

Numerik I

Bernhard Schmitt

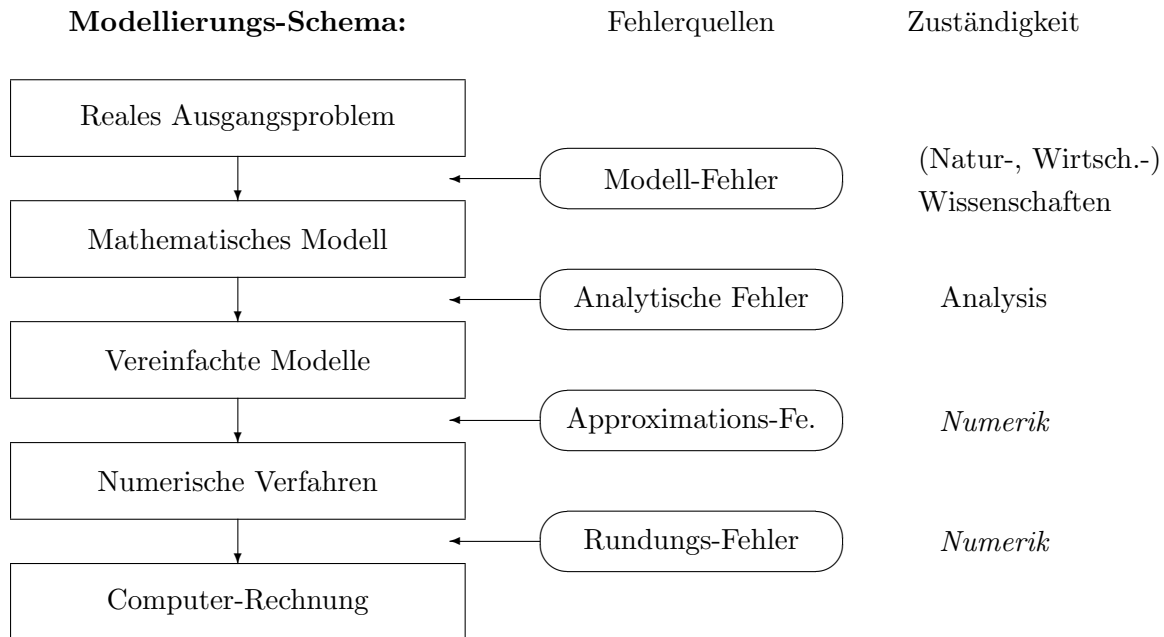
Sommer-Semester 2012

Inhaltsverzeichnis

| | | |
|----------|---|-----------|
| 1 | Einleitung | 1 |
| 2 | Interpolation von Funktionen | 3 |
| 2.1 | Interpolations-Polynome | 3 |
| | Newton-Interpolation | 5 |
| | Interpolationsfehler, optimale Knoten | 9 |
| 2.2 | Spline-Funktionen | 13 |
| | Konvergenz bei Splinefunktionen | 18 |
| 3 | Rechnerarithmetik und Kondition | 23 |
| 3.1 | Zahldarstellung und Rundungsfehler | 23 |
| 3.2 | Konditionszahlen für Algorithmen | 26 |
| 4 | Lineare Gleichungssysteme | 31 |
| 4.1 | Beispiele | 31 |
| 4.2 | Matrizen und Normen | 33 |
| 4.3 | Der Gauß-Algorithmus | 36 |
| | LR-Zerlegung | 38 |
| | Pivotisierung | 39 |

1 Einleitung

Die Numerik befaßt sich mit der "Lösung" analytisch-mathematischer Modelle auf Computern (Wettervorhersage, Fahrzeugentwicklung, Finanz-Modelle). Reale Probleme lassen sich nur ausnahmsweise exakt durch (endliche!) numerische Modelle darstellen. Die meisten Probleme können dagegen nur durch endliche Modelle approximiert werden, wobei die Qualität der Aussagen in der Regel mit der Zahl der Freiheitsgrade (Parameter) des Modells wächst. Realistische Computer-Modelle arbeiten daher meist mit großen Datenmengen. Ein Modellierungs-Prozeß besteht aus mehreren Stufen, in denen jeweils bestimmte Fehler in Kauf genommen werden. Die Numerik entwickelt Verfahren zur Realisierung bzw. Lösung mathematischer Modelle mit der Aufgabe, (nur) in dieser letzten Modellierungs-Stufe auf dem Computer beherrschbare Fehler zu garantieren.



Fehlerquellen, Fehlerbehandlung

Die Genauigkeit von mathematischen Modellen realer Probleme muß von den zuständigen Wissenschaften durch empirische Verfahren überprüft werden. Um "ideale" Modelle mathematisch zugänglich zu machen, sind oft noch weitere analytische Vereinfachungen mit Fehleruntersuchungen erforderlich. Zur numerischen Lösung solcher Modelle sind in doppelter Hinsicht Approximationen durchzuführen, zunächst durch numerische Verfahren, die i.d.R. nicht exakt sind, dann bei der Rechnung auf Computern, da dort reelle Zahlen *nicht exakt darstellbar* sind. Die hier auftretenden Fehler lassen sich zwar immer theoretisch abschätzen, die praktische Umsetzung von **A**bschätzungen ist aber meist nur schwer möglich. Die Beherrschung von Fehlern hat also eine entscheidende Bedeutung in der Numerik, man nutzt dazu verschiedene Zugänge:


- Theoretische Analyse der Fehlerquellen und ihrer Auswirkungen, damit Konstruktion und

Verwendung von *Fehlerschätzungen* in den Verfahren, die dann auch mit Fehlersteuerung kombiniert werden können. Ein Versagen der Verfahren ist aber (hoffentlich nur in Ausnahmefällen) möglich!

- In Teilbereichen (nicht-/ lineare Algebra) sind exakte Fehler *einschließungen* auf Computern durchführbar durch sichere Arithmetiken (Sprache PASCAL-SC): dies liefert exakte Aussagen, ist aber sehr aufwändig und hat nur einen begrenzten Einsatzbereich.

Informationsquellen

Für die Lösung praktischer Probleme durch numerische Verfahren sollte man außer den üblichen Literatur-Quellen auch vorhandene Software heranziehen. Ein Überblick:

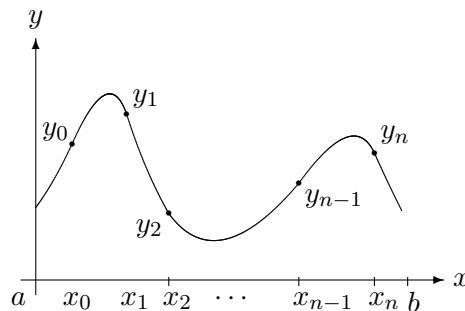
| Form | Art | Namen/Quellen |
|------------------------------------|--|---|
| <i>Literatur klassisch:</i> | Lehrbücher und Zeitschriften | Bibliotheken der Univ., elektronische Zeitschr. |
| <i>Literatur-Besprechungen:</i> | Literatur-Datenbanken mit Kurzreferaten zu mathematischen Veröffentlichungen im "Zentralblatt für Mathematik" und "Mathematical Reviews". | <i>Zentralblatt für Mathematik</i> im WWW über Uni-Netz |
| <i>Software kommerziell:</i> | Software-Bibliotheken (Unterprogramme), fertige Algorithmen für numerische Standardprobleme, Programmier-Oberflächen mit eingebauten numerischen Algorithmen | NAG auf Großrechner MATLAB/ Octave |
| <i>Freie Software im Internet:</i> | Bibliotheken/Spezial-Software für viele Standard-Probleme (BLAS, LINPACK, LAPACK) | elib (elib.zib-berlin.de), Netlib (www.netlib.org) |
| <i>WWW-Lexika</i> | Mitmach-Lexika brauchbar für Schnell-Information, Qualität aber unsicher!! | Wikipedia  |
| <i>Suchmaschinen</i> | Durchsucht nur Web-Oberfläche, (zu) viele Treffer, Qualität unklar | Google, etc. |

Bei konkreten Anwendungsproblemen sind auch für solche Standard-Programme fundierte Numerik-Kenntnisse erforderlich, um Auswahl und Einsatz sinnvoll durchführen zu können.

2 Interpolation von Funktionen

2.1 Interpolations-Polynome

Bei der Beobachtung von sich stetig ändernden Größen (Bewegung, Finanzdaten) können immer nur endlich viele diskrete Werte gemessen werden. Zur Berechnung von Zwischenwerten verwendet man einfache Ersatzfunktionen, die die vorhandenen Daten berücksichtigen. Polynome bieten sich für numerische Rechnungen als Ersatzfunktionen an, da sie nur die Operationen Addition und Multiplikation verwenden. Die Einbeziehung der Messdaten geschieht am einfachsten durch *Interpolation*, wobei die Ersatzfunktion in bestimmten Parameterwerten ("Stützstellen") die Datenwerte annimmt. Später werden auch Splinefunktionen behandelt, welche sich stückweise aus Polynomen zusammensetzen.



Die Aufgabenstellung wird jetzt präzisiert.

Definition 2.1.1 Gegeben sei ein Gitter $\{x_0, \dots, x_n\}$ paarweise verschiedener Stützstellen,

$$a \leq x_0, \dots, x_n \leq b, \quad x_i \neq x_j \text{ für } i \neq j, \quad (2.1.1)$$

und ein Datenvektor $y := (y_0, \dots, y_n)^\top$. Ein Polynom p mit

$$p(x_i) = y_i, \quad i = 0, 1, \dots, n. \quad (2.1.2)$$

heißt (Lagrange-) Interpolationspolynom.

Die Interpolations-Bedingungen (2.1.2) führen im Prinzip auf ein lineares Gleichungssystem für die Koeffizienten a_j eines Polynoms vom Grad n in der Potenz-Darstellung

$$p_n(x) = \sum_{j=0}^n a_j x^j, \quad \Pi_n := \{p_n : a_j \in \mathbb{R}, j = 0, \dots, n\}. \quad (2.1.3)$$

Das Lagrange-Interpolationspolynom kann aber direkt konstruiert werden:

Satz 2.1.2 Zu einem Gitter (2.1.1) und beliebigen Werten y_i , $i = 0, \dots, n$, existiert genau ein Interpolationspolynom vom Grad n , das die Forderungen (2.1.2) erfüllt. Es hat die Form

$$p := p_y := \sum_{i=0}^n y_i L_i, \quad L_i(x) := \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x - x_j}{x_i - x_j}. \quad (2.1.4)$$

Die Abbildung $\mathbb{R}^{n+1} \rightarrow \Pi_n$, $y \mapsto p_y$ ist linear.

Die Polynome L_i bilden nach Konstruktion eine Kardinal-Basis,

$$L_i \in \Pi_n, \quad L_i(x_k) = \delta_{ik} = \begin{cases} 1 & \text{für } i = k, \\ 0 & \text{für } i \neq k \end{cases},$$

in den Stützstellen x_j verschwindet daher genau ein Summand von p_y nicht.

Beweis zu Satz 2.1.2 Wegen der Kardinalität der L_i gilt für p aus (2.1.4)

$$p \in \Pi_n, \quad p(x_k) = \sum_{i=0}^n y_i L_i(x_k) = y_k, \quad k = 0, \dots, n.$$

Zur Eindeutigkeit: wenn zwei Polynome $p, \bar{p} \in \Pi_n$ beide (2.1.2) erfüllen, so folgt

$$(p - \bar{p})(x_i) = 0, \quad i = 0, \dots, n.$$

Das Differenzpolynom vom Grad n hat also $n+1$ Nullstellen und nach dem Fundamentalsatz der Algebra sind p und \bar{p} identisch. ■

Den Rechenaufwand bei der Interpolation gibt man abhängig vom Polynomgrad n an. Wenn die Differenzen $x - x_j$ getrennt berechnet werden, sind zur Auswertung von (2.1.4) jeweils $n^2 + 2n$ Additionen, $n(n+1)$ Multiplikationen und $n(n+1)$ Divisionen nötig. Daher ist der

Rechenaufwand für eine Auswertung von (2.1.4): $3n^2 + \mathcal{O}(n)$ Operationen.

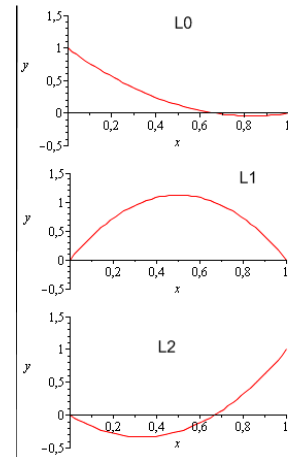
Dabei gibt man üblicherweise nur den am schnellsten wachsenden Term $3n^2$ an. Für theoretische Überlegungen ist die explizite Darstellung (2.1.4) mit dem Kardinalsystem L_i sehr bequem und ein Ansatz für andere Verfahren etwa bei der numerischen Integration (Quadratur). Für die praktische Beurteilung muß man den genauen Einsatzbereich präzisieren. Ist nur ein einziger Datensatz mit $y_i \in \mathbb{R}$, $i = 0, \dots, n$, zu interpolieren, ist die Darstellung (2.1.4) zu teuer und wird kaum verwendet. Bei vektorwertigen Daten $y_i \in \mathbb{R}^d$, $i = 0, \dots, n$, (z.B. Bahndaten einer Differentialgleichung) sieht das anders aus, $p(x) = \sum_{i=0}^n L_i(x)y_i$ beschreibt eine Kurve im \mathbb{R}^d , die teurere Berechnung der $L_i(x)$ amortisiert sich über die Dimension d insbesondere für $d \gg n$.

Beispiel 2.1.3 Es wird die Interpolation zum Gitter $x_0 = 0$, $x_1 = \frac{2}{3}$, $x_2 = 1$ betrachtet. Die zugehörigen Lagrange Polynome (2.1.4) sind daher

$$\begin{aligned} L_0(x) &= \frac{(x - 2/3)(x - 1)}{(0 - 2/3)(0 - 1)} = \left(\frac{3}{2}x - 1\right)(x - 1) = \frac{3}{2}x^2 - \frac{5}{2}x + 1, \\ L_1(x) &= \frac{(x - 0)(x - 1)}{(2/3 - 0)(2/3 - 1)} = \frac{9}{2}x(1 - x) = \frac{9}{2}(x - x^2), \\ L_2(x) &= \frac{x(x - 2/3)}{1(1 - 2/3)} = x(3x - 2) = 3x^2 - 2x. \end{aligned}$$

Für abstrakte Datenwerte y_0, y_1, y_2 oder vorgegebene $y^\top = (1, \frac{1}{3}, \frac{1}{2})$ bekommt man das Interpolationspolynom (2.1.4) explizit in der Form

$$\begin{aligned} p(x) &= \left(\frac{3}{2}x - 1\right)(x - 1)y_0 + \frac{9}{2}x(1 - x)y_1 + x(3x - 2)y_2 \\ &= \left(\frac{3}{2}x - 1\right)(x - 1) + \frac{3}{2}x(1 - x) + \frac{1}{2}x(3x - 2) = \frac{3}{2}x^2 - 2x + 1. \end{aligned}$$



Newton-Interpolation

Für einfache (skalare) Daten ist ein induktiver Ansatz für das Interpolationspolynom günstiger als die Lagrange-Darstellung. Der Ansatz erlaubt im praktischen Einsatz die einfache Hinzunahme zusätzlicher Stützstellen:

$$p_n(x) = p_{n-1}(x) + (x - x_0) \cdots (x - x_{n-1})a_n.$$

Nach Ansatz verschwindet der zweite Summand in den ersten n Punkten x_0, \dots, x_{n-1} . Wenn also p_{n-1} die Interpolationsaufgabe in den ersten n Punkten erfüllt, $p_{n-1}(x_i) = y_i$, $i = 0, \dots, n-1$, tut dies auch p_n . Den Wert a_n kann man dann aus der zusätzlichen Interpolationsbedingung $p_n(x_n) = y_n$ berechnen, zweckmäßiger ist aber ein weiter unten formulierter Algorithmus. Beginnend mit $p_0(x) \equiv y_0 = a_0$ bekommt p_n induktiv also die folgende Gestalt

$$\begin{aligned} p_n(x) &= a_0 + (x - x_0)a_1 + \dots + (x - x_0) \cdots (x - x_{n-1})a_n \\ &= a_0 + (x - x_0) \left(\cdots \left(a_{n-2} + (x - x_{n-2}) \left(a_{n-1} + (x - x_{n-1})a_n \right) \right) \cdots \right) \end{aligned} \quad (2.1.5)$$

Dies ist die *Newton-Darstellung* des Interpolationspolynoms. Die Klammerung in (2.1.5) deutet bereits das Berechnungsverfahren zur Auswertung $p_n(x)$ an:

Algorithmus 2.1.4 Horner-Schema zur Auswertung von $p_n(x)$ bei bekannten a_i .

$$\begin{aligned} q &:= a_n; \\ \text{für } k &= n-1, n-2, \dots, 0: \{ q := a_k + (x - x_k)q; \} \\ p_n(x) &= q. \end{aligned}$$

Rechenaufwand Hornerschema: $3n$ Operationen

Die Koeffizienten a_i in (2.1.5) berechnet man über folgenden Algorithmus, dessen Begründung einige Zusatzüberlegungen erfordert.

Definition 2.1.5 (und Algorithmus) Die k -ten dividierten Differenzen oder k -ten Differenzenquotienten der Daten y bezgl. der Stützstellen x werden bestimmt durch den Algorithmus

$$\begin{aligned} &\text{für } i = 0, 1, \dots, n: \{ y[x_i] := y_i; \} \\ &\text{für } k = 1, \dots, n: \{ \\ &\quad \text{für } i = 0, \dots, n-k: \{ \\ &\quad \quad y[x_i, \dots, x_{i+k}] := \frac{y[x_{i+1}, \dots, x_{i+k}] - y[x_i, \dots, x_{i+k-1}]}{x_{i+k} - x_i}; \} \\ &\quad \} \end{aligned} \quad (2.1.6)$$

Das Verfahren (2.1.6) erzeugt folgendes Dreieckschema spaltenweise ($k = 1, 2, \dots, n$).

| x_i | $y_i = y[x_i]$ | $k = 1$ | $k = 2$ | $k = n$ |
|----------|-------------------------|-------------------------------------|--|---|
| x_0 | <u>y_0</u> | | | |
| x_1 | y_1 | <u>$y[x_0, x_1]$</u> | | |
| x_2 | y_2 | <u>$y[x_1, x_2]$</u> | <u>$y[x_0, x_1, x_2]$</u> | |
| \vdots | \vdots | \vdots | \ddots | |
| x_n | y_n | <u>$y[x_{n-1}, x_n]$</u> | <u>$y[x_{n-2}, x_{n-1}, x_n]$</u> | \cdots <u>$y[x_0, \dots, x_n]$</u> |

(2.1.7)

Im Newtonpolynom (2.1.5) werden nur die unterstrichenen Differenzen $y[x_0, \dots, x_i]$ benötigt, s.u., (2.1.11). Daher genügt zur Durchführung ein lineares Feld $[0, 1, \dots, n]$, in dem der Reihe nach zunächst die Werte y_n, \dots, y_1 rückwärts durch die ersten dividierten Differenzen, dann die ersten dividierten Differenzen $y[x_1, x_2], \dots, y[x_{n-1}, x_n]$ durch die zweiten überschrieben werden, usw. Am Ende bleiben die in (2.1.11) benötigten Werte übrig. Es ist auch eine zeilenweise Berechnung möglich, wenn man mit dem höchsten Index beginnt. Dabei kann dann bei Hinzunahme zusätzlicher Interpolationspunkte das Differenzenschema einfach ergänzt werden.

Rechenaufwand für (2.1.6): $3(n + (n - 1) + \dots + 1) = \frac{3}{2}n^2 + \mathcal{O}(n)$ Operationen.

Der Zusammenhang der Differenzenquotienten (2.1.6) mit den im Newtonpolynom (2.1.5) benötigten Koeffizienten a_i ergibt sich am einfachsten aus dem folgenden Neville-Algorithmus und dem anschließenden Satz 2.1.7.

Definition 2.1.6 (und Neville-Algorithmus) *Es seien $p_{ik} \in \Pi_k$ Abschnittspolynome mit*

$$p_{ik}(x_j) = y_j, \quad j = i, \dots, i + k. \tag{2.1.8}$$

Man erhält die Werte dieser p_{ik} und den des Polynoms p_n zu (2.1.2) an einer Stelle x durch

$$\begin{aligned}
 &\text{für } i = 0, 1, \dots, n: \{ p_{i0}(x) := y_i; \} \\
 &\text{für } k = 1, \dots, n: \{ \\
 &\quad \text{für } i = 0, \dots, n - k: \{ \\
 &\quad \quad p_{ik}(x) := \frac{(x - x_i)p_{i+1,k-1}(x) + (x_{i+k} - x)p_{i,k-1}(x)}{x_{i+k} - x_i}; \} \} \\
 &p_n(x) = p_{0n}(x).
 \end{aligned} \tag{2.1.9}$$

Der Neville-Algorithmus liefert (mit $\mathcal{O}(n^2)$ -Aufwand) nur einen einzigen Polynomwert und ist daher nur in Sonderfällen von praktischem Interesse. Er stellt aber die Verbindung zu den dividierten Differenzen aus (2.1.6) her.

Satz 2.1.7 *a) Die in (2.1.9) berechneten Polynome und die in (2.1.8) definierten sind gleich.
 b) Die dividierten Differenzen in (2.1.6) stimmen mit den höchsten Koeffizienten der Teilpolynome in (2.1.8) überein:*

$$y[x_i, \dots, x_{i+k}] = \frac{1}{k!} p_{ik}^{(k)}, \quad 0 \leq i \leq i + k \leq n. \tag{2.1.10}$$

Dieser Wert ist unabhängig von der Reihenfolge der Wertepaare $(x_i, y_i), \dots, (x_{i+k}, y_{i+k})$.

c) Das Lagrange-Interpolationspolynom (2.1.2) ist in der Newtonform (2.1.5) darstellbar als

$$p_n(x) = y[x_0] + (x - x_0)y[x_0, x_1] + \dots + (x - x_0) \cdots (x - x_{n-1})y[x_0, \dots, x_n]. \quad (2.1.11)$$

Beweis Durch Induktion über k , die Aussagen zu (2.1.8) bis (2.1.11) sind offensichtlich richtig für $k = 0$.

a) Unter der Induktionsvoraussetzung, dass $p_{i,k-1}(x)$ und $p_{i+1,k-1}(x)$ ihre jeweiligen Interpolationsbedingungen in (2.1.8) erfüllen, gilt für das in (2.1.9) definierte $p_{ik}(x)$ die Aussage

$$p_{ik}(x_j) = y_j, \quad j = i, \dots, i+k.$$

Für $j = i$ und $j = i+k$ treten nämlich in (2.1.9) nur der zweite bzw. erste Summand im Zähler auf, für die Fälle $j = i+1, \dots, i+k-1$ kann man aufgrund der Induktionsvoraussetzung den Faktor y_j ausklammern, die Vorfaktoren summieren zu eins.

b) Nun wird angenommen, (2.1.10) sei richtig für $k-1$. Dann gilt nach (2.1.5) und (2.1.8)

$$p_{ik}(x) = p_{i,k-1}(x) + (x - x_i) \cdots (x - x_{i+k-1}) \cdot a_{ik}. \quad (2.1.12)$$

Die k -te Ableitung dieser Identität zeigt $p_{ik}^{(k)} = k! a_{ik}$ und die Behauptung (2.1.10) ist daher äquivalent zu

$$a_{ik} = y[x_i, \dots, x_{i+k}]. \quad (2.1.13)$$

Um dies zu zeigen, betrachtet man die höchsten Koeffizienten a_{ik} im Neville-Algorithmus (2.1.9). Mit der Induktionsvoraussetzung erhält man tatsächlich die Differenzen-Rekursion (2.1.6) für die Werte (2.1.13):

$$a_{ik} = \frac{a_{i+1,k-1} - a_{i,k-1}}{x_{i+k} - x_i} \stackrel{(I.V.)}{=} \frac{y[x_{i+1}, \dots, x_{i+k}] - y[x_i, \dots, x_{i+k-1}]}{x_{i+k} - x_i} \stackrel{(2.1.6)}{=} y[x_i, \dots, x_{i+k}].$$

c) Ist der Spezialfall von (2.1.12) und (2.1.13) für $i = 0$ und $k = 0, \dots, n$. ■

Beispiel 2.1.8 Newton-Polynom durch vier Interpolationspunkte (Werte in der Tabelle):

| i | x_i | y_i | | | |
|-----|-------|-------|---|----------------|---------------|
| 0 | -1 | 2 | | | |
| 1 | 0 | 4 | 2 | | |
| 2 | 2 | 6 | 1 | $-\frac{1}{3}$ | |
| 3 | 3 | 12 | 6 | $\frac{5}{3}$ | $\frac{1}{2}$ |

Die eingerahmten Werte sind die Koeffizienten im Interpolationspolynom

$$p_3(x) = 2 + 2(x+1) - \frac{1}{3}(x+1)x + \frac{1}{2}(x+1)x(x-2).$$

Bei den besprochenen Verfahren werden keinerlei spezielle Eigenschaften der verwendeten Daten y_j berücksichtigt. Wenn diese allerdings als *Funktionswerte* einer glatten Funktion f interpretiert werden können, läßt sich die Abweichung von p_n und f abschätzen.

Satz 2.1.9 Es sei $f \in C^{n+1}[a, b]$ und $y_j = f(x_j)$, $j = 0, \dots, n$, mit einem Gitter (2.1.1).

a) Es gilt

$$f[x_i, \dots, x_{i+k}] := y[x_i, \dots, x_{i+k}] = \frac{1}{k!} f^{(k)}(\xi), \quad (2.1.14)$$

mit $\xi \in (\min_{j=i}^{i+k} x_j, \max_{j=i}^{i+k} x_j)$, $0 \leq i \leq i+k \leq n$.

b) Für $f(x) = c_k x^k + \dots \in \Pi_k$ ist

$$f[x_0, \dots, x_n] = \begin{cases} c_k & \text{für } k = n, \\ 0 & \text{für } k < n. \end{cases}$$

c) Für den Interpolationsfehler gilt mit dem Knotenpolynom $\omega_{n+1}(x) := (x - x_0) \dots (x - x_n)$ und einem $\xi \in (a, b)$ die Darstellung

$$f(x) - p_n(x) = \omega_{n+1}(x) f[x_0, \dots, x_n, x] = \omega_{n+1}(x) \frac{f^{(n+1)}(\xi)}{(n+1)!}. \quad (2.1.15)$$

Beweis Die Interpolationsfehler werden mit $r_{ik}(x) := f(x) - p_{ik}(x)$ bezeichnet.

a) Nach (2.1.10) ist $y[x_i, \dots, x_{i+k}] = p_{ik}^{(k)}(x)/k!$. Nun verschwindet r_{ik} in den Punkten $x = x_i, \dots, x_{i+k}$. Nach dem iterierten Satz von Rolle existiert $\xi \in (\min_{j=i}^{i+k} x_j, \max_{j=i}^{i+k} x_j)$ mit

$$0 = r_k^{(k)}(\xi) = f^{(k)}(\xi) - p_{ik}^{(k)}(\xi).$$

b) Dies ist ein Spezialfall von a).

c) Der Interpolationsfehler wird an einer festen Stelle $x = \bar{x}$ dargestellt und das erweiterte Interpolationspolynom $p_{0,n+1}$ mit der zusätzlichen Stützstelle $x_{n+1} := \bar{x}$, $y_{n+1} := f(\bar{x})$, betrachtet. Nach Satz 2.1.7 gilt hierfür

$$p_{0,n+1}(x) = p_{0n}(x) + (x - x_0) \dots (x - x_n) f[x_0, \dots, x_n, x_{n+1} = \bar{x}].$$

Wegen $p_{0,n+1}(\bar{x}) = f(\bar{x})$ folgt in \bar{x} für den Fehler

$$r_{0n}(\bar{x}) = f(\bar{x}) - p_{0n}(\bar{x}) = (\bar{x} - x_0) \dots (\bar{x} - x_n) f[x_0, \dots, x_n, \bar{x}].$$

Die Definition von ω_{n+1} und (2.1.14) liefern schließlich (2.1.15). ■

Wegen der Beziehung (2.1.14) liegt es nahe, den Grenzübergang von mehreren Knoten in eine gemeinsame Stelle zu betrachten. Bekanntlich ist

$$\lim_{x_{i+1} \rightarrow x_i} f[x_i, x_{i+1}] = f'(x_i) \quad \text{für } f \in C^1[a, b].$$

Analoge Aussagen gelten in (2.1.14), wenn mehr als zwei Knoten zusammenrücken. Das Interpolationsproblem ist allerdings nur dann sinnvoll gestellt, wenn an einer Stelle Funktions- und Ableitungswerte bis zu einer bestimmten Ordnung lückenlos vorgeschrieben sind. Die Definition 2.1.1 kann daher verallgemeinert werden.

Definition 2.1.10 Gegeben sei ein Gitter $\{x_0, \dots, x_n\}$ beliebiger Stützstellen $x_i \in [a, b]$, zusammenfallende Stützstellen seien zusammenhängend numeriert ($x_i = x_j$ für $i < j \Rightarrow x_k = x_i$ für $i \leq k \leq j$). Beim Hermite-Interpolationsproblem wird zu einem Datenvektor (y_0, \dots, y_n) ein Polynom $p \in \Pi_n$ gesucht mit

$$p^{(j)}(x_i) = y_{i+j}, \quad x_i = x_{i+1} = \dots = x_{i+j}, \quad 0 \leq i \leq n.$$

Für die Hermite-Interpolationsaufgabe sind also die folgenden Werte vorzugeben,

$$f(x_i), f'(x_i), \dots, f^{(j)}(x_i), \quad \text{wenn } x_i = x_{i+1} = \dots = x_{i+j}. \quad (2.1.16)$$

Im Einklang mit (2.1.14) definiert man mit diesen Ableitungen Differenzenquotienten durch

$$f[x_i, \dots, x_{i+k}] := \frac{1}{k!} f^{(k)}(x_i), \quad 0 \leq k \leq j.$$

Für alle solchen *Mehrfachknoten* (2.1.16) werden diese Werte in das Differenzentableau (2.1.7) eingetragen. Die noch fehlenden Differenzenquotienten berechnet man dann nach (2.1.6).

Beispiel 2.1.11 Differenzentableau für $n = 4$, $x_1 = x_2 = x_3$:

$$\begin{array}{l|l} x_0 & y_0 \\ x_1 & y_1 \quad y[x_0, x_1] \\ x_1 & y_1 \quad y'_1 \quad y[x_0, x_1, x_1] \\ x_1 & y_1 \quad y'_1 \quad \frac{1}{2}y''_1 \quad y[x_0, x_1, x_1, x_1] \\ x_4 & y_4 \quad y[x_1, x_4] \quad y[x_1, x_1, x_4] \quad y[x_1, x_1, x_1, x_4] \quad y[x_0, x_1, x_1, x_1, x_4] \end{array}$$

Bemerkung: Man kann das Hornerschema aus Algorithmus 2.1.3 verallgemeinern zur Berechnung von Ableitungswerten von Polynomen.

Satz 2.1.9 erlaubt eine Gegenüberstellung von Taylor- und Newton-Entwicklung für $f \in C^{n+1}[a, b]$. Mit den analog zu ω_{n+1} definierten Polynomen $\omega_k(x) = (x - x_0) \cdots (x - x_{k-1})$, vgl. (2.1.15), erhält man für die Entwicklung nach

$$\begin{aligned} \text{Taylor:} \quad f(x) &= \sum_{k=0}^n (x - x_0)^k \frac{f^{(k)}(x_0)}{k!} + (x - x_0)^{n+1} \frac{f^{(n+1)}(\xi)}{(n+1)!}, \\ \text{Newton:} \quad f(x) &= \sum_{k=0}^n \omega_k(x) f[x_0, \dots, x_k] + \omega_{n+1}(x) \frac{f^{(n+1)}(\eta)}{(n+1)!}, \end{aligned}$$

$\xi, \eta \in (a, b)$. Insbesondere geht für $x_j \rightarrow x_0$, $j = 1, \dots, n$, die Newton-Darstellung über in die Taylor-Darstellung.

Interpolationsfehler, optimale Knoten

Wenn die Plazierung der “Eingangsdaten” (x_i, y_i) nicht fest vorgegeben ist, kann der Interpolationsfehler ohne Mehraufwand durch geeignete Wahl der Stützstellen x_i stark reduziert werden. Dazu setzt man an der Darstellung (2.1.15) des Interpolationsfehlers an. Wenn eine *gleichmäßig* gute Approximation im Intervall $[a, b]$ erzielt werden soll, benutzt man die Supremumnorm

$$\|g\|_\infty := \sup\{|g(x)| : x \in [a, b]\}, \quad g \in C[a, b]$$

des Fehlers und versucht, diese möglichst klein zu machen. Für eine “glatte” Funktion $f \in C^{n+1}[a, b]$ erhält man hier aus der Fehlerdarstellung (2.1.15) mit $\omega_{n+1}(x) = (x - x_0) \cdots (x - x_n)$ die Schranke

$$\|f - p_n\|_\infty \leq \|\omega_{n+1}\|_\infty \frac{\|f^{(n+1)}\|_\infty}{(n+1)!}. \quad (2.1.17)$$

Bei festem Polynomgrad n ist der Anteil $\|f^{(n+1)}\|_\infty$ durch die Funktion f vorgegeben. Dagegen kann aber die Norm $\|\omega_{n+1}\|_\infty$ des Knotenpolynoms durch geeignete Wahl der Stützstellen beeinflusst werden. Es ist sogar eine optimale Wahl möglich mit Knoten x_0, \dots, x_n , die

$$\|\omega_{n+1}\|_\infty = \min_{\xi_j \in [a,b]} \max_{x \in [a,b]} |x - \xi_0| |x - \xi_1| \cdots |x - \xi_n| \quad (2.1.18)$$

erfüllen. In diesem Zusammenhang spielen folgende Polynome eine Rolle

Definition 2.1.12 *Die Funktionen*

$$T_n(x) := \cos(n \arccos x), \quad |x| \leq 1, \quad n = 0, 1, \dots, \quad (2.1.19)$$

heißen Tschebyscheff-Polynome und die Nullstellen von T_{n+1}

$$x_k := x_{k,n} := \cos\left(\frac{2k+1}{2n+2}\pi\right), \quad k = 0, \dots, n \quad (2.1.20)$$

Tschebyscheff-Punkte oder -Knoten.

Satz 2.1.13 *Die in (2.1.19) definierten Funktionen T_n sind Polynome vom Grad n . Sie können mit $T_0 = 1$, $T_1(x) = x$, rekursiv berechnet werden durch die Drei-Term-Rekursion*

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x), \quad n = 1, 2, \dots \quad (2.1.21)$$

Die Punkte $x_k = x_{k,n}$ aus (2.1.20) sind die Nullstellen des Polynoms T_{n+1} . Die Polynome T_n haben die Form

$$T_n(x) = 2^{n-1}x^n + \dots, \quad T_n \in \Pi_n, \quad (2.1.22)$$

ihre Supremum-Norm auf $[-1, 1]$ ist eins, für $n \in \mathbb{N}_0$ gilt

$$\|T_n\|_{[-1,1],\infty} = \max_{x \in [-1,1]} |T_n(x)| = 1. \quad (2.1.23)$$

Beweis Mit $t := \arccos x$ folgen die Aussagen für T_0, T_1 unmittelbar aus (2.1.19). Die Rekursion in (2.1.21) ergibt sich aus dem Additionstheorem

$$\cos(n+1)t + \cos(n-1)t = 2 \cos t \cdot \cos nt.$$

An (2.1.21) erkennt man, dass die $T_n \in \Pi_n$ die Form (2.1.22) besitzen und an (2.1.19), dass die x_k aus (2.1.20) die Nullstellen von T_{n+1} sind und (2.1.23) gilt. ■

Geeignet skalierte Tschebyscheff-Polynome führen tatsächlich auf optimale Interpolationsknoten.

Satz 2.1.14 *Die optimalen Knoten in (2.1.18) sind die Tschebyscheff-Punkte*

$$x_k := \frac{a+b}{2} - \frac{b-a}{2} \cos\left(\frac{2k+1}{2n+2}\pi\right), \quad k = 0, \dots, n. \quad (2.1.24)$$

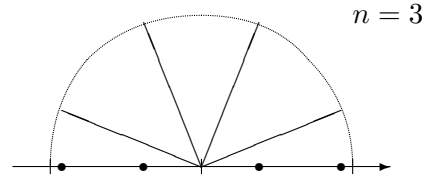
Der Minimalwert der Norm ist

$$\|\omega_{n+1}\|_\infty = \max_{x \in [a,b]} |\omega_{n+1}(x)| = 2 \left(\frac{b-a}{4}\right)^{n+1}. \quad (2.1.25)$$

Bemerkung: Die Knoten ergeben sich aus den in (2.1.20) genannten durch affine Transformation des Intervalls

$$\begin{cases} [-1, 1] & \rightarrow [a, b] \\ \xi & \mapsto \frac{a+b}{2} - \frac{b-a}{2}\xi = x \end{cases} \quad (2.1.26)$$

Die Punkte $\xi_k = \cos \frac{2k+1}{2n+2}\pi = \operatorname{Re} \exp \left(i \frac{2k+1}{2n+2}\pi \right)$ sind Realteile der $(4n+4)$ -ten komplexen Einheitswurzeln. Sie liegen am Rand des Intervalls $[-1, 1]$ *viel dichter* als im Innern.



Beweis des Satzes: Beim Standardintervall $[a, b] = [-1, 1]$ gilt $x_k = -\xi_k$, $k = 0, \dots, n$, also

$$T_{n+1}(x_k) = \cos \left((n+1) \arccos x_k \right) = \cos \left((n+1) \frac{2k+1}{2n+2}\pi \right) = \cos \left(\frac{\pi}{2} + k\pi \right) = 0.$$

Daher ist ω_{n+1} , bis auf eine Konstante, das Tschebyscheff-Polynom. Ein Vergleich des höchsten Koeffizienten mit (2.1.22) liefert genauer

$$\omega_{n+1}(x) = x^{n+1} + \dots = 2^{-n} T_{n+1}(x).$$

Die Funktion $\cos(n+1)\varphi$ nimmt ihre Extrema ± 1 in den Stellen $\varphi_j = \frac{j\pi}{n+1}$ an, das Polynom ω_{n+1} seine Extrema $\pm 2^{-n}$ daher in $t_j = \cos \frac{j\pi}{n+1}$, $j = 0, \dots, n+1$ (*Alternanden*). Daher gilt

$$\|\omega_{n+1}\|_{\infty} = 2^{-n}.$$

Dies ist tatsächlich der Minimalwert dieser Norm, gäbe es nämlich ein Polynom der Form $p_{n+1} = x^{n+1} + \dots$ mit *kleinerer* Norm, dann müßte für dessen Differenz zu ω_{n+1} in den Stellen t_j gelten

$$\omega_{n+1}(t_j) - p_{n+1}(t_j) \begin{cases} < 0 & \text{für } n+1-j \text{ gerade} \\ > 0 & \text{für } n+1-j \text{ ungerade} \end{cases}, \quad j = 0, \dots, n+1.$$

Das Polynom $\omega_{n+1} - p_{n+1}$ vom Grad n (!) hätte dann aber mindestens $n+1$ Nullstellen. Dies führt zu einem Widerspruch. Durch die Transformation (2.1.26) wird

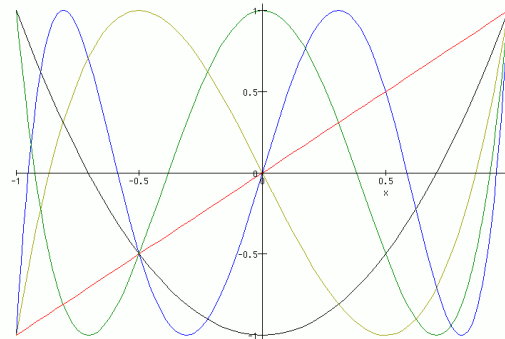
$$T_{n+1}(\xi) = T_{n+1} \left(\frac{a+b-2x}{b-a} \right) = 2^n \left(\frac{-2}{b-a} \right)^{n+1} x^{n+1} + \dots,$$

die Normierung des höchsten Koeffizienten ergibt damit

$$\omega_{n+1}(x) = \left(-\frac{b-a}{2} \right)^{n+1} 2^{-n} T_{n+1}(\xi).$$

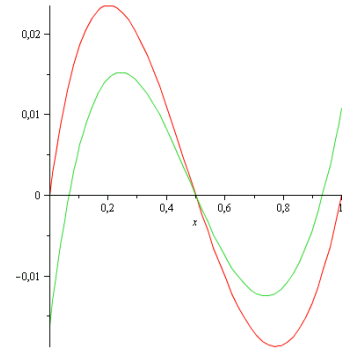
Daraus folgt die Schranke (2.1.25). ■

Die nebenstehende Graphik der Tschebyscheffpolynome T_1, \dots, T_5 zeigt die auch im Beweis verwendeten Eigenschaften dieser Polynomfamilie. Die Polynome von (un-) geradem Grad sind (un-) gerade Funktionen, die expliziten Formeln sind $T_2 = 2x^2 - 1$, $T_3(x) = 4x^3 - 3x$, $T_4(x) = 8x^4 - 8x^2 + 1$, $T_5(x) = 16x^5 - 20x^3 + 5x$.



Beispiel 2.1.15 Die Funktion $f(x) = \sin(\frac{\pi}{2}x)$ soll im Intervall $[0, 1]$ durch ein quadratisches Polynom approximiert werden. Dazu wird die Interpolation in den Stellen $x^T = (0, \frac{1}{2}, 1)$ verglichen mit der in Tschebyscheffknoten $\hat{x} = (\frac{1}{4}(2 - \sqrt{3}), \frac{1}{2}, \frac{1}{4}(2 + \sqrt{3}))$. Die beiden folgenden Tabellen zeigen oben die Ergebnisse zu äquidistanten Punkten, unten die zu Tschebyscheffknoten und in der Grafik die Interpolationsfehler.

| | | | |
|--|------------|------------|-------------|
| 0 | 0 | | |
| 0.5 | 0.70710678 | 1.41421356 | |
| 1 | 1 | 0.58578643 | -0.82842712 |
| $p(x) = x(1.82842712 - 0.82842712x)$ | | | |
| Fehler $\cong 0.0235$ | | | |
| | | | |
| 0.06698729 | 0.10502933 | | |
| 0.5 | 0.70710678 | 1.39043829 | |
| 0.93301270 | 0.99446912 | 0.66363490 | -0.83924026 |
| $\hat{p}(x) = -0.01622158 + 1.86627686x - 0.83924026x^2$ | | | |
| Fehler = 0.0162 | | | |



Wegen des niedrigen Polynomgrads ist der Genauigkeitsunterschied in diesem Beispiel nur gering. Dies ändert sich bei höheren Polynomgraden. Nach dem Satz von Weierstraß kann jede stetige Funktion auf einem kompakten Intervall beliebig genau durch Polynome approximiert werden. Dieses Ergebnis läßt sich aber nicht unbesehen auf Interpolationspolynome übertragen, selbst wenn immer mehr Stützstellen verwendet werden. Mit der Angabe (2.1.25) hat die Fehleraussage (2.1.17) für Tschebyscheffknoten die Form

$$\|p_n - f\|_\infty \leq 2 \left(\frac{b-a}{4} \right)^{n+1} \frac{\|f^{(n+1)}\|}{(n+1)!}. \quad (2.1.27)$$

Der Vorfaktor bei $f^{(n+1)}$ ist dabei kleiner als bei allen anderen Stützstellen, etwa bei äquidistanten, $x_i = a + (b-a)i/n$. Bei der Formel (2.1.27) ist aber zu beachten, dass sich bei Anhebung des Polynomgrads n auch die Ordnung der Ableitung $f^{(n+1)}$ erhöht. Schon lange bekannt ist das Beispiel von Runge

$$f(x) := \frac{1}{1 + 25x^2}, \quad x \in [-1, 1],$$

wo bei äquidistanten Stützstellen das Anwachsen der Ableitungen für $n \rightarrow \infty$ verhindert, dass der Fehler klein wird. Die Folge der Interpolationspolynome (p_n) ($n \rightarrow \infty$) divergiert in diesem Beispiel sogar in Randnähe bei ± 1 . Bei Tschebyscheffknoten ist wegen der Alternandeneigenschaft (vgl. Beweis) der Fehlerverlauf viel gleichmäßiger. Für $f \in C^{m+1}[-1, 1]$, $m \geq 1$ läßt sich hier sogar allgemein Konvergenz nachweisen in der Form [Schaback, 10.4.5]

$$\|p_n - f\|_\infty = o\left(\frac{\log n}{n^m}\right), \quad n \rightarrow \infty.$$

2.2 Spline-Funktionen

Der Fehler bei Interpolation einer Funktion $f \in C^{k+1}[\alpha, \beta]$ durch ein Polynom p_k vom Grad k mit Stützstellen $\xi_0, \dots, \xi_k \in [\alpha, \beta]$ hat nach (2.1.17) die Gestalt ($x \in [\alpha, \beta]$)

$$|f(x) - p_k(x)| \leq |(x - \xi_0) \cdots (x - \xi_k)| \frac{\|f^{(k+1)}\|_\infty}{(k+1)!} \leq \frac{(\beta - \alpha)^{k+1}}{(k+1)!} \|f^{(k+1)}\|_\infty. \quad (2.2.1)$$

Im letzten Paragraph wurde ein Beispiel genannt, bei dem für $k \rightarrow \infty$ i.a. keine Konvergenz $\|f - p_k\|_\infty \rightarrow 0$ garantiert werden kann. Dagegen folgt aus (2.2.1) trivialerweise

$$|f - p_k| \rightarrow 0 \quad \text{für} \quad (\beta - \alpha) \rightarrow 0, \quad k \text{ fest.}$$

Dies nutzt man aus durch Unterteilung des Gesamtintervalls $[\alpha, \beta]$ mit einem Gitter

$$\begin{aligned} \Delta : \quad & \alpha = x_0 < x_1 < \dots < x_n = \beta, \\ & h_i := x_{i+1} - x_i, \quad i = 0, \dots, n-1, \quad H := \max_{i=0}^{n-1} h_i, \end{aligned} \quad (2.2.2)$$

und approximiert die Funktion f in den Teilintervallen $[x_i, x_{i+1}]$ durch Polynome festen Grades.

Definition 2.2.1 Zu einem Gitter Δ nach (2.2.2) und $m, k \in \mathbb{N}_0$, $0 \leq m < k$, wird der Raum S_k^m aller Spline-Funktionen s bezeichnet, die

- a) lokal, in jedem Teilintervall $(x_i, x_{i+1}]$, Polynome vom Grad k sind: $s|_{(x_i, x_{i+1}]} \in \Pi_k$,
 b) global m -mal stetig differenzierbar sind: $s \in C^m[\alpha, \beta]$.

Der wichtigste Fall ist der der *kubischen Splines* ($k = 3$) mit $m \in \{1, 2\}$. Splines werden außer zur Approximation von Meßdaten hauptsächlich zur Kurven- und Flächendarstellung in der Computer-Graphik (CAD) und als Finite-Elemente-Methode (FEM) zur Lösung von partiellen Differentialgleichungen eingesetzt. Zur Interpolation wird definiert:

Definition 2.2.2 Mit einem Gitter Δ nach (2.2.2) und $f \in C^1[\alpha, \beta]$ ist die Funktion $s \in S_3^2$ der *kubische Interpolationsspline*, wenn sie die Interpolationsbedingungen

$$s(x_i) = f(x_i), \quad i = 0, 1, \dots, n, \quad (2.2.3)$$

erfüllt und eine der Randbedingungen

$$\begin{aligned} a) \quad & s''(\alpha) = s''(\beta) = 0, \\ b) \quad & s'(\alpha) = f'(\alpha), \quad s'(\beta) = f'(\beta). \end{aligned} \quad (2.2.4)$$

Zur Darstellung von Splines sind zwei Methoden verbreitet. Die *Bézier-Darstellung* (2.2.5) ist flexibler, wenn sowohl C^2 - als auch C^1 - oder C^0 -Splines verwendet werden sollen. Allerdings benutzt diese Darstellung bei C^2 -Splines unnötig viele Parameter. Dies wird bei Verwendung

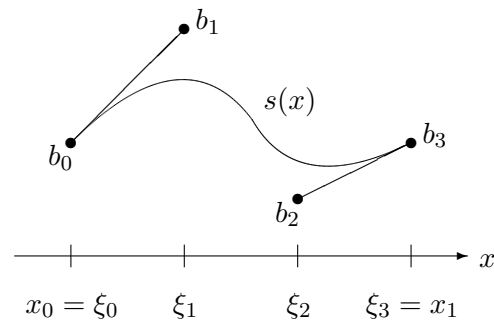
einer Basis von S_3^2 aus sogenannten *B-Splines* vermieden. Die *Bézier-Bernstein-Darstellung* von kubischen Polynomen auf dem Standardintervall $[0, h]$, $h > 0$, lautet

$$s(x) = \frac{1}{h^k} \sum_{j=0}^k \binom{k}{j} x^j (h-x)^{k-j} b_j \tag{2.2.5}$$

mit Koeffizienten b_0, \dots, b_k . Diese Darstellung besitzt folgende Eigenschaften:

$$\left. \begin{aligned} (a) \quad & b_j \equiv 1 \Rightarrow s(x) = \frac{1}{h^k} \sum_{j=0}^k \binom{k}{j} x^j (h-x)^{k-j} = \frac{1}{h^k} (x+h-x)^k \equiv 1 \\ (b) \quad & s(0) = b_0, \quad s(h) = b_k \\ (c) \quad & s'(x) = \frac{1}{h^k} [-k(h-x)^{k-1} b_0 + k(h-kx)(h-x)^{k-2} b_1 + x(\dots)] \Rightarrow \\ & s'(0) = \frac{b_1 - b_0}{h/k}, \quad s'(h) = \frac{b_k - b_{k-1}}{h/k} \\ (d) \quad & s''(0) = \frac{k(k-1)}{h^2} (b_0 - 2b_1 + b_2), \quad s''(h) = \frac{k(k-1)}{h^2} (b_{k-2} - 2b_{k-1} + b_k). \end{aligned} \right\} \tag{2.2.6}$$

Die Eigenschaft (c) führt bei $k = 3$ auf eine einfache *geometrische* Interpretation der Koeffizienten b_j . Werden die Paare $(\xi_j, b_j)^\top$ mit $\xi_j = jh/3$ betrachtet, dann ist die Gerade $(\xi_0, b_0)^\top (\xi_1, b_1)^\top$ die Tangente an s in $x = \xi_0 = 0$ und $(\xi_2, b_2)^\top (\xi_3, b_3)^\top$ die Tangente an s in $x = \xi_3 = h$. Durch b_0, b_3 werden also die Werte von s in den Randpunkten beeinflusst, durch b_1, b_2 die dortigen Ableitungen. Man bezeichnet die Koeffizienten oft als *Kontrollpunkte*.



Der Spline s wird auf dem Gesamtintervall $[\alpha, \beta]$ durch lokale Darstellungen (2.2.5) zusammengesetzt. Dazu wird ein Zusatzgitter eingeführt,

| | | | | |
|---------|---------|---------|---------|---------|
| y_0 | y_1 | y_2 | \dots | |
| x_0 | x_1 | x_2 | \dots | |
| b_0 | b_1 | b_2 | b_3 | b_4 |
| ξ_0 | ξ_1 | ξ_2 | ξ_3 | ξ_4 |
| b_5 | b_6 | \dots | | |
| ξ_5 | ξ_6 | | | |

$\xi_{ik+j} := x_i + \frac{j}{k} h_i,$
 $j = 0, \dots, k-1,$
 $i = 0, \dots, n-1,$
(2.2.7)

Die **Bézier-Darstellung** wird nur für äquidistante Gitter behandelt. Dies ist aber bei einer der wichtigsten Anwendungen, der Parameterdarstellung von Kurven $(s_1(x), s_2(x), \dots)$ im \mathbb{R}^2 bzw. \mathbb{R}^3 keine starke Einschränkung. In diesem Fall, $h_j \equiv h$, gibt es für die verschiedenen Stetigkeitsbedingungen einfache geometrische Interpretationen.

- C^0 ist erfüllt mit $b_{3j} = y_j$
- 1. Ableitung, deren Stetigkeit bedeutet nach (2.2.6,c)

$$s'(x_{j-}) = \frac{3}{h} (b_{3j} - b_{3j-1}) \stackrel{!}{=} \frac{3}{h} (b_{3j+1} - b_{3j}) = s'(x_{j+}) \iff \frac{1}{2} (b_{3j-1} + b_{3j+1}) = b_{3j} = y_j. \tag{2.2.8}$$

Da auch $\xi_{3j} = \frac{1}{2}(\xi_{3j-1} + \xi_{3j+1})$ gilt, ist also der \mathbb{R}^2 -Punkt $(\xi_{3j}, b_{3j})^\top$ Mittelpunkt der Verbindungsstrecke von $(\xi_{3j-1}, b_{3j-1})^\top$ und $(\xi_{3j+1}, b_{3j+1})^\top$.

- 2. Ableitung: Nach (2.2.6,d) ist die Bedingung $s''(x_{j-}) \stackrel{!}{=} s''(x_{j+})$ äquivalent mit

$$\frac{1}{h^2}(b_{3j-2} - 2b_{3j-1} + b_{3j}) = \frac{1}{h^2}(b_{3j} - 2b_{3j+1} + b_{3j+2}) \quad (2.2.9)$$

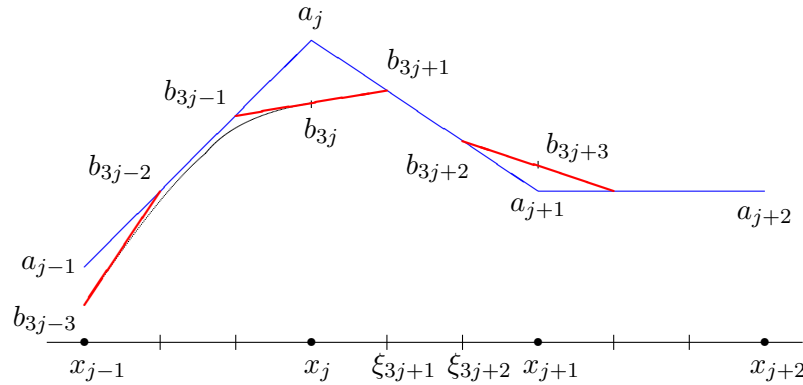
Hier fällt b_{3j} weg und es ist sinnvoll, spezielle Teilausdrücke als neue Parameter einzuführen. Aus (2.2.9) wird so:

$$\begin{aligned} 2b_{3j-1} - b_{3j-2} &= 2b_{3j+1} - b_{3j+2} =: a_j, & \text{außerdem ist} \\ 2\xi_{3j-1} - \xi_{3j-2} &= 2\xi_{3j+1} - \xi_{3j+2} = \xi_{3j} = x_j. \end{aligned}$$

Daher liegt der Punkt $(x_j, a_j)^\top$ im Schnittpunkt der Geraden durch $(\xi_\nu, b_\nu)^\top$ mit $\nu = 3j-2, 3j-1$ sowie $\nu = 3j+1, 3j+2$. Umgekehrt dritteln die Punkte $(\xi_{3j+\nu}, b_{3j+\nu})^\top$, $\nu = 1, 2$, die Strecke von $(x_j, a_j)^\top$ nach $(x_{j+1}, a_{j+1})^\top$, denn

$$\left. \begin{aligned} 2b_{3j+1} - b_{3j+2} &= a_j \\ -b_{3j+1} + 2b_{3j+2} &= a_{j+1} \end{aligned} \right\} \Rightarrow \begin{cases} b_{3j+1} = \frac{1}{3}(2a_j + a_{j+1}) \\ b_{3j+2} = \frac{1}{3}(a_j + 2a_{j+1}) \end{cases}. \quad (2.2.10)$$

Diese Beziehungen der Bézier-Koeffizienten beim C^2 -Spline erlauben folgende geometrische Interpretation (Hilfslinien: C^1 rot, C^2 blau), die analog auch für Kurven $(s_1(x), s_2(x))^\top$ gilt. Der Spline ist hier nur links von x_j skizziert:



Mit (2.2.10) können auch die C^1 -Bedingungen (2.2.8) noch umformuliert werden,

$$2b_{3j} = b_{3j-1} + b_{3j+1} = \frac{1}{3}(a_{j-1} + 4a_j + a_{j+1}).$$

Daher erfüllen die Koeffizienten a_j des C^2 -Interpolationssplines das Gleichungssystem

$$a_{j-1} + 4a_j + a_{j+1} = 6y_j, \quad j = 1, \dots, n-1, \quad (2.2.11)$$

das je nach Randbedingung (2.2.4) zu ergänzen ist durch

| | | |
|------------------------------|---------------------------------|--|
| $a_0 = y_0,$ | $a_n = y_n,$ | bei $s''(a) = s''(b) = 0$, bzw. bei $s'(a) = y'_0, s'(b) = y'_n$. |
| $2a_0 + a_1 = 3y_0 + hy'_0,$ | $a_{n-1} + 2a_n = 3y_n - hy'_n$ | |

(2.2.12)

Denn nach (2.2.6,d) gilt $\frac{1}{6}h^2s''(\alpha) = b_0 - 2b_1 + b_2 = y_0 - a_0$ und nach (2.2.10) $hs'(\alpha) = 3(b_1 - b_0) = 2a_0 + a_1 - 3y_0$. Die Bézier-Koeffizienten $\{b_\nu\}$ ergeben sich aus den $\{a_j\}$ nach (2.2.10).

Satz 2.2.3 Gegeben sei ein Gitter Δ , (2.2.2), mit konstanter Schrittweite h . Dann existiert ein eindeutiger Interpolationsspline zu Definition 2.2.2 (Abkürzung $y_i = f(x_i)$, $y'_i = f'(x_i)$). Für den durch (2.2.11), (2.2.12) bestimmten Koeffizientenvektor a gilt $\|a\|_\infty \leq 3\|y\|_\infty$ bzw. $\|a\|_\infty \leq 3\|y\|_\infty + h \max\{|y'_0|, |y'_n|\}$.

Beweis Die Matrix des Gleichungssystems (2.2.11), (2.2.12) ist eine symmetrische Tridiagonalmatrix

$$A = \begin{pmatrix} \gamma & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & & 1 & 4 & 1 \\ & & & & 1 & \gamma \end{pmatrix}. \quad (2.2.13)$$

Bei der Randbedingung (2.2.4,a) sind a_0, a_n bekannt und die Dimension daher $n - 1$ mit $\gamma = 4$. Das System zu (2.2.4,b) hat dagegen Dimension $n + 1$ mit $\gamma = 2$. In beiden Fällen ist die Matrix A in (2.2.13) regulär und die Norm der Lösung kann folgendermaßen abgeschätzt werden. Denn es gilt

$$4|a_j| - 2\|a\|_\infty \leq |4a_j + a_{j-1} + a_{j+1}| = 6|y_j| \leq 6\|y\|_\infty \quad \forall j = 1, \dots, n-1.$$

Bei Randbedingung a) ist auch $|a_0|, |a_n| \leq \|y\|_\infty$ und da die maximale Komponente von a in einer dieser Ungleichungen auftritt, folgt $\|a\|_\infty \leq \max\{1, \frac{6}{2}\}\|y\|_\infty = 3\|y\|_\infty$. Demnach hat die homogene Gleichung nur die triviale Lösung $a = 0$ und A ist daher regulär. Bei Randbedingung b) hat man in den zusätzlichen Randgleichungen $i \in \{0, n\}$ analog

$$2|a_i| = |3y_i \pm hy'_i - a_{i\pm 1}| \leq 3\|y\|_\infty + h\|y'\|_\infty + \|a\|_\infty.$$

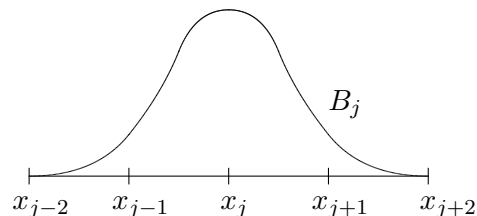
Diese Ungleichung deckt den Fall $\|a\|_\infty = |a_i|, i \in \{0, n\}$ ab und zeigt die Behauptung. ■

Die Eigenschaften der Matrix (2.2.13) werden in § 4 wieder aufgegriffen. Dort zeigt sich auch, dass das Tridiagonalsystem (2.2.11) mit dem Gauß-Algorithmus einfach gelöst werden kann mit einem geringen Aufwand $O(n)$. Die Auswertung eines Splines s in Bézier-Darstellung erfolgt, z.B., mit der Formel (2.2.5), nach (2.2.6a) ist sie eine konvexe Linearkombination (Betragssumme der Vorfaktoren der b_j ist eins) und daher numerisch stabil.

B-Splines: Die wesentlichen Parameter des Splines s sind die Größen a_j . Die Koeffizienten b_i und Funktionswerte $s(x)$ des Splines berechnet man daraus durch lineare Operationen (2.2.10), (2.2.5). Daher existiert eine Basisdarstellung der Form

$$s(x) = \sum_{j=0}^n a_j B_j(x), \quad x \in [\alpha, \beta]. \quad (2.2.14)$$

Für jedes j , $0 \leq j \leq n$, entspricht hier die Funktion B_j dem Spline, den man für den speziellen Koeffizientenvektor $(a_m)_m = (\delta_{jm})_m$ erhält.



Dieser Spline hat im Innern des Intervalls die im Bild gezeigte Gestalt (vgl. die geometrische Interpretation oben). Der Träger von B_j für $2 \leq j \leq n-2$, ist $[x_{j-2}, x_{j+2}]$, umfaßt also $4 = k + 1$ Teilintervalle. In Randnähe ergeben sich etwas unterschiedliche Formen.

Diese Basisfunktionen werden *B-Splines* genannt. Zur Vereinfachung der Definition am Rand erweitert man das Gitter um $2k = 6$ Punkte, bei äquidistanten Punkten ist also $x_i = x_0 + ih$, $i = -3, \dots, n + 3$. In diesem Fall sind die B-Splines verschobene Kopien eines Standardsplines, $B_j(x) = \hat{B}(x - x_j)$, aus dem sie durch Translation hervorgehen. Es gilt die einfache Darstellung mit der gestreckten Variablen $\xi := x/h$

$$\hat{B}(x) = \begin{cases} \frac{1}{6}(\xi + 2)^3 & \xi \in (-2, -1] \\ \frac{2}{3} - \xi^2 - \frac{1}{2}\xi^3 & \xi \in (-1, 0], \\ \frac{2}{3} - \xi^2 + \frac{1}{2}\xi^3 & \xi \in (0, 1], \\ \frac{1}{6}(2 - \xi)^3 & \xi \in (1, 2], \\ 0 & \xi \notin (-2, 2]. \end{cases} \quad (2.2.15)$$

B-Splines können auch auf beliebigen Gittern definiert werden. Auch diese allgemeinen B-Splines bilden eine sogenannte *lokale Zerlegung der Eins* (vgl. auch (2.2.6,a)) mit Hilfe derer wichtige Beweisprinzipien einsetzbar sind. Die Eigenschaften dieser Zerlegung enthält der folgende, für (2.2.15) elementar nachprüfbare, Satz.

Satz 2.2.4 *Die B-Splines bilden eine lokale Zerlegung der Eins, es gilt:*

$$\begin{aligned} B_i(x) &= 0 \quad \forall x \notin (x_{i-2}, x_{i+2}), \\ B_i(x) &> 0 \quad \forall x \in (x_{i-2}, x_{i+2}), \\ \sum_{i=-1}^{n+1} B_i(x) &= \sum_{i=j-1}^{j+2} B_i(x) \equiv 1, \text{ wenn } x \in (x_j, x_{j+1}], \quad 0 \leq j < n. \end{aligned} \quad (2.2.16)$$

Die Eigenschaften (2.2.16) legen umgekehrt den B-Spline B_i eindeutig fest, er stimmt daher mit dem aus (2.2.15) überein. Für einen Spline in B-Spline-Darstellung kann ein stabiles Rekursionschema angegeben werden, wobei bei jeder Auswertung höchstens 4 Koeffizienten a eingehen (vgl. Stoer, §2.4.5).

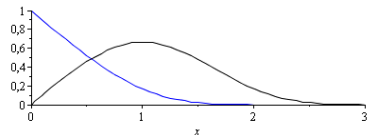
Beispiel 2.2.5 Zur Darstellung von Interpolationssplines zu Definition 2.2.2 kann man die Standard-B-Splines aus (2.2.15) einfach modifizieren. Man betrachtet den ergänzten Ansatz $s(x) = \sum_{j=-1}^{n+1} a_j B_j(x)$ und hat bei natürlichen Randbedingungen am linken Rand die Bedingung

$$0 \stackrel{!}{=} s''(x_0+) = a_{-1} B_{-1}''(x_0+) + a_0 B_0''(x_0+) + a_1 B_1''(x_0+) = \frac{1}{h^2}(a_{-1} - 2a_0 + a_1).$$

Damit eliminiert man $a_{-1} = 2a_0 - a_1$ und erhält

$$s(x) = a_0 \underbrace{(\hat{B}(x - x_0) + 2\hat{B}(x - x_{-1}))}_{=: \tilde{B}_0(x)} + a_1 \underbrace{(\hat{B}(x - x_1) - \hat{B}(x - x_{-1}))}_{=: \tilde{B}_1(x)} + a_2 \hat{B}(x - x_2) + \dots$$

So bekommt man die minimale Darstellung (2.2.14) mit den Basisfunktionen \tilde{B}_0, \tilde{B}_1 , welche die natürliche Randbedingung $\tilde{B}_j''(x_0+) = 0$ schon erfüllen (Bilder rechts).



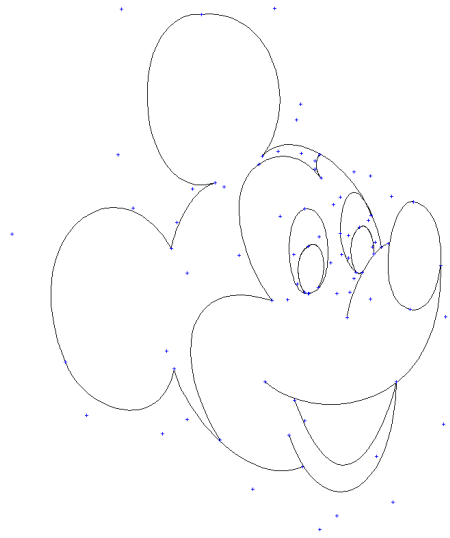
B-Spline- und Bézier-Darstellung bieten **praktische Vorteile** bei der Handhabung, die darauf beruhen, dass die benutzten Basisfunktionen die erwähnte *lokale Zerlegung der Eins* bilden:

Der Funktionswert $s(x)$ an einer beliebigen Stelle x im Intervall ist eine **konvexe** Linearkombination von $k + 1 = 4$ Koeffizienten.

Konsequenzen aus der Eigenschaft "Zerlegung der Eins":

- Bei Auswertung des Splines treten keine zusätzlichen Rundungsfehler auf, da sogar $\sum |B_i(x)| \equiv 1$ gilt: $|\sum_j a_j B_j - \sum_j \tilde{a}_j B_j| \leq \max_j |a_j - \tilde{a}_j|$.
- Die Koeffizienten schließen, als Punkte der Ebene betrachtet, den Funktionsgraphen ein. Graphik-Algorithmen wie Skalierung, Clipping (Abschneiden bei Fenstern, Verdeckung) können eine Vorentscheidung anhand der Koeffizienten treffen.
- Bei Koordinatentransformationen (Zoom) sind alle Koeffizienten gleich zu behandeln.
- Änderungen einzelner Koeffizienten haben nur *lokale* Änderungen des Splines zur Folge (\rightarrow interaktiver Entwurf, CAD).

Beispiel 2.2.6 \mathbb{R}^2 -Graphik aus kubischen Bézier-Splines mit Kontrollpunkten:



Konvergenz bei Splinefunktionen

Bisher wurde nur die praktische Handhabung von Splinefunktionen behandelt. Für den Fehler zwischen dem kubischen Interpolations-Spline aus Def. 2.2.2 und der interpolierten Funktion f kann man nach der Vorüberlegung das Verhalten $O(h^4)$, $h \rightarrow 0$, erwarten. Dies gilt aber nur bei Randbedingung Def. 2.2.2 b). Bei den Interpolationssplines konvergieren sogar noch die Ableitungen gegen die der Funktion. Der Nachweis erfordert folgenden Hilfssatz, der das Maximum einer an zwei Stellen "eingespannten" Funktion durch ihre Krümmung abschätzt.

Satz 2.2.7 *Verschwindet eine Funktion $g \in C^2[\alpha, \beta]$ in den Randpunkten, $g(\alpha) = g(\beta) = 0$, gilt*

$$\|g\|_\infty \leq \frac{(\beta - \alpha)^2}{8} \|g''\|_\infty, \quad \|g'\|_\infty \leq (\beta - \alpha) \|g''\|_\infty.$$

Beweis Taylorentwicklung um $x \in (\alpha, \beta)$ ergibt

$$\begin{aligned} 0 = g(\alpha) &= g(x) + (\alpha - x)g'(x) + \frac{1}{2}(\alpha - x)^2g''(\xi_1), \quad \xi_1 \in (\alpha, \beta), \\ 0 = g(\beta) &= g(x) + (\beta - x)g'(x) + \frac{1}{2}(\beta - x)^2g''(\xi_2), \quad \xi_2 \in (\alpha, \beta). \end{aligned}$$

Multiplikation der Gleichungen mit $(\beta - x)$ bzw. $(x - \alpha)$ und Addition eliminiert $g'(x)$ und man erhält

$$0 = (\beta - x + x - \alpha)g(x) + \frac{1}{2}(\beta - x)(x - \alpha)[(x - \alpha)g''(\xi_1) + (\beta - x)g''(\xi_2)].$$

Daraus folgt dann die erste Ungleichung,

$$|g(x)| \leq \frac{(\beta - x)(x - \alpha)}{2(\beta - \alpha)} (|x - \alpha| + |\beta - x|) \|g''\|_\infty \leq \frac{(\beta - \alpha)^2}{8} \|g''\|_\infty.$$

Die zweite Ungleichung folgt aus dem Satz von Rolle, da ein $x_0 \in [\alpha, \beta]$ existiert mit $g'(x_0) = 0$. Der Mittelwertsatz zeigt dann

$$|g'(x)| = |g'(x_0) + (x - x_0)g''(\xi)| = |x - x_0| \|g''(\xi)\| \leq (\beta - \alpha) \|g''\|_\infty. \quad \blacksquare$$

Gute Konvergenzaussagen zum Interpolationsspline gibt es nur bei Randbedingung b), sie werden im folgenden Satz für äquidistantes Gitter Δ formuliert. Ein ähnliches Ergebnis gilt aber auch für beliebige Gitter. Da die Fehlerschranke für die Funktionswerte vereinfacht die Form $\|s - f\| = O(h^4)$ ($h \rightarrow 0$) hat, sagt man, der Spline konvergiere mit *Ordnung 4*.

Satz 2.2.8 Gegeben sei $f \in C^4[\alpha, \beta]$ und ein Gitter Δ mit konstanter Weite h . Dann gelten für den Interpolationsspline $s \in S_3^2$, der die Bedingungen (2.2.4-b) erfüllt,

$$s(x_j) = f(x_j), \quad j = 0, \dots, n, \quad s'(x_0) = f'(x_0), \quad s'(x_n) = f'(x_n),$$

und seine Ableitungen mit (von h und f unabhängigen) Konstanten $c_i > 0$ die Fehlerschranken

$$\|s^{(i)} - f^{(i)}\|_\infty \leq c_i h^{4-i} \|f^{(4)}\|_\infty, \quad 0 \leq i \leq 3.$$

Beweis Die Funktionswerte werden mit $f_j := f(x_j)$ abgekürzt. Nach (2.2.9) ist $\mu_j := \frac{1}{6}s''(x_j) = \frac{1}{h^2}(b_{3j} - 2b_{3j+1} + b_{3j+2}) = \frac{1}{h^2}(f_j - a_j)$. Der Beweis beruht auf der Tatsache, dass diese Momente μ_j gute Näherungen der zweiten Ableitung $\hat{\mu}_j := \frac{1}{6}f''(x_j)$ der Funktion f sind. Daher wird zusätzlich zu s eine weitere Splinefunktion $\hat{s} \in S_3^0$ eingeführt mit den Koeffizienten

$$\hat{a}_j = f_j - h^2 \hat{\mu}_j, \quad \hat{\mu}_j := \frac{1}{6}f''(x_j), \quad j = 0, \dots, n. \quad (2.2.17)$$

Diese erfüllt die Bedingungen $\hat{s}(x_j) = f(x_j)$, $\hat{s}''(x_j) = f''(x_j)$, $j = 0, \dots, n$, hat also stetige Funktionswerte und 2. Ableitungen. Daher gilt (2.2.10) auch für die Koeffizienten von \hat{s} , also

$$\hat{b}_{3j+1} = \frac{1}{3}(2\hat{a}_j + \hat{a}_{j+1}), \quad \hat{b}_{3j+2} = \frac{1}{3}(\hat{a}_j + 2\hat{a}_{j+1}).$$

Im Vergleich mit den Koeffizienten von s folgt

$$|b_{3j+\nu} - \hat{b}_{3j+\nu}| \leq \|a - \hat{a}\|_\infty, \quad \nu = 1, 2. \quad (2.2.18)$$

In der Bézierdarstellung (2.2.5) in $(x_j, x_{j+1}]$,

$$s(x) - \hat{s}(x) = \frac{3}{h^3} \sum_{\nu=1}^2 (x - x_j)^\nu (x_{j+1} - x)^{3-\nu} (b_{3j+\nu} - \hat{b}_{3j+\nu}),$$

ergibt sich daher für die Ableitungen der Differenz $s - \hat{s}$ die Schranke

$$\begin{aligned} |s^{(i)}(x) - \hat{s}^{(i)}(x)| &\leq \frac{3}{h^3} \sum_{\nu=1}^2 \left| \frac{d^i}{dx^i} [(x - x_j)^\nu (x_{j+1} - x)^{3-\nu}] \right| \max_{\nu} |b_{3j+\nu} - \hat{b}_{3j+\nu}| \\ &\leq \frac{\tilde{c}_i}{h^i} \max_{\nu} |b_{3j+\nu} - \hat{b}_{3j+\nu}|, \end{aligned}$$

mit $\tilde{c}_0 = \frac{3}{4}$, $\tilde{c}_1 = 3$, $\tilde{c}_2 = 18$, $\tilde{c}_3 = 36$. Mit dieser Aussage und (2.2.18) wurde also bisher gezeigt

$$\|s^{(i)} - \hat{s}^{(i)}\|_{\infty} \leq \tilde{c}_i h^{-i} \|a - \hat{a}\|_{\infty}. \quad (2.2.19)$$

Die Koeffizienten a_j des Interpolationssplines stammen aus dem Gleichungssystem $Aa = r$, (2.2.11), (2.2.12). Für die Differenz $a - \hat{a}$ gilt daher $A(a - \hat{a}) = r - A\hat{a}$. Da A nach Satz 2.2.3 regulär ist, ist also zu zeigen, dass der Defekt $r - A\hat{a}$ sehr klein ist. Für $1 \leq j < n$ ist

$$\begin{aligned} r_j - (A\hat{a})_j &= 6f_j - (f_{j-1} + 4f_j + f_{j+1}) + h^2(\hat{\mu}_{j-1} + 4\hat{\mu}_j + \hat{\mu}_{j+1}) \\ &= -f_{j-1} + 2f_j - f_{j+1} + h^2(\hat{\mu}_{j-1} + 4\hat{\mu}_j + \hat{\mu}_{j+1}) \end{aligned} \quad (2.2.20)$$

Bei Taylorentwicklung um x_j entfallen bei $f_{j-1} + f_{j+1}$ und $\hat{\mu}_{j-1} + \hat{\mu}_{j+1}$ Anteile mit ungeraden h -Potenzen und man erhält für die auftretenden Ausdrücke

$$\begin{aligned} f_{j-1} - 2f_j + f_{j+1} &= f_j + \frac{1}{2}h^2 f_j'' + \frac{1}{24}h^4 f^{(4)}(\xi_1) - 2f_j + f_j + \frac{1}{2}h^2 f_j'' + \frac{1}{24}h^4 f^{(4)}(\xi_1) \\ &= h^2 f''(x_j) + \frac{1}{24}h^4 (f^{(4)}(\xi_1) + f^{(4)}(\xi_2)), \\ \mu_{j-1} + 4\mu_j + \mu_{j+1} &= \frac{1}{6} \left(f_j'' + \frac{1}{2}h^2 f^{(4)}(\xi_3) + 4f_j'' + f_j'' + \frac{1}{2}h^2 f^{(4)}(\xi_4) \right) \\ &= f''(x_j) + \frac{1}{12}h^2 (f^{(4)}(\xi_3) + f^{(4)}(\xi_4)), \end{aligned} \quad (2.2.21)$$

mit geeigneten Zwischenstellen ξ_ℓ . Die Defekte (2.2.20) sind daher beschränkt durch $\frac{1}{4}h^4 \|f^{(4)}\|_{\infty}$. In den Randgleichungen gelten analoge Ergebnisse und mit Satz 2.2.3 folgt $\|a - \hat{a}\|_{\infty} = \|A^{-1}(r - A\hat{a})\|_{\infty} \leq ch^4 \|f^{(4)}\|_{\infty}$. Mit (2.2.19) führt das auf

$$\|s^{(i)} - \hat{s}^{(i)}\|_{\infty} \leq \frac{1}{4} \tilde{c}_i h^{4-i} \|f^{(4)}\|_{\infty}. \quad (2.2.22)$$

Zum Abschluß des Beweises ist nun nur noch zu zeigen, dass eine solche Abschätzung auch für die Differenz $\hat{s} - f$ gilt. Nach Konstruktion verschwinden $\hat{s} - f$ und $\hat{s}'' - f''$ in allen Gitterpunkten x_j , $j = 0, \dots, n$. Daher besitzt in jedem Teilintervall $(x_j, x_{j+1}]$ die Funktion

| | | | |
|---------------|-----------------|-------------------|---------------------|
| $\hat{s} - f$ | $\hat{s}' - f'$ | $\hat{s}'' - f''$ | $\hat{s}''' - f'''$ |
| 2 Nullstellen | 1 Nullstelle | 2 Nullstellen | 1 Nullstelle |

Aus Satz 2.2.7 folgen wegen $s^{(4)} \equiv 0$ rekursiv die Schranken

$$\begin{aligned} |\hat{s}'' - f''| &\leq \frac{1}{8}h^2 \|f^{(4)}\|_{\infty}, & |\hat{s}''' - f'''| &\leq h \|f^{(4)}\|_{\infty} \quad \Rightarrow \\ |\hat{s} - f| &\leq \frac{1}{64}h^4 \|f^{(4)}\|_{\infty}, & |\hat{s}' - f'| &\leq \frac{1}{8}h^3 \|f^{(4)}\|_{\infty}. \end{aligned}$$

Der Gesamtfehler bestimmt sich nun aus der Dreieckungleichung bei $s^{(i)} - f^{(i)} = s^{(i)} - \hat{s}^{(i)} + \hat{s}^{(i)} - f^{(i)}$. ■

Zusammenfassend kann also mit Hilfe des kubischen Interpolationssplines die Approximation einer Funktion $f \in C^4$ mit einem Fehler $O(h^4)$ erfolgen. Der Rechenaufwand wächst linear mit n und wird durch die Lösung des linearen Gleichungssystems (2.2.11) verursacht. Mit geringfügigen Einbußen kann auch dieses System noch eingespart werden. Im Beweis des Konvergenzsatzes wurde ausgenutzt, dass ein Spline mit den explizit gegebenen Koeffizienten (2.2.17) auch nur einen Fehler $O(h^4)$ besitzt. Da die Koeffizienten a_j auch die einer B-Spline-Darstellung (2.2.14) sind, nutzt man dies aus bei der Konstruktion *Lokaler Spline-Approximationen*

$$s(x) := \sum_{j=-1}^{n+1} a_j(f) B_j(x), \quad (2.2.23)$$

auf einem erweiterten Gitter, wobei $a_j(f)$ explizit mit Hilfe von f gegeben ist. Bei der Analyse spielt die Eigenschaft aus Satz 2.2.4 (*lokale Zerlegung der Eins*) eine zentrale Rolle. Als Beispiel dafür wird die einfache Approximation mit $a_i(f) = f(x_i)$ betrachtet.

Satz 2.2.9 *Die Funktion f sei Lipschitz-stetig auf \mathbb{R} mit der Lipschitz-Konstanten L . Mit den Eigenschaften (2.2.16) der Basis-Funktionen B_j gilt für den Spline*

$$s(x) := \sum_{j=-1}^{n+1} f(x_j) B_j(x)$$

im Intervall $[\alpha, \beta] = [x_0, x_n]$ die Konvergenzaussage

$$\|s - f\|_\infty \leq 2hL.$$

Beweis Für $x \in (x_j, x_{j+1}]$ ist $1 = \sum_{i=j-1}^{j+2} B_i(x)$ und daher

$$|s(x) - f(x)| = \left| \sum_{i=j-1}^{j+2} (f(x_i) - f(x)) B_i(x) \right| \leq \max_{|x-y| \leq 2h} |f(x) - f(y)| \leq 2hL. \quad \blacksquare$$

Für eine bessere Approximation von f kann man aber bei den Werten (2.2.17) die zweite Ableitung gemäß (2.2.21) durch Differenzen ersetzen und bekommt dann

$$a_j(f) := f(x_j) - \frac{h^2}{3} f[x_{j-1}, x_j, x_{j+1}] = \frac{1}{6} \left(-f(x_{j-1}) + 8f(x_j) - f(x_{j+1}) \right). \quad (2.2.24)$$

Dieser Spline interpoliert zwar die Werte $f(x_j)$ *nicht* mehr (exakt), der Aufwand zur Berechnung ist jetzt aber konstant, unabhängig von n . Aus (2.2.15) folgt außerdem für die Funktionswerte dieses lokalen Splines

$$\begin{aligned} s(x_j) &= \frac{1}{6} (a_{j-1}(f) + 4a_j(f) + a_{j+1}(f)) \\ &= f(x_j) - \frac{1}{36} (f_{j-2} - 4f_{j-1} + 6f_j - 4f_{j+1} + f_{j+2}) \\ &= f(x_j) - \frac{2}{3} h^4 f[x_{j-2}, x_{j-1}, x_j, x_{j+1}, x_{j+2}]. \end{aligned}$$

Da für $f \in C^4$ die dividierte Differenz $f[..]$ beschränkt ist für $h \rightarrow 0$, ist die Abweichung dieses lokalen Splines von den Funktionswerten ebenfalls nur $O(h^4)$. Es werden dabei 4 zusätzliche Funktionswerte außerhalb des Gitters x_0, \dots, x_n verwendet.

3 Rechnerarithmetik und Kondition

Das grundsätzliche Problem bei numerischen Rechnungen ist, dass in einem Computer nur endlich viele Zahldarstellungen (Maschinenzahlen) zur Approximation reeller Zahlen zur Verfügung stehen. Bei (fast) jeder Daten-Eingabe und -Verknüpfung treten daher Fehler auf, die das Endergebnis verfälschen. Zur quantitativen Beschreibung der damit verbundenen Risiken wurde der Begriff der *Kondition* eines Problems entwickelt.

3.1 Zahldarstellung und Rundungsfehler

Die gängige (Dezimal-) Darstellung reeller Zahlen, z.B. $e=2.71828\dots$, entspricht der Angabe der Koeffizienten in der Reihen-Entwicklung

$$e = 2 \cdot 10^0 + 7 \cdot 10^{-1} + 1 \cdot 10^{-2} + 8 \cdot 10^{-3} + \dots$$

zur *Basis* 10. Auf Computern ist eine Zahldarstellung günstiger zu einer Basis B , die eine Zweierpotenz ist, z.B., $B = 2$, $B = 16$. Jede reelle Zahl x ist darstellbar in der Form

$$x = \pm(x_m B^m + x_{m-1} B^{m-1} + \dots + x_1 B + x_0 + x_{-1} B^{-1} + \dots), \quad (3.1.1)$$

mit $x_j \in \{0, 1, \dots, B-1\}$, wenn $1 < B \in \mathbb{N}$. Diese Darstellung ist nur dann eindeutig, wenn von zwei möglichen Darstellungen die endliche gewählt wird (man beachte: $1.999\dots = 2$). Um bei der Zahldarstellung mit einer endlichen Zahlmenge M möglichst viele Größenordnungen abdecken zu können, wird die *Gleitpunktdarstellung* verwendet ("floating point representation"). Dann läßt sich nämlich eine Abbildung $\mathbb{R} \rightarrow M$ mit kleinem *relativem* Fehler konstruieren dadurch, dass man den höchste Exponent m und die ersten ℓ Ziffern $x_m, \dots, x_{m-\ell+1}$ angibt. Der zulässige Exponentenbereich muß dabei natürlich auch beschränkt werden. Außerdem ist die Standardisierung üblich, dass die erste (Nachkomma-) Stelle nicht null ist:

Definition 3.1.1 Die Menge M der normalisierten Gleitpunktzahlen enthält das Element $x = 0$ und die Zahlen

$$x = \pm 0.\xi_1 \xi_2 \dots \xi_\ell \cdot B^d := \pm(\xi_1 B^{-1} + \xi_2 B^{-2} + \dots + \xi_\ell B^{-\ell}) B^d, \quad (3.1.2)$$

mit $\xi_j \in \{0, 1, \dots, B-1\}$, $\xi_1 \neq 0$, $d \in \mathbf{Z}$, $|d| < e$.

Die Zahl $\xi_1 \xi_2 \dots \xi_\ell$ heißt *Mantisse*, für unsere Analysen wird ein unbeschränkter Exponentenbereich ($e = \infty$) angenommen.

Beispiel 3.1.2 Binärdarstellung, $B = 2$. Da wegen der Normalisierung $\xi_1 \neq 0$ gilt, braucht der Wert $\xi_1 = 1$ nicht gespeichert zu werden. Im IEEE-Standard wird eine *real*-Zahl (double precision) in 64 Bit gespeichert mit 11 Bit Exponent und 52(+1) Bit Mantisse. Der Exponent wird verschoben, mit der Binärdarstellung $d + e = d_{10}..d_1 d_0$ ist die Darstellung

$$\pm 0.\xi_1 \xi_2 \dots \xi_\ell \cdot B^d \quad \mapsto \quad \boxed{\pm d_{10}..d_4} \mid \boxed{d_3..d_0 \xi_2.. \xi_5} \mid \boxed{\xi_6 \dots \xi_{13}} \mid \dots \mid \boxed{\dots \xi_\ell}$$

(Die Speicherung dieser 8 Byte in PCs erfolgt aber in umgekehrter Reihenfolge) Hier gilt also $\ell = 53, e = 1023$. Die Zahlgenauigkeit beträgt ca. 15 Dezimalstellen, die Größe der Zahlen ist durch $2^d \cong 10^{308}$ beschränkt.

Zentraler Bestandteil jeder Computerarithmetik ist die Zuordnung von reellen Zahlen zu Gleitpunktzahlen, also die Abbildung

$$rd: \begin{cases} \mathbb{R} & \mapsto M \\ x & \mapsto rd(x) \end{cases}, \text{ die Rundung.}$$

Eine triviale Variante dieser Rundung ist das Abschneiden der Darstellung (3.1.1): $\xi_1 \dots \xi_\ell = x_m \dots x_{m-\ell+1}$. Günstiger ist aber die gewöhnliche Rundung, bei der die Funktion rd jeweils auf die nächstgelegene Maschinenzahl abbildet. Bei Dezimalzahlen etwa wird ab 5 aufwärts gerundet, allgemein wird mit $x_1 \neq 0$ definiert.

$$rd(\pm 0.x_1 x_2 \dots x_\ell x_{\ell+1} \dots \cdot B^d) := \begin{cases} \pm 0.x_1 x_2 \dots x_\ell \cdot B^d & \text{falls } x_{\ell+1} < B/2 \\ \pm(0.x_1 x_2 \dots x_\ell + B^{-\ell}) \cdot B^d & \text{falls } x_{\ell+1} \geq B/2 \end{cases}. \quad (3.1.3)$$

Bei dieser Rundung gilt für den *relativen Fehler* folgende Aussage.

Satz 3.1.3 Für $x \neq 0$ gilt mit (3.1.3)

$$\left| \frac{rd(x) - x}{x} \right| \leq \frac{1}{2} B^{1-\ell} =: \mathcal{E} \quad \iff \quad rd(x) = x(1 + \epsilon), \quad |\epsilon| \leq \mathcal{E}.$$

Die Größe \mathcal{E} heißt *Maschinengenauigkeit*, bei $B = 2$ ist $\mathcal{E} = 2^{-\ell}$.

Beweis In (3.1.3) ist $|x| \geq B^{d-1}$. In beiden Fällen gilt dort

$$|rd(x) - x| \leq \frac{B}{2} B^{-\ell-1} B^d \leq \frac{1}{2} B^{1-\ell} B^{d-1} \leq |x| \cdot \mathcal{E}. \quad \blacksquare$$

Im IEEE-Standard von oben ist $\mathcal{E} = 2^{-53} \cong 10^{-16}$. Da das Ergebnis der Verknüpfung von zwei Maschinenzahlen i.a. nicht wieder eine solche ist, werden bei jeder Verknüpfung neue Rundungsfehler erzeugt. Daher muß man bei der Analyse Maschinenoperationen von reellen Verknüpfungen unterscheiden:

Definition 3.1.4 Für $a, b \in M$ und $*$ $\in \{+, -, \cdot, /\}$ bezeichne (mit $b \neq 0$ im Fall der Division)

$$a \tilde{*} b = gl(a * b)$$

das Ergebnis der entsprechenden (geräteabhängigen) Gleitpunktoperation.

Der IEEE-Standard für Rechner schreibt vor, dass der Fehler bei diesen Operationen den Rundungsfehler des korrekten Ergebnisses nicht übersteigt. Dies kann durch Verwendung eines Hilfsregisters (“Akkumulator”) mit doppelter Stellenzahl 2ℓ garantiert werden. Dort wird die Operation mit 2ℓ Stellen ausgeführt und dann auf ℓ Stellen gerundet. Daher gilt für die Gleitpunktoperationen $*$ $\in \{+, -, \cdot, /\}$ mit $a, b \in M$ ($b \neq 0$ bei Division) die Beziehung

$$a \tilde{*} b = (a * b)(1 + \delta), \quad |\delta| \leq \mathcal{E} = \frac{1}{2} B^{1-\ell}. \quad (3.1.4)$$

Die wichtigste Konsequenz der Rundung ist der Verlust der *Assoziativität* bei Gleitpunktoperationen. Eine wichtige Aufgabe der Fehleranalyse ist daher die Identifikation *günstiger* Anordnungen bei mehrfachen Verknüpfungen. Vor der Entwicklung einer allgemeinen Begriffsbildung werden einige elementare Spezialfälle diskutiert.

Beispiel 3.1.5 $B = 10$, $\ell = 3$, $a = 0.721$, $b = 0.457_{102}$, $c = -0.456_{102}$. Die beiden Varianten

$$\begin{aligned} (a\tilde{+}b)\tilde{+}c &= (0.464_{102})\tilde{+}(-0.456_{102}) = 0.800_{100}, \\ a\tilde{+}(b\tilde{+}c) &= 0.721\tilde{+}(0.1) = 0.821_{100}. \end{aligned}$$

liefern offensichtlich verschiedene Ergebnisse.

Der in der ersten Version im Beispiel beobachtete Effekt wird *Auslöschung* genannt: Besitzen zwei Zahlen $a, b \in M$ gleiche Exponenten und gleiche führende Ziffern, so ist bei Subtraktion das Ergebnis *exakt*! Sind in den Operanden aber *alte* Fehler vorhanden, werden diese wegen der Betrachtung relativer Fehler sehr verstärkt, da dann $|a - b| \ll |a| + |b|$ ist in (3.1.4). Dies muß bei der Fehlerfortpflanzung besonders beachtet werden.

Als Anwendung wird zunächst die Bildung mehrfacher Produkte und Summen betrachtet,

$$w_n := a_1 * a_2 * \dots * a_n, \quad n \in \mathbb{N}, \quad (3.1.5)$$

mit $* \in \{+, \cdot\}$. Das numerische Ergebnis \widetilde{w}_n werde dabei sequentiell berechnet mit gerundeten Werten \tilde{a}_j ,

$$\widetilde{w}_n := gl(\dots gl(gl(\tilde{a}_1 * \tilde{a}_2) * \tilde{a}_3) \dots) = gl(\widetilde{w}_{n-1} * \tilde{a}_n), \quad (3.1.6)$$

wobei $\tilde{a}_i = a_i(1 + \delta_i)$, $|\delta_i| \leq \mathcal{E}$ ist. Für die weiteren relativen Fehler gelte ebenfalls $|\epsilon_i| \leq \mathcal{E}$.

Produkt: Hier gilt induktiv

$$\begin{aligned} \widetilde{w}_2 &= gl(\tilde{a}_1 \cdot \tilde{a}_2) = gl(a_1(1 + \delta_1)a_2(1 + \delta_2)) = a_1a_2(1 + \delta_1)(1 + \delta_2)(1 + \epsilon_2) \\ &\dots \\ \widetilde{w}_n &= gl(\widetilde{w}_{n-1} \cdot \tilde{a}_n) = \widetilde{w}_{n-1} \cdot a_n(1 + \delta_n)(1 + \epsilon_n) \\ &= a_1a_2 \dots a_n(1 + \delta_1) \dots (1 + \delta_n)(1 + \epsilon_2) \dots (1 + \epsilon_n) = w_n(1 + \gamma_n) \end{aligned}$$

mit

$$(1 - \mathcal{E})^{2n-1} \leq 1 + \gamma_n \leq (1 + \mathcal{E})^{2n-1}. \quad (3.1.7)$$

Ausdrücke dieser Form treten jetzt öfter auf. Nach dem Mittelwertsatz gilt $(1 + x)^m = 1 + mx(1 + \xi)^{m-1}$ mit $|\xi| \leq |x|$. Daraus folgt

$$|(1 + x)^m - 1| \leq m|x|(1 + |x|)^{m-1} \leq m|x|e^{(m-1)|x|} \leq 1.06m|x| \quad \text{für } (m-1)|x| \leq 0.05. \quad (3.1.8)$$

Daher gilt für die Produktbildung in (3.1.7) für γ_n die Aussage

$$|\gamma_n| \leq 1.06(2n-1)\mathcal{E}, \quad \text{falls } 2(n-1)\mathcal{E} \leq 0.05. \quad (3.1.9)$$

Summe: Bei der Addition ist die Situation komplizierter, hier ist

$$\begin{aligned}
\widetilde{w}_2 &= gl(\widetilde{a}_1 + \widetilde{a}_2) = (a_1(1 + \delta_1) + a_2(1 + \delta_2))(1 + \epsilon_2) \\
&= a_1(1 + \delta_1)(1 + \epsilon_2) + a_2(1 + \delta_2)(1 + \epsilon_2) = a_1(1 + \sigma_{21}) + a_2(1 + \sigma_{22}) \\
&\dots \\
\widetilde{w}_n &= gl(\widetilde{w}_{n-1} + \widetilde{a}_n) = (\widetilde{w}_{n-1} + a_n(1 + \delta_n))(1 + \epsilon_n) \\
&= \left(a_1(1 + \sigma_{n-1,1}) + \dots + a_{n-1}(1 + \sigma_{n-1,n-1}) \right) (1 + \epsilon_n) + a_n(1 + \delta_n)(1 + \epsilon_n) \\
&= a_1(1 + \sigma_{n1}) + a_2(1 + \sigma_{n2}) + \dots + a_n(1 + \sigma_{nn}),
\end{aligned}$$

woraus sich $(1 + \sigma_{ni}) = (1 + \delta_i)(1 + \epsilon_i) \dots (1 + \epsilon_n)$ ergibt. Aus (3.1.8) folgt daher

$$|\sigma_{ni}| \leq 1.06(n - i + 2)\mathcal{E}, \quad \text{wenn } (n - i + 1)\mathcal{E} \leq 0.05. \quad (3.1.10)$$

Bemerkung: Bei der Produktbildung ist in der Fehlerschranke keine Bevorzugung einer bestimmten Anordnung der Operanden erkennbar. Bei der Summe dagegen liefern die ersten Summanden den größten relativen Fehlerbeitrag. Daher ist zu erwarten, dass der absolute Fehler in \widetilde{w}_n kleiner wird, wenn zuerst die *betragsskleinsten* Koeffizienten a_i summiert werden, da dann die Schranke

$$|\widetilde{w}_n - w_n| \leq 1.06\mathcal{E} \sum_{i=1}^n (n - i + 2)|a_i|$$

minimal wird. Daher sollte man, z.B., bei der Summation (monoton) konvergenter Reihen mit den letzten, d.h. kleinsten, Koeffizienten beginnen (*Rückwärts-Summation*). Ohne Vorkenntnis der Größenverhältnisse bekommt man günstigere Fehler durch Zweier-Summation, wobei rekursiv jeweils Paare von Koeffizienten addiert werden (Übung).

3.2 Konditionszahlen für Algorithmen

In der Abschätzung (3.1.10) läßt sich erkennen, wie sich einzelne Summanden (Eingabegrößen) auf den Fehler auswirken. Allerdings betrifft diese Schranke den absoluten Fehler im Ergebnis, während für die Gleitpunktdarstellung der relative Fehler von Interesse ist. Im folgenden werden für allgemeine Rechenvorschriften analoge Kenngrößen (Konditionszahlen) hergeleitet, welche die relative Empfindlichkeit eines Ergebnisses gegenüber Störungen der Eingangsgrößen beschreiben. Wegen der geringen Größe der Maschinengenauigkeit \mathcal{E} vernachlässigt man dabei Produkte von Maschinenfehlern $\mathcal{O}(\mathcal{E}^2)$ und schreibt $\mathcal{E} + \mathcal{O}(\mathcal{E}^2) \doteq \mathcal{E}$, ähnlich zu (3.1.8), wo diese Anteile durch die Konstante 1.06 aufgefangen wurden: $\mathcal{E} + \mathcal{O}(\mathcal{E}^2) \leq 1.06\mathcal{E}$.

Einen numerischen Algorithmus kann man als eine Folge von Operationen bzw. Abbildungen $F = \phi^{[N]} \circ \dots \circ \phi^{[2]} \circ \phi^{[1]}$ modellieren, die auf einem Vektor von Eingabedaten $x = (x_1, \dots, x_n)^\top \in \mathbb{R}^n$ operieren. Betrachtet man nur einen Ausgabewert, gilt also $F : \mathbb{R}^n \rightarrow \mathbb{R}$. Als ersten Schritt sollte man sich die Empfindlichkeit der Gesamtabbildung F bei Auswertung an einem (etwa

durch Rundungsfehler) leicht veränderten Punkt $\tilde{x} \in \mathbb{R}^n$ ansehen. Wegen der Aussage von Satz 3.1.3 wird dabei von einem kleinen relativen Fehler ausgegangen in der Form

$$\tilde{x}_i = (1 + \epsilon_i)x_i, \quad |\epsilon_i| \ll 1, \quad i = 1, \dots, n. \quad (3.2.1)$$

Für $x_i \neq 0$ ist diese Darstellung äquivalent mit $\epsilon_i = (\tilde{x}_i - x_i)/x_i$. Analog betrachtet man auch im Ergebnis den relativen Fehler $\epsilon_F = (F(\tilde{x}) - F(x))/F(x)$. Vernachlässigt man Produkte von Fehlertermen, kann die Empfindlichkeit des Ergebnisses F gegenüber Störungen einzelner Eingabedaten x_i durch folgende Zahlen charakterisiert werden. Die Vektoren $e^{(i)} \in \mathbb{R}^n$ bezeichnen dabei die Einheitsvektoren.

Definition 3.2.1 Für die Funktion $F : \mathbb{R}^n \rightarrow \mathbb{R}$ werden die Konditionszahlen $\kappa_i \in \mathbb{R}_+$ an der Stelle $x \in \mathbb{R}^n$ mit $F(x) \neq 0$ definiert durch

$$\kappa_i := \limsup_{h \rightarrow 0} \left| \frac{F(x + hx_i e^{(i)}) - F(x)}{hF(x)} \right|, \quad i = 1, \dots, n. \quad (3.2.2)$$

Beispiel 3.2.2 Für die arithmetischen Grundoperationen gelten folgende Zusammenhänge:

$$\begin{aligned} F(x_1, x_2) &:= x_1 x_2, & \epsilon_F &\doteq \epsilon_1 + \epsilon_2, & \kappa_1 &= \kappa_2 = 1, \\ F(x_1, x_2) &:= x_1 / x_2, & \epsilon_F &\doteq \epsilon_1 - \epsilon_2, & \kappa_1 &= \kappa_2 = 1, \\ F(x_1, x_2) &:= x_1 + x_2, & \epsilon_F &\doteq \frac{x_1}{x_1 + x_2} \epsilon_1 + \frac{x_2}{x_1 + x_2} \epsilon_2, & \kappa_1 &= \frac{|x_1|}{|x_1 + x_2|}, \quad \kappa_2 = \frac{|x_2|}{|x_1 + x_2|}. \end{aligned}$$

Nach (3.2.2) liest man die Konditionszahlen an den Vorfaktoren der ϵ_i ab.

Für allgemeine, glatte Funktionen F können die Konditionszahlen über Ableitungen bestimmt werden. Die zu einem Algorithmus gehörige Abbildung F ist in einem Bereich $D \in \mathbb{R}^n$ normalerweise aber nur dann glatt, wenn die auftretenden Fallunterscheidungen für alle $x \in D$ gleich ausfallen.

Satz 3.2.3 Die Funktion $F : \mathbb{R}^n \rightarrow \mathbb{R}$ sei stetig differenzierbar in einer Umgebung der Stelle $x \in \mathbb{R}^n$ mit $F(x) \neq 0$. Dann haben die in (3.2.2) definierten Konditionszahlen die Werte

$$\kappa_i = \left| \frac{x_i}{F(x)} \frac{\partial F}{\partial x_i}(x) \right|, \quad i = 1, \dots, n.$$

Für den relativen Fehler $\epsilon_F = (F(\tilde{x}) - F(x))/F(x)$ bei Auswertung in einem gestörten Argument \tilde{x} , (3.2.1) mit $|\epsilon_i| \leq h$, $i = 1, \dots, n$, gilt dann

$$|\epsilon_F| \leq \sum_{i=1}^n \kappa_i |\epsilon_i| + o(h), \quad h \rightarrow 0. \quad (3.2.3)$$

Anmerkung: Die Ungleichung (3.2.3) ist scharf, da bei geeigneter Vorzeichenverteilung der ϵ_i Gleichheit gilt.

Beweis Wegen der stetigen Differenzierbarkeit von F gilt mit $\tilde{x}_i - x_i = x_i \epsilon_i$ die Darstellung

$$F(\tilde{x}) - F(x) = \sum_{i=1}^n \frac{\partial F}{\partial x_i}(x) x_i \epsilon_i + o(h).$$

Daraus folgt wegen $o(h)/h \rightarrow 0 (h \rightarrow 0)$ durch die spezielle Wahl $\epsilon_j = h\delta_{ij}$, $j = 1, \dots, n$, zunächst die Darstellung der Konditionszahl κ_i und anschließend aus der Dreiecksgleichung die Formel (3.2.3). ■

Im obigen Beispiel zeigt sich, dass die Gleitpunktdarstellung die Punktoperationen begünstigt. Bei Addition/Subtraktion können dagegen im schon erwähnten Fall der *Auslöschung*, nämlich für $|x_1 + x_2| \ll \max\{|x_1|, |x_2|\}$, d.h., bei Subtraktion gleich großer Zahlen, sehr große Konditionszahlen auftreten.

Die in (3.2.2) definierten Konditionszahlen beschreiben die Störungsempfindlichkeit von F komponentenweise. Zur Vereinfachung kann man durch Verwendung von Normen eine einheitliche Konditionszahl definieren. Dies ist insbesondere bei linearen Abbildungen F angebracht und wird im nächsten Abschnitt über Lineare Gleichungssysteme eingeführt.

Bei der fehlerbehafteten Ausführung von Operationen in einem Algorithmus reicht die Betrachtung der Gesamtabbildung F natürlich nicht aus, auch die Einzelschritte müssen überprüft werden. Nicht nur die Assoziativität von Summe und Produkt geht durch Rundungsfehler verloren, auch andere Gesetze gelten im allgemeinen nicht mehr (exakt). Daher können verschiedene, analytisch äquivalente Ausdrücke, wie z.B. die der binomischen Formel $a^2 - b^2 = (a + b)(a - b)$, sehr unterschiedliche numerische Ergebnisse liefern. Als wichtiges Beispiel werde die Funktion

$$F(x) := \sqrt{1 + x^2} - x \quad (3.2.4)$$

betrachtet. Für $x \geq 0$ ist $0 \leq F(x) \leq 1$. Da $|F'(x)| = \left| \frac{x}{\sqrt{1+x^2}} - 1 \right| \leq 1$ ist für $x \geq 0$, führt eine Störung im Argument x bei exakter Rechnung nur zu einer geringen absoluten Verfälschung im Funktionswert F . Auch für die Konditionszahl ergibt sich nach Satz 3.2.3 die harmlose Schranke

$$\kappa_x = \frac{|xF'(x)|}{|F(x)|} = \frac{|x|}{\sqrt{1+x^2}} \leq 1.$$

Werden allerdings die Teilschritte in (3.2.4) nur mit endlicher Genauigkeit ausgeführt, sieht das Ergebnis schlechter aus. Selbst wenn nur bei der Addition in (3.2.4) ein relativer Fehler ϵ , $|\epsilon| \leq \mathcal{E}$, erzeugt wird, gilt

$$\tilde{F}(x) = \sqrt{1 + \tilde{x}^2} - x = \sqrt{(1 + x^2)(1 + \epsilon)} - x = F(x) + \frac{\epsilon}{2}\sqrt{1 + x^2} + \mathcal{O}(\epsilon^2).$$

Für große $x \rightarrow \infty$ ist also ein großer absoluter Fehler $\cong \frac{1}{2}x\epsilon$ zu erwarten, der relative Fehler ist wegen $|F| \leq 1$ noch größer. In der Formel (3.2.4) tritt nämlich bei der Differenzbildung der beiden großen Ausdrücke $\sqrt{1 + x^2} \cong x$ und x eine *Auslöschung* führender Stellen auf. Diese läßt sich vermeiden durch Umformung in die äquivalente Form

$$F(x) = \sqrt{1 + x^2} - x = \frac{1 + x^2 - x^2}{\sqrt{1 + x^2} + x} = \frac{1}{x + \sqrt{1 + x^2}}, \quad x \geq 0. \quad (3.2.5)$$

Für $x < 0$ ist dagegen (3.2.4) die günstigere Form, eine Implementierung von F für $x \in \mathbb{R}$ sollte daher beide Formen mit einer Fallunterscheidung verwenden.

Beispiel 3.2.4 Formel (3.2.4) wird mit 4-ziffriger Dezimalrechnung in $x = 9.999$ ausgewertet, bei exakter Rechnung ist $F(9.999) = 0.04988\dots$ Dagegen ist $\tilde{x} \cdot \tilde{x} = 99.98$, $1 + \tilde{x} \cdot \tilde{x} = 101.0$, $\sqrt{101} =$

10.05 und daher $\tilde{F}(x) = 0.05100$. Dies entspricht einem relativen Fehler von $0.022 \cong 44 \cdot \mathcal{E}$. Die letzte, äquivalente Form in (3.2.5) ergibt dagegen den korrekten 4-stelligen Wert 0.04988. Für $x \geq 100$ schließlich liefert 4-stellige Rechnung in (3.2.4) immer den Wert $\tilde{F}(x) = 0$.

Das Vorgehen beim Ausdruck (3.2.4) soll nun verallgemeinert werden. Bei einer allgemeinen Struktur der Abbildung $F = \phi^{[N]} \circ \dots \circ \phi^{[2]} \circ \phi^{[1]}$ müssen nun Fehler bei jeder einzelnen Auswertung berücksichtigt werden. Um die Darstellung in vertretbarem Rahmen zu halten, wird nur der Fall $N = 2$ mit

$$F = \psi \circ \phi \quad \text{und} \quad \phi : \mathbb{R}^n \rightarrow \mathbb{R}^m, \quad \psi : \mathbb{R}^m \rightarrow \mathbb{R}$$

betrachtet, also die Vorschrift $z := \psi(y)$, $y := \phi(x)$ für $z = F(x)$ mit glatten Funktionen ϕ, ψ . Indizes bei den Funktionen bezeichnen die Komponenten, $\phi = (\phi_1, \dots, \phi_m)^\top$. Das Ergebnis kann dann induktiv auf eine beliebige Zahl von Teilschritten erweitert werden. Statt der exakten Berechnung werden bei Gleitpunktrechnung Werte

$$\tilde{z} = \tilde{\psi}(\tilde{y}), \quad \tilde{y} = \tilde{\phi}(\tilde{x})$$

geliefert. Für die Maschinen-Implementierung $\tilde{\phi}$ wird wie früher die Fehleraussage

$$\tilde{\phi}_i(x) = (1 + \epsilon_i^{(1)})\phi_i(x), \quad i = 1, \dots, m \quad \iff \quad \tilde{\phi}(x) = (I + E_1)\phi(x)$$

angenommen für alle x . Die Verfälschung entspricht also bei der Abbildung $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^m$ der Multiplikation mit einer Diagonalmatrix $I + E_1 = \text{diag}(1 + \epsilon_1^{(1)}, \dots, 1 + \epsilon_m^{(1)})$. Bei $\tilde{\psi}$ gelte $\tilde{\psi}(y) = (1 + \epsilon^{(2)})\psi(y)$. Alle Einzelfehler $\epsilon_i^{(j)}$ seien so klein, dass ihre Produkte vernachlässigbar sind (z.B. $|\epsilon_i^{(j)}| \leq c\mathcal{E}$). Man erhält so mit $\epsilon^{(2)}\psi(\tilde{y}) \doteq \epsilon^{(2)}\psi(y) = \epsilon^{(2)}z$ die Beziehungen

$$\begin{aligned} \tilde{z} - z &= \tilde{\psi}(\tilde{y}) - \psi(y) = \tilde{\psi}(\tilde{y}) - \psi(\tilde{y}) + \psi(\tilde{y}) - \psi(y) \\ &\doteq \epsilon^{(2)}z + \psi'(y)(\tilde{y} - y), \\ \tilde{y} - y &= \tilde{\phi}(\tilde{x}) - \phi(\tilde{x}) + \phi(\tilde{x}) - \phi(x) \doteq E_1 y + \phi'(x)(\tilde{x} - x). \end{aligned}$$

Die Ableitung in der letzten Gleichung ist die Funktionalmatrix $\phi'(x) \in \mathbb{R}^{m \times n}$. Auch für den Startfehler hat man mit (3.2.1) die Darstellung $\tilde{x} - x = E_0 x$ mit einer entsprechenden Diagonalmatrix E_0 . Setzt man die Fehlerdarstellungen ineinander ein, tritt u.a. das Produkt $\psi'(y)\phi'(x)$ auf, das nach der Kettenregel wegen $y = \phi(x)$ gerade die Ableitung

$$F'(x) = \text{grad}(\psi \circ \phi)(x) = \psi'(\phi(x))\phi'(x)$$

darstellt. Daher ergibt sich für den Gesamtfehler die Aussage

$$\tilde{z} - z \doteq F'(x)E_0 x + \psi'(y)E_1 y + \epsilon^{(2)}z. \quad (3.2.6)$$

Diese läßt sich so interpretieren, dass der in einem Zwischenschritt des Algorithmus auftretende Fehler mit der Ableitung der *Restabbildung* multipliziert wird. Für den Startfehler E_0 ist dieses die Gesamtabbildung $F = \psi \circ \phi$, für den Fehler E_1 aber nur noch die Abbildung ψ . Für den relativen Fehler bekommt man mit $z = \psi(y) = F(x)$ die Formel

$$\epsilon_z = \frac{\tilde{z} - z}{z} = \frac{F'(x)}{F(x)} E_0 x + \frac{\psi'(y)}{\psi(y)} E_1 y + \epsilon^{(2)}.$$

Diese Aussage kann als Verallgemeinerung von (3.2.3) geschrieben werden, wenn man die Konditionszahlen aus Satz 3.2.3 auf die Variablen y_i, x_j bezieht:

$$\kappa_{y_i} = \left| \frac{y_i}{\psi(y)} \frac{\partial \psi}{\partial y_i}(y) \right|, \quad \kappa_{x_j} = \left| \frac{x_j}{F(x)} \frac{\partial F}{\partial x_j}(x) \right|.$$

Aus (3.2.6) folgt dann insgesamt

$$|\epsilon_z| \leq \sum_{j=1}^n \kappa_{x_j} |\epsilon_j^{(0)}| + \sum_{i=1}^m \kappa_{y_i} |\epsilon_i^{(1)}| + |\epsilon^{(2)}|. \quad (3.2.7)$$

Auch hier tritt bei einer geeigneten Vorzeichenverteilung Gleichheit auf. Merkgel:

Der im k -ten Schritt eines Algorithmus auftretende Rundungsfehler wirkt sich proportional zur Konditionszahl der *Restabbildung* auf den relativen Endfehler aus

Daher darf also in einem guten numerischen Algorithmus keine einzige Restabbildung eine große Konditionszahl besitzen.

Beispiel 3.2.5 Die in (3.2.4) betrachtete Funktion $F(x)$ ist die positive Lösung der quadratischen Gleichung $F^2 + 2xF - 1 = 0$, in der bei ungünstiger Auswertung also erhebliche Fehler auftreten können. Die Funktion F läßt sich vereinfachend aus den beiden Abbildungen

$$\phi(x) = \begin{pmatrix} 1 + x^2 \\ x \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \quad \psi(y) = \sqrt{y_1} - y_2$$

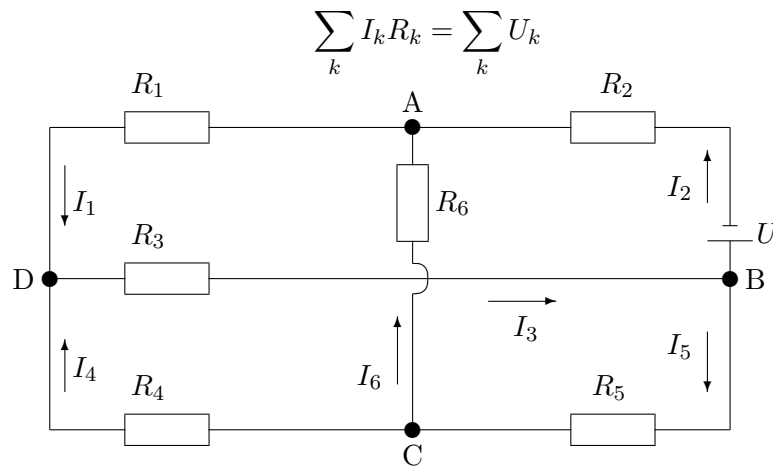
zusammensetzen. Die Konditionszahl κ_x für den Eingangsfehler wurde schon bestimmt, sie ist harmlos, $|\kappa_x| \leq 1$. Die Ableitung von ψ ist $\text{grad}\psi(y) = (\frac{1}{2\sqrt{y_1}}, -1)$. Für die κ_{y_i} ergibt sich daher mit $y = \phi(x)$:

$$\kappa_{y_1} = \frac{|y_1|}{2\psi(y)\sqrt{y_1}} = \frac{\sqrt{1+x^2}}{2F(x)} = \frac{\sqrt{1+x^2}}{2}(\sqrt{1+x^2} + x), \quad \kappa_{y_2} = \frac{|y_2|}{\psi(y)} = |x|(\sqrt{1+x^2} + x).$$

Im beiden Ausdrücken wurde der Kehrwert $1/F$ mit Hilfe von (3.2.5) ersetzt und zeigt das starke Anwachsen $\kappa_{y_i} \rightarrow \infty$, $x \rightarrow +\infty$. Für negative x gilt dagegen $F(x) > \sqrt{1+x^2} > x$, dort sind alle Konditionszahlen durch eins beschränkt und die ursprüngliche Formel (3.2.4) gut konditioniert.

Eine entsprechende, aufwändigere Analyse zeigt das umgekehrte Bild für die Darstellung (3.2.4): diese hat für $x > 0$ eine gute, für $x < 0$ dagegen eine schlechte Kondition. Bei einer Implementierung dieser speziellen Funktion F muss daher durch Fallunterscheidung die jeweils günstigere Form ausgewählt werden.

b) *Maschenregel*: Für jeden geschlossenen Teilstromkreis gilt (unter Berücksichtigung der Richtungen)



Für das abgebildete Netzwerk ergibt sich so das folgende LGS:

$$\begin{array}{l} \text{Knoten} \\ A: \\ B: \\ C: \\ D: \end{array} \begin{array}{r} -I_1 + I_2 + I_6 = 0 \\ -I_2 + I_3 - I_5 = 0 \\ -I_4 + I_5 - I_6 = 0 \\ I_1 - I_3 + I_4 = 0 \end{array} \quad (4.1.3)$$

$$\begin{array}{l} \text{Maschen} \\ ACB: \\ ADB: \\ ADC: \\ BDC: \end{array} \begin{array}{r} I_2 R_2 - I_5 R_5 - I_6 R_6 = U \\ I_1 R_1 + I_2 R_2 + I_3 R_3 = U \\ I_1 R_1 - I_4 R_4 + I_6 R_6 = 0 \\ -I_3 R_3 - I_4 R_4 - I_5 R_5 = 0 \end{array} \quad (4.1.4)$$

Die Gleichungen sind offensichtlich *linear abhängig*, überflüssige sind zu eliminieren. Die Summe der Gleichungen in (4.1.3) verschwindet, auch in (4.1.4) ist eine überflüssig. Damit bleiben sechs Gleichungen für sechs Unbekannte. In Matrixform schreibt sich (4.1.3,4.1.4) also kompakt als

$$\begin{array}{l} A \\ B \\ C \\ ACB \\ ADB \\ ADC \end{array} \begin{pmatrix} -1 & 1 & 0 & 0 & 0 & 1 \\ 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 1 & -1 \\ 0 & R_2 & 0 & 0 & -R_5 & -R_6 \\ R_1 & R_2 & R_3 & 0 & 0 & 0 \\ R_1 & 0 & 0 & -R_4 & 0 & R_6 \end{pmatrix} \begin{pmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ U \\ U \\ 0 \end{pmatrix} \quad (4.1.5)$$

Bemerkung: Ähnliche Strukturen tauchen auch bei anderen Problemen auf, etwa Transportproblemen in Graphen. Die Matrix des Teilsystems (4.1.3) ist die sog. *Inzidenzmatrix* des zum Netzwerk gehörigen Graphen.

Die Numerik befaßt sich mit der effizienten und numerisch stabilen Auflösung solcher Systeme, die bei heute üblichen Anwendungen insbesondere sehr große Dimensionen n aufweisen. Ein Ansatzpunkt ist dabei die Identifikation von Klassen von Matrizen, deren spezielle Struktur effizientere Verfahren ermöglicht (z.B., Definitheit, dünne Besetzung, s.u.).

4.2 Matrizen und Normen

Vektoren im \mathbb{R}^n bzw. \mathbb{C}^n (Sammelbegriff \mathbb{K}^n) werden als Spaltenvektoren betrachtet

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}.$$

Für quantitative Aussagen werden Normen verwendet. Mit $p \in \mathbb{R}$, $p \geq 1$, sind die Hölder-Normen definiert durch

$$\|x\|_p := \left(\sum_{j=1}^n |x_j|^p \right)^{1/p}, \quad (4.2.1)$$

Die wichtigsten Vertreter sind

$$\begin{aligned} \|x\|_1 &= \sum_{j=1}^n |x_j| && \text{Summennorm} \\ \|x\|_2 &= \sqrt{\sum_{j=1}^n |x_j|^2} && \text{Euklidnorm} \\ \|x\|_\infty &= \max_{j=1}^n |x_j|, && \text{Maximumnorm} \end{aligned} \quad (4.2.2)$$

Mit diesen Normen $\|\cdot\|_p$ ist \mathbb{K}^n ein Banachraum. Da alle Normen im \mathbb{K}^n äquivalent sind, existieren Konstanten mit

$$\|x\|_p \leq c_{pq} \|x\|_q \quad \forall x \in \mathbb{K}^n, \quad p, q \geq 1.$$

Diese Konstanten c_{pq} hängen im allgemeinen aber von der Dimension n ab. Lineare Abbildungen von $\mathbb{K}^n \rightarrow \mathbb{K}^m$ werden (in der Einheits-Basis) durch Matrizen

$$A = \left(a_{ij} \right)_{i=1, j=1}^{m, n} \in \mathbb{K}^{m \times n}$$

beschrieben. Jedes Paar von Vektornormen in $\mathbb{K}^m, \mathbb{K}^n$ (mit gleichem Index p) induziert eine zugehörige Matrixnorm, die mit dem gleichen Symbol bezeichnet wird

$$\|A\|_p := \sup_{x \neq 0} \frac{\|Ax\|_p}{\|x\|_p} = \sup_{\|x\|_p=1} \|Ax\|_p. \quad (4.2.3)$$

In (4.2.3) wird das Supremum als Maximum angenommen (Grund?). Die durch (4.2.2) induzierten Normen lassen sich explizit angeben:

$$\begin{aligned} \|A\|_1 &= \max_{j=1}^n \sum_{i=1}^m |a_{ij}| && \text{Spaltensummennorm,} \\ \|A\|_2 &= \varrho(A^*A)^{1/2} && \text{Spektralnorm.} \\ \|A\|_\infty &= \max_{i=1}^m \sum_{j=1}^n |a_{ij}| && \text{Zeilensummennorm,} \end{aligned} \quad (4.2.4)$$

Dabei ist der Spektralradius $\varrho(A^*A) = \max_j \lambda_j(A^*A)$ der maximale Eigenwert von A^*A , wobei $A^* := \bar{A}^T$ ist. Für induzierte Matrixnormen überträgt sich die Dreieckungleichung von der

Vektornorm: $\|A+B\|_p \leq \|A\|_p + \|B\|_p$. Eine Matrixnorm heißt *verträglich* mit einer Vektornorm, wenn

$$\|Ax\| \leq \|A\| \|x\| \quad \forall x \in \mathbb{K}^n, A \in \mathbb{K}^{m \times n}. \quad (4.2.5)$$

Induzierte Normen sind verträglich. Als obere Schranke für $\|A\|_2$ betrachtet man oft eine einfachere, die

$$\|A\|_F := \left(\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2 \right)^{1/2} \quad \text{Frobeniusnorm.} \quad (4.2.6)$$

Sie ist verträglich mit der Euklidnorm, denn aus der Cauchy-Schwarz-Ungleichung folgt

$$\|Ax\|_2 \leq \|A\|_F \|x\|_2.$$

Allerdings ist sie nicht die induzierte Norm, da z.B. $\|I\|_F = \sqrt{n} > \|I\|_2 = 1$ gilt, wobei $I = (\delta_{ij})$ die Einheitsmatrix bezeichnet. Für induzierte Normen und für $\|\cdot\|_F$ gilt die Produktformel

$$\|AB\| \leq \|A\| \|B\| \quad (4.2.7)$$

In dieser Vorlesung werden nur Normen benutzt, die (4.2.7) erfüllen. Wenn es also unterschiedliche Normen gibt, stellt sich die Frage nach einer besten (minimalen) Matrix-Norm. Für quadratische Matrizen gibt es leider nur eine größte untere Schranke (Infimum), die allerdings selbst keine Norm ist. Diese Größe ist für $A \in \mathbb{K}^{n \times n}$ der schon erwähnte *Spektralradius*

$$\varrho(A) := \max\{|\lambda_j(A)| : \lambda_j(A) \text{ EW zu } A\}. \quad (4.2.8)$$

Wegen der fehlenden Definitheit, etwa bei $\varrho\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = 0$, ist $\varrho(A)$ aber sicherlich keine Norm. Der Zusammenhang mit Normen wird im nächsten Satz behandelt. Da die Aussagen von Eigenschaften der Matrix abhängt, werden einige der auch später verwendeten Begriffe und ihr Bezug zur Jordan-Normalform hier zusammengestellt:

| Eigenschaft | Definition | Jordan-Normalform $A = X\Lambda X^{-1}$ |
|---------------------------|---------------------------------------|--|
| A diagonalisierbar | \exists Eigenvektor-Basis | Λ diagonal |
| A normal | $A^*A = AA^*$ | Λ komplex diagonal, X unitär: $X^* = X^{-1}$ |
| A hermitesch | $A^* = A$ | Λ reell diagonal, X unitär |
| A herm. positiv definit | $A^* = A, x^*Ax > 0 \forall x \neq 0$ | Eigenwerte positiv reell |

Satz 4.2.1 a) Für jede Matrixnorm $\|\cdot\|$ gilt $\varrho(A) \leq \|A\|$.

b) Für jede Matrix A und jedes $\varepsilon > 0$ existiert eine (spezielle) Norm mit

$$\varrho(A) \leq \|A\|_M \leq \varrho(A) + \varepsilon. \quad (4.2.9)$$

c) Für diagonalisierbare Matrizen A kann in (4.2.9) $\varepsilon = 0$ gewählt werden, für hermitesche, reell-symmetrische und normale Matrizen gilt insbesondere $\|A\|_2 = \varrho(A)$.

Beweis a) x sei Eigenvektor von A , $\Rightarrow |\lambda|\|x\| = \|\lambda x\| = \|Ax\| \leq \|A\|\|x\|$.

b) Die Jordan-Normalform von A sei

$$\Lambda = X^{-1}AX = \begin{pmatrix} \lambda_1 & \delta_1 & & \\ & \lambda_2 & \delta_2 & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix}$$

mit $\delta_i \in \{0, 1\}$, $i = 1, \dots, n-1$. Eine weitere Ähnlichkeitstransformation mit der Diagonalmatrix $P = \text{diag}(1, \varepsilon, \dots, \varepsilon^{n-1})$ führt Λ über in

$$\tilde{\Lambda} = P^{-1}\Lambda P = (XP)^{-1}A(XP).$$

Da $\tilde{\Lambda}$ die gleiche Hauptdiagonale wie Λ , in der Nebendiagonale aber Elemente $\varepsilon\delta_j$ hat, gilt

$$\|\tilde{\Lambda}\|_\infty = \max_{j=1}^n \{|\lambda_j| + \varepsilon|\delta_j|\} \begin{cases} \leq \varrho(\tilde{\Lambda}) + \varepsilon = \varrho(A) + \varepsilon, & \text{wenn ein } \delta_j \neq 0, \\ = \varrho(A), & \text{wenn alle } \delta_j = 0. \end{cases}$$

Für die durch die Vektornorm $\|x\|_M := \|M^{-1}x\|_\infty$ induzierte Matrixnorm

$$\|B\|_M := \|M^{-1}BM\|_\infty, \quad M := XP,$$

gilt also b). Bei diagonalisierbaren Matrizen ist $\delta_j \equiv 0$. Dies zeigt die erste Behauptung in c).

Die in c) genannten Spezialfälle mit der Norm $\|\cdot\|_2$ folgen aus der Orthogonalität der Eigenvektoren bei normalen Matrizen, denn unitäre Matrizen $X^{-1} = X^*$ sind *isometrisch*, $\|Xy\|_2 = \|y\|_2 \forall y$. Für diese ist $\|X\Lambda X^{-1}\|_2 = \|\Lambda\|_2$. ■

Der Satz erleichtert folgende wichtige Regularitätsaussage für Matrizen der Form $A = I - B$.

Satz 4.2.2 (Neumannreihe) Gegeben sei die Matrix $B \in \mathbb{R}^{n \times n}$ und es gelte

$$\varrho(B) < 1. \tag{4.2.10}$$

a) Dann ist das Gleichungssystem $(I - B)x = b$ eindeutig lösbar,

b) Dann gilt $(I - B)^{-1} = \sum_{j=0}^{\infty} B^j$,

c) Dann gibt es eine Matrixnorm so, dass $\|B\| < 1$ ist. In dieser Norm gilt die Schranke

$$\|(I - B)^{-1}\| \leq \frac{1}{1 - \|B\|}. \tag{4.2.11}$$

Bemerkung: Dieser Satz ist die Basis vieler Störungsaussagen, (4.2.11) gilt natürlich in jeder Norm, für die $\|B\| < 1$ ist.

Beweis a) Da $I - B$ regulär ist genau dann, wenn die Gleichung $x = Bx$ nur die triviale Lösung $x = 0$ hat, d.h. $\lambda = 1$ kein EW von B ist, folgt die Existenz von $(I - B)^{-1}$ aus (4.2.10).

b) Mit den Teilsummen $\sum_{k=0}^m B^k$ gilt die Identität

$$S_m := (I - B) \sum_{k=0}^m B^k = I - B + B - B^2 + \dots - B^{m+1} = I - B^{m+1}.$$

Nach Voraussetzung und Satz 4.2.1 existiert eine Norm mit $\|B\| < 1$. In dieser Norm gilt $\|S_m - I\| = \|B^{m+1}\| \leq \|B\|^{m+1} \rightarrow 0$ für $m \rightarrow \infty$ und somit $S_m \rightarrow I$, also

$$S_m \rightarrow (I - B) \sum_{k=0}^{\infty} B^k = I, \quad m \rightarrow \infty.$$

Durch Betrachtung von $(I - B)^{-1}S_m$ folgt die Behauptung b). Aus der Reihe b) ergibt sich mit Dreieck- und Produkt-Ungleichung die Schranke

$$\|(I - B)^{-1}\| = \left\| \sum_{k=0}^{\infty} B^k \right\| \leq \sum_{k=0}^{\infty} \|B^k\| \leq \sum_{k=0}^{\infty} \|B\|^k = \frac{1}{1 - \|B\|}. \quad \blacksquare$$

4.3 Der Gauß-Algorithmus

Zur Lösung von linearen Gleichungssystemen

$$\sum_{j=1}^n a_{ij}x_j = b_i, \quad i = 1, \dots, n \quad \Leftrightarrow \quad Ax = b, \quad (4.3.1)$$

$A \in \mathbb{K}^{n \times n}$, gibt es sowohl *direkte* als auch *iterative* Verfahren. Bei den direkten Verfahren formt man das gegebene System in einer endlichen Anzahl von Schritten in eine leicht auflösbare Form um, ändert also insbesondere die Matrix A . In der Vorlesung werden zwei direkte Verfahren behandelt, die mit verschiedenen Umformungen arbeiten. Im Vordergrund steht dabei die Interpretation als *Faktorisierung* der Matrix A . Bei iterativen Verfahren konstruiert man dagegen eine Folge von Näherungen, die gegen die Lösung konvergiert. Dabei wird die Matrix nicht verändert, was in vielen Fällen ("dünnbesetzte" Matrizen) vorteilhaft sein kann.

Der Gauß-Algorithmus ist das Standardverfahren zur direkten Auflösung und nutzt die Tatsache, dass sich die Lösung x von (4.3.1) nicht ändert, wenn Gleichungen linear kombiniert werden. Ziel ist die Konstruktion eines Systems in Dreieckform,

$$\left. \begin{array}{cccc} r_{11}x_1 & +r_{12}x_2 & +\dots & +r_{1n}x_n & = & b'_1 \\ & r_{22}x_2 & +\dots & +r_{2n}x_n & = & b'_2 \\ & & \ddots & \vdots & \vdots & \\ & & & r_{nn}x_n & = & b'_n \end{array} \right\} \Leftrightarrow Rx = b', \quad (4.3.2)$$

das man beginnend mit x_n direkt auflösen kann, falls alle $r_{ii} \neq 0$ sind. Die Matrix A sei im folgenden regulär. Man erzeugt ein Dreieckssystem (4.3.2) (im ursprünglichen Speicherplatz der Matrix A) in $n - 1$ Durchgängen. Im ersten wird

1. für $a_{11} = 0$ eine Zeile mit $a_{i1} \neq 0$ gesucht und diese dann mit der ersten vertauscht (*Pivot-Schritt*, i -te Zeile = Pivot-Zeile);
2. jeweils dasjenige Vielfache der ersten Zeile zu den restlichen Zeilen $i = 2, \dots, n$ addiert, das dort den Koeffizienten bei x_1 zu Null macht (*Eliminationsschritt*). Bei Behandlung der i -ten Zeile ist dabei $-a_{i1}/a_{11}$ der Vorfaktor der 1. Zeile.

Eliminationsteil des Algorithmus müssen dann nur die die rechte Seite b betreffenden Schritte (unterstrichene Anweisung in der 3. Zeile) und die Auflösung wiederholt werden. Dazu wurden die Größen ℓ_{ik} gespeichert. Der Aufwand ist dann

$$\sum_{k=1}^{n-1} 2(n-k) = n(n-1) \text{ Operationen für } b^{(n)}.$$

Die abschließende Auflösung des Dreiecksystems $Rx = b^{(n)}$ benötigt ebenfalls

$$1 + \sum_{i=1}^{n-1} (1 + 2(n-i)) = n^2 \text{ Operationen.}$$

Rechenaufwand Der Gauß-Algorithmus benötigt i.w. $\frac{2}{3}n^3$ arithmetische Operationen. Die nochmalige Auflösung mit neuer rechter Seite b erfordert nur $2n^2$ Operationen.

LR-Zerlegung

Die reine Umformung beim System $A^{(k)}x = b^{(k)}$ in einem Eliminationsschritt ohne Pivotisierung,

für $i = k + 1, \dots, n$: {
 $b_i^{(k+1)} := b_i^{(k)} - \ell_{ik}b_k^{(k)}$;
 für $j := k + 1, \dots, n$: {
 $a_{ij}^{(k+1)} := a_{ij}^{(k)} - \ell_{ik}a_{kj}^{(k)}$; } } }

entspricht der Multiplikation dieses Systems mit einer speziellen Matrix:

$$A^{(k)}x = b^{(k)} \quad \rightarrow \quad A^{(k+1)}x = L_k^{-1}A^{(k)}x = L_k^{-1}b^{(k)} = b^{(k+1)},$$

wobei sich L_k^{-1} nur durch eine Rang-1-Matrix von der Einheitsmatrix unterscheidet,

$$L_k^{-1} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -\ell_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & -\ell_{nk} & & 1 \end{pmatrix} = I - \ell^{(k)}e^{(k)\top}, \quad \ell^{(k)} := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \ell_{k+1,k} \\ \vdots \\ \ell_{nk} \end{pmatrix}. \quad (4.3.5)$$

Wegen $e^{(k)\top}\ell^{(k)} = 0$ ($e^{(k)}$ = k -ter Einheitsvektor) gilt $(I - \ell^{(k)}e^{(k)\top})(I + \ell^{(k)}e^{(k)\top}) = I$, also $L_k = I + \ell^{(k)}e^{(k)\top}$. Somit wird die obere Dreieckmatrix $R = A^{(n)}$ erzeugt durch Multiplikation mit L -Matrizen:

$$R = A^{(n)} = L_{n-1}^{-1}A^{(n-1)} = \dots = L_{n-1}^{-1} \dots L_1^{-1}A \quad \Leftrightarrow \\ A = L_1L_2 \dots L_{n-1}R =: LR. \quad (4.3.6)$$

Da die Menge der unteren bzw. oberen Dreiecksmatrizen abgeschlossen ist unter Matrixmultiplikation, ist L eine *untere* und R eine *obere* Dreiecksmatrix:

$$A = LR = \begin{pmatrix} 1 & 0 & \dots & 0 \\ * & 1 & & \\ \vdots & & \ddots & \\ * & * & \dots & 1 \end{pmatrix} \cdot \begin{pmatrix} * & * & \dots & * \\ 0 & * & & * \\ & & \ddots & \vdots \\ 0 & & & * \end{pmatrix} \quad (4.3.7)$$

Die Faktoren L (ohne Hauptdiagonale) und R können bei (4.3.4) wieder in A gespeichert werden.

Satz 4.3.2 *Der Gaußalgorithmus (4.3.4) (ohne Pivotisierung) erzeugt eine LR-Zerlegung (4.3.7) der Matrix A . Der Rechenaufwand dafür ist $\frac{2}{3}n^3$ Operationen. Anschließend reduziert sich jede Lösung eines LGS $Ax = b$ auf*

$$b = Ax = \underbrace{L(Rx)}_{b^{(n)}} \Leftrightarrow Lb^{(n)} = b, Rx = b^{(n)}, \quad (4.3.8)$$

also auf die Auflösung von zwei LGSen mit Dreiecksmatrizen mit insgesamt $2n^2$ Operationen.

Bemerkung: Der Gauß-Algorithmus (4.3.4) besteht aus 3 geschachtelten Schleifen mit der Basisoperation $a_{ij} := a_{ij} - a_{ik}a_{kj}$ und den Laufindizes i, j, k . Verblüffenderweise kann für jede Permutation dieser Schleifen eine korrekte Variante des Verfahrens angegeben werden. Praktische Bedeutung hat diese Erkenntnis, da die Effizienz dieser Varianten stark sprach- und maschinenabhängig sein kann (Matrizen zeilen-/spaltenweise gespeichert? Parallel-Verarbeitung?).

Zum Beispiel kann die LR-Zerlegung direkt erzeugt werden ohne die Zwischenmatrizen $A^{(k)}$. Die Identität $A = L \cdot R$ wird dabei als Gleichungssystem für L und R aufgefaßt:

$$a_{ij} = \sum_{k=1}^{\min\{i,j\}} l_{ik}r_{kj} \iff \begin{cases} a_{ij} = \sum_{k=1}^{j-1} l_{ik}r_{kj} + l_{ij}r_{jj}, & i > j, \\ a_{ij} = \sum_{k=1}^{i-1} l_{ik}r_{kj} + r_{ij}, & i \leq j. \end{cases} \quad (4.3.9)$$

Diese Gleichungen kann man direkt nach den l_{ij} (obere Gleichungen) bzw. r_{ij} (untere Gleichungen) auflösen. Auch hier sind noch unterschiedliche Reihenfolgen möglich und führen auf die ijk - und jik -Variante des Gauß-Algorithmus. Ein Vorteil dieser Varianten ist eine höhere erreichbare Genauigkeit, wenn die Summen in (4.3.9) mit erhöhter Genauigkeit berechnet werden.

Pivotisierung

Im allgemeinen muß beim Gauß-Algorithmus mit Zeilenvertauschungen (*Pivotisierung*) verhindert werden, dass zu kleine Hauptdiagonalelemente auftreten. Mögliche Pivotstrategien sind unter Effizienzgesichtspunkten zu analysieren. Andererseits kann man den Zusatzaufwand für Vertauschungen einsparen bei Klassen von Matrizen, wo Pivotisierung unnötig ist. Interessant ist hier, dass bestimmte Strukturen bzw. Eigenschaften der Matrix A beim Gauß-Algorithmus unverändert bleiben.