0, \ldots, n$ die Werte $y_k := f(x_k)$ einer Funktion $f : [a; b] \to \mathbb{R}$.

a) Polynominterpolation: Sei $\varphi_k(x) := x^k$ für $k = 0, 1, \ldots$.
Gesucht ist $p(x) := \sum_{k=0}^{n} a_k\, x^k$ mit $p(x_j) = y_j$ für $j = 0, \ldots, n$.

b) Trigonometrische Interpolation:
Die Funktionsterme seien $1,\ \cos x,\ \sin x,\ \cos 2x,\ \sin 2x,\ \ldots$. Gesucht ist

$$t(x) := \frac{1}{2}A_0 + \sum_{k=1}^{\frac{n}{2}}(A_k \cos\, kx +, B_k \sin\, kx) \qquad \text{(falls } n \text{ gerade ist)},$$

$$t(x) := \frac{1}{2}A_0 + \sum_{k=1}^{\frac{n-1}{2}}(A_k \cos\, kx +, B_k \sin\, kx) + \frac{1}{2}A_{\frac{n+1}{2}} \cos\, \tfrac{n+1}{2}x$$
(falls n ungerade ist)

mit $t(x_j) = y_j$ für $j = 0 \ldots, n$.

c) Approximation durch kubische Splines:

Sei $\kappa_j(x) := \begin{cases} 1 & \text{für } x_{j-1} \le x \le x_j\,, \\ 0 & \text{sonst}\,. \end{cases}$

und

$$e_j(x) := \kappa_j(x),\ g_j(x) := x \cdot \kappa_j(x),\ q_j(x) := x^2 \cdot \kappa_j(x),\ k_j(x) := x \cdot \kappa_k(x)$$

für $j = 1, \ldots, n$.

Gesucht ist eine *zweimal stetig differenzierbare* Funktion S des Typs

$$S(x) := \sum_{j=1}^{n}(A_j e_j(x) + B_j g_j(x) + C_j q_j(x) + D_j k_j(x))$$

mit $S(x_j) = y_j$ für $j = 0, \ldots, n$, die einer der folgenden Zusatzbedingungen genügt:

(1) $S''(a) = S''(b) = 0$ bzw.
(2) $S'(a) = S'(b)$ und $S''(a) = S''(b)$ (falls $y_0 = y_n$) bzw.
(3) $S'(a) = f'(a)$ und $S'(b) = f'(b)$ (falls $f'(a)$ und $f'(b)$ bekannt)\,.

Der Graph von S besteht aus kubischen Parabelbögen, die an den Stützstellen „glatt“ ineinander übergehen.

Da der Fall b) zu aufwendig ist (in der Literatur findet man ihn unter „Fourierkoeffizientenbestimmung“, „Fast-Fourier-Transform“, ...), sollen hier nur die Fälle a) und c) behandelt werden.

3.2 Polynominterpolation

3.2.1 Die Methode von LAGRANGE

Dass das Interpolationsproblem mit Polynomen lösbar ist, besagt der Satz:

Satz 3.1 *Wenn $n+1$ reelle Werte $y_0, \ldots, y_n \in \mathbb{R}$ und $n+1$ verschiedene reelle Stützstellen $x_0, \ldots, x_n \in \mathbb{R}$ gegeben sind, so gibt es **genau ein** reelles Polynom p maximal n-ten Grades mit $p(x_j) = y_j$ für $j = 0, \ldots, n$.*

Beweis:

① Die Koeffizienten des gesuchten Polynoms p ergeben sich aus dem LGS

$$\begin{aligned} \sum_{k=0}^{n} x_0^k a_k &= y_0 \\ \sum_{k=0}^{n} x_1^k a_k &= y_1 \\ &\vdots \\ \sum_{k=0}^{n} x_n^k a_k &= y_n \end{aligned}$$

mit den Unbekannten $a_0, \ldots, a_n$.

② Die Matrix

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & & & & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{pmatrix}$$

hat die von 0 verschiedene Determinante

$$\prod_{k=1}^{n} \left(\prod_{r=0}^{k-1} (x_k - x_r) \right) \quad (\text{VANDERMONDesche Determinante}),$$

woraus die eindeutige Lösbarkeit des LGS folgt.

□

Nun muss man glücklicherweise nicht ein solches System lösen:

Interpolation nach Lagrange:
Bei dem Interpolationsproblem aus Satz 3.1 bilde man die Polynome

$$p_v(x) := \prod_{\substack{k=0 \\ k \neq v}}^{n} (x - x_k) \Big/ \prod_{\substack{k=0 \\ k \neq v}}^{n} (x_v - x_k)$$

und setze

$$p(x) := \sum_{k=0}^{n} y_k \, p_k(x) \, .$$

Offensichtlich gilt $p(x_j) = y_j$ für $j = 0, \ldots, n$ wegen $p_v(x_j) = \begin{cases} 1 & \text{für } v = j \, , \\ 0 & \text{für } v \neq j \, . \end{cases}$

Beispiel: Gegeben ist eine Tabelle für x_j und y_j:

x	-1	0	1
y	1	0	2

Gesucht ist das Interpolationspolynom maximal zweiten Grades.

$$\begin{aligned} p_0(x) &= \frac{(x-0)(x-1)}{(-1-0)(-1-1)} = \frac{1}{2}(x^2-x) \\ p_1(x) &= \frac{(x+1)(x-1)}{(0+1)(0-1)} = -x^2+1 \\ p_2(x) &= \frac{(x+1)(x-0)}{(1+1)(1-0)} = \frac{1}{2}(x^2+x) \end{aligned}$$

Also erhält man:

$$p(x) = 1\cdot\frac{1}{2}(x^2-x) + 0\cdot(-x^2+1) + 2\cdot\frac{1}{2}(x^2+x) = \frac{3}{2}x^2 + \frac{1}{2}x\,.$$

3.2.2 Die Methode von NEWTON

Das Verfahren von LAGRANGE hat den Nachteil, dass man bei Hinzunahme einer weiteren Stützstelle alle Teilpolynome p_k neu bestimmen muss. Dies wird vermieden durch ein auf NEWTON zurück gehendes Verfahren:

Interpolation nach Newton (dividierte Differenzen):
Bei dem Interpolationsproblem aus Satz 3.1 bilde man die Polynome

$$p_0(x) := 1 \quad \text{und} \quad p_v := \prod_{k=0}^{v-1}(x-x_k) \ \text{ für } v = 1\ldots,n$$

und setze

$$p(x) = y_0 + f_1^{(1)}p_1(x) + f_2^{(2)}p_2(x),+,\ldots\, f_n^{(n)}p_n(x)$$

mit

$$\begin{aligned} f_j^{(1)} &:= \frac{y_j - y_0}{x_j - x_0} && \text{für } j = 1,\ldots,n \\ f_j^{(2)} &:= \frac{f_j^{(1)} - f_1^{(1)}}{x_j - x_1} && \text{für } j = 2,\ldots,n \\ f_j^{(3)} &:= \frac{f_j^{(2)} - f_2^{(2)}}{x_j - x_2} && \text{für } j = 3,\ldots,n \\ &\vdots \\ f_n^{(n)} &:= \frac{f_n^{(n-1)} - f_{n-1}^{(n-1)}}{x_n - x_{n-1}} \end{aligned}$$

Die Idee bei diesem Verfahren ist die rekursive Bestimmung des Interpolationspolynoms. Die Polynome $p_0, p_1, \ldots, p_n$ sind dabei definiert, wie bei der Beschreibung auf der vorherigen Seite:

(0) $q_0(x) := y_0 \cdot p_0(x)$ ist konstant und hat überall (also auch bei x_0) den Wert y_0.

(1) Gesucht ist ein Polynom q_1, das bei x_0 den Wert 0 und bei x_1 den Wert y_1 annimmt. Also setze man $q_1(x) := q_0(x) + c \cdot p_1(x)$. Dann hat q_1 bei x_0 den Wert y_0 und aus der Gleichung $y_1 = q_1(x_1) = q_0(x_1) + c \cdot p_1(x_1)$ für c folgt

$$c = \frac{y_1 - q_0(x_0)}{p_1(x_1)}.$$

(2) Gesucht ist ein Polynom q_2, das bei x_0, x_1 den Wert 0 und bei x_2 den Wert y_2 annimmt. Also setze man $q_2(x) := q_1(x) + c \cdot p_2(x)$. Dann hat q_2 bei x_0, x_1 die vorgeschriebenen Werte und aus der Gleichung $y_2 = q_2(x_2) = q_1(x_2) + c \cdot p_2(x_2)$ für c folgt

$$c = \frac{y_2 - q_1(x_2)}{p_2(x_2)}.$$

$\vdots$

(k+1) Bestimmt seien die Polynome $q_0, q_1, \ldots, q_k$. Gesucht ist ein Polynom q_{k+1}, das bei $x_0, \ldots, x_k$ den Wert 0 und bei x_{k+1} den Wert y_{k+1} annimmt. Also setze man $q_{k+1}(x) := q_k(x) + c \cdot p_{k+1}(x)$. Dann hat q_{k+1} bei $x_0, \ldots, x_k$ die vorgeschriebenen Werte und c ergibt sich aus der Gleichung $y_{k+1} = q_k(x_{k+1}) + c \cdot p_{k+1}(x_{k+1})$ zu

$$c = \frac{y_{k+1} - q_k(x_{k+1})}{q_k(x_{k+1})}$$

$\vdots$

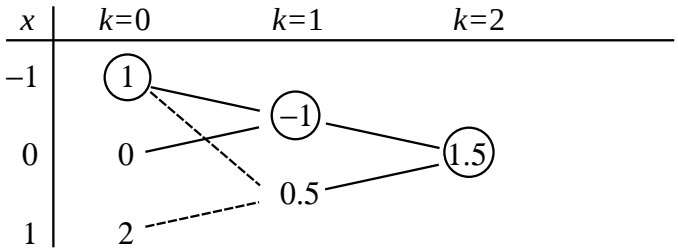
Man kann zeigen, dass c jeweils gleich dem auf der vorherigen Seite definierten Faktor $f_k^{(k)}$ ist.

Beispiel: Wir gehen vom letzten Beispiel aus, bei dem die Tabelle

x	-1	0	1
y	1	0	2

gegeben war.

Für das NEWTONsche Interpolationsverfahren schreiben wir die Tabelle um und tragen in der Spalte unter $k = 0$ die Funktionswerte y_j ein:



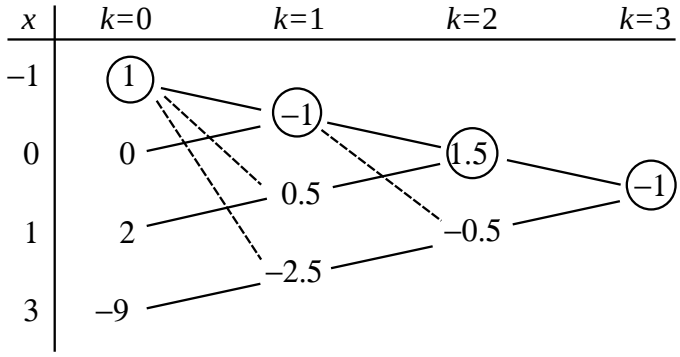
In den Spalten für $k = 1$ und $k = 2$ stehen jeweils die Quotienten $f_j^{(k)}$. Damit ergibt sich mit

$$\begin{aligned} p_0(x) &= 1 \\ p_1(x) &= x + 1 \\ p_2(x) &= x^2 + x \end{aligned}$$

das Interpolationspolynom in der Form

$$q_2(x) \;=\; 1 + (-1) \cdot (x+1) + 1.5 \cdot (x^2 + x) \;=\; 1.5\,x^2 \,+\, 0.5\,x$$

Wird noch ein Punkt hinzugefügt, so bleibt q_2 verwendbar und es kommt nur ein zusätzlicher Summand dazu, z.B. im Falle $x_3 = 3$ und $y_3 = -2$:



Also ist

$$\begin{aligned} q_3(x) &= q_2(x) \,+\, (-1) \cdot (x^2 + x)(x - 1) \\ &= 1.5\,x^2 \,+\, 0.5\,x \,-\, (x^3 - x) \\ &= -x^3 \,+\, 1.5\,x^2 \,+\, 1.5\,x \end{aligned}$$

das neue Interpolationspolynom.

Analysiert man die Bestimmung der Faktoren $f_k^{(k)}$ genauer, so stellt man fest, dass sie sich auch in folgender Form bestimmen lassen:

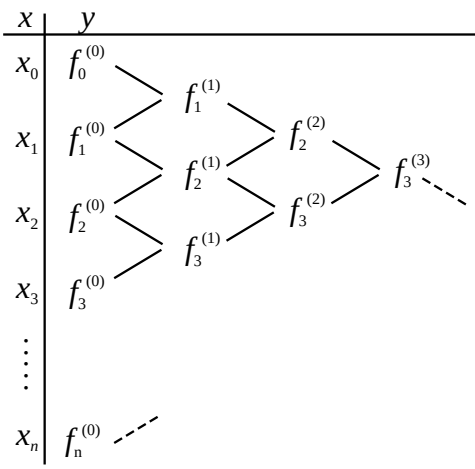
Man setze

$$f_j^{(0)} := y_j \qquad (j = 0, \ldots, n)$$

und berechne für $k = 1, \ldots, n$ jeweils

$$f_j^{(k)} := \frac{f_j^{(k-1)} - f_{j-1}^{(k-1)}}{x_j - x_{j-k}} \qquad (j = k, \ldots, n) \; .$$

Das zugehörige Schema ist dann:



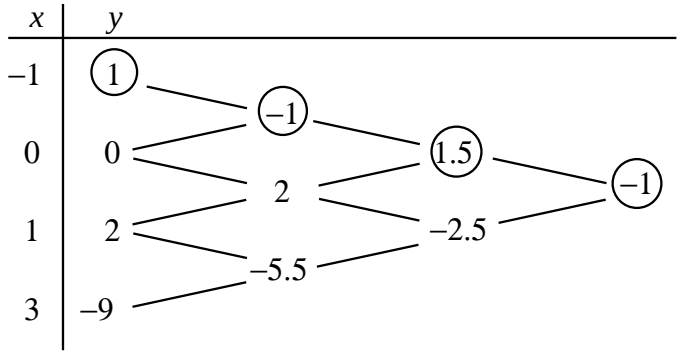
Also:

$$f_j^{(1)} = \frac{f_j^{(0)} - f_{j-1}^{(0)}}{x_j - x_{j-1}} \qquad (j = 1, \ldots, n)$$

$$f_j^{(2)} = \frac{f_j^{(1)} - f_{j-1}^{(1)}}{x_j - x_{j-2}} \qquad (j = 2, \ldots, n)$$

u.s.w.

Im letzten Beispiel würden daher die ***dividierten Differenzen*** wie folgt bestimmt:



Bei der Interpolation von Funktionswerten bringt man das Interpolationspolynom im Allgemeinen ***nicht*** in die explizite Form. Man verwendet stattdessen seine faktorisierte Darstellung und ordnet häufig sogar die Stützstellen je nach Interpolationsstelle um. Es gibt daher in der Literatur eine Vielzahl von „Interpolationsformeln“, die auf einem solchen Vorgehen beruhen. Wir führen diese Formeln nicht auf und beschränken uns auf ein Beispiel.

Beispiel: Zu den in der Tabelle

x	11	13	14	18
y	1342	2210	2758	5850

angegebenen Werten einer Funktion f soll ein polynomial interpolierter Wert an der Stelle 13,5 berechnet werden. Da der Funktionswert zwischen 13 und 14 liegt, wählen wir $x_0 = 13, x_1 = 14, x_2 = 18$ und $x_3 = 11$. Dann ergiebt sich folgende Tabelle mit dividierten Differenzen:

x	y			
13	**2210**			
		548		
14	2758		**45**	
		773		**1**
18	5850		43	
		644		
11	1342			

Also ergibt sich mit

$$q_3(x) \;=\; 2210 + 548(x-13) + 45(x-13)(x-14) + (x-13)(x-14)(x-18)$$

der interpolierte Wert

$$f(13.5) \approx 2210 + 548 \cdot 0.5 - 45 \cdot 0.5 \cdot 0.5 - 0.5 \cdot 0.5 \cdot 4.5 = 2437.875\,.$$

Eine Vereinfachung ergibt sich auch dann, wenn der Abstand zwischen benachbarten Stützstellen konstant ist:

Im Falle *äquidistanter* Stützstellen mit $x_1 = x_0 + h, x_2 = x_0 + 2h, \ldots\ x_n = x_0 + nh$ geht die NEWTONsche Interpolationsformel über in:

$$\begin{aligned} q_n(x) \;=\; y_0 \;+\; \frac{\Delta y_0}{h}(x-x_0) \;+\; \frac{\Delta^2 y_0}{2!\,h^2}(x-x_0)(x-x_1) \\ + \frac{\Delta^3 y_0}{3!h^3}(x-x_0)(x-x_1)(x-x_3) \;+\; \ldots \\ + \frac{\Delta^n y_0}{n!h^n}(x-x_0)(x-x_1)\cdots(x-x_n) \end{aligned}$$

Hier werden in der Differenzentabelle nur die Differenzen $\Delta y_j := y_{j+1} - y_j$ der Funktionswerte und die *iterierten Differenzen* $\Delta^k y_j := \Delta^{k-1} y_{j+1} - \Delta^{k-1} y_j$ in der Tabelle eingetragen.

3.2.3 Fehlerabschätzung für die Polynominterpolation

Angenommen, eine (stetige) Funktion f wird im Intervall $[a;b]$ mit Hilfe der Stützstellen

$$a =: x_0 \ < \ x_1 \ < \ x_2 \ < \ \ldots \ < \ x_n := b$$

durch das Interpolationspolynom q_n angenähert. Wie groß kann dann der Fehler $\delta(x)$:= $f(x) - q_n(x)$ betragsmäßig im Intervall höchstens werden? Offensichtlich gilt $\delta(x_k) = 0$ für $k = 0, \ldots, n$ an den Nullstellen des sogenannten ***Knotenpolynoms*** κ, das in der Form

$$\kappa(x) := \ (x - x_0)(x - x_1) \ldots (x - x_n)$$

definiert ist. Wenn f oft genug differenzierbar ist, kann man den Fehler $|\delta(\overline{x})|$ für alle $\overline{x} \in [a;b]$ nach oben abschätzen:

Satz 3.2 (Fehler bei der Polynominterpolation)

Voraussetzung: *Es sei f eine auf dem Intervall $[a;b]$ stetige und im Inneren des Intervalls mindestens $(n+1)$-mal stetig differenzierbare Funktion von $\mathbb{R}$ nach $\mathbb{R}$. Gegeben sei das Interpolationspolynom p für f mit den Stützstellen*

$$a =: x_0 \ < \ x_1 \ < \ x_2 \ < \ \ldots \ < \ x_n := b\,.$$

Behauptung: *Zu jedem $\overline{x} \in]a;b[$ gibt es $\xi \in]a;b[$ so, dass für die $(n+1)$-te Ableitung $f^{(n+1)}$ von f gilt:*

$$f(\overline{x}) \ - \ p(\overline{x}) \ = \ \frac{f^{(n+1)}(\xi)}{(n+1)!}\,\kappa(\overline{x})$$

Beweis:

① Sei $\overline{x} \in]a;b[$ gegeben. Dann kann man $c \in \mathbb{R}$ so wählen, dass $f(\overline{x}) - p(\overline{x}) = c \cdot \kappa(\overline{x})$ gilt. Dann hat die Funktion d mit

$$d(x) := f(x) - p(x) - c \cdot \kappa(x)$$

mindestens $n+2$ Nullstellen in $]a;b[$.

② Aus der stetigen Differenzierbarkeit von d und dem ersten Mittelwertsatz der Differentialrechnung folgt, dass d' mindestens $n+1$ Nullstellen in $]a;b[$ besitzt.

③ Setzt man diese Argumentation fort, so erhält man die Existenz mindestens einer Nullstelle $\xi \in]a;b[$ von $d^{(n+1)}$. Es gibt also $\xi \in]a;b[$ mit

$$0 = d^{(n+1)}(\xi) = f^{(n+1)}(\xi) - p^{(n+1)}(\xi) - c \cdot \kappa^{(n+1)}(\xi)\,.$$

④ Da die $(n+1)$-te Ableitung von p das Nullpolynom ist und $\kappa^{(n+1)}(x) \equiv (n+1)!$ gilt, folgt daraus $c = \dfrac{1}{(n+1)!}\, f^{(n+1)}(\xi)$.

Also gilt $f(\overline{\mathrm{x}}) - p(\overline{\mathrm{x}}) = \dfrac{1}{(n+1)!}\, f^{(n+1)}(\xi)$ wegen ①.

□

Wenn man also den Interpolationsfehler klein halten will, sollte man die Stützstellen so wählen, dass das Knotenpolynom eine möglichst kleine *Amplitude* im Intervall $]a;b[$ besitzt (d.h. der maximale Betrag von κ soll in diesem Intervall möglichst klein sein).

Eine gute Wahl erhält man durch Ausnutzen von Eigenschaften der TSCHEBYSCHEFF-Polynome:

Unter allen Polynomen $(n+1)$-ten Grades mit Höchstkoeffizient 1 hat das TSCHEBYSCHEFF-Polynom T_{n+1} das kleinste Betragsmaximum im Intervall $[-1;+1]$. Für dieses Polynom gilt $\max\limits_{-1 \le x \le 1} |T_{n+1}(x)| = \dfrac{1}{2^n}$.

Die TSCHEBYSCHEFF-Polynome sind durch

$$\begin{aligned} T_0(x) &:= 1 \\ T_m(x) &:= \frac{\cos(m \arccos x)}{2^{m-1}} \qquad (m = 1, 2, \ldots) \end{aligned}$$

definiert.

Die Nullstellen $x_0^{(n)}, \ldots, x_n^{(n)}$ des Polynoms T_{n+1} lassen sich in der Form

$$x_k^{(n)} = \cos \frac{(2k+1)\pi}{2n+2}, \quad k = 0, 1, \ldots, n$$

bestimmen. Sie liegen an den Enden des Intervalls $[-1;+1]$ dichter zusammen als in der Intervallmitte.

Wir bezeichnen die Nullstellen von T_{n+1} nun der Kürze halber mit $t_0, \ldots, t_n$ und geben an, wie man unter der Vorraussetzung $a < b$ die Stützstellen wählen sollte:

$$\begin{aligned} x_0 &:= a \\ x_k &:= \frac{t_k - t_0}{t_n - t_0}(b-a) + a, \quad k = 1, \ldots, n-1 \\ x_n &:= b \end{aligned}$$

Dann hat man ein „optimales“ Knotenpolynom.

Beispiel: Die Sinusfunktion soll im Intervall $\mathcal{S} := [0; 1.8]$ durch ein Polynom dritten Grades interpoliert werden (Werte und Argumente vierstellig).

a) Bei äquidistanter Unterteilung von $\mathcal{S}$ erhält man die Wertetabelle

x	0.0000	0.6000	1.200	1.800
y	0.000	0.5646	0.9320	0.9738

und es ergibt sich das Interpolationspolynom q mit

$$q(x) := 0.9410 \cdot x - 0.2740 \cdot x(x - 0.6000) - 0.1187x(x - 0.6000)(x - 1.200)\,.$$

Der Fehler $\delta(x) := \sin x - q(x)$ liegt hier zwischen -0.0045 und 0.0025:

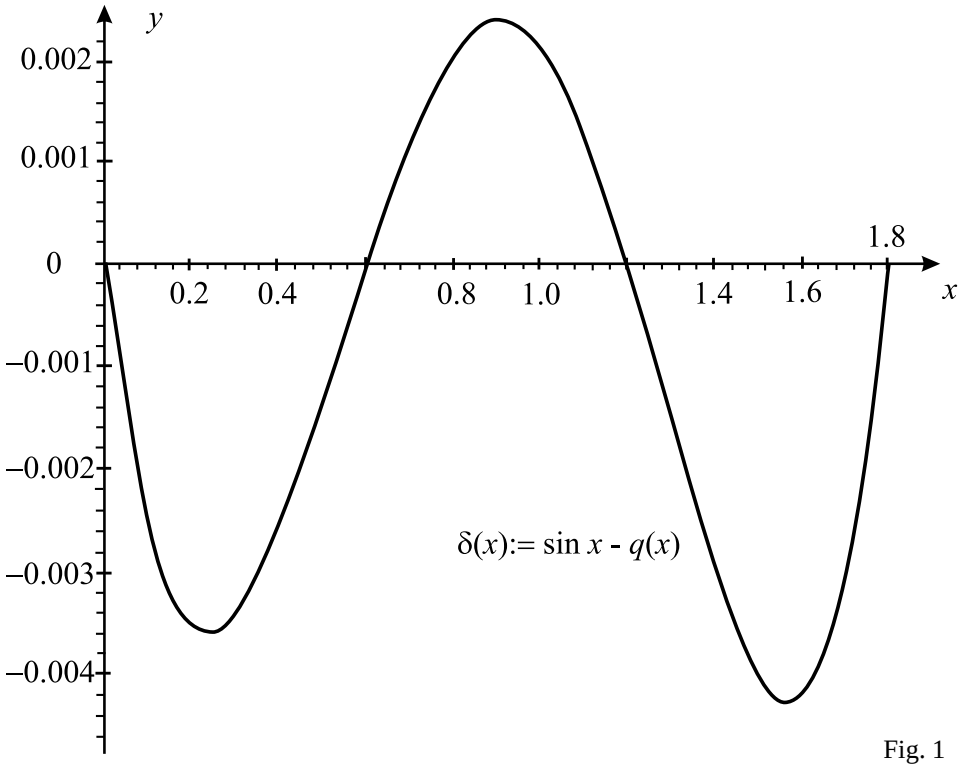


Fig. 1

b) Wird das transformierte TESCHEBYSCHEFF-Polynom vierten Grades als Knotenpolynom gewählt, so sind dessen Nullstellen zu transformieren und es ergeben sich folgende Stützstellen:

Nullstellen von T_3			Stützstellen für $[0; 1]$		
t_0	=	-0.9239	x_0	=	0.0000
t_1	=	-0.3827	x_1	=	0.5272
t_2	=	0.3827	x_2	=	1.273
t_3	=	0.9239	x_3	=	1.800

Daraus folgt die Wertetabelle

x	0.0000	0.5272	1.273	1.800
y	0.000	0.2887	0.9560	0.9738

und es ergibt sich das Interpolationspolynom q mit

$$q(x) := 0.9543 \cdot x - 0.2727 \cdot x(x-0.5272) - 0.09878x(x-0.5272)(x-1.273)\,.$$

Jetzt liegt der Fehler zwischen -0.0038 und 0.0038:

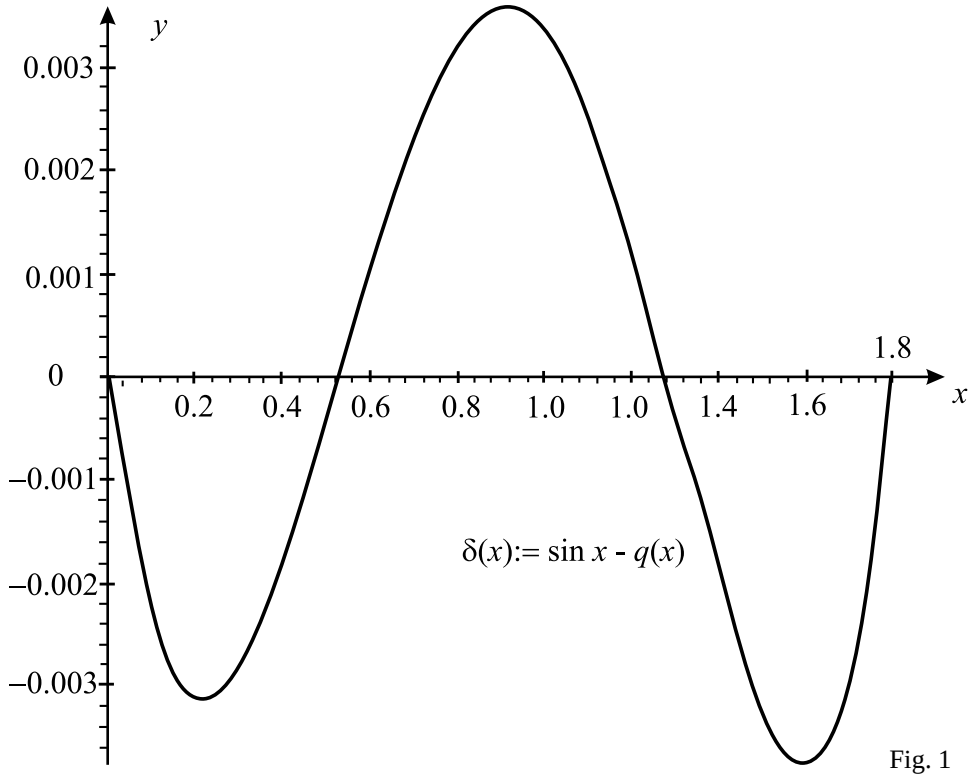


Fig. 1

3.3 Approximation durch kubische Splines

Es sei die Wertetabelle einer Funktion f in der Form

x	x_0	x_1	x_2	$\ldots$	x_n
y	y_0	y_1	y_2	$\cdots$	y_n

gegeben.

Die Regeln für die Splineinterpolation sind dann folgendermaßen zu handhaben:

In jedem der n Teilintervalle $[x_{k-1}; x_k]$ $(k = 1, \ldots, n)$ setzt man ein Polynom S_k des Typs $S_k(x) = y_{k-1} + b_k(x - x_{k-1}) + c_k(x - x_{k-1})^2 + d_k(x - x_{k-1})^3$ an und verlangt:

$$\begin{array}{lrcll} (1) & S_k(x_k) & = & y_k & (k = 1, \ldots, n) \\ (2) & S_k'(x_k) & = & S_{k+1}'(x_k) & (k = 1, \ldots, n-1) \\ (3) & S_k''(x_k) & = & S_{k+1}''(x_k) & (k = 1, \ldots, n-1) \end{array}$$

Danach wird in der Form

$$S(x) := \begin{cases} S_1(x) & \text{falls} \quad x_0 \leq x < x_1 \\ S_2(x) & \text{falls} \quad x_1 \leq x < x_2 \\ \quad \vdots & \\ S_n(x) & \text{falls} \quad x_{n-1} \leq x \leq x_n \end{cases}$$

eine ***Spline-Interpolierende*** S definiert.

Da man insgesamt nur $3n - 2$ Gleichungen für $3n$ Koeffizienten hat hat, muss man zur Sicherstellung der eindeutigen Lösbarkeit des resultierenden LGS zwei weitere Bedingungen verlangen. Wir wiederholen hier die bereits auf Seite 72 angegebenen Empfehlungen:

(4) $S''(a) = S''(b) = 0$ bzw.

(4′) $S'(a) = S'(b)$ und $S''(a) = S''(b)$ (falls $y_0 = y_n$) bzw.

(4″) $S'(a) = f'(a)$ und $S'(b) = f'(b)$ (falls $f'(a)$ und $f'(b)$ bekannt).

Offensichtlich wird in (4′) so etwas wie ein „periodischer“ Ansatz gemacht.

Die Funktion S ist zweimal stetig differenzierbar in $[a; b]$. Setzt man erst einmal das LGS zu (1), (2) und (3) an, so erhält für $k = 1, \ldots, n-1$ die Gleichungen

$$\begin{array}{rcrcrcrcl} y_{k-1} & + & b_k(x_k - x_{k-1}) & + & c_k(x_k - x_{k-1})^2 & + & d_{3k}(x_k - x_{k-1})^3 & = & y_k \\ & & b_k & + & 2c_k(x_k - x_{k-1}) & + & 3d_k(x_k - x_{k-1})^2 & = & b_{k+1} \\ & & & & 2c_k & + & 6d_k(x_k - x_{k-1}) & = & 2c_{k+1}\,, \end{array}$$

die sich mit Hilfe der Abkürzungen $h_k := x_k - x_{k-1}$ und $s_k := \frac{y_k - y_{k-1}}{x_k - x_{k-1}}$ und teilweise Elimination von Variablen in der Form

$$\begin{array}{rcrcrcrcl} b_k & + & c_k h_k & + & d_k h_k^2 & & & = & s_k \\ & & c_k h_k & + & 2d_k h_k^2 & - & b_{k+1} & = & -s_k \\ & & & & d_k h_k^2 & + & b_{k+1} - c_{k+1} h_k & = & s_k \end{array}$$

schreiben lassen. Daraus resultiert ein System der Form

$$\begin{array}{lll}
b_1+c_1h_1+d_1h_1^2 & \dots & = s_1 \\
\quad c_1h_1+2d_1h_1^2-b_2 & \dots & =-s_1 \\
\qquad d_1h_1^2+b_2-c_2h_1 & \dots & = s_1 \\
\qquad\quad b_2+c_2h_2+d_2h_2^2 & \dots & = s_2 \\
\qquad\qquad c_2h_2+2d_2h_2^2-b_3 & \dots & =-s_2 \\
\qquad\qquad\quad d_2h_2^2+b_3-c_3h_2 & \dots & = s_2 \\
\qquad\qquad\qquad \dots & & \\
\qquad\qquad\qquad\quad \dots & & ,
\end{array}$$

bei dem die letzten vier Gleichungen

$$\begin{array}{ll}
b_{n-1}+c_{n-1}h_{n-1}+d_{n-1}h_{n-1}^2 & = s_{n-1} \\
\quad c_{n-1}h_{n-1}+2d_{n-1}h_{n-1}^2-b_n & =-s_{n-1} \\
\qquad d_{n-1}h_{n-1}^2+b_n-c_nh_{n-1} & = s_{n-1} \\
\qquad\quad b_n+c_nh_n+d_nh_n^2 & = s_n
\end{array}$$

lauten. Zu diesen Gleichungen müssen dann noch zwei Gleichungen dazu genommen werden. Sie lauten im Fall

(4): $c_1 = 0$ und $c_n + 3d_nh_n = 0$,

(4′): $b_1 - b_n - 2c_nh_n - 3d_nh_n^2 = 0$ und $c_1 - c_n - 3d_nh_n = 0$,

(4″): $b_1 = f'(a)$ und $b_n + 2c_nh_n + 3d_nh_n^2 = f'(b)$.

Bei großem n ist es von Vorteil, dass die Matrix des LGS bis auf die Störung durch die hinzugenommenen Gleichungen nur in der Hauptdiagonalen und deren beiden oberen Parallelen besetzt ist.

Beispiel: Durch die Punkte $A = (-2; 0)$, $C = (0; 1)$ und $B = (2; 2)$ soll eine Kurve gelegt werden. Für die Splinefunktion S soll gelten $S'(-1) = 1$ und $S'(2) = 0$.

Dies führt auf $y_0 = 0$, $y_1 = 1$, $y_2 = 2$ und das LGS

$$\begin{array}{ll}
b_1+2c_1+4d_1 & = 1/2 \\
\quad 2c_1+8d_1-b_2 & =-1/2 \\
\qquad 4d_1+b_2-2c_2 & = 1/2 \\
\qquad\quad b_2+2c_2+4d_2 & = 1/2 \\
b_1 & = 1 \\
\qquad\quad b_2+4c_2+12d_2 & = 0\,,
\end{array}$$

aus dem $d_2 = -\frac{1}{8}, c_2 = \frac{1}{4}, b_2 = \frac{1}{2}, d_1 = \frac{1}{8}, c_1 = -\frac{1}{2}$ folgt.

Also sind die Terme der beiden kubischen Polynome

$$S_1(x) := (x+2) - \frac{1}{2}(x+2)^2 + \frac{1}{8}(x+2)^3 \quad \text{und} \quad S_2(x) := 1 + \frac{1}{2}x + \frac{1}{4}x^2 - \frac{1}{8}x^3$$

und es ergibt sich folgender Graph:

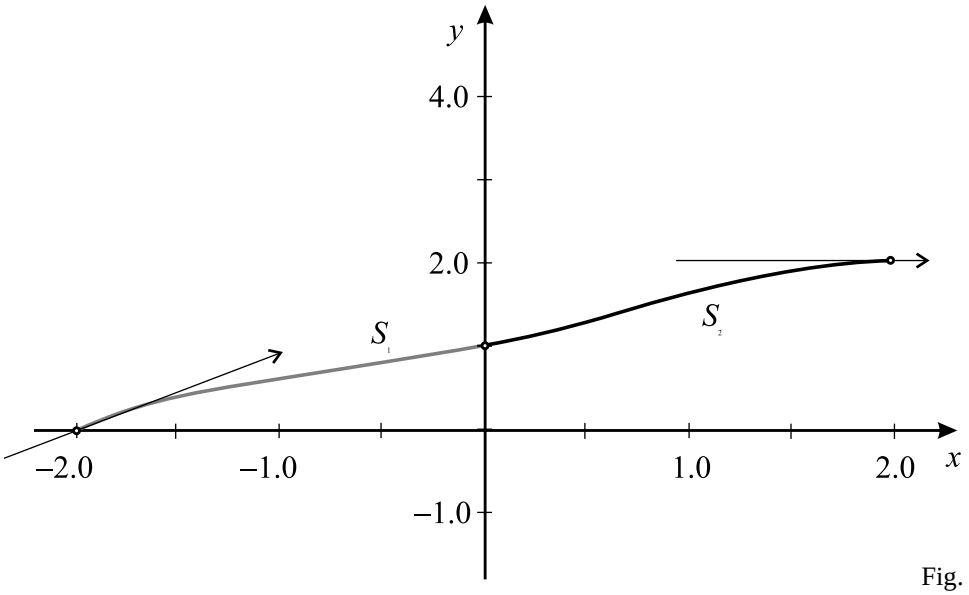


Fig. 1

Literaturverzeichnis

[1] Poloshii, G. N.: Mathematisches Praktikum, Zürich und Frankfurt/Main: Harri Deutsch 1964

[2] Schuppar, B.: Elementare Numerische Mathematik, Braunschweig/Wiesbaden: Vieweg 1999

[3] Stoer, J.: Numerische Mathematik 1, 7. Aufl., Berlin: Springer 1994

[4] Toernig, W.; Spellucci, P.: Numerische Mathematik für Ingenieure und Physiker, Band 1: Numerische Methoden der Algebra, 2. Aufl., Berlin: Springer 1988

[5] Toernig, W.; Spellucci, P.: Numerische Mathematik für Ingenieure und Physiker, Band 2: Numerische Methoden der Analysis, 2. Aufl., Berlin: Springer 1990