Einführung in die Numerische Mathematik

Vorlesungsskriptum Sommersemester 2018

R. Verfürth

Fakultät für Mathematik, Ruhr-Universität Bochum

Inhaltsverzeichnis

Einleitung

Die numerische Mathematik entwickelt und untersucht Methoden zur numerischen, d.h. computerunterstützten Lösung praktischer Probleme. Dabei besteht eine beständige Interaktion zwischen dem Anwender, der seine Kenntnisse über das Anwendungsproblem und dessen (mathematische) Modellierung einfließen lässt, anderen Bereichen der Mathematik, die z.B. Aussagen über Existenz, Eindeutigkeit und Eigenschaften der Lösungen des mathematischen Problems liefern, und der Numerik. Wesentliche Gesichtspunkte bei der Entwicklung oder Auswahl eines numerischen Verfahrens sind Effizienz und Stabilität des Verfahrens, d.h. die Frage, welchen Aufwand – computer- und manpower – das Verfahren erfordert und wie empfindlich die Lösung von Fehlern bei der Bestimmung der Eingabedaten und den durchzuführenden Rechenoperationen abhängt. Bei der Beurteilung eines Verfahrens spielt natürlich die ursprüngliche Fragestellung eine erhebliche Rolle. So ist es z.B. sinnlos, mit einem numerischen Verfahren die Temperatur im Innern eines Stahlblockes auf 1/1000 Grad genau zu bestimmen, wenn die gemessene Oberflächentemperatur nur auf 1 – 10 Grad genau bekannt ist!

Eine wesentliche Rolle bei der Entwicklung und Beurteilung eines numerischen Verfahrens spielt die Tatsache, dass man in einem Computer stets nur mit Zahldarstellungen mit endlich vielen Ziffern arbeiten kann. Daher wollen wir im Folgenden kurz auf diese endlichen Zahldarstellungen und die daraus resultierenden Rundungsfehler eingehen.

Zahlen werden in *normalisierter Form*

$$\sigma\, 0.a_1 \dots a_t\, b^c$$

dargestellt. Dabei ist

$b \in \mathbb{N}^*$	die *Basis*,
$t \in \mathbb{N}^*$	die *Mantissenlänge*,
$a_i \in \mathbb{N}_{b-1}$	die *Ziffern* mit $a_1 \neq 0$,
$c \in \mathbb{Z}$	der *Exponent*,
$\sigma \in \{-1, +1\}$	das *Vorzeichen*.

Gebräuchliche Basen sind

$b = 10$	*Dezimalsystem*,
$b = 2$	*Dualsystem*,
$b = 16$	*Hexadezimalsystem*.

Übliche Mantissenlängen bezogen auf die Basis 10 sind:

$$t = 8 \quad \text{oder} \quad t = 15.$$

Der Einfachheit halber beschränken wir uns im Folgenden auf das Dezimalsystem. Mit $\mathrm{rd}(x)$ bezeichnen wir die endliche Darstellung der Zahl x. Für

$$x = \sigma\, 0.a_1 \dots a_t a_{t+1} \dots 10^c$$

ist also

$$\mathrm{rd}(x) = \begin{cases} \sigma\, 0.a_1 \dots a_t\, 10^c & \text{falls } a_{t+1} < 5, \\ \sigma\, 0.a_1 \dots (a_t + 1)\, 10^c & \text{falls } a_{t+1} \geq 5. \end{cases}$$

Mithin ist

$$|x - \mathrm{rd}(x)| \leq 5 \cdot 10^{c-t-1} \quad \text{und} \quad \frac{|x - rd(x)|}{|x|} \leq 5 \cdot 10^{-t}.$$

Die Zahl

$$\mathrm{eps} = 5 \cdot 10^{-t}$$

heißt *Maschinengenauigkeit*.

Im Folgenden bezeichnen wir mit $\approx$ Ergebnisse, die bis auf Terme der Ordnung eps^2 oder höher genau sind. Für die Addition und Multiplikation von zwei Zahlen ergibt sich damit z.B.

$$\begin{aligned} \mathrm{rd}(x + y) &= [x(1 + \mathrm{eps}) + y(1 + \mathrm{eps})]\,(1 + \mathrm{eps}) \\ &= (x + y)(1 + \mathrm{eps})^2 \\ &\approx (x + y)(1 + 2\,\mathrm{eps}) \end{aligned}$$

und

$$\begin{aligned} \mathrm{rd}(x \cdot y) &= [x(1 + \mathrm{eps}) \cdot y(1 + \mathrm{eps})]\,(1 + \mathrm{eps}) \\ &= xy(1 + \mathrm{eps})^3 \\ &\approx xy(1 + 3\,\mathrm{eps}), \end{aligned}$$

d.h., der relative Fehler beträgt $2\,\mathrm{eps}$ bzw. $3\,\mathrm{eps}$.

Beispiel 1. Für $b = 10$ und $t = 8$ erhält man mit

$$a = 0.23371258 \cdot 10^{-4},\ b = 0.33678429 \cdot 10^2,\ c = -0.33677811 \cdot 10^2$$

die Ergebnisse

$$(a + b) + c = 0.33678452 \cdot 10^2 - 0.33677811 \cdot 10^2 = 0.64100000 \cdot 10^{-3}$$

und

$$a + (b + c) = 0.23371258 \cdot 10^{-4} + 0.61800000 \cdot 10^{-3} = 0.64137126 \cdot 10^{-3}$$

gegenüber dem exakten Ergebnis $0.641371258 \cdot 10^{-3}$. Der relative Fehler beträgt also $3 \cdot 10^{-4}$ bzw. $2 \cdot 10^{-9}$. Dieses Beispiel zeigt, dass es ratsam ist, Zahlen möglichst gleicher Größenordnung zu addieren, um Auslöschung zu vermeiden.

Beispiel 2. Zu lösen ist das lineare Gleichungssystem (LGS)

$$0.780x_1 + 0.563x_2 = 0.217$$
$$0.913x_1 + 0.659x_2 = 0.254$$

mit $b = 10$ und $t = 3$. Die exakte Lösung lautet $x = (1, -1)^t$.
Wir erhalten mit:

- Cramersche Regel: LGS nicht lösbar, da $\det A = 0$,
- Gauß-Elimination: $x = (0.278, 0.0)^t$,
- Gauß-Elimination mit Spalten- oder totaler Pivotisierung: LGS nicht lösbar.

Diese völlig unzureichenden Ergebnisse liegen an der Kondition von ca. $2 \cdot 10^6$ der Matrix. Daher werden die Ergebnisse unabhängig vom verwendeten Lösungsalgorithmus erst besser, wenn man zu einer 8- oder 9-stelligen Arithmetik übergeht.

Beispiel 3. Zu berechnen ist $\ln(2)$. Aus

$$-\ln(1-x) = \sum_{n=1}^{\infty} \frac{x^n}{n} \quad \forall x \in [-1, 1)$$

folgt für $x = -1$

$$\ln(2) = \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{n} = \sum_{n=1}^{m} \frac{(-1)^{n-1}}{n} + R_m$$

mit

$$\frac{1}{2m} \leq |R_m| \leq \frac{1}{m}.$$

Damit folgt für $b = 10$ und $t = 8$

$$\left| \ln(2) - \mathrm{rd}\left(\sum_{n=1}^{m} \frac{(-1)^{n-1}}{n} \right) \right| \leq \underbrace{|R_m|}_{\text{Abbruchfehler}} + \underbrace{\left| \sum_{n=1}^{m} \frac{(-1)^{n-1}}{n} - \mathrm{rd}\left(\sum_{n=1}^{m} \frac{(-1)^{n-1}}{n} \right) \right|}_{\text{Rundungsfehler}}$$
$$\leq \frac{1}{m} + m \cdot 10^{-8}.$$

Die rechte Seite wird für $m = 10^4$ minimal und liefert den Gesamtfehler $2 \cdot 10^{-4}$. Dieses Beispiel zeigt das typische Verhalten, dass mit größer werdendem m der Abbruchfehler abnimmt, während der Rundungsfehler zunimmt. Insgesamt kann der Gesamtfehler nicht unter eine von der verwendeten Arithmetik abhängigen Schranke – hier $2 \cdot 10^{-4}$ – gedrückt werden.

Andererseits folgt aus

$$\frac{1}{2}\ln\left(\frac{1+x}{1-x}\right) = \sum_{n=1}^{\infty} \frac{x^{2n-1}}{2n-1} \quad \forall x \in [-1,1)$$

für $x = \frac{1}{3}$

$$\ln(2) = 2\sum_{n=1}^{\infty} \frac{3^{-(2n-1)}}{2n-1} = 2\sum_{n=1}^{m} \frac{3^{-(2n-1)}}{2n-1} + \widetilde{R}_m$$

mit

$$\left|\widetilde{R}_m\right| \leq \frac{2}{(2m+1)3^{2m+1}} \sum_{k=0}^{\infty} 3^{-2k} = \frac{2}{3(2m+1)}\frac{9}{8} \cdot 9^{-m} \leq 10^{-7}$$

für $m \geq 6$. Also erhalten wir mit diesem Verfahren schon mit 6 Summanden einen Abbruch- und Gesamtfehler von 10^{-7}. Dieses Verfahren ist also offensichtlich viel genauer und effizienter als das erste.

Zusammenfassend kann man folgende Forderungen an ein gutes numerisches Verfahren aufstellen:

- *Ökonomie*: Der Aufwand sollte möglichst gering sein. Ein Maß hierfür ist die Zahl der benötigten arithmetischen Operationen.
- *Stabilität*: Ein Verfahren sollte möglichst unempfindlich gegen Rundungsfehler sein und a priori und a posteriori Abschätzungen des Fehlers liefern.
- *Anwendungsbereich*: Er sollte möglichst groß sein.

KAPITEL I

Numerische Interpolation

In diesem Kapitel behandeln wir das Problem, eine unbekannte Funktion in bestimmten Punkten, in denen ihr Wert bekannt ist, zu interpolieren. Zunächst betrachten wir eine allgemeine Formulierung dieses Problems und geben eine Reihe von Beispielen an. Danach untersuchen wir die folgenden Spezialfälle ausführlicher:

- Lagrange-Interpolation,
- Spline-Interpolation,
- trigonometrische Interpolation.

I.1. Das allgemeine Interpolationsproblem

Wir betrachten das Interpolationsproblem zunächst in einer sehr allgemeinen abstrakten Form.

DEFINITION I.1.1 (Allgemeines Interpolationsproblem). Seien X ein normierter Vektorraum, $V \subset X$ ein N-dimensionaler Untervektorraum und $\ell_1, \ldots, \ell_N \in \mathcal{L}(X, \mathbb{R})$ stetige lineare Funktionale. Dann lautet das *allgemeine Interpolationsproblem*:
Finde zu $f \in X$ ein $u \in V$ mit

$$\ell_i(u) = \ell_i(f) \quad \forall 1 \leq i \leq N. \tag{I.1.1}$$

Das Interpolationsproblem heißt *wohl gestellt*, wenn (I.1.1) zu jedem $f \in X$ eine eindeutige Lösung besitzt.

Selbstverständlich sind in der Praxis nur die Daten $\ell_i(f)$, $1 \leq i \leq N$, bekannt.

Der folgende Satz gibt allgemeine Kriterien an, wann ein Interpolationsproblem wohl gestellt ist.

SATZ I.1.2 (Wohlgestelltheit des Interpolationsproblems). (1) *Das Interpolationsproblem* (I.1.1) *ist genau dann wohl gestellt, wenn für eine Basis $v_1, \ldots, v_N$ von V die Matrix $A = (\ell_i(v_j))_{1 \leq i,j \leq N}$ regulär ist.* (2) *Das Interpolationsproblem* (I.1.1) *ist höchstens dann wohl gestellt, wenn die linearen Funktionale $\ell_1, \ldots, \ell_N$ linear unabhängig sind.*

BEWEIS. *ad (1):* (I.1.1) ist äquivalent zu:
Finde $\alpha_1, \ldots, \alpha_N \in \mathbb{R}$ mit

$$\ell_i \left(\sum_{j=1}^{N} \alpha_j v_j \right) = \sum_{j=1}^{N} \ell_i(v_j) \alpha_j = \ell_i(f) \quad \forall 1 \leq i \leq N.$$

Hieraus folgt die Behauptung.
ad (2): Ist offensichtlich. □

Das folgende Beispiel gibt einige Interpolationsprobleme an, die in den abstrakten Rahmen von Definition I.1.1 passen. Einige dieser Probleme werden wir in den nächsten Paragraphen näher untersuchen.

BEISPIEL I.1.3. *(1) (Lagrange-Interpolation in $\mathbb{R}$)*

$$\begin{aligned} X &= C([a,b],\mathbb{R}), a<b, a,b\in\mathbb{R},\\ V &= \mathbb{P}_n = \operatorname{span}\{x^j : 0\le j\le n\}, n\in\mathbb{N},\\ N &= n+1,\\ \ell_i(f) &= f(x_i),\ 0\le i\le n,\\ a &\le x_0 < \ldots < x_n \le b. \end{aligned}$$

Die Punkte $x_0, \ldots, x_n$ heißen *Knoten*. Das Problem ist wohl gestellt.
(2) (Hermite-Interpolation in $\mathbb{R}$)

$$\begin{aligned} X &= C^1([a,b],\mathbb{R}), a<b, a,b\in\mathbb{R},\\ V &= \mathbb{P}_{2n+1}, n\in\mathbb{N},\\ N &= 2n+2,\\ \ell_{2i}(f) &= f(x_i),\ 0\le i\le n,\\ \ell_{2i+1}(f) &= f'(x_i),\ 0\le i\le n,\\ a &\le x_0 < \ldots < x_n \le b. \end{aligned}$$

Das Problem ist wohl gestellt.
(3) (Kubische Spline-Interpolation)

$$\begin{aligned} X &= C^2([a,b],\mathbb{R}), a<b, a,b\in\mathbb{R},\\ V &= \{v\in X : v|_{[x_{i-1},x_i]}\in\mathbb{P}_3, 1\le i\le n, v''(x_0)=v''(x_n)=0\},\\ &\quad a=x_0<x_1<\ldots<x_n=b,\\ N &= n+1, n\in\mathbb{N}^*,\\ \ell_i(f) &= f(x_i),\ 0\le i\le n. \end{aligned}$$

Das Problem ist wohl gestellt.
(4) (Trigonometrische Interpolation)

$$\begin{aligned} X &= \{f\in C(\mathbb{R},\mathbb{R}) : f \text{ ist } 2\pi\text{-periodisch}\},\\ V &= \operatorname{span}\{1,\cos(kx),\sin(kx) : 1\le k\le n\},\\ N &= 2n+1, n\in\mathbb{N},\\ \ell_i(f) &= f(x_i),\ 0\le i\le 2n,\\ x_0 &< \ldots < x_{2n}, |x_{2n}-x_0| < 2\pi. \end{aligned}$$

Das Problem ist wohl gestellt.
(5) (Lagrange-Interpolation in $\mathbb{R}^2$)

$$\begin{aligned} X &= C(I,\mathbb{R}), I = [0,1]\times[0,1], \\ V &= Q_n = \operatorname{span}\{x_1^{\alpha_1}x_2^{\alpha_2} : 0 \le \alpha_i \le n\}, \\ N &= (n+1)^2, n \in \mathbb{N}^*, \\ \ell_{ij}(f) &= f\left(\frac{i}{n}, \frac{j}{n}\right),\ 0 \le i,j \le n. \end{aligned}$$

Das Problem ist wohl gestellt.
(6) (Taylor-Polynom)

$$\begin{aligned} X &= C^n(I,\mathbb{R}), I \subset \mathbb{R} \text{ offenes Intervall}, n \in \mathbb{N}, \\ V &= \mathbb{P}_n, \\ N &= n+1, \\ \ell_i(f) &= f^{(i)}(x_0),\ 0 \le i \le n, x_0 \in I. \end{aligned}$$

Das Problem ist wohl gestellt.
(7) (Momentenproblem)

$$\begin{aligned} X &= (C[a,b],\mathbb{R}), a < b, a, b \in \mathbb{R}, \\ V &= \mathbb{P}_n, \\ N &= n+1, \\ \ell_i(f) &= \int_a^b x^i f(x)dx,\ 0 \le i \le n. \end{aligned}$$

Das Problem ist wohl-gestellt.
(8) (Gemischte Lagrange-Hermite-Interpolation)

$$\begin{aligned} X &= C^1([-1,1],\mathbb{R}), \\ V &= \mathbb{P}_2, \\ N &= 3, \\ \ell_1(f) &= f(-1),\ \ell_2(f) = f'(0),\ \ell_3(f) = f(1). \end{aligned}$$

Das Problem ist nicht wohl gestellt.

BEWEIS. *ad (1):* Sei $p \in \mathbb{P}_n$ eine Lösung des homogenen Problems. Dann ist p durch

$$\omega(x) = \prod_{i=0}^{n}(x - x_i) \in \mathbb{P}_{n+1} \setminus \mathbb{P}_n$$

teilbar. Also ist $p = 0$.
ad (2): Sei $p \in \mathbb{P}_{2n+1}$ eine Lösung des homogenen Problems. Dann ist p durch $\omega^2 \in \mathbb{P}_{2n+2} \setminus \mathbb{P}_{2n+1}$ teilbar. Also ist $p = 0$.
ad (3): Siehe Paragraph I.3.

ad (4): Sei

$$p(x) = \frac{a_0}{2} + \sum_{k=1}^{n} \{a_k \cos(kx) + b_k \sin(kx)\}$$

eine Lösung des homogenen Problems. Dann gilt

$$\begin{aligned} p(x) &= \frac{a_0}{2} + \sum_{k=1}^{n} \left\{ a_k \frac{1}{2}(e^{ikx} + e^{-ikx}) + b_k \frac{1}{2i}(e^{ikx} - e^{-ikx}) \right\} \\ &= \frac{a_0}{2} + \sum_{k=1}^{n} \frac{1}{2}(a_k - ib_k)e^{ikx} + \sum_{k=1}^{n} \frac{1}{2}(a_k + ib_k)e^{-ikx} \\ &= \sum_{m=-n}^{n} \alpha_m e^{imx} \end{aligned}$$

mit

$$\alpha_m = \frac{1}{2}\left[a_{|m|} - i\,\mathrm{sgn}(m) b_{|m|}\right].$$

Also ist

$$p(x) = e^{-inx} \sum_{m=-n}^{n} \alpha_m e^{i(m+n)x} = e^{-inx} \sum_{k=0}^{2n} \alpha_{k-n} e^{ikx} = e^{-inx} q(e^{ix})$$

mit

$$q(z) = \sum_{k=0}^{2n} \alpha_{k-n} z^k.$$

Dann hat q die $2n+1$ verschiedenen komplexen Nullstellen $z_k = e^{ix_k}$, $0 \le k \le 2n$. Also ist $q = 0$ und damit $p = 0$.
ad (5): Die Funktionen

$$q_{ij}(x) = \prod_{\substack{k=0 \\ k \neq i}}^{n} \left(x_1 - \frac{k}{n}\right) \prod_{\substack{l=0 \\ l \neq j}}^{n} \left(x_2 - \frac{l}{n}\right) \in Q_n$$

bilden eine Basis von Q_n, bzgl. derer die Matrix A aus Satz I.1.2 eine nicht verschwindende Diagonalmatrix ist.
ad (6): Das Taylor-Polynom $T_n(f; x_0)$ leistet das Gewünschte.
ad (7): Sei p eine Lösung des homogenen Problems. Dann gilt

$$\begin{aligned} p \in \mathbb{P}_n \quad \text{und} \quad \int_a^b x^i p(x) dx = 0 \quad \forall 0 \le i \le n \\ \Longrightarrow \int_a^b p(x)^2 dx = 0 \\ \Longrightarrow p = 0. \end{aligned}$$

ad (8): $p_1 = 0$ und $p_2(x) = x^2 - 1$ lösen das homogene Problem. □

I.2. Lagrange-Interpolation

In diesem Paragraphen betrachten wir das Interpolationsproblem aus Beispiel I.1.3(1) (S. 10). Dabei ist stets $I = [a, b]$, $a, b \in \mathbb{R}$, $a < b$, und $a \leq x_0 < \ldots < x_n \leq b$, $n \in \mathbb{N}$.

Definition I.2.1 (Knotenpolynom, Grundpolynome). Die Polynome

$$\omega(x) = \prod_{i=0}^{n}(x - x_i) \in \mathbb{P}_{n+1} \setminus \mathbb{P}_n$$

und

$$\lambda_i(x) = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j} = \frac{\omega(x)}{(x - x_i)\omega'(x_i)} \in \mathbb{P}_n,\ 0 \leq i \leq n,$$

heißen das *Knotenpolynom* bzw. die *Lagrangeschen Grundpolynome* zu den Knoten $x_0, \ldots, x_n$.

Mit Hilfe der Lagrangeschen Grundpolynome erhalten wir eine einfache explizite Darstellung des Lagrangeschen Interpolationspolynoms. Diese Darstellung ist für theoretische Untersuchungen insbesondere für Fehlerabschätzungen günstig, für die praktische Rechnung aber weniger geeignet.

Satz I.2.2 (Interpolationsformel von Lagrange). *Das Lagrangesche Interpolationspolynom $L_n f$ zu $f \in C(I, \mathbb{R})$ und den Knoten $x_0, \ldots, x_n$ ist gegeben durch*

$$L_n f = \sum_{i=0}^{n} f(x_i)\lambda_i.$$

Beweis. Offensichtlich gilt für $0 \leq i, j \leq n$

$$\lambda_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{für } i = j, \\ 0 & \text{für } i \neq j. \end{cases}$$

Zusammen mit Beispiel I.1.3(1) (S. 10) folgt hieraus die Behauptung. □

Der folgende Satz gibt eine Abschätzung für die Genauigkeit der Lagrange-Interpolation.

Satz I.2.3 (Fehlerabschätzung für die Lagrange-Interpolation). *Sei $f \in C^{n+1}(I, \mathbb{R})$. Dann gibt es zu jedem $x^* \in I$ ein $\eta^* = \eta^*(x^*) \in I$ mit*

$$f(x^*) - L_n f(x^*) = \frac{1}{(n+1)!} f^{(n+1)}(\eta^*)\omega(x^*).$$

Insbesondere gilt

$$\|f - L_n f\|_{C(I,\mathbb{R})} \leq \frac{(b-a)^{n+1}}{(n+1)!} \left\|f^{(n+1)}\right\|_{C(I,\mathbb{R})}$$

und

$$\|f - L_n f\|_{C(I,\mathbb{R})} \leq h^{n+1} \left\|f^{(n+1)}\right\|_{C(I,\mathbb{R})}$$

mit

$$h = \max_{0\leq i\leq n+1} |x_i - x_{i-1}| \quad \textit{und} \quad x_{-1} = a,\ x_{n+1} = b.$$

BEWEIS. O.E. ist $x^* \notin \{x_0, \ldots, x_n\}$, da die erste Aussage sonst offensichtlich gilt. Definiere

$$g(x) = f(x) - L_n f(x) - \frac{1}{\omega(x^*)} \left[f(x^*) - L_n f(x^*)\right] \omega(x).$$

Die Funktion g hat die Nullstellen x^*, x_0, ..., x_n. Aus dem Satz von Rolle folgt, dass $g^{(n+1)}$ eine Nullstelle $\eta^* \in I$ besitzt. Da die $(n+1)$-te Ableitung von $L_n f$ verschwindet, ergibt sich

$$0 = g^{(n+1)}(\eta^*) = f^{(n+1)}(\eta^*) - \frac{(n+1)!}{\omega(x^*)} \left[f(x^*) - L_n f(x^*)\right].$$

Hieraus folgt die erste Behauptung.
Die zweite Behauptung folgt aus der ersten und der offensichtlichen Abschätzung

$$|\omega(x)| \leq (b-a)^{n+1} \quad \forall x \in I.$$

Die dritte Behauptung schließlich folgt aus der ersten und der verschärften Abschätzung

$$\frac{1}{(n+1)!} |\omega(x)| \leq h^{n+1} \frac{i!(n+1-i)!}{(n+1)!} \leq h^{n+1}$$

wenn $x_{i-1} \leq x \leq x_i$ ist. □

Aus Satz I.2.3 folgt $\lim_{n\to\infty} \|f - L_n f\|_{C(I,\mathbb{R})} = 0$, wenn $f \in C^\infty(I,\mathbb{R})$ ist und $\sup_{n\in\mathbb{N}} \left\|f^{(n)}\right\|_{C(I,\mathbb{R})} < \infty$ gilt. Diese Bedingung ist insbesondere für analytische Funktionen erfüllt. Ist f dagegen nicht analytisch, z.B. $f \in C^k(I,\mathbb{R})$ mit festem k, erlaubt Satz I.2.3 keine Rückschlüsse über die Konvergenz der Lagrangeschen Interpolationspolynome für Knotenzahl $n \to \infty$. Der folgende Satz zeigt, dass der Fehler so gar beliebig groß werden kann. Andererseits zeigt Satz I.2.3, dass bei fester Knotenzahl n der Fehler des Lagrangeschen Interpolationspolynoms beliebig klein wird, wenn die Intervalllänge klein wird. Dies werden wir im nächsten Paragraphen bei der Spline-Interpolation ausnutzen.

SATZ I.2.4 (Satz von Faber). *Zu jeder Knotenmatrix* $a \leq x_0^{(n)} < \ldots < x_n^{(n)} \leq b$, $n \in \mathbb{N}$ *gibt es ein* $f \in C(I,\mathbb{R})$, *so dass für die Folge der zugehörigen Lagrangeschen Interpolationspolynome gilt*

$$\limsup_{n\to\infty} \|f - L_n f\|_{C(I,\mathbb{R})} = \infty.$$

BEWEISIDEE. Der Satz von Faber folgt aus dem Prinzip der gleichmäßigen Beschränktheit und der Abschätzung

$$\limsup_{n\to\infty} \|L_n\|_{\mathcal{L}(C(I,\mathbb{R}),C(I,\mathbb{R}))} = \infty. \qquad \square$$

BEISPIEL I.2.5. Zur Illustration des Satzes von Faber geben wir in den Tabellen I.2.1 und I.2.2 den maximalen Fehler des Lagrangeschen Interpolationspolynoms der Funktionen $|x|$ auf dem Intervall $[-a,a] = [-1,1]$ und $\arctan x$ auf dem Intervall $[-a,a] = [-10,10]$ in n äquidistanten Knoten $a\frac{2i+1-n}{n-1}$, $0 \le i \le n-1$, bzw. in n Čebyšev Knoten $-a\cos(\frac{i\pi}{n-1})$, $0 \le i \le n-1$, an. Zum Vergleich geben wir auch den maximalen Fehler des interpolierenden Splines aus Paragraph I.3 für äquidistante Knoten an. Diese Werte zeigen die Überlegenheit der Spline-Interpolation. Die Abbildungen I.2.1 und I.2.2 zeigen für einige Fälle die Funktion (grün), die Interpolierende (blau) und die Fehler (rot). Man beachte die unterschiedlichen Skalierungen.

TABELLE I.2.1. Maximaler Fehler der Lagrange- und Spline-Interpolation der Funktion $|x|$ auf $[-1,1]$ in n Knoten

n	Lagrange		Spline
	äquidistant	Čebyšev	äquidistant
5	0.1472	0.1422	0.0858
9	0.3157	0.0737	0.0425
17	11.1371	0.0372	0.0213
33	105717.8079	0.0186	0.0106

TABELLE I.2.2. Maximaler Fehler der Lagrange- und Spline-Interpolation der Funktion $\arctan x$ auf $[-10,10]$ in n Knoten

n	Lagrange		Spline
	äquidistant	Čebyšev	äquidistant
5	0.5116	0.6067	0.4837
9	0.3052	0.3480	0.2213
17	4.6190	0.1241	0.0549
33	6132.8420	0.0186	0.0050

Satz I.2.2 löst das Lagrangesche Interpolationsproblem vollständig. Allerdings ist die dort gegebene Darstellung für die praktische Rechnung aus mehreren Gründen ungeeignet:

- Die Auswertung von $L_n f$ erfordert $O(n^2)$ arithmetische Operationen pro Auswertungspunkt.

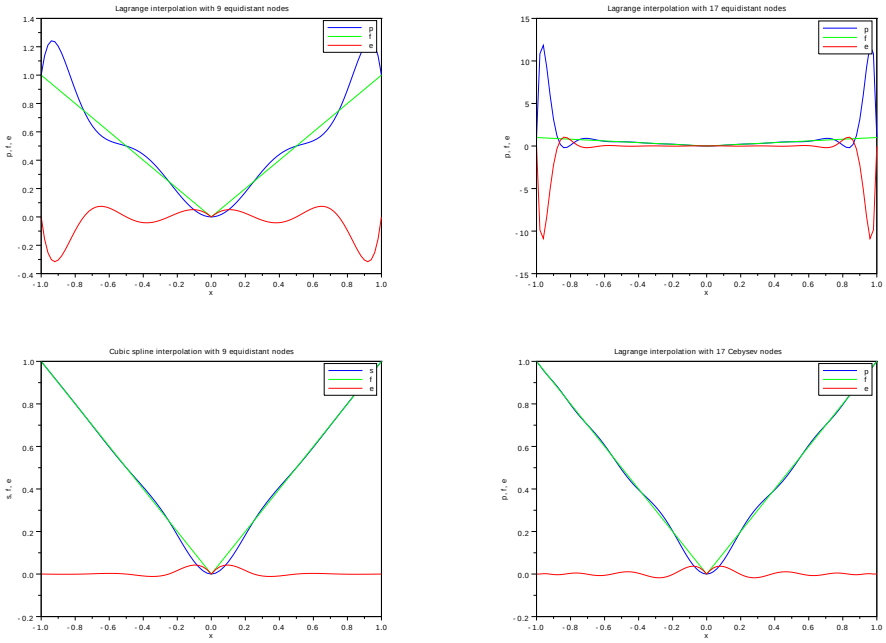

ABBILDUNG I.2.1. Interpolation der Funktion $|x|$ auf $[-1,1]$ (o.l.: Lagrange-Interpolation in 9 äquidistanten Knoten, o.r.: Lagrange-Interpolation in 17 äquidistanten Knoten, u.l.: kubische Spline-Interpolation in 9 äquidistanten Knoten, u.r.: Lagrange-Interpolation in 17 Čebyšev-Knoten; grün: Funktion, blau: Interpolierende, rot: Fehler)

- Bei der Auswertung der Lagrangeschen Grundpolynome treten kleine Nenner auf, so dass die Berechnung von $L_n f$ sehr anfällig für Rundungsfehler ist.

Wir geben im Folgenden zwei für die Praxis besser geeignete Algorithmen an. Beide vermeiden die Anfälligkeit für Rundungsfehler. Der erste Algorithmus ist gut geeignet, wenn $L_n f$ nur an wenigen Punkten ausgewertet werden muss. Der zweite hingegen ist bei der Auswertung von $L_n f$ in $m \gg n$ Punkten wesentlich effizienter, da er den Aufwand von $\mathrm{O}(mn^2)$ auf $\mathrm{O}(n^2 + mn)$ arithmetische Operationen reduziert.

SATZ I.2.6 (Neville-Polynom). *Für $0 \le i \le n$ und $0 \le k \le n - i$ bezeichne $p_{i,k} \in \mathbb{P}_k$ das Lagrangesche Interpolationspolynom von f zu den Knoten $x_i, \ldots, x_{i+k}$. Dann gilt für alle $0 \le i \le n$*

$$p_{i,0}(x) = f(x_i) \tag{I.2.1}$$

und für alle $0 \le i \le n-1$, $1 \le k \le n - i$

$$p_{i,k}(x) = \frac{(x - x_i)p_{i+1,k-1}(x) - (x - x_{i+k})p_{i,k-1}(x)}{x_{i+k} - x_i}. \tag{I.2.2}$$

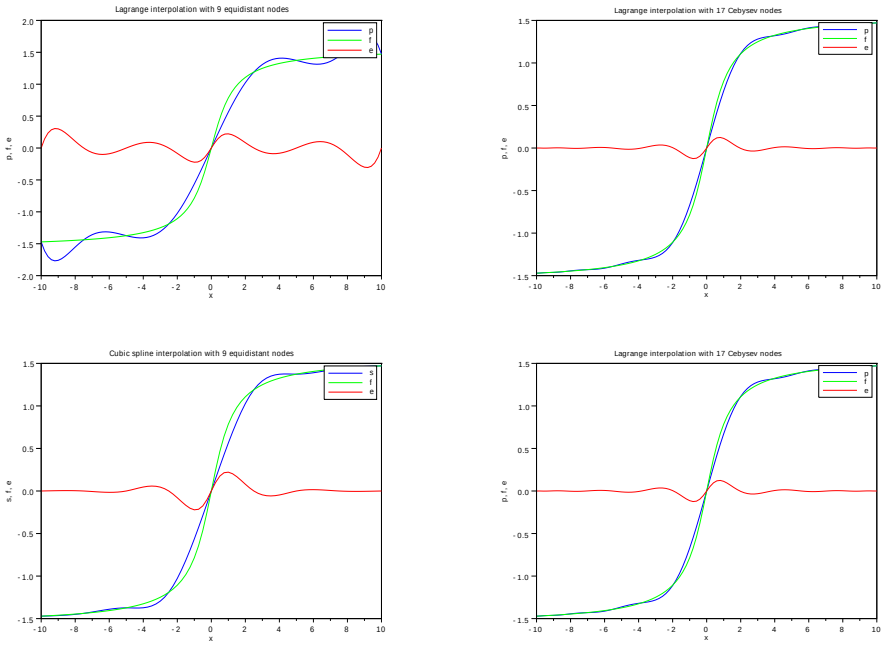

ABBILDUNG I.2.2. Interpolation der Funktion $\arctan x$ auf $[-10, 10]$ (o.l.: Lagrange-Interpolation in 9 äquidistanten Knoten, o.r.: Lagrange-Interpolation in 17 äquidistanten Knoten, u.l.: kubische Spline-Interpolation in 9 äquidistanten Knoten, u.r.: Lagrange-Interpolation in 17 Čebyšev-Knoten; grün: Funktion, blau: Interpolierende, rot: Fehler)

BEWEIS. Wir beweisen die Behauptung durch Induktion über k.
$\underline{k = 0:}$ Offensichtlich ist (I.2.1) richtig.
$\underline{k - 1 \to k:}$ Bezeichne die rechte Seite von (I.2.2) mit q. Dann ist $q \in \mathbb{P}_k$. Für $i + 1 \le j \le i + k - 1$ folgt aus der Induktionsannahme

$$q(x_j) = \frac{(x_j - x_i)f(x_j) - (x_j - x_{i+k})f(x_j)}{x_{i+k} - x_i} = f(x_j).$$

Weiter ist offensichtlich

$$q(x_i) = f(x_i), \quad q(x_{i+k}) = f(x_{i+k}).$$

Zusammen mit Satz I.2.2 folgt hieraus die Behauptung. □

Satz I.2.6 führt auf Algorithmus I.2.1 zur Berechnung von $L_n f(x)$.

Wertet man $L_n f$ mit Hilfe von Algorithmus I.2.1 in m Punkten aus benötigt man $O(mn^2)$ arithmetische Operationen. Daher ist Algorithmus I.2.1 nur für kleine Werte von m empfehlenswert. Für größere Werte von m ist der Algorithmus, den wir im Folgenden entwickeln, wesentlich effizienter.

Algorithmus I.2.1 Interpolationsformel von Neville-Aitken

Gegeben: Knoten $x_0, \ldots, x_n$, Daten $y_0 = f(x_0), \ldots, y_n = f(x_n)$, Auswertungspunkt x
Gesucht: $y = L_n f(x)$

1: **for** $i = 0, 1, \ldots, n$ **do**
2: $\quad p_{i,0} \leftarrow y_i$
3: **end for**
4: **for** $k = 1, \ldots, n$ **do**
5: $\quad$ **for** $i = 0, 1, \ldots, n-k$ **do**
6: $\quad\quad p_{i,k} \leftarrow \frac{(x-x_i)p_{i+1,k-1} - (x-x_{i+k})p_{i,k-1}}{x_{i+k}-x_i}$
7: $\quad$ **end for**
8: **end for**
9: $y \leftarrow p_{0,n}$

DEFINITION I.2.7 (Dividierte Differenzen). Die *k-te dividierte Differenz* $f[x_i, \ldots, x_{i+k}]$ von f zu den Knoten $x_i, \ldots, x_{i+k}$ ist definiert als der Höchstkoeffizient von $p_{i,k}$.

SATZ I.2.8 (Eigenschaften der dividierten Differenzen). *Die dividierten Differenzen haben folgende Eigenschaften:*

(1) $f[x_i, \ldots, x_{i+k}]$ *ist unabhängig von der Reihenfolge der Knoten.*
(2) $f[x_i] = f(x_i)$ *für alle* $0 \le i \le n$ *und*

$$f[x_i, \ldots, x_{i+k}] = \frac{f[x_{i+1}, \ldots, x_{i+k}] - f[x_i, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}$$

für alle $0 \le i \le n$, $1 \le k \le n - i$.
(3) *Ist* $f \in C^n(I, \mathbb{R})$, *so gibt es ein* $\eta \in I$ *mit*

$$f[x_0, \ldots, x_n] = \frac{1}{n!} f^{(n)}(\eta).$$

(4) $p[x_0, \ldots, x_n] = 0$ *für alle* $p \in \mathbb{P}_{n-1}$.

BEWEIS. *ad (1):* Die Polynome $p_{i,k}$ hängen nicht von der Reihenfolge der Knoten ab.
ad (2): Folgt durch Koeffizientenvergleich aus Satz I.2.6.
ad (3): Sei $g = f - p_{0,n} \in C^n(I, \mathbb{R})$. Da g die $n+1$ Nullstellen $x_0, \ldots, x_n$ hat, folgt aus dem Satz von Rolle die Existenz eines $\eta \in I$ mit

$$0 = g^{(n)}(\eta) = f^{(n)}(\eta) - n! f[x_0, \ldots, x_n].$$

ad(4): Folgt direkt aus Teil (3). □

BEMERKUNG I.2.9. Wegen Satz I.2.8 (3) liefern die dividierten Differenzen gute Approximationen für die Ableitung einer gegebenen Funktion und werden zur numerischen Differentiation benutzt.

Satz I.2.8 (2) führt unmittelbar auf Algorithmus I.2.2 zur Berechnung der dividierten Differenzen.

Algorithmus I.2.2 Berechnung dividierter Differenzen

Gegeben: Knoten $x_0, \ldots, x_n$, Daten $y_0 = f(x_0), \ldots, y_n = f(x_n)$
Gesucht: $\triangle_{i,k} = f[x_i, \ldots, x_{i+k}]$, $0 \le i \le n$, $0 \le k \le n - i$

```
for i = 0, 1, ..., n do
    △_{i,0} ← y_i
end for
for k = 1, ..., n do
    for i = 0, 1, ..., n − k do
        △_{i,k} ← (△_{i+1,k−1} − △_{i,k−1}) / (x_{i+k} − x_i)
    end for
end for
```

Mit Hilfe der dividierten Differenzen erhalten wir eine alternative Darstellung von $L_n f$, die rekursiv ausgewertet werden kann und dann $O(n)$ arithmetische Operationen pro Auswertungspunkt erfordert.

Satz I.2.10 (Interpolationsformel von Newton). *Das Lagrangesche Interpolationspolynom zu f und den Knoten $x_0, \ldots, x_n$ ist gegeben durch*

$$L_n f(x) = f(x_0) + \sum_{i=1}^{n} f[x_0, \ldots, x_i] \prod_{j=0}^{i-1} (x - x_j).$$

Beweis. Wir beweisen die Behauptung durch Induktion über n.
$\underline{n = 0:}$ Ist offensichtlich.
$\underline{n \to n+1:}$ Aus der Induktionsvoraussetzung folgt

$$f(x_0) + \sum_{i=1}^{n+1} f[x_0, \ldots, x_i] \prod_{j=0}^{i-1} (x - x_j) = L_n f(x) + f[x_0, \ldots, x_{n+1}] \prod_{j=0}^{n} (x - x_j)$$

und damit für $0 \le \mu \le n$

$$f(x_0) + \sum_{i=1}^{n+1} f[x_0, \ldots, x_i] \prod_{j=0}^{i-1} (x_\mu - x_j) = f(x_\mu) = p_{0,n+1}(x_\mu).$$

Da nach Definition I.2.7

$$f(x_0) + \sum_{i=1}^{n+1} f[x_0, \ldots, x_i] \prod_{j=0}^{i-1} (x - x_j) - p_{0,n+1} \in \mathbb{P}_n$$

ist, folgt hieraus die Behauptung. □

Aus Satz I.2.10 ergibt sich Algorithmus I.2.3 zur Berechnung von $L_n f(x)$ bei bekannten dividierten Differenzen.

Algorithmus I.2.3 Interpolationsformel von Newton

Gegeben: Knoten $x_0, \ldots, x_n$, dividierte Differenzen $\triangle_{0,0}, \ldots, \triangle_{0,n}$, Auswertungspunkt x
Gesucht: $y = L_n f(x)$

1: $y \leftarrow \triangle_{0,n}$
2: **for** $k = n-1, n-2, \ldots, 0$ **do**
3: $\quad y \leftarrow \triangle_{0,k} + (x - x_k)y$
4: **end for**

BEISPIEL I.2.11. Für $x_0 = -1$, $x_1 = -\frac{1}{3}$, $x_2 = \frac{1}{3}$, $x_3 = 1$ und $f(x) = |x|$ erhalten wir folgendes Schema

$$\begin{array}{c|cccc} -1 & 1 & & & \\ & & -1 & & \\ -\frac{1}{3} & \frac{1}{3} & & \frac{3}{4} & \\ & & 0 & & 0 \\ \frac{1}{3} & \frac{1}{3} & & \frac{3}{4} & \\ & & 1 & & \\ 1 & 1 & & & \end{array}$$

und

$$\begin{aligned} L_n f(x) &= 1 - (x+1) + \frac{3}{4}(x+1)(x+\frac{1}{3}) + 0 \cdot (x+1)(x+\frac{1}{3})(x-\frac{1}{3}) \\ &= -x + \frac{3}{4}(x^2 + \frac{4}{3}x + \frac{1}{3}) \\ &= \frac{3}{4}x^2 + \frac{1}{4}. \end{aligned}$$

I.3. Spline-Interpolation

In diesem Paragraphen betrachten wir das Interpolationsproblem aus Beispiel I.1.3(3) (S. 10). Es ist motiviert durch die Sätze I.2.3 (S. 13) und I.2.4 (S. 14), die zeigen, dass man die Genauigkeit der Interpolierenden nicht durch Erhöhung des Grades, wohl aber durch Verkleinerung des Intervalls verbessern kann.

Im Folgenden ist stets $a = x_0 < x_1 < \ldots < x_n = b$. Außerdem setzen wir zur Abkürzung:

$$\begin{aligned} I_k &= [x_k, x_{k+1}], \quad 0 \le k \le n-1, \\ h_k &= x_{k+1} - x_k, \quad 0 \le k \le n-1, \\ h &= \max_{0 \le k \le n-1} h_k, \\ \mu_k &= \frac{h_{k-1}}{h_k + h_{k-1}}, \quad 1 \le k \le n-1, \\ \lambda_k &= \frac{h_k}{h_k + h_{k-1}}, \quad 1 \le k \le n-1, \end{aligned}$$

und

$$M_n = \begin{pmatrix} 2 & \lambda_1 & & & & & \\ \mu_2 & 2 & \lambda_2 & & & & \\ & \ddots & \ddots & \ddots & & 0 & \\ & & \ddots & \ddots & \ddots & & \\ & 0 & & \ddots & \ddots & \ddots & \\ & & & & \mu_{n-2} & 2 & \lambda_{n-2} \\ & & & & & \mu_{n-1} & 2 \end{pmatrix} \in \mathbb{R}^{(n-1)\times(n-1)}.$$

Splines sind Funktionen, die stückweise Polynome und global hinreichend oft differenzierbar sind.

DEFINITION I.3.1 (Splines). *(1)* Für $k, n \in \mathbb{N}^*$ bezeichnet

$$S_n^k = \{u \in C^{k-1}(I, \mathbb{R}) : u|_{I_l} \in \mathbb{P}_k, 0 \le l \le n-1\}$$

den Raum der *Splines der Ordnung k zur Unterteilung $a = x_0 < \ldots < x_n = b$.*
(2) Der Raum der *natürlichen kubischen Splines* ist durch

$$\widehat{S}_n^3 = \{u \in S_n^3 : u''(a) = u''(b) = 0\}$$

definiert.
(3) $u \in \widehat{S}_n^3$ heißt *interpolierender natürlicher kubischer Spline* von $f \in C(I, \mathbb{R})$ zu den Knoten $x_0, \ldots, x_n$, wenn gilt

$$u(x_i) = f(x_i) \quad \forall 0 \le i \le n.$$

BEMERKUNG I.3.2 (Eigenschaften von Splines). *(1)* Wegen $\dim S_n^k = n + k$ kann ein $u \in S_n^3$ durch die Interpolationsbedingungen aus Definition I.3.1 (3) nicht eindeutig bestimmt sein. Neben den Randbedingungen $u''(a) = u''(b) = 0$, auf die wir uns hier der Einfachheit halber beschränken, kann man andere sinnvolle Randbedingungen wie z.B.

$$u'(a) = f'(a) \quad \text{und} \quad u'(b) = f'(b)$$

oder

$$u'(a) = u'(b) \quad \text{und} \quad u''(a) = u''(b)$$

stellen.
(2) Man kann zeigen, dass die Splines der Ordnung k den Ausdruck $\int_a^b \left|y^{(k-1)}(x)\right|^2 \mathrm{d}x$ in einem geeigneten Unterraum von $C^{k-1}(I, \mathbb{R})$ minimieren.

Wir wollen zeigen, dass die Interpolationsaufgabe aus Definition I.3.1 (3) in $\widehat{S}_n^3$ eindeutig lösbar ist. Dazu müssen wir einige Eigenschaften der Matrix M_n nachweisen.

SATZ I.3.3 (Eigenschaften von M_n). (1) *Für jedes $z \in \mathbb{R}^{n-1}$ gilt*

$$\|z\|_\infty \le \|M_n z\|_\infty .$$

(2) *M_n ist regulär.*
(3) *Es ist $\|M_n\|_\infty \le 3$ und $\|M_n^{-1}\|_\infty \le 1$, wobei $\|\cdot\|_\infty$ die zur Maximumsnorm gehörige Matrixnorm, d.h. die Zeilensummennorm, bezeichnet.*

BEWEIS. *ad (1):* Offensichtlich gilt für alle $1 \leq k \leq n-1$

$$0 < \mu_k, \lambda_k < 1, \quad \mu_k + \lambda_k = 1.$$

Sei $z \in \mathbb{R}^{n-1}$ und $1 \leq i \leq n-1$, so dass

$$|z_i| = \|z\|_\infty$$

ist. Setzen wir zur Abkürzung $z_{-1} = 0$ und $z_n = 0$, so folgt

$$\begin{aligned} \|M_n z\|_\infty &\geq |(M_n z)_i| \\ &= |\mu_i z_{i-1} + 2z_i + \lambda_i z_{i+1}| \\ &\geq 2\,|z_i| - \mu_i\,|z_{i-1}| - \lambda_i\,|z_{i+1}| \\ &\geq \|z\|_\infty\,[2 - \mu_i - \lambda_i] \\ &= \|z\|_\infty\,. \end{aligned}$$

ad (2): Wegen Teil (1) besitzt das homogene Problem nur die triviale Lösung.
ad (3): Folgt direkt aus Teil (1) und dessen Beweis. □

Nach diesen Vorbereitungen wollen wir zeigen, dass es zu gegebenen Daten genau einen interpolierenden natürlichen kubischen Spline gibt. Der Beweis beruht auf folgender Idee:
Die zweite Ableitung eines jeden kubischen Splines ist stückweise, d.h. auf den Teilintervallen, eine lineare Funktion. Damit ist ihr Wert auf jedem Intervall $[x_k, x_{k+1}]$ eindeutig festgelegt durch ihre Werte in den Endpunkten x_k und x_{k+1}. Man betrachtet nun die zweiten Ableitungen in den Punkten $x_1, \ldots, x_{n-1}$ als Parameter, integriert auf jedem Teilintervall die entsprechende lineare Funktion zweimal und beachtet die Interpolationsbedingungen in den Punkten $x_0, \ldots, x_n$. Damit erhält man eine stetige stückweise kubische Funktion, deren stückweise zweite Ableitung stetig ist. Zur Lösung der Interpolationsaufgabe muss man daher nur noch die Stetigkeit der stückweisen ersten Ableitung nachweisen. Dies liefert Bedingungen an die noch zu bestimmenden zweiten Ableitungen in den Punkten $x_1, \ldots, x_{n-1}$.

SATZ I.3.4 (Kubische Spline-Interpolation; Funktion `cubic_spline` in `Numerics`). *Zu jedem $f \in C(I, \mathbb{R})$ gibt es genau einen interpolierenden natürlichen kubischen Spline u. Für $0 \leq k \leq n-1$ gilt*

$$\begin{aligned} u|_{I_k} &= f(x_k) \\ &\quad + (x - x_k)\left\{\frac{f(x_{k+1}) - f(x_k)}{h_k} - \frac{1}{3}m_k h_k - \frac{1}{6}m_{k+1}h_k\right\} \\ &\quad + \frac{1}{2}m_k(x - x_k)^2 \\ &\quad + \frac{1}{6}\frac{m_{k+1} - m_k}{h_k}(x - x_k)^3. \end{aligned} \tag{I.3.1}$$

Dabei ist $m_0 = 0$, $m_n = 0$ *und* $m = (m_1, \ldots, m_{n-1})^t \in \mathbb{R}^{n-1}$ *ist die eindeutige Lösung des linearen Gleichungssystems*

$$M_n m = d \tag{I.3.2}$$

mit

$$d_k = \frac{6}{h_k + h_{k-1}}\{\delta_k - \delta_{k-1}\} \quad , 1 \le k \le n-1,$$

und

$$\delta_k = \frac{f(x_{k+1}) - f(x_k)}{h_k} \quad , 0 \le k \le n-1.$$

Beweis. Wegen Satz I.3.3 besitzt das Gleichungssystem (I.3.2) eine eindeutige Lösung. Seien m_0, …, m_n und u wie angegeben. Aus (I.3.1) folgt durch leichte Rechnung

$$\begin{aligned}
u(x_k) &= f(x_k) && \forall 0 \le k \le n,\\
u''(x_0) &= u''(x_n) = 0,\\
u''(x_k - 0) &= u''(x_k + 0) = m_k && \forall 1 \le k \le n-1,\\
u'(x_k + 0) &= \delta_k - \frac{1}{3}m_k h_k - \frac{1}{6}m_{k+1}h_k, && \forall 1 \le k \le n-1,\\
u'(x_k - 0) &= \delta_{k-1} + \frac{1}{6}m_{k-1}h_{k-1} + \frac{1}{3}m_k h_{k-1} && \forall 1 \le k \le n-1.
\end{aligned}$$

Damit folgt aus (I.3.2) für $1 \le k \le n-1$

$$\begin{aligned}
\frac{6}{h_{k-1}+h_k}[u'(x_k+0) - u'(x_k-0)] &= d_k - \mu_k m_{k-1} - 2m_k - \lambda_k m_{k+1}\\
&= [d - M_n m]_k\\
&= 0.
\end{aligned}$$

Also ist $u \in \widehat{S}_n^3$ und erfüllt die Interpolationsbedingungen. Hieraus folgt die Behauptung. □

Der folgende Satz gibt eine Fehlerabschätzung für die Spline-Interpolation.

Satz I.3.5 (Fehlerabschätzung für die kubische Spline-Interpolation). *Sei* $f \in C^4(I, \mathbb{R})$ *mit* $f''(a) = f''(b) = 0$ *und* u *der interpolierende natürliche kubische Spline zu* f *und den Knoten* x_0, …, x_n. *Dann gilt die Fehlerabschätzung*

$$\|f - u\|_{C(I,\mathbb{R})} \le h^4 \left\|f^{(4)}\right\|_{C(I,\mathbb{R})}.$$

Beweis. Sei $0 \le k \le n-1$. Da das lineare Lagrangesche Interpolationspolynom von $f - u$ zu den Knoten x_k, x_{k+1} das Null-Polynom ist, liefert Satz I.2.3 (S. 13)

$$\|f - u\|_{C(I_k,\mathbb{R})} \le \frac{1}{2}h_k^2 \|f'' - u''\|_{C(I_k,\mathbb{R})}. \tag{I.3.3}$$

Sei p_k das lineare Lagrangesche Interpolationspolynom von f'' zu den Knoten x_k, x_{k+1}. Da p_k und $u''|_{I_k}$ lineare Polynome sind, folgt aus Satz I.2.3 (S. 13)

$$\begin{aligned}\|f''-u''\|_{C(I_k,\mathbb{R})} &\le \|f''-p_k\|_{C(I_k,\mathbb{R})} + \|p_k-u''\|_{C(I_k,\mathbb{R})}\\ &\le \frac{1}{2}h_k^2\left\|f^{(4)}\right\|_{C(I_k,\mathbb{R})}\\ &\quad + \max_{l=k,k+1}\left|f''(x_l)-\underbrace{u''(x_l)}_{=m_l}\right|.\end{aligned}\tag{I.3.4}$$

Aus Satz I.3.3 (1) ergibt sich wegen $f''(x_0)=m_0=0$ und $f''(x_n)=m_n=0$

$$\begin{aligned}&\max_{0\le l\le n}|f''(x_l)-m_l|\\ &\quad=\max_{1\le l\le n-1}|f''(x_l)-m_l|\\ &\quad\le \max_{1\le l\le n-1}\left|\mu_l f''(x_{l-1})+2f''(x_l)+\lambda_l f''(x_{l+1})\right.\\ &\qquad\left.-\frac{6}{h_{l-1}+h_l}\left[\frac{f(x_{l+1})-f(x_l)}{h_l}-\frac{f(x_l)-f(x_{l-1})}{h_{l-1}}\right]\right|.\end{aligned}\tag{I.3.5}$$

Sei $1\le l\le n-1$. Durch Taylor-Entwicklung um x_l folgt, dass es Zahlen $\theta_1,\dots,\theta_4\in(0,1)$ gibt mit

$$\begin{aligned}&\left|\mu_l f''(x_{l-1})+2f''(x_l)+\lambda_l f''(x_{l+1})\right.\\ &\qquad\left.-\frac{6}{h_{l-1}+h_l}\left[\frac{f(x_{l+1})-f(x_l)}{h_l}-\frac{f(x_l)-f(x_{l-1})}{h_{l-1}}\right]\right|\\ &\quad=\left|\frac{1}{2}\frac{h_{l-1}^3}{h_{l-1}+h_l}f^{(4)}(x_l-\theta_1h_{l-1})\right.\\ &\qquad+\frac{1}{2}\frac{h_l^3}{h_{l-1}+h_l}f^{(4)}(x_l+\theta_2h_l)\\ &\qquad-\frac{1}{4}\frac{h_l^3}{h_{l-1}+h_l}f^{(4)}(x_l+\theta_3h_l)\\ &\qquad\left.-\frac{1}{4}\frac{h_{l-1}^3}{h_{l-1}+h_l}f^{(4)}(x_l-\theta_4h_{l-1})\right|\\ &\quad\le\frac{3}{2}h^2\left\|f^{(4)}\right\|_{C(I_{l-1}\cup I_l,\mathbb{R})}.\end{aligned}\tag{I.3.6}$$

Aus der Abschätzung (I.3.3) – (I.3.6) folgt die Behauptung. □

Bemerkung I.3.6. *(1)* Man kann in Satz I.3.5 auf die Voraussetzung $f''(a)=f''(b)=0$ verzichten, wenn man den interpolierenden Spline u in S_n^3 sucht und die zusätzlichen Interpolationsbedingungen

$$u'(a)=f'(a),\quad u'(b)=f'(b)$$

stellt. Die Sätze I.3.4 und I.3.5 bleiben dann gültig, wobei das Gleichungssystem (I.3.2) entsprechend durch Gleichungen für m_0 und m_n zu ergänzen ist.
(2) Aus dem Beweis von Satz I.3.5 ergibt sich die Abschätzung

$$\|f-u\|_{C(I,\mathbb{R})} \le \max_{0\le k\le n-1} h_k^4 \left\|f^{(4)}\right\|_{C(I_k,\mathbb{R})}.$$

Daher sollte die Schrittweite dort klein sein, wo $f^{(4)}$ groß ist. Ein Indikator für $\left\|f^{(4)}\right\|_{C(I_k,\mathbb{R})}$ ist

$$\frac{2}{h_k+h_{k-1}}\left|\frac{m_{k+1}-m_k}{h_k}-\frac{m_k-m_{k-1}}{h_{k-1}}\right|.$$

TABELLE I.3.1. Maximaler Fehler der Spline-Interpolation der Funktionen $|x|$ und $\arctan(x)$ in n äquidistanten Knoten auf dem Intervall $[-1,1]$

n	$\lvert x\rvert$	$\arctan(x)$
5	0.0858	0.00614
9	0.0425	0.00155
17	0.0213	0.00038
33	0.0106	0.00009

BEISPIEL I.3.7. Zur Illustration von Satz I.3.5 geben wir in Tabelle I.3.1 die maximalen Fehler der interpolierenden Splines der Funktionen $|x|$ bzw. $\arctan(x)$ auf dem Intervall $[-1,1]$ in n äquidistanten Knoten an. Sie zeigen deutlich den Einfluss der mangelnden Glattheit der interpolierten Funktion.

I.4. Bézier-Darstellung von Polynomen und Splines

In vielen Anwendungen wie dem Computer Aided Design, der Seitenbeschreibungssprache `Postscript` oder der `picture`-Umgebung von `LaTex` wird die Bézier-Darstellung von Polynomen und Splines benutzt. Sie beruht auf den *Bernstein-Polynomen*.

DEFINITION I.4.1 (Bernstein-Polynome). Die *Bernstein-Polynome* $B_{n,k}$ sind für $n \ge 0$ und $0 \le k \le n$ definiert durch

$$B_{n,k}(x) = \binom{n}{k} x^k (1-x)^{n-k}.$$

Für jedes n bilden die Bernstein-Polynome $B_{n,0}$, …, $B_{n,n}$ eine Basis des Raumes $\mathbb{P}_n$. Jedes Polynom $p \in \mathbb{P}_n$ kann daher eindeutig dargestellt werden in der Form

$$p(x) = \sum_{k=0}^{n} b_k B_{n,k}(x).$$

Dies ist die so genannte *Bézier-Darstellung* von p. Die Koeffizienten b_0, …, b_n heißen die *Bézier-Punkte* von p.

Bemerkung I.4.2 (Ableitung der Bernstein-Polynome). Mit der Konvention $B_{n,k}(x) = 0$ für $k < 0$ und $k > n$ folgt

$$B'_{n,k}(x) = n\left[B_{n-1,k-1}(x) - B_{n-1,k}(x)\right].$$

Damit ergibt sich aus der Bézier-Darstellung

$$p(x) = \sum_{k=0}^{n} b_k B_{n,k}(x)$$

von p die Bézier-Darstellung

$$p'(x) = \sum_{\ell=0}^{n-1} n\left(b_{\ell+1} - b_\ell\right) B_{n-1,\ell}(x).$$

von p'. Insbesondere ist

$$p'(0) = n\left(b_1 - b_0\right) \quad \text{und} \quad p'(1) = n\left(b_n - b_{n-1}\right).$$

Die Auswertung eines Polynoms in Bézier-Darstellung erfolgt mit dem *Algorithmus von de Casteljau*. Zu seiner Beschreibung definieren wir ausgehend von den Bézier-Punkten b_0, …, b_n die Hilfspolynome

$$b_{r,s}(x) = \sum_{k=r}^{s} b_k B_{s-r,k-r}(x), \quad 0 \le r \le s \le n.$$

Dann ist offensichtlich $b_{0,n}(x) = p(x)$ und $b_{r,r}(x) = b_r$, $0 \le r \le n$. Aus den Identitäten

$$\begin{aligned} B_{j+1,0}(x) &= (1-x)B_{j,0}(x), \\ B_{j+1,i}(x) &= (1-x)B_{j,i}(x) + xB_{j,i-1}(x), \\ B_{j+1,j+1}(x) &= xB_{j,j}(x), \end{aligned}$$

erhält man dann die Rekursionsformel

$$b_{r,s}(x) = (1-x)b_{r,s-1}(x) + xb_{r+1,s}(x).$$

Sie erlaubt die rekursive Berechnung von $b_{0,n}(x)$ aus den Bézier-Punkten $b_k = b_{k,k}(x)$, $0 \le k \le n$. Damit ergibt sich Algorithmus I.4.1:

Für die praktische Rechnung ordnet man die Werte $b_{r,s}$ in einem Dreiecksschema an. Die Werte mit gleichem ersten Index stehen in derselben Diagonalen; die Werte mit dem gleichen zweiten Index stehen

Algorithmus I.4.1 Algorithmus von de Casteljau

Gegeben: n Grad des Polynoms p, Bézier-Punkte des Polynoms b_0, …, b_n, Auswertungspunkt $x \in [0,1]$
Gesucht: $p(x)$ Wert des Polynoms p im Punkt x

for $k = 0, 1, \ldots, n$ **do**
    $b_{k,k} \leftarrow b_k$
end for
for $j = 1, 2, \ldots, n$ **do**
    for $k = 0, 1, \ldots, n-j$ **do**
        $s \leftarrow k + j$, $b_{k,s} \leftarrow (1-x)b_{k,s-1} + xb_{k+1,s}$
    end for
end for
$p(x) \leftarrow b_{0,n}$

in derselben Zeile.

$$
\begin{array}{ccccccccc}
b_{0,0} & & & & & & & & \\
 & \searrow & & & & & & & \\
b_{1,1} & \to & b_{0,1} & & & & & & \\
 & \searrow & & \searrow & & & & & \\
b_{2,2} & \to & b_{1,2} & \to & b_{0,2} & & & & \\
 & \searrow & & \searrow & & \searrow & & & \\
b_{3,3} & \to & b_{2,3} & \to & b_{1,3} & \to & b_{0,3} & & \\
\vdots & & \vdots & & \vdots & & \vdots & \ddots &
\end{array}
$$

Das Symbol $\begin{array}{ccc} a & & \\ & \searrow & \\ b & \to & c \end{array}$ steht dabei für die Rechenvorschrift $c = (1-x)a + xb$.

Beispiel I.4.3. Für das kubische Polynom p mit den Bézier-Punkten $b_0 = 2$, $b_1 = 10$, $b_2 = 7$, $b_3 = 0$ und den Auswertungspunkt $x = 0.4$ erhalten wir folgendes Schema

$$
\begin{array}{cccc}
2 & & & \\
10 & 5.2 & & \\
7 & 8.8 & 6.64 & \\
0 & 4.2 & 6.96 & 6.768
\end{array}
$$

und den Funktionswert $p(0.4) = 6.768$.

Beispiel I.4.4 (Bézier-Kurven in `LaTex`). In `LaTex` können mit der `picture`-Umgebung einfache Graphiken erstellt werden. Diese Umgebung stellt insbesondere den Befehl `\qbezier`$(u_1, u_2)(v_1, v_2)(w_1, w_2)$ zur Verfügung, hinter dem sich die Bézier-Darstellung eines quadratischen Polynomes verbirgt. Um dies einzusehen, setze

$$\mathbf{u} = (u_1, u_2)^t, \quad \mathbf{v} = (v_1, v_2)^t, \quad \mathbf{w} = (w_1, w_2)^t$$

und

$$\mathbf{p}(x) = \mathbf{u}B_{2,0}(x) + \mathbf{v}B_{2,1}(x) + \mathbf{w}B_{2,2}(x).$$

Dann ist gemäß Bemerkung I.4.2

$$\begin{aligned} \mathbf{p}(0) &= \mathbf{u}, & \mathbf{p}(1) &= \mathbf{w}, \\ \mathbf{p}'(0) &= 2(\mathbf{v}-\mathbf{u}), & \mathbf{p}'(1) &= 2(\mathbf{w}-\mathbf{v}). \end{aligned}$$

Also ist $\mathbf{v}$ der Schnittpunkt der Tangenten an die Kurve $x \mapsto \mathbf{p}(x)$ in den Punkten $\mathbf{u} = \mathbf{p}(0)$ und $\mathbf{w} = \mathbf{p}(1)$. Der Polygonzug mit den Ecken $\mathbf{u}$, $\mathbf{v}$ und $\mathbf{w}$ wird in diesem Zusammenhang als das *Kontrollpolygon* der Kurve $x \mapsto \mathbf{p}(x)$ bezeichnet. Der Algorithmus von de Casteljau hat für dieses spezielle Beispiel die Form

$$\begin{array}{ccccccc} \mathbf{u} & & & & & & \\ & \searrow & & & & & \\ \mathbf{v} & \to & \mathbf{y} = (1-x)\mathbf{u} + x\mathbf{v} & & & & \\ & \searrow & & & \searrow & & \\ \mathbf{w} & \to & \mathbf{z} = (1-x)\mathbf{v} + x\mathbf{w} & \to & \mathbf{p} = (1-x)\mathbf{y} + x\mathbf{z} \end{array}$$

und lässt sich graphisch wie folgt interpretieren: $\mathbf{y}$ teilt die Strecke $\overline{\mathbf{uv}}$ im Verhältnis $1-x : x$, $\mathbf{z}$ teilt die Strecke $\overline{\mathbf{vw}}$ im Verhältnis $1-x : x$ und $\mathbf{p}$ teilt die Strecke $\overline{\mathbf{yz}}$ im Verhältnis $1-x : x$. Abbildung I.4.1 illustriert dieses Vorgehen am Beispiel $\mathbf{u} = (0,0)^t$, $\mathbf{v} = (0,4)^t$, $\mathbf{w} = (4,4)^t$ für die Werte $x = \frac{1}{4}$, $x = \frac{1}{2}$ und $x = \frac{3}{4}$.

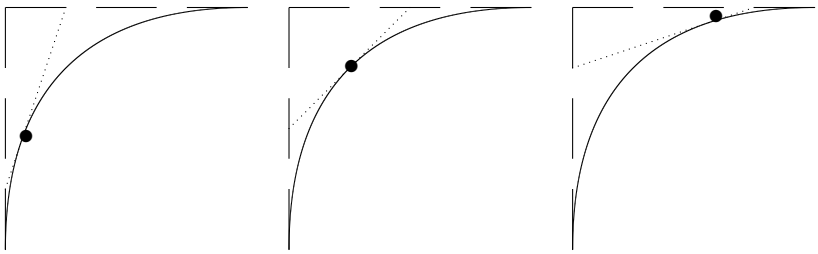

ABBILDUNG I.4.1. Bézier-Kurve mit Kontrollpolygon (gestrichelt), Hilfsstrecke des de Casteljau Algorithmus (gepunktet) und Auswertungspunkt (•) für die Auswertungspunkte $x = \frac{1}{4}$ (links), $x = \frac{1}{2}$ (Mitte) und $x = \frac{3}{4}$ (rechts)

Wir betrachten nun die Bézier-Darstellung eines kubischen Splines u zu der Unterteilung $a = x_0 < x_1 < \ldots < x_n = b$. Auf dem Teilintervall $I_k = [x_k, x_{k+1}]$, $0 \le k \le n-1$, der Länge $h_k = x_{k+1} - x_k$ schreiben wir dazu

$$u|_{I_k}(x) = \sum_{\ell=0}^{3} b_{k,\ell} B_{3,\ell}\left(\frac{x - x_k}{h_k}\right).$$

Dann gilt

$$\begin{aligned}
(u|_{I_k})(x_k) &= b_{k,0},\\
(u|_{I_k})(x_{k+1}) &= b_{k,3},\\
(u|_{I_k})'(x_k) &= \frac{3}{h_k}(b_{k,1}-b_{k,0}),\\
(u|_{I_k})'(x_{k+1}) &= \frac{3}{h_k}(b_{k,3}-b_{k,2}),\\
(u|_{I_k})''(x_k) &= \frac{6}{h_k^2}(b_{k,2}-2b_{k,1}+b_{k,0}),\\
(u|_{I_k})''(x_{k+1}) &= \frac{6}{h_k^2}(b_{k,3}-2b_{k,2}+b_{k,1}).
\end{aligned}$$

Anders als im vorigen Abschnitt betrachten wir jetzt die Interpolationsaufgabe:
Finde zu den Knoten $x_0 < \ldots < x_n$ und den Daten $f(x_0)$, …, $f(x_n)$ und $f'(x_0)$, $f'(x_n)$ einen kubischen Spline u mit

$$u(x_k) = f(x_k), \quad k = 0, 1, \ldots, n$$

und

$$u'(x_0) = f'(x_0), \quad u'(x_n) = f'(x_n).$$

Man beachte, dass wir jetzt in den Randpunkten $x_0 = a$ und $x_n = b$ Bedingungen an u' statt an u'' im vorigen Abschnitt stellen.

Wir bezeichnen jetzt mit $M_k = u'(x_k)$, $0 \le k \le n$, die Werte der ersten Ableitung von u in den Knoten. Aus der Interpolationsbedingung $u(x_k) = f(x_k)$ folgt für jedes $k \in \{0, \ldots, n-1\}$

$$b_{k,0} = f(x_k), \quad b_{k,3} = f(x_{k+1}).$$

Aus den Formeln für $(u|_{I_k})'$ erhalten wir dann

$$\begin{aligned}
b_{k,1} = b_{k,0} + \frac{h_k}{3}M_k \quad &= f(x_k) + \frac{h_k}{3}M_k,\\
b_{k,2} = b_{k,3} - \frac{h_k}{3}M_{k+1} \quad &= f(x_{k+1}) - \frac{h_k}{3}M_{k+1}.
\end{aligned}$$

Damit sind die Bézier-Punkte $b_{k,i}$, $0 \le k \le n-1$, $0 \le i \le 3$, durch die Daten $f(x_0)$, …, $f(x_n)$ und die Werte M_0, …, M_n ausgedrückt. Aus der Stetigkeit von u'' in den Punkten x_1, …, x_{n-1} erhalten wir schließlich für $k = 1, \ldots, n-1$ die Bedingung

$$\begin{aligned}
\frac{6}{h_{k-1}^2}(b_{k-1,3} - 2b_{k-1,2} + b_{k-1,1}) &= (u|_{I_{k-1}})''(x_k)\\
&= (u|_{I_k})''(x_k)\\
&= \frac{6}{h_k^2}(b_{k,2} - 2b_{k,1} + b_{k,0}).
\end{aligned}$$

Dies liefert nach einigen Umformungen für $1 \leq k \leq n-1$ die Bestimmungsgleichungen

$$\lambda_k M_{k-1} + 2M_k + \mu_k M_{k+1} = \frac{3}{h_k + h_{k-1}} \left[h_{k-1} \frac{f(x_{k+1}) - f(x_k)}{h_k} + h_k \frac{f(x_k) - f(x_{k-1})}{h_{k-1}} \right].$$

Dabei ist zu beachten, dass $M_0 = f'(x_0)$ und $M_n = f'(x_n)$ durch die Interpolationsbedingungen festgelegt sind.

Insgesamt erhalten wir damit Algorithmus I.4.2. Die Auswertung des Splines auf den Teilintervallen in Schritt 11 von Algorithmus I.4.2 erfolgt dabei mit dem Algorithmus von de Casteljau, Algorithmus I.4.1.

Algorithmus I.4.2 Kubische Spline-Interpolation in Bézier-Darstellung

Gegeben: Knoten $x_0 < \ldots < x_n$, Daten $f(x_0)$, …, $f(x_n)$ und $f'(x_0)$, $f'(x_n)$

Gesucht: Kubischer Spline u in Bézier-Darstellung mit $u'(x_0) = f'(x_0)$, $u(x_0) = f(x_0)$, $u(x_1) = f(x_1)$, …, $u(x_n) = f(x_n)$, $u'(x_n) = f'(x_n)$

1: **for** $k = 0, \ldots, n-1$ **do**
2: $\quad h_k \leftarrow x_{k+1} - x_k$, $\delta_k \leftarrow \frac{f(x_{k+1}) - f(x_k)}{h_k}$
3: **end for**
4: **for** $k = 1, \ldots, n-1$ **do**
5: $\quad D_k \leftarrow \frac{3}{h_k + h_{k-1}} [h_{k-1}\delta_k + h_k \delta_{k-1}]$
6: $\quad \mu_k \leftarrow \frac{h_{k-1}}{h_k + h_{k-1}}$, $\lambda_k \leftarrow \frac{h_k}{h_k + h_{k-1}}$
7: **end for**
8: $M_0 \leftarrow f'(x_0)$, $M_n \leftarrow f'(x_n)$
9: Löse das lineare Gleichungssystem

$$\begin{pmatrix} 2 & \mu_1 & & & 0 \\ \lambda_2 & 2 & \mu_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \lambda_{n-2} & 2 & \mu_{n-2} \\ 0 & & & \lambda_{n-1} & 2 \end{pmatrix} \begin{pmatrix} M_1 \\ M_2 \\ \vdots \\ M_{n-2} \\ M_{n-1} \end{pmatrix} = \begin{pmatrix} D_1 - \lambda_1 f'(x_0) \\ D_2 \\ \vdots \\ D_{n-2} \\ D_{n-1} - \mu_{n-1} f'(x_n) \end{pmatrix}.$$

10: **for** $k = 0, \ldots, n-1$ **do**
11: $\quad u|_{I_k}(x) \leftarrow \sum_{\ell=0}^{3} b_{k,\ell} B_{3,\ell} \left(\frac{x - x_k}{h_k} \right)$
12: **end for**

I.5. Trigonometrische Interpolation

In diesem Paragraphen betrachten wir das Interpolationsproblem aus Beispiel I.1.3(4) (S. 10). Dabei beschränken wir uns auf den Spezialfall äquidistanter Knoten $x_k = \frac{2\pi k}{2n+1}$, $0 \leq k \leq 2n$, $n \in \mathbb{N}$.

Der folgende Satz zeigt, dass das Interpolationsproblem wohl gestellt ist und gibt eine explizite Formel für die Lösung.

SATZ I.5.1 (Darstellung des trigonometrischen Interpolationspolynomes). *Sei*

$$p(x) = \frac{a_0}{2} + \sum_{k=1}^{n} [a_k \cos(kx) + b_k \sin(kx)]$$

das trigonometrische Interpolationspolynom von $f \in C([0, 2\pi], \mathbb{R})$ *zu den Knoten* $x_k = \frac{2\pi k}{2n+1}$, $0 \le k \le 2n$. *Dann gilt*

$$a_k = \frac{2}{2n+1} \sum_{l=0}^{2n} f(x_l) \cos(kx_l), \quad 0 \le k \le n,$$

$$b_k = \frac{2}{2n+1} \sum_{l=0}^{2n} f(x_l) \sin(kx_l), \quad 1 \le k \le n.$$

BEWEIS. Bezeichne mit $(\cdot\,,\cdot)$ das euklidische Skalarprodukt auf dem $\mathbb{R}^{2n+1}$. Für $k \in \mathbb{Z}$ und $l \in \mathbb{Z}^*$ definiere die Vektoren $c_k, s_l \in \mathbb{R}^{2n+1}$ durch

$$c_{k,j} = \cos(kx_{j-1}), \quad s_{l,j} = \sin(lx_{j-1}).$$

Aus der Identität

$$\sum_{j=0}^{2n} e^{ikx_j} = \sum_{j=0}^{2n} \left(e^{i\frac{2\pi k}{2n+1}}\right)^j = \delta_{k,0}(2n+1)$$

und den Additionstheoremen für den Sinus und Cosinus folgt für alle relevanten Indizes

$$(c_k, c_l) = \frac{1}{2}\left[(c_{k+l}, c_0) + (c_{k-l}, c_0)\right] = \frac{2n+1}{2}\left[\delta_{k,0}\delta_{l,0} + \delta_{k,l}\right],$$

$$(s_k, s_l) = \frac{1}{2}\left[(c_{k-l}, c_0) - (c_{k+l}, c_0)\right] = \frac{2n+1}{2}\delta_{k,l},$$

$$(c_k, s_l) = \frac{1}{2}\left[(s_{l+k}, c_0) + (s_{l-k}, c_0)\right] = 0.$$

Definiere die Vektoren $P, F \in \mathbb{R}^{2n+1}$ durch

$$P_i = p(x_{i-1}), \quad F_i = f(x_{i-1}), \qquad 1 \le i \le 2n+1.$$

Dann gilt $P = F$ und

$$P = \frac{a_0}{2}c_0 + \sum_{k=1}^{n}[a_k c_k + b_k s_k].$$

Zusammen mit den obigen Orthogonalitätsbeziehungen folgt hieraus

$$\frac{2n+1}{2}a_k = (P, c_k) = (F, c_k) \quad \forall 0 \le k \le n$$

$$\frac{2n+1}{2}b_k = (P, s_k) = (F, s_k) \quad \forall 1 \le k \le n$$

und damit die Behauptung. □

Damit ist dieses Interpolationsproblem gelöst. Bleibt noch das Problem, die Formeln aus Satz I.5.1 und – bei bekannten Koeffizienten – das Interpolationspolynom effizient auszuwerten. Dies bereitet der folgende Satz vor.

SATZ I.5.2. *Seien* $z \in \mathbb{R} \setminus \pi\mathbb{Z}$ *und*

$$\begin{aligned} u_j &= \frac{1}{\sin z} \sum_{k=j}^{N-1} y_k \sin((k-j+1)z), \quad 0 \le j \le N-1, \\ u_N &= 0, \\ u_{N+1} &= 0 \end{aligned}$$

mit $y_k \in \mathbb{R}$, $0 \le k \le N-1$, *und* $N \in \mathbb{N}^*$. *Dann gilt:*

(1) $u_j = y_j + 2(\cos z)u_{j+1} - u_{j+2}$ *für* $j = N-1, N-2, \dots, 0$,

(2) $\displaystyle\sum_{k=1}^{N-1} y_k \sin kz = u_1 \sin z$,

(3) $\displaystyle\sum_{k=0}^{N-1} y_k \cos kz = y_0 + u_1 \cos z - u_2$.

BEWEIS. *ad (1):* Wegen $\sin 0 = 0$ und $\sin(\alpha+\beta) + \sin(\alpha-\beta) = 2\sin\alpha\cos\beta$ folgt

$$\begin{aligned} & y_j + 2(\cos z)u_{j+1} - u_{j+2} \\ &= \frac{1}{\sin z}\Big[y_j \sin z + \sum_{k=j+1}^{N-1} y_k[2\cos z \sin(k-j)z - \sin(k-j-1)z]\Big] \\ &= \frac{1}{\sin z}[y_j \sin z + \sum_{k=j+1}^{N-1} y_k \sin(k-j+1)z] \\ &= u_j. \end{aligned}$$

ad (2): Folgt direkt aus der Definition.
ad (3): Wegen $\sin 0 = 0$ und $\sin(\alpha-\beta) = \sin\alpha\cos\beta - \sin\beta\cos\alpha$ folgt

$$\begin{aligned} & y_0 + u_1 \cos z - u_2 \\ &= y_0 + \frac{1}{\sin z} \sum_{k=1}^{N-1} y_k[\cos z \sin(kz) - \sin(k-1)z] \\ &= y_0 + \sum_{k=1}^{N-1} y_k \cos(kz). \end{aligned}$$

□

Setzen wir $N = 2n+1$, $y_l = f(x_l)$ und $z = \frac{2\pi k}{2n+1}$, $1 \le k \le n$, so liefert Satz I.5.2 Algorithmus I.5.1 zur Berechnung der Koeffizienten des trigonometrischen Interpolationspolynoms.

Algorithmus I.5.1 Algorithmus von Goertzel zur Berechnung der Koeffizienten des trigonometrischen Interpoaltionspolynomes

Gegeben: Daten $y_l = f(\frac{2\pi l}{2n+1})$, $0 \le l \le 2n$
Gesucht: Koeffizienten a_k, b_l, $0 \le k \le n$, $1 \le l \le n$

$a_0 \leftarrow \dfrac{2}{2n+1} \sum_{l=0}^{2n} y_l$
for $k = 1, \ldots, n$ **do**
    $z \leftarrow \dfrac{2\pi k}{2n+1}$, $c \leftarrow \cos(z)$, $cc \leftarrow 2c$
    $s \leftarrow \sin z$, $u_{2n+1} \leftarrow 0$, $u_{2n+2} \leftarrow 0$
    for $j = 2n, 2n-1, \ldots, 1$ **do**
        $u_j \leftarrow y_j + cc \cdot u_{j+1} - u_{j+2}$
    end for
    $a_k \leftarrow (y_0 + cu_1 - u_2)\dfrac{2}{2n+1}$
    $b_k \leftarrow s \cdot u_1 \cdot \dfrac{2}{2n+1}$
end for

Ganz analog kann man das Polynom

$$p(x) = \frac{a_0}{2} + \sum_{k=1}^{n} [a_k \cos(kx) + b_k \sin(kx)]$$

an einer gegebenen Stelle x^* auswerten.

Algorithmus I.5.2 Algorithmus von Goertzel zur Auswertung des trigonometrischen Interpolationspolynomes

Gegeben: Koeffizienten a_k, b_l, $0 \le k \le n$, $1 \le l \le n$, Auswertungspunkt x

Gesucht: Wert $p = \frac{a_0}{2} + \sum_{k=1}^{n} [a_k \cos(kx) + b_k \sin(kx)]$

$u_{n+2} \leftarrow 0$, $u_{n+1} \leftarrow 0$
$c \leftarrow \cos(x)$, $cc \leftarrow 2c$, $s \leftarrow \sin(x)$
for $j = n, n-1, \ldots, 1$ **do**
    $u_j \leftarrow b_j + cc \cdot u_{j+1} - u_{j+2}$
end for
$r \leftarrow s \cdot u_1$
for $j = n, n-1, \ldots, 1$ **do**
    $u_j \leftarrow a_j + cc \cdot u_{j+1} - u_{j+2}$
end for
$q \leftarrow \frac{1}{2}a_0 + c \cdot u_1 - u_2$
$p \leftarrow q + r$

BEMERKUNG I.5.3. *(1)* Die Algorithmen I.5.1 und I.5.2 sind für kleine Werte von z bzw. x numerisch instabil. Zur Vermeidung dieser Instabilität gibt es eine einfache Modifikation von Reinsch, auf die wir hier aber aus Zeitgründen nicht eingehen [3, §2.3.3].
(2) Die Berechnung der Koeffizienten a_k, b_l mit Algorithmus I.5.1 erfordert $O(n^2)$ Operationen. Die Auswertung des trigonometrischen Interpolationspolynoms mit Algorithmus I.5.2 erfordert $O(n)$ Operationen.

Zum Abschluss dieses Paragraphen wollen wir noch kurz das Prinzip der *schnellen Fourier-Transformation* (kurz *FFT*) erläutern [1, §8.5.3], [2, Algorithmus 7.28], [3, §2.3.3]. Dazu betrachten wir das folgende Interpolationsproblem:
Finde zu gegebenem $f \in C([0, 2\pi], \mathbb{R})$ und $N = 2^m$

$$p(x) = \frac{A_0}{2} + \sum_{k=1}^{N-1} A_k \cos(kx) + \frac{A_N}{2} \cos(Nx) + \sum_{k=1}^{N-1} B_k \sin(kx) \tag{I.5.1}$$

mit

$$p(x_j) = f(x_j), \quad x_j = \frac{2\pi j}{2N},\ 0 \le j \le 2N-1.$$

Wie in Beispiel I.1.3(4) (S. 10) kann man zeigen, dass dieses Interpolationsproblem wohl gestellt ist. Wie in Satz I.5.1 folgt

$$A_k = \frac{1}{N} \sum_{j=0}^{2N-1} f(x_j) \cos(kx_j), \quad 0 \le k \le N,$$

$$B_k = \frac{1}{N} \sum_{j=0}^{2N-1} f(x_j) \sin(kx_j), \quad 1 \le k \le N-1.$$

Diese Formeln können mit Algorithmus I.5.1 in $O(N^2)$ Operationen ausgewertet werden. Ebenso kann p bei gegebenen Koeffizienten mit Algorithmus I.5.2 in $O(N^2)$ Operationen an einer Stelle x^* ausgewertet werden. Die schnelle Fourier-Transformation reduziert den Aufwand für diese beiden Aufgaben auf jeweils $O(Nm) = O(N \log_2 N)$ Operationen. Dazu muss allerdings x^* einer der Knoten x_j, $0 \le j \le 2N-1$, sein

Man fragt sich an dieser Stelle natürlich zu Recht, welchen Sinn es macht, eine Funktion in einigen Punkten zu interpolieren und anschließend das Interpolationspolynom genau in den Interpolationsknoten auszuwerten. Ein wichtiger Anwendungsbereich, der wesentlich auf einer derartigen Vorgehensweise beruht, ist die Datenkompression und Datenübertragung. Zur prinzipiellen Beschreibung der Vorgehensweise betrachten wir einen MP3-Spieler:
Am Anfang steht ein analoges Eingangssignal, hier Musik. Dieses entspricht der Funktion f. Für die Übertragung oder Speicherung wird das Signal digitalisiert. Dies entspricht der Interpolation. Ergebnis sind die Koeffizienten des Interpolationspolynoms. Wegen des großen Frequenzbereiches des Eingangssignals ist die Zahl N der Koeffizienten groß.

Um Speicher- und Übertragungskapazität einzusparen, werden die digitalen Daten komprimiert. Dabei werden „uninteressante“ Koeffizienten ausgeblendet und zu Null gesetzt. Dieser Ausblendprozess basiert auf einem komplexen, nichtlinearen Optimierungsproblem. Er reduziert zwar nicht den Grad N des Polynomes, aber wesentlich die Zahl der zu speichernden oder zu übertragenden Koeffizienten. Beim Empfänger, hier Musikhörer, müssen die digitalen Daten wieder in ein analoges Ausgangssignal, hier Musik, umgewandelt werden. Dies entspricht der Auswertung des Polynomes. Da sich der Grad nicht geändert hat, muss das Polynom in den ursprünglichen Interpolationsknoten ausgewertet werden.

Zur weiteren Erläuterung der Idee der schnellen Fourier-Transformation nehmen wir an, dass $m \geq 1$ ist, und setzen $M = \frac{N}{2}$. Bezeichne mit q, r die beiden trigonometrischen Polynome vom Grad M, d.h. von der Form (I.5.1) mit N ersetzt durch M, die die Interpolationsbedingungen

$$\begin{aligned} q(x_{2j}) &= (-1)^j f(x_{2j}), \\ r(x_{2j}) &= (-1)^j f(x_{2j+1}) \end{aligned}$$

für $0 \leq 1 \leq 2M-1$ erfüllen. Wegen

$$\begin{aligned} \cos(Mx_{2j}) &= (-1)^j, \qquad \sin(Mx_{2j}) = 0, \\ \cos(Mx_{2j+1}) &= 0, \qquad \sin(Mx_{2j+1}) = (-1)^j, \end{aligned}$$

für $0 \leq j \leq 2M-1$ folgt aus der eindeutigen Lösbarkeit des Interpolationsproblems

$$p(x) = \cos(Mx)q(x) + \sin(Mx)r\left(x - \frac{\pi}{N}\right). \tag{I.5.2}$$

Damit ist das Interpolationsproblem zu N auf zwei Interpolationsprobleme zu $M = \frac{N}{2}$ reduziert. Diese Reduktion kann rekursiv fortgesetzt werden.

Nehme nun an, dass die Koeffizienten von q und r bekannt sind. Wegen

$$\begin{aligned} p(x_{2j}) &= (-1)^j q(x_{2j}) \\ p(x_{2j+1}) &= (-1)^j r(x_{2j}) \end{aligned}$$

für $0 \leq j \leq 2M-1$ ist die Auswertung von p in $2N$ Punkten auf die Auswertung von q und r in jeweils $2M$ Punkten reduziert. Auch diese Reduktion kann rekursiv fortgesetzt werden.

Damit diese beiden rekursiven Prozesse effizient durchgeführt werden können, benötigen wir Beziehungen zwischen den Koeffizienten von

p, q und r, die mit geringem Aufwand ausgewertet werden können. Dazu schreiben wir

$$
\begin{aligned}
q(x) &= \frac{a_0}{2} + \sum_{k=1}^{M-1} a_k \cos(kx) + \frac{a_M}{2}\cos(Mx) \\
&\quad + \sum_{k=1}^{M-1} b_k \sin(kx), \\
r(x) &= \frac{\widetilde{\alpha}_0}{2} + \sum_{k=1}^{M-1} \widetilde{\alpha}_k \cos(kx) + \frac{\widetilde{a}_M}{2}\cos(Mx) \\
&\quad + \sum_{k=1}^{M-1} \widetilde{\beta}_k \sin(kx), \\
r\left(x - \frac{\pi}{N}\right) &= \frac{\alpha_0}{2} + \sum_{k=1}^{M-1} \alpha_k \cos(kx) \\
&\quad + \sum_{k=1}^{M-1} \beta_k \sin(kx) + \frac{\beta_M}{2}\sin(Mx).
\end{aligned}
\tag{I.5.3}
$$

Setze

$$\widetilde{\beta}_0 = 0, \quad \beta_0 = 0, \quad b_0 = 0.$$

Dann folgt mit den Additionstheoremen für Sinus und Cosinus durch Koeffizientenvergleich in (I.5.3)

$$
\begin{aligned}
\alpha_k &= \widetilde{\alpha}_k \cos\Big(\frac{k\pi}{N}\Big) - \widetilde{\beta}_k \sin\Big(\frac{k\pi}{N}\Big), \quad 0 \le k \le M-1, \\
\beta_k &= \widetilde{\beta}_k \cos\Big(\frac{k\pi}{N}\Big) + \widetilde{\alpha}_k \sin\Big(\frac{k\pi}{N}\Big), \quad 1 \le k \le M,
\end{aligned}
$$

bzw.

$$
\begin{aligned}
\widetilde{\alpha}_k &= \alpha_k \cos\Big(\frac{k\pi}{N}\Big) + \beta_k \sin\Big(\frac{k\pi}{N}\Big), \quad 0 \le k \le M, \\
\widetilde{\beta}_k &= \beta_k \cos\Big(\frac{k\pi}{N}\Big) - \alpha_k \sin\Big(\frac{k\pi}{N}\Big), \quad 1 \le k \le M-1.
\end{aligned}
$$

Ebenso folgt aus (I.5.2) durch Koeffizientenvergleich mit (I.5.1) und (I.5.3)

$$
\begin{aligned}
A_{M\pm l} &= \frac{1}{2}a_l \mp \frac{1}{2}\beta_l, \quad 0 \le l \le M, \\
B_{M\pm l} &= \frac{1}{2}\alpha_l \pm \frac{1}{2}b_l, \quad 0 \le l \le M-1,
\end{aligned}
$$

bzw.

$$
\begin{aligned}
a_l &= A_{M+l} + A_{M-l}, \quad 0 \le l \le M, \\
\beta_l &= B_{M+l} - B_{M-l}, \quad 1 \le l \le M-1, \\
\alpha_l &= B_{M+l} + B_{M-l}, \quad 0 \le l \le M-1, \\
\beta_l &= A_{M-l} - A_{M+l}, \quad 1 \le l \le M.
\end{aligned}
$$

KAPITEL II

Numerische Integration

In diesem Kapitel behandeln wir Verfahren zur numerischen Berechnung bestimmter Integrale. Nach einer allgemeinen Einführung beschränken wir uns auf den Spezialfall eindimensionaler Integrale. Wir betrachten drei Typen von Verfahren:

- Newton-Cotes-Verfahren,
- Gauß-Verfahren, insbesondere die Legendre-Formeln,
- Extrapolationsverfahren, insbesondere das Romberg-Verfahren.

Zum Abschluss gehen wir noch kurz auf die Behandlung spezieller Integrale, insbesondere von solchen mit singulären Integranden ein.

II.1. Das allgemeine Integrationsproblem

Zunächst geben wir eine allgemeine Definition des Integrationsproblemes und von Quadraturformeln.

DEFINITION II.1.1 (Allgemeine Quadraturformel). Sei $M \subset \mathbb{R}^d$ messbar. Eine Funktion $\omega \in L^1(M, \mathbb{R})$ mit $\omega > 0$ heißt *Gewichtsfunktion*. Wir definieren für alle $f \in C(M, \mathbb{R})$

$$I_{M,\omega}(f) = \int_M f(x)\omega(x)\mathrm{d}x.$$

Ein Ausdruck der Form

$$Q_{M,\omega,n}(f) = \sum_{k=0}^{n} c_k f(x_k)$$

mit $c_k \in \mathbb{R}, x_k \in M$ heißt *Quadraturformel* (für $I_{M,\omega}$).
Ist aus dem Zusammenhang die spezielle Wahl von M oder ω, insbesondere $\omega = 1$, klar, so lassen wir den Index M oder ω bei $I_{M,\omega}$ und $Q_{M,\omega,n}$ fort.
Die Punkte x_k heißen *Knoten*, die Zahlen c_k *Gewichte* der Quadraturformel.
Die Quadraturformel heißt *offen*, wenn alle Knoten im Innern von M sind; sonst heißt sie *abgeschlossen*.
Die Quadraturformel hat die *Ordnung* $K \in \mathbb{N}$, wenn gilt

$$I_{M,\omega}(p) = Q_{M,\omega,n}(p)$$

für alle Polynome vom Grad $\leq K$. Ist $n+1 \leq \binom{K+d}{d}$, so sprechen wir von einer *Newton-Cotes-Formel*; ist $2n+2 \leq \binom{K+d}{d}$, sprechen wir von einer *Gaußschen Formel*.

BEMERKUNG II.1.2 (Stetigkeit der Integration und der Quadraturformel, maximale Ordnung einer Quadraturformel). *(1)* Es ist $I_{M,\omega}$, $Q_{M,\omega,n} \in \mathcal{L}(C(M,\mathbb{R}),\mathbb{R})$ und

$$\|I_{M,\omega}\|_{\mathcal{L}(C(M,\mathbb{R}),\mathbb{R})} = \|\omega\|_{L^1(M,\mathbb{R})},$$
$$\|Q_{M,\omega,n}\|_{\mathcal{L}(C(M,\mathbb{R}),\mathbb{R})} = \sum_{k=0}^{n} |c_k|.$$

(2) Ist M irgendein Intervall, ω irgendeine Gewichtsfunktion und $Q_{M,\omega,n}$ irgendeine Quadraturformel mit $n+1$ Knoten für $I_{M,\omega}$, so hat $Q_{M,\omega,n}$ höchstens die Ordnung $2n+1$.

BEWEIS. *ad (1):* Die Linearität der Operatoren ist offensichtlich. Gleiches gilt für die Beziehung „$\leq$“ für die Operatornormen. Für die Beziehung „$\geq$“ wähle $f = 1$ bzw. $f \in C(M,\mathbb{R})$ mit $\|f\|_{C(M,\mathbb{R})} = 1$ und $f(x_k) = \operatorname{sgn}(c_k)$.
ad (2): Seien $x_0, \ldots, x_n$ die Knoten der Quadraturformel. Dann ist

$$p(x) = \prod_{i=0}^{n} (x - x_i)^2$$

ein Polynom vom Grad $2n+2$ mit

$$I_{M,\omega}(p) > 0 \quad \text{und} \quad Q_{M,\omega,n}(p) = 0.$$

Also kann die Ordnung nicht $2n+2$ sein. □

Als nächstes betrachten wir einige konkrete Beispiele.

BEISPIEL II.1.3. *(1)* $Q_{M,\omega,0}(f) = c_0 f(x_0)$ mit $x_0 \in M$ und $c_0 = \int_M \omega$ hat die Ordnung $K = 0$. Ist insbesondere x_0 der Schwerpunkt von M, so spricht man von der *Mittelpunktsregel*.
(2) Sei $\omega = 1$ und $M = \{x \in \mathbb{R}^2 : x_i \geq 0, x_1 + x_2 \leq 1\}$. Definiere

$$x_0 = \left(\frac{1}{3}, \frac{1}{3}\right), x_1 = \left(\frac{1}{2}, \frac{1}{2}\right), x_2 = \left(0, \frac{1}{2}\right), x_3 = \left(\frac{1}{2}, 0\right)$$

und

$$Q_0(f) = \frac{1}{2} f(x_0), \quad Q_1(f) = \frac{1}{6} \sum_{i=1}^{3} f(x_i).$$

Dann haben Q_0 und Q_1 die Ordnung 1 bzw. 2.
(3) Seien $\omega = 1$, $M = [a,b] \times [c,d]$ und

$$Q_1(f) = \sum_{i=0}^{n} c_i f(x_i), \quad Q_2(f) = \sum_{j=0}^{m} d_j f(y_j)$$

zwei Quadraturformeln der Ordnung K für $I_{[a,b]}$ bzw. $I_{[c,d]}$. Dann ist

$$Q(f) = \sum_{i=0}^{n} \sum_{j=0}^{m} c_i d_j f(x_i, y_j)$$

eine Quadraturformel der Ordnung K für I_M.
(4) $Q_0(f) = f(\frac{1}{2})$ *(Mittelpunktsregel)* und

$$Q_1(f) = T(f) = \frac{1}{2}[f(0) + f(1)] \quad \textit{(Trapezregel)}$$

sind Quadraturformeln der Ordnung 1 für $I_{[0,1]}$.
(5) Die wichtigsten Gewichtsfunktionen und zugehörigen Intervalle im Fall $d = 1$ sind

$$\begin{aligned}
&\omega(x) = 1,\ M \subset \mathbb{R} \text{ beliebig} && \textit{(Legendre)},\\
&\omega(x) = (1-x^2)^{-\frac{1}{2}},\ M = (-1,1) && \textit{(Čebyšev 1. Art)},\\
&\omega(x) = (1-x^2)^{\frac{1}{2}},\ M = (-1,1) && \textit{(Čebyšev 2. Art)},\\
&\omega(x) = e^{-x},\ M = \mathbb{R}_+^* && \textit{(Laguerre)},\\
&\omega(x) = e^{-x^2},\ M = \mathbb{R} && \textit{(Hermite)}.
\end{aligned}$$

Der folgende Satz überträgt für affine Transformationen den Transformationssatz für Integrale auf Quadraturformeln.

Satz II.1.4 (Affine Transformation von Quadraturformeln). *Seien* $b \in \mathbb{R}^d$, $B \in \mathcal{GL}(\mathbb{R}^d)$, $\varphi(x) = b + Bx$ *und*

$$Q_{M,\omega,n}(f) = \sum_{k=0}^{n} c_k f(x_k)$$

eine Quadraturformel für $I_{M,\omega}$. *Dann ist*

$$Q_{\varphi(M),\omega\circ\varphi^{-1},n}(f) = \sum_{k=0}^{n} |\det B|\, c_k f(\varphi(x_k))$$

eine Quadraturformel für $I_{\varphi(M),\omega\circ\varphi^{-1}}$, *die die gleiche Ordnung hat wie* $Q_{M,\omega,n}$.

Beweis. Die Abbildung

$$C(M,\mathbb{R}) \ni f \mapsto f \circ \varphi^{-1} \in C(\varphi(M),\mathbb{R})$$

ist ein Isomorphismus, der Polynome in Polynome gleichen Grades überführt. Aus dem Transformationssatz für Integrale folgt

$$\begin{aligned}
&I_{\varphi(M),\omega\circ\varphi^{-1}}(f\circ\varphi^{-1}) - Q_{\varphi(M),\omega\circ\varphi^{-1},n}(f\circ\varphi^{-1})\\
&\quad = |\det B|\,[I_{M,\omega}(f) - Q_{M,\omega,n}(f)].
\end{aligned}$$

□

Wegen Satz II.1.4 kann man sich bei der Konstruktion von Quadraturformeln auf einfache „Referenzgebiete“ beschränken. Für $d \geq 2$ sind diese häufig der *Referenzwürfel* $[0,1]^d$ oder der *Referenzsimplex* $\{x \in \mathbb{R}^d, x_i \geq 0, \sum_{i=1}^d x_i \leq 1\}$. Komplizierte Gebiete behandelt man,

indem man sie in einfache Teilgebiete zerlegt, die man auf ein Referenzgebiet transformiert. Wir führen diesen Ansatz im Folgenden nur für $d = 1$ aus. Der Einfachheit halber beschränken wir uns dabei auf die Gewichtsfunktion $\omega = 1$. Die folgenden Sätze gelten mit den offensichtlichen Modifikationen aber auch für allgemeine Gewichtsfunktionen.

DEFINITION II.1.5 (Zusammengesetzte Quadraturformel). Seien $a, b \in \mathbb{R}$, $a < b$, $m \in \mathbb{N}^*$, $h = \frac{b-a}{m}$ und

$$Q(f) = \sum_{k=0}^{n} c_k f(x_k)$$

eine Quadraturformel für $I_{[0,1],1}$. Dann heißt

$$Q_m(f) = h \sum_{j=1}^{m} \sum_{k=0}^{n} c_k f(a + (j - 1 + x_k)h)$$

die zu Q gehörige *zusammengesetzte Quadraturformel*.

Der folgende Satz gibt eine allgemeine Fehlerabschätzung für zusammengesetzte Quadraturformeln.

SATZ II.1.6 (Fehlerabschätzung für zusammengesetzte Quadraturformeln). *Seien $a, b \in \mathbb{R}$, $a < b$, $m \in \mathbb{N}^*$, $h = \frac{b-a}{m}$,*

$$Q(f) = \sum_{k=0}^{n} c_k f(x_k)$$

eine Quadraturformel der Ordnung K für $I_{[0,1],1}$ und Q_m die zugehörige zusammengesetzte Quadraturformel. Dann gilt für alle $f \in C^{K+1}([a, b], \mathbb{R})$

$$\left| \int_a^b f(x)\mathrm{d}x - Q_m(f) \right| \leq (b - a) \left\{ 1 + \sum_{k=0}^{n} |c_k| \right\} \frac{h^{K+1}}{(K + 1)!} \left\| f^{(K+1)} \right\|_{C([a,b],\mathbb{R})}.$$

BEWEIS. Für $1 \leq j \leq m$ sei $I_j = [a + (j - 1)h, a + jh]$ und

$$\varphi_j(t) = a + (j - 1 + t)h, \quad t \in [0, 1],$$

die affine Transformation von $[0, 1]$ auf I_j. Dann ist

$$Q_m(f) = \sum_{j=1}^{m} Q_{\varphi_j([0,1])}(f \circ \varphi_j^{-1}).$$

Hieraus folgt mit der Dreiecksungleichung

$$\begin{aligned}\left|\int_a^b f(x)\mathrm{d}x - Q_m(f)\right| &= \left|\sum_{j=1}^m \left\{\int_{I_j} f - Q_{\varphi_j([0,1])}(f\circ\varphi_j^{-1})\right\}\right| \\ &\leq \sum_{j=1}^m \left|\left\{\int_{I_j} f - Q_{\varphi_j([0,1])}(f\circ\varphi_j^{-1})\right\}\right|.\end{aligned}$$

Da Q die Ordnung K hat, folgt für jedes j und jedes $p \in \mathbb{P}_K$

$$\int_{I_j} p = Q_{\varphi_j([0,1])}(p\circ\varphi_j^{-1})$$

und

$$\begin{aligned}&\left|\left\{\int_{I_j} f - Q_{\varphi_j([0,1])}(f\circ\varphi_j^{-1})\right\}\right| \\ &\quad= \left|\left\{\int_{I_j} f - \int_{I_j} p + Q_{\varphi_j([0,1])}(p\circ\varphi_j^{-1}) - Q_{\varphi_j([0,1])}(f\circ\varphi_j^{-1})\right\}\right| \\ &\quad\leq \left|\int_{I_j} f - \int_{I_j} p\right| + \left|Q_{\varphi_j([0,1])}(p\circ\varphi_j^{-1}) - Q_{\varphi_j([0,1])}(f\circ\varphi_j^{-1})\right|.\end{aligned}$$

Wegen Bemerkung II.1.2 ist

$$\left|\int_{I_j} f - \int_{I_j} p\right| \leq h\,\|f-p\|_{C(I_j,\mathbb{R})}$$

und wegen Satz II.1.4

$$\left|Q_{\varphi_j([0,1])}(p\circ\varphi_j^{-1}) - Q_{\varphi_j([0,1])}(f\circ\varphi_j^{-1})\right| \leq h\left(\sum_{k=0}^n |c_k|\right)\|f-p\|_{C(I_j,\mathbb{R})}.$$

Für jedes j wählen wir nun p als das Lagrangesche Interpolationspolynom zu f in $K+1$ äquidistanten Knoten auf I_j. Aus der Fehlerabschätzung von Satz I.2.3 (S. 13) folgt dann

$$\begin{aligned}\|f-p\|_{C(I_j,\mathbb{R})} &\leq \frac{h^{K+1}}{(K+1)!}\left\|f^{(K+1)}\right\|_{C(I_j,\mathbb{R})} \\ &\leq \frac{h^{K+1}}{(K+1)!}\left\|f^{(K+1)}\right\|_{C([a,b],\mathbb{R})}.\end{aligned}$$

Kombinieren wir diese Abschätzungen, erhalten wir

$$\begin{aligned}&\left|\int_a^b f(x)\mathrm{d}x - Q_m(f)\right| \\ &\quad\le \sum_{j=1}^{m} h\left(1+\sum_{k=0}^{n}|c_k|\right)\frac{h^{K+1}}{(K+1)!}\left\|f^{(K+1)}\right\|_{C([a,b],\mathbb{R})} \\ &\quad\le mh\left(1+\sum_{k=0}^{n}|c_k|\right)\frac{h^{K+1}}{(K+1)!}\left\|f^{(K+1)}\right\|_{C([a,b],\mathbb{R})}.\end{aligned}$$

Wegen $mh = b - a$ beweist dies die Behauptung. □

Bemerkung II.1.7. Falls die Gewichte c_k alle nicht-negativ sind, kann der Faktor $1 + \sum_{k=0}^{n} |c_k|$ in Satz II.1.6 wegen $\sum_{k=0}^{n} c_k = 1$ durch 2 ersetzt werden.

II.2. Newton-Cotes-Formeln

In diesem Paragraphen betrachten wir Newton-Cotes-Formeln für beschränkte Intervalle und die Gewichtsfunktion $\omega = 1$. Wegen Satz II.1.4 (S. 39) können wir uns dabei auf das Intervall $[0, 1]$ beschränken. Der folgende Satz charakterisiert Newton-Cotes-Formeln eindeutig in Abhängigkeit von den Knoten.

Satz II.2.1 (Charakterisierung von Newton-Cotes Formeln). *Die Quadraturformel*

$$Q_n(f) = \sum_{k=0}^{n} c_k f(x_k)$$

ist genau dann eine Newton-Cotes-Formel für $I_{[0,1]}$, wenn für die Gewichte gilt

$$c_k = \int_0^1 \lambda_k(x)\mathrm{d}x = \int_0^1 \prod_{\substack{i=0\\ i\neq k}}^{n} \frac{x - x_i}{x_k - x_i}\mathrm{d}x, \quad 0 \le k \le n.$$

Beweis. „$\Longrightarrow$“: Ist Q_n eine Newton-Cotes-Formel, so folgt für $0 \le k \le n$

$$\int_0^1 \lambda_k(x)\mathrm{d}x = Q_n(\lambda_k) = \sum_{j=0}^{n} c_j \lambda_k(x_j) = c_k.$$

„$\Longleftarrow$“: Sei $p \in \mathbb{P}_n$. Dann ist gemäß Beispiel I.1.3(1) (S. 10) und Satz I.2.2 (S. 13)

$$p(x) = \sum_{i=0}^{n} p(x_i)\lambda_i(x).$$

Damit folgt

$$\int_0^1 p(x)\mathrm{d}x - Q_n(p) = \sum_{i=0}^{n} p(x_i) \left\{ \int_0^1 \lambda_i(x)\mathrm{d}x - \underbrace{Q_n(\lambda_i)}_{=c_i} \right\} = 0. \quad \square$$

TABELLE II.2.1. Abgeschlossene Newton-Cotes-Formeln

n	x_i	c_i	Ordnung	Name
1	$0, 1$	$\frac{1}{2}, \frac{1}{2}$	1	Trapezregel
2	$0, \frac{1}{2}, 1$	$\frac{1}{6}, \frac{4}{6}, \frac{1}{6}$	3	Simpsonregel
3	$0, \frac{1}{3}, \frac{2}{3}, 1$	$\frac{1}{8}, \frac{3}{8}, \frac{3}{8}, \frac{1}{8}$	3	Keplerregel
4	$0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1$	$\frac{7}{90}, \frac{32}{90}, \frac{12}{90}, \frac{32}{90}, \frac{7}{90}$	4	Milneregel

BEISPIEL II.2.2 (`midpoint_rule`, `trapezoidal_rule`, `simpson_rule` in `Numerics`). *(1)* Tabelle II.2.1 gibt die Knoten und Gewichte von abgeschlossenen Newton-Cotes-Formeln zu äquidistanten Knoten.
(2) Tabelle II.2.2 gibt die Knoten und Gewichte von offenen Newton-Cotes-Formeln zu äquidistanten Knoten.
(3) Die wichtigsten zusammengesetzten Newton-Cotes-Formeln mit $h = \frac{1}{m}$ sind:

$$Q_m(f) = h \sum_{j=1}^{m} f\left(\left(-\frac{1}{2} + j\right) h\right) \quad \textit{(Mittelpunktsregel)},$$

$$Q_m(f) = \frac{h}{2} \left\{ f(0) + 2 \sum_{j=1}^{m-1} f(jh) + f(1) \right\} \quad \textit{(Trapezregel)},$$

$$Q_m(f) = \frac{h}{6} \left\{ f(0) + 2 \sum_{j=1}^{m-1} f(jh) + 4 \sum_{j=1}^{m} f\left(\frac{1}{2}(2j-1)h\right) + f(1) \right\} \quad \textit{(Simpsonregel)}.$$

BEMERKUNG II.2.3 (Verhalten von Newton-Cotes Formeln). *(1)* Bei den Newton-Cotes-Formeln zu äquidistanten Knoten treten ab $n = 7$ negative Gewichte auf.
(2) Wegen des Satzes von Faber, Satz I.2.4 (S. 14), gibt es zu jeder Knotenmatrix $((x_k^{(n)})_{0\le k\le n})_{n\in\mathbb{N}}$ eine Funktion $f \in C([0,1], \mathbb{R})$, so dass für die Folge $(Q_n)_{n\in\mathbb{N}}$ der zugehörigen Newton-Cotes-Formeln gilt

$$\limsup_{n\to\infty} \left| \int_0^1 f\mathrm{d}x - Q_n(f) \right| = \infty.$$

TABELLE II.2.2. Offene Newton-Cotes-Formeln

n	x_i	c_i	Ordnung	Name
0	$\frac{1}{2}$	1	1	Mittelpunktsregel
1	$\frac{1}{4}, \frac{3}{4}$	$\frac{1}{2}, \frac{1}{2}$	1	
2	$\frac{1}{6}, \frac{1}{2}, \frac{5}{6}$	$\frac{3}{8}, \frac{2}{8}, \frac{3}{8}$	3	
3	$\frac{1}{8}, \frac{3}{8}, \frac{5}{8}, \frac{7}{8}$	$\frac{13}{48}, \frac{11}{48}, \frac{11}{48}, \frac{13}{48}$	3	

Wegen Satz II.1.6 (S. 40) ist es daher viel effizienter zusammengesetzte Formeln relativ niedriger Ordnung zu benützen.

BEISPIEL II.2.4. Tabelle II.2.3 zeigt die Ergebnisse der zusammengesetzten Mittelpunkts-, Trapez- und Simpsonregel angewandt auf das Integral $\int_0^1 \frac{3}{2}\sqrt{x} = 1$.

TABELLE II.2.3. Mittelpunkts-, Trapez- und Simpsonregel angewandt auf das Integral $\int_0^1 \frac{3}{2}\sqrt{x} = 1$

h	Mittelpunkt	Trapez	Simpson
1	1.06066	0.75000	0.95711
$\frac{1}{2}$	1.02452	0.99530	0.98479
$\frac{1}{4}$	1.00947	0.96493	0.99462
$\frac{1}{8}$	1.00355	0.98720	0.99810
$\frac{1}{16}$	1.00131	0.99537	0.99933
$\frac{1}{32}$	1.00047	0.99834	0.99976
$\frac{1}{64}$	1.00017	0.99941	0.99992
$\frac{1}{128}$	1.00006	0.99979	0.99997

II.3. Gaußsche Formeln

In diesem Paragraphen konstruieren wir Gaußsche Quadraturformeln. Wegen Bemerkung II.1.2(2) (S. 38) haben diese Formeln die maximal mögliche Ordnung. Wegen Satz II.2.1 (S. 42) müssen wir nur noch die Knoten bestimmen. Wie wir sehen werden, sind diese die Nullstellen geeigneter Orthogonalpolynome. Daher müssen wir zuerst einige Eigenschaften solcher Polynome herleiten.

Dazu bezeichnen wir mit $[a, b]$, $a, b \in \mathbb{R}$, $a < b$, ein beliebiges Intervall, $\omega \in L^1([a, b], \mathbb{R})$, $\omega > 0$, eine Gewichtsfunktion und $(\cdot\,,\,\cdot)_\omega$ das zu ω gehörige Skalarprodukt auf $C([a, b], \mathbb{R})$

$$(\varphi\,,\,\psi)_\omega = \int_a^b \varphi(x)\psi(x)\omega(x)\mathrm{d}x.$$

SATZ II.3.1 (Orthogonalpolynome). *Es gibt genau eine Folge von Polynomen $(q_n)_{n\in\mathbb{N}}$ mit den Eigenschaften:*

(i) $q_n \in \mathbb{P}_n$ *für alle* $n \in \mathbb{N}$,
(ii) q_n *hat den Höchstkoeffizienten* 1 *für alle* $n \in \mathbb{N}$,
(iii) $(q_n\,,\,q_m)_\omega = 0$ *für alle* $n \neq m$.

Es ist

(a) $q_0(x) = 1$,
(b) $q_1(x) = x - \alpha_0$,
(c) $q_{n+1}(x) = (x-\alpha_n)q_n(x) - \beta_{n-1}q_{n-1}(x)$ *für alle* $n \in \mathbb{N}^*$

mit

$$\alpha_n = \frac{(xq_n\,,\,q_n)_\omega}{(q_n\,,\,q_n)_\omega} \qquad \forall n \in \mathbb{N},$$

$$\beta_{n-1} = \frac{(xq_n\,,\,q_{n-1})_\omega}{(q_{n-1}\,,\,q_{n-1})_\omega} \quad \forall n \in \mathbb{N}^*.$$

BEWEIS. *Eindeutigkeit:* Seien $(q_n)_{n\in\mathbb{N}}$ und $(\widetilde{q}_n)_{n\in\mathbb{N}}$ zwei Folgen mit den Eigenschaften (i) – (iii). Aus (i), (ii) folgt $q_0 = \widetilde{q}_0$. Für $n \in \mathbb{N}^*$ folgt aus (i), (ii) $q_n - \widetilde{q}_n \in \mathbb{P}_{n-1}$ und aus (iii)

$$(q_n - \widetilde{q}_n\,,\,q_n - \widetilde{q}_n)_\omega = (q_n\,,\,q_n - \widetilde{q}_n)_\omega - (\widetilde{q}_n\,,\,q_n - \widetilde{q}_n)_\omega = 0$$

und somit $q_n = \widetilde{q}_n$.
Existenz: Offensichtlich sind die Zahlen α_n, β_{n-1} und die Polynome q_n wohl definiert. Weiter erfüllen letztere offensichtlich die Bedingungen (i), (ii). Die Orthogonalitätsbeziehung (iii) ist äquivalent zu

(iv) $(q_n\,,\,q_m)_\omega = 0$ für alle $n \in \mathbb{N}^*$ und alle m mit $0 \leq m < n$.

Wir zeigen (iv) durch Induktion über n.
$\underline{n = 1:}$ Folgt direkt aus der Definition von α_0.
$\underline{n \to n+1:}$ Aus der Induktionsvoraussetzung folgt für $0 \leq m \leq n-2$ wegen $xq_m \in \mathbb{P}_{n-1}$

$$(q_{n+1}\,,\,q_m)_\omega = (q_n\,,\,xq_m)_\omega - \alpha_n\,(q_n\,,\,q_m)_\omega - \beta_{n-1}\,(q_{n-1}\,,\,q_m)_\omega = 0.$$

Weiter folgt aus der Definition von α_n und β_{n-1} und $(q_n\,,\,q_{n-1})_\omega = 0$

$$\begin{aligned}(q_{n+1}\,,\,q_{n-1})_\omega &= (q_n\,,\,xq_{n-1})_\omega - \alpha_n\,(q_n\,,\,q_{n-1})_\omega - \beta_{n-1}\,(q_{n-1}\,,\,q_{n-1})_\omega \\ &= 0,\\ (q_{n+1}\,,\,q_n)_\omega &= (xq_n\,,\,q_n)_\omega - \alpha_n\,(q_n\,,\,q_n)_\omega - \beta_{n-1}\,(q_{n-1}\,,\,q_n)_\omega \\ &= 0\end{aligned}$$

und damit die Behauptung. □

SATZ II.3.2 (Nullstellen von Orthogonalpolynomen). *Die Folge $(q_n)_{n\in\mathbb{N}}$ erfülle die Bedingungen (i) – (iii) aus Satz II.3.1. Dann besitzt q_n, $n \in \mathbb{N}^*$, genau n verschiedene, reelle Nullstellen mit Vorzeichenwechsel in (a, b).*

BEWEIS. Seien $n \in \mathbb{N}^*$ und $a < x_1 < \ldots < x_m < b$ die Nullstellen von q_n mit Vorzeichenwechsel in (a, b). Angenommen, es ist $m < n$. Dann ist

$$p(x) = \prod_{i=1}^{m}(x - x_i) \in \mathbb{P}_{n-1}$$

und daher wegen (iii)

$$(q_n\,,\,p)_\omega = 0.$$

Andererseits besitzt $q_n p$ keinen Vorzeichenwechsel in (a, b) und ist ungleich Null. Damit folgt

$$(q_n\,,\,p)_\omega = \int_a^b q_n(x)p(x)\omega(x)\mathrm{d}x \neq 0.$$

Dies ist ein Widerspruch. Also gilt $m = n$. □

Nach diesen Vorbereitungen können wir jetzt Gaußsche Quadraturformeln eindeutig charakterisieren.

SATZ II.3.3 (Charakterisierung von Gaußschen Formeln). *Die Quadraturformel*

$$Q(f) = \sum_{k=0}^{n} c_k f(x_k)$$

ist genau dann eine Gaußsche Formel für $I_{[a,b],\omega}$, wenn die Knoten die Nullstellen des Polynoms q_{n+1} aus Satz II.3.2 sind und die Gewichte c_k durch

$$c_k = \int_a^b \lambda_k(x)\omega(x)\mathrm{d}x = \int_a^b \omega(x) \prod_{\substack{i=0 \\ i\neq k}}^{n} \frac{x - x_i}{x_k - x_i}\mathrm{d}x$$

gegeben sind.

BEWEIS. Die Formel für die Gewichte folgt wie in Beweis von Satz II.2.1 (S. 42). Bleibt nur die Formel für die Knoten zu zeigen.
„$\Longrightarrow$“: Für $0 \leq k \leq n$ folgt

$$c_k = Q(\lambda_k^2) = I_{[a,b],\omega}(\lambda_k^2) > 0$$

und

$$0 = (q_{n+1}\,,\,\lambda_k)_\omega = I_{[a,b],\omega}(\lambda_k q_{n+1}) = Q(q_{n+1}\lambda_k) = c_k q_{n+1}(x_k).$$

Also ist $q_{n+1}(x_k) = 0$.
„$\Longleftarrow$“: Sei $p \in \mathbb{P}_{2n+1}$. Dann gibt es Polynome $q, r \in \mathbb{P}_n$ mit $p = qq_{n+1} + r$. Damit folgt

$$\begin{aligned} I_{[a,b],\omega}(p) &= (q\,,\,q_{n+1})_\omega + I_{[a,b],\omega}(r) = I_{[a,b],\omega}(r) \\ &= Q(r) = Q(qq_{n+1} + r) = Q(p). \end{aligned}$$

□

BEISPIEL II.3.4 (`gauss_rule` in `Numerics`). Betrachte $[a, b] = [0, 1]$ und $\omega = 1$. Dann ist $q_0(x) = 1$,

$$\alpha_0 = \int_0^1 x \mathrm{d}x = \frac{1}{2}$$

und

$$q_1(x) = x - \frac{1}{2}.$$

Damit ergibt sich für $n = 0$ die Gaußsche Formel

$$Q(f) = f\left(\frac{1}{2}\right).$$

Dies ist die Mittelpunktsregel.
Weiter ist

$$\begin{aligned}(q_1\,,\, q_1)_\omega &= \int_0^1 \left(x - \frac{1}{2}\right)^2 \mathrm{d}x &&= \frac{1}{12},\\ (xq_1\,,\, q_1)_\omega &= \int_0^1 x\left(x - \frac{1}{2}\right)^2 \mathrm{d}x &&= \frac{1}{24},\\ (xq_1\,,\, q_0)_\omega &= \int_0^1 x\left(x - \frac{1}{2}\right) \mathrm{d}x &&= \frac{1}{12}\end{aligned}$$

und somit

$$\alpha_1 = \frac{1}{2}, \quad \beta_0 = \frac{1}{12}$$

und

$$q_2(x) = \left(x - \frac{1}{2}\right)\left(x - \frac{1}{2}\right) - \frac{1}{12} = x^2 - x + \frac{1}{6}.$$

Offensichtlich hat q_2 die Nullstellen $\frac{1}{2} \pm \frac{1}{\sqrt{12}} = \frac{3\pm\sqrt{3}}{6}$. Damit ergibt sich für $n = 2$ die Gaußsche Formel zu

$$Q(f) = \frac{1}{2}\left(f\left(\frac{3-\sqrt{3}}{6}\right) + f\left(\frac{3+\sqrt{3}}{6}\right)\right).$$

BEISPIEL II.3.5. Tabelle II.3.1 zeigt die Ergebnisse der zusammengesetzten Gaußschen Formeln mit 2, 3 und 4 Knoten angewandt auf das Integral $\int_0^1 \frac{3}{2}\sqrt{x} = 1$.

Satz II.3.3 löst das Problem, zu gegebenem Intervall und gegebener Gewichtsfunktion eine Gaußsche Formel zu konstruieren, vollständig. Für die praktische Rechnung ist das Ergebnis aber nicht geeignet, da die Bestimmung der Knoten und Gewichte über die Nullstellen von q_{n+1} und die angegebene Formel numerisch nicht stabil und geeignet ist. Im Folgenden zeigen wir für den Spezialfall $[a, b] = [-1, 1]$ und $\omega = 1$, dass die Knoten und Gewichte der entsprechenden Gaußschen Formel leicht aus den Eigenwerten und Eigenvektoren geeigneter Tridiagonalmatrizen berechnet werden können. Dabei lassen wir den Index ω bei $(\cdot\,,\,\cdot)_\omega$ fort.

TABELLE II.3.1. Zusammengesetzte Gaußsche Formeln mit 2, 3 und 4 Knoten angewandt auf das Integral $\int_0^1 \frac{3}{2}\sqrt{x} = 1$

h	2 Knoten	3 Knoten	4 Knoten
1	1.01083	1.00377	1.00174
$\frac{1}{2}$	1.00386	1.00133	1.00062
$\frac{1}{4}$	1.00137	1.00047	1.00022
$\frac{1}{8}$	1.00048	1.00017	1.00008
$\frac{1}{16}$	1.00017	1.00006	1.00003
$\frac{1}{32}$	1.00006	1.00002	1.00001
$\frac{1}{64}$	1.00002	1.00001	1.00000
$\frac{1}{128}$	1.00001	1.00000	1.00000

DEFINITION II.3.6 (Legendre Polynome). Die gemäß Satz II.3.1 zum Intervall $[-1, 1]$ und zur Gewichtsfunktion $\omega = 1$ eindeutig bestimmten Orthogonalpolynome heißen *Legendre-Polynome* und werden mit L_n bezeichnet.

Der folgende Satz gibt eine alternative Darstellung der Legendre-Polynome.

SATZ II.3.7 (Formel von Rodriguez). *Für $n \in \mathbb{N}$ ist*

$$L_n(x) = \frac{n!}{(2n)!}\frac{d^n}{dx^n}\left[(x^2-1)^n\right].$$

BEWEIS. Bezeichne die rechte Seite in obiger Gleichung mit q_n. Offensichtlich sind die Bedingungen (i) und (ii) von Satz II.3.1 erfüllt. Da $(x^2-1)^n$ die n-fachen Nullstellen ± 1 hat, gilt für alle $0 \le m \le n-1$

$$\frac{d^m}{dx^m}\left[(x^2-1)^n\right]|_{x=\pm 1} = 0.$$

Damit folgt für $0 \le m \le n$ durch n-malige partielle Integration

$$\begin{aligned}
&(q_n\,,\,q_m)\\
&= \int_{-1}^{1} \frac{n!}{(2n)!}\frac{d^n}{dx^n}\left[(x^2-1)^n\right]\frac{m!}{(2m)!}\frac{d^m}{dx^m}\left[(x^2-1)^m\right]\mathrm{d}x\\
&= -\int_{-1}^{1} \frac{n!}{(2n)!}\frac{d^{n-1}}{dx^{n-1}}\left[(x^2-1)^n\right]\frac{m!}{(2m)!}\frac{d^{m+1}}{dx^{m+1}}\left[(x^2-1)^m\right]\mathrm{d}x\\
&= (-1)^n \int_{-1}^{1} \frac{n!}{(2n)!}\left[(x^2-1)^n\right]\frac{m!}{(2m)!}\frac{d^{n+m}}{dx^{n+m}}\left[(x^2-1)^m\right]\mathrm{d}x.
\end{aligned}$$

Also ist $(q_n\,,\,q_m) = 0$ für $0 \le m \le n-1$ und

$$\begin{aligned}(q_n\,,\,q_n) &= (-1)^n \int_{-1}^{1} \left[\frac{n!}{(2n)!}\right]^2 (x-1)^n (x+1)^n (2n)!\mathrm{d}x \\ &= \frac{(n!)^4}{[(2n)!]^2}\frac{2^{2n+1}}{2n+1}.\end{aligned}$$

□

Mit Hilfe von Satz II.3.7 können wir die Zahlen α_n und β_{n-1} aus Satz II.3.1 explizit angeben.

Satz II.3.8 (Rekursionsformel für die Legendre-Polynome). *Es ist*

$$L_0(x) = 1, \quad L_1(x) = x$$

und für $n \in \mathbb{N}^*$

$$L_{n+1}(x) = xL_n(x) - \frac{n^2}{4n^2-1}L_{n-1}(x).$$

Beweis. Aus Satz II.3.7 folgt für alle $x \in [-1,1]$ und $n \in \mathbb{N}$

$$L_n(-x) = (-1)^n L_n(x).$$

Also ist $\alpha_n = 0$ für alle $n \in \mathbb{N}$. Aus Satz II.3.1 und dem Beweis von Satz II.3.7 folgt für alle $n \in \mathbb{N}^*$

$$\begin{aligned}\beta_{n-1} &= \frac{(xL_{n-1}\,,\,L_n)}{(L_{n-1}\,,\,L_{n-1})} = \frac{(L_n\,,\,L_n)}{(L_{n-1}\,,\,L_{n-1})} \\ &= \frac{[n!]^4}{[(2n)!]^2}\frac{2^{2n+1}}{2n+1}\frac{[(2n-2)!]^2}{[(n-1)!]^4}\frac{2n-1}{2^{2n-1}} \\ &= \frac{n^2}{4n^2-1}.\end{aligned}$$

□

Mit Hilfe von Satz II.3.8 zeigen wir als nächstes, dass die Legendre-Polynome die charakteristischen Polynome geeigneter Tridiagonalmatrizen sind. Damit ist die Bestimmung der Knoten der Gaußschen Quadraturformeln auf $[-1,1]$ zur Gewichtsfunktion $\omega = 1$ auf die Berechnung der Eigenwerte dieser Matrizen reduziert.

Satz II.3.9 (Darstellung der Legendre-Polynome als charakteristische Polynome). *Definiere die Matrizen* $A_n \in \mathbb{R}^{n\times n}$ *durch* $A_1 = 0$ *und für* $n \ge 2$

$$A_n = \begin{pmatrix} 0 & -\mu_1 & & & \\ -\mu_1 & 0 & -\mu_2 & & \\ & \ddots & \ddots & \ddots & \\ & & & & -\mu_{n-1} \\ & & & -\mu_{n-1} & 0 \end{pmatrix}$$

mit

$$\mu_k = \frac{k}{\sqrt{4k^2-1}}$$

für alle $k \in \mathbb{N}^*$. Dann gilt für alle $n \in \mathbb{N}^*$

$$L_n(x) = \det(xI - A_n).$$

BEWEIS. Definiere für $n \in \mathbb{N}^*$

$$q_n(x) = \det(xI - A_n).$$

Es reicht, zu zeigen, dass die q_n die Rekursionsformel aus Satz II.3.8 erfüllen.
Die Entwicklung der Determinante nach der letzten Zeile liefert

$$\begin{aligned} q_{n+1}(x) &= xq_n(x) - \mu_n \det \begin{pmatrix} x & \mu_1 & & & \\ \mu_1 & x & \mu_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \mu_{n-2} & x & 0 \\ & & & \mu_{n-1} & \mu_n \end{pmatrix} \\ &= xq_n(x) - \mu_n^2 q_{n-1}(x) \\ &= xq_n(x) - \frac{n^2}{4n^2-1} q_{n-1}(x). \end{aligned}$$

□

Nach diesen Vorbereitungen können wir zeigen, dass die Knoten und Gewichte der Gaußschen Quadraturformeln auf $[-1, 1]$ zur Gewichtsfunktion $\omega = 1$ gegeben sind durch die Eigenwerte und die ersten Komponenten der Eigenvektoren der Matrix A_n aus Satz II.3.9.

SATZ II.3.10 (Bestimmung der Knoten und Gewichte). *Sei*

$$Q(f) = \sum_{k=0}^{n} c_k f(x_k)$$

die Gaußsche Quadraturformel zu $I_{[-1,1],1}$. Dann sind die Knoten x_k die Eigenwerte der Matrix A_{n+1} aus Satz II.3.9. Für die Gewichte c_k gilt

$$c_k = 2 \left| z_1^{(k)} \right|^2,$$

wobei $z^{(k)}$ ein Eigenvektor von A_{n+1} zum Eigenwert x_k mit $\left\| z^{(k)} \right\|_2 = 1$ ist.

BEWEIS. Wegen Satz II.3.3 muss nur noch die Formel für die Gewichte gezeigt werden. Definiere dazu die Vektoren $\widetilde{z}^{(k)} \in \mathbb{R}^{n+1}$, $0 \le k \le n$, durch

$$\widetilde{z}_i^{(k)} = \nu_{i-1} L_{i-1}(x_k), \quad 1 \le i \le n+1,$$

mit

$$\nu_0 = 1, \quad \nu_i = -\frac{\nu_{i-1}}{\mu_i}, \ 1 \le i \le n,$$

wobei $x_0, \ldots, x_n$ die Eigenwerte von A_{n+1} sind. Dann folgt

$$(A_{n+1}\widetilde{z}^{(k)})_1 = -\mu_1\nu_1 L_1(x_k) = -\mu_1\nu_1 x_k = \nu_0 x_k = x_k \widetilde{z}_1^{(k)}.$$

Aus Satz II.3.8 folgt für $2 \leq j \leq n$

$$\begin{aligned}(A_{n+1}\widetilde{z}^{(k)})_j &= -\mu_{j-1}\nu_{j-2}L_{j-2}(x_k) - \mu_j\nu_j L_j(x_k)\\ &= -\mu_{j-1}\nu_{j-2}L_{j-2}(x_k) - \mu_j\nu_j x_k L_{j-1}(x_k)\\ &\quad + \mu_j\nu_j\mu_{j-1}^2 L_{j-2}(x_k)\\ &= \nu_{j-1}x_k L_{j-1}(x_k)\\ &= x_k\widetilde{z}_j^{(k)}.\end{aligned}$$

Wegen $L_{n+1}(x_k) = 0$ gilt diese Beziehung auch für $j = n+1$.
Mithin sind die Vektoren $\widetilde{z}^{(k)}$ Eigenvektoren von A_{n+1} zu den Eigenwerten $x_0, \ldots, x_n$. Da A_{n+1} symmetrisch ist, sind sie paarweise orthogonal. Definiere $B \in \mathbb{R}^{(n+1)\times(n+1)}$ durch

$$B_{ij} = \widetilde{z}_i^{(j-1)} \quad 1 \leq i, j \leq n+1.$$

Aus Satz II.3.7 folgt für $0 \leq i \leq n$

$$\sum_{k=0}^{n} c_k L_i(x_k) = \int_{-1}^{1} L_i(x)\mathrm{d}x = 2\delta_{0,i}$$

und daher

$$B\underline{c} = 2e_1$$

mit $\underline{c} = (c_0, \ldots, c_n)^t$. Hieraus folgt für $0 \leq k \leq n$

$$2 = 2\widetilde{z}_1^{(k)} = 2(\widetilde{z}^{(k)}, e_1) = \left(\widetilde{z}^{(k)}\,,\, B\underline{c}\right) = c_k \left\|\widetilde{z}^{(k)}\right\|_2^2$$

und damit die Behauptung. □

Beispiel II.3.11. Für $n = 1$ und $n = 2$ erhält man so die Quadraturformeln

$$Q(f) = f\left(-\frac{\sqrt{3}}{3}\right) + f\left(\frac{\sqrt{3}}{3}\right)$$

und

$$Q(f) = \frac{5}{9}f\left(-\sqrt{\frac{3}{5}}\right) + \frac{8}{9}f(0) + \frac{5}{9}f\left(\sqrt{\frac{3}{5}}\right).$$

II.4. Extrapolation

Ausgangspunkt für das Quadraturverfahren dieses Paragraphen ist die zusammengesetzte Trapezregel

$$T_h(f) = \frac{h}{2}\left\{f(0) + 2\sum_{i=1}^{m-1} f(ih) + f(1)\right\}, \quad h = \frac{1}{m}, m \in \mathbb{N}^*.$$

In Satz II.4.5 (S. 55) werden wir zeigen, dass für den Fehler der zusammengesetzten Trapezregel eine *asymptotische Fehlerentwicklung* der Form

$$T_h(f) = \int_0^1 f(x)\mathrm{d}x + \sum_{l=1}^{n} \alpha_l(f)h^{2l} + r_{2n+2}(f)h^{2n+2} \tag{II.4.1}$$

gilt. Dies ist eine wesentlich schärfere Aussage als die Fehlerabschätzung von Satz II.1.6 (S. 40).

Ausgehend von der Fehlerentwicklung (II.4.1) kann man die Idee des neuen Quadraturverfahrens wie folgt beschreiben:
Betrachte die Entwicklungen (II.4.1) für $T_h(f)$ und für $T_{\frac{h}{2}}(f)$

$$T_h(f) = \int_0^1 f(x)\mathrm{d}x + \sum_{l=1}^{n} \alpha_l(f)h^{2l} + r_{2n+2}(f)h^{2n+2},$$
$$T_{\frac{h}{2}}(f) = \int_0^1 f(x)\mathrm{d}x + \sum_{l=1}^{n} \alpha_l(f)2^{-2l}h^{2l} + r_{2n+2}(f)2^{-2n-2}h^{2n+2}.$$

Dann ist offensichtlich

$$4T_{\frac{h}{2}}(f) - T_h(f) = 3\int_0^1 f(x)\mathrm{d}x + \sum_{l=2}^{n} \widetilde{\alpha}_l(f)h^{2l} + \widetilde{r}_{2n+2}(f)h^{2n+2}$$

mit geeigneten Zahlen $\widetilde{\alpha}_2(f)$, …, $\widetilde{\alpha}_n(f)$ und $\widetilde{r}_{2n+2}(f)$. Also ist

$$\frac{1}{3}\left(4T_{\frac{h}{2}}(f) - T_h(f)\right)$$

eine Quadraturformel der Ordnung 4, die ebenfalls eine asymptotische Fehlerentwicklung besitzt. Daher können wir die neue Quadraturformel zu h und $\frac{h}{2}$ wieder geeignet zusammenfassen und erhalten so eine Quadraturformel der Ordnung 6.
Diese Vorgehensweise nennt man *Extrapolation*.

Zum Nachweis der Fehlerentwicklung (II.4.1) benötigen wir die *Bernoulli-Polynome* und deren Eigenschaften.

Definition II.4.1 (Bernoulli-Polynome). Die Folge $(B_k)_{k\in\mathbb{N}^*}$, $B_k \in \mathbb{P}_k$, der *Bernoulli-Polynome* ist definiert durch

(1) $B_1(x) = x - \frac{1}{2}$,
(2) $B'_{k+1}(x) = (k+1)B_k(x)$ für alle $k \in \mathbb{N}^*$,
(3) $B_{2l+1}(0) = B_{2l+1}(1) = 0$ für alle $l \in \mathbb{N}^*$.

Die Zahlen $\beta_k = B_k(0)$ heißen *Bernoulli-Zahlen*.

Bemerkung II.4.2. Die Bernoulli-Polynome sind durch die Bedingung (1) – (3) aus Definition II.4.1 eindeutig bestimmt.

BEWEIS. Sei $l \in \mathbb{N}^*$. Aus (2) folgt

$$B_{2l}(x) = 2l \cdot \int_0^x B_{2l-1}(t)\mathrm{d}t + c_{2l}$$

$$B_{2l+1}(x) = 2l \cdot (2l+1) \cdot \int_0^x \int_0^t B_{2l-1}(s)\mathrm{d}s\mathrm{d}t + (2l+1)c_{2l}x + c_{2l+1}.$$

Aus (3) folgt

$$c_{2l+1} = 0, \quad c_{2l} = -2l \cdot \int_0^1 \int_0^t B_{2l-1}(s)\mathrm{d}s\mathrm{d}t. \qquad \square$$

SATZ II.4.3 (Eigenschaften der Bernoulli-Polynome). *Die Bernoulli-Polynome haben die folgenden Eigenschaften:*

(1) $(-1)^k B_k(1-x) = B_k(x)$ *für alle* $k \in \mathbb{N}$,
(2) $\int_0^1 B_k(x)\mathrm{d}x = \frac{1}{k+1}[B_{k+1}(1) - B_{k+1}(0)] = 0$ *für alle* $k \in \mathbb{N}$,
(3) $(-1)^m B_{2m-1}(x) > 0$ *für alle* $0 < x < \frac{1}{2}$, $m \in \mathbb{N}^*$,
(4) $(-1)^m[B_{2m}(x) - \beta_{2m}] > 0$ *für alle* $0 < x < 1$, $m \in \mathbb{N}^*$,
(5) $(-1)^{m+1}\beta_{2m} > 0$ *für alle* $m \in \mathbb{N}^*$.

BEWEIS. *ad (1):* Definiere $r_k \in \mathbb{P}_k$ für $k \in \mathbb{N}^*$ durch

$$r_k(x) = (-1)^k B_k(1-x).$$

Wie man leicht nachrechnet, erfüllen die r_k die Bedingungen (1) – (3) aus Definition II.4.1. Damit folgt die Behauptung aus Bemerkung II.4.2.
ad (2): Die erste Gleichung folgt aus Bedingung (2) von Definition II.4.1. Die zweite Gleichung folgt aus Bedingung (3) von Definition II.4.1 und der ersten Behauptung von Satz II.4.3.
ad (3) – (5): Beweis durch Induktion über m:
$\underline{m = 1:}$ (3) ist offensichtlich:

$$(-1)B_1(x) = \frac{1}{2} - x > 0 \quad \forall 0 < x < \frac{1}{2}.$$

(4) folgt für $0 < x \leq \frac{1}{2}$ aus

$$\begin{aligned}(-1)[B_2(x) - \beta_2] &= (-1)\int_0^x B_2'(t)\mathrm{d}t \\ &= 2\int_0^x (-1)B_1(t)\mathrm{d}t > 0 \quad \forall 0 < x \leq \frac{1}{2}.\end{aligned}$$

Für $x > \frac{1}{2}$ folgt die Behauptung aus (1).
(5) folgt aus (2) und (4):

$$0 < \int_0^1 (-1)[B_2(x) - \beta_2]\mathrm{d}x = (-1)^2\beta_2.$$

$\underline{m \to m+1:}$ Angenommen, B_{2m+1} besitzt einen Vorzeichenwechsel in $(0, \frac{1}{2})$. Dann gibt es ein $x^* \in (0, \frac{1}{2})$ mit $B_{2m+1}(x^*) = 0$. Wegen (1) ist

$B_{2m+1}(\frac{1}{2}) = 0$. Hieraus und aus Bedingung (3) von Definition II.4.1 folgt mit dem Satz von Rolle, dass es ein $x^{**} \in (0, \frac{1}{2})$ gibt mit

$$0 = B''_{2m+1}(x^{**}) = (2m+1)2mB_{2m-1}(x^{**})$$

im Widerspruch zur Induktionsannahme. Also hat $(-1)^{m+1}B_{2m+1}$ einheitliches Vorzeichen in $(0, \frac{1}{2})$. Wegen $B_{2m+1}(0) = 0$ ist dieses gleich demjenigen von $(-1)^{m+1}B'_{2m+1}(0)$ also gleich dem von $(-1)^{m+1}\beta_{2m}$. Damit folgt (3) aus der Induktionsannahme.
(4) und (5) folgen wie im Falle $m = 1$. □

Aus Satz II.4.3 folgt folgende Darstellung des Integrals.

Satz II.4.4 (Euler-Mac Laurinsche Summenformel). *Zu jedem $g \in C^{2n+2}([0,1], \mathbb{R})$ gibt es ein $\eta \in (0,1)$ mit*

$$\int_0^1 g(x)\mathrm{d}x = \frac{1}{2}[g(0)+g(1)] + \sum_{l=1}^{n} \frac{\beta_{2l}}{(2l)!}[g^{(2l-1)}(0) - g^{(2l-1)}(1)] - \frac{\beta_{2n+2}}{(2n+2)!}g^{(2n+2)}(\eta).$$

Beweis. Mittels $2n$-maliger partieller Integration folgt

$$\begin{aligned}
&\int_0^1 g(x)\mathrm{d}x \\
&= \int_0^1 g(x)B'_1(x)\mathrm{d}x \\
&= [gB_1]_0^1 - \int_0^1 g'(x)B_1(x)\mathrm{d}x \\
&= \frac{1}{2}[g(1)+g(0)] - \frac{1}{2}\int_0^1 g'(x)B'_2(x)\mathrm{d}x \\
&= \frac{1}{2}[g(0)+g(1)] - \frac{1}{2}[g'B_2]_0^1 + \frac{1}{2}\int_0^1 g''(x)B_2(x)\mathrm{d}x \\
&= \frac{1}{2}[g(0)+g(1)] + \frac{1}{2}\beta_2[g'(0)-g'(1)] + \frac{1}{6}\int_0^1 g''(x)B'_3(x)\mathrm{d}x \\
&= \frac{1}{2}[g(0)+g(1)] + \sum_{l=1}^{n}\frac{\beta_{2l}}{(2l)!}[g^{(2l-1)}(0) - g^{(2l-1)}(1)] + r_{2n+1}
\end{aligned}$$

mit

$$\begin{aligned}
r_{2n+1} &= -\frac{1}{(2n+1)!}\int_0^1 g^{(2n+1)}(x)B_{2n+1}(x)\mathrm{d}x \\
&= -\frac{1}{(2n+2)!}\int_0^1 g^{(2n+1)}(x)[B_{2n+2}(x) - \beta_{2n+2}]'\mathrm{d}x \\
&= \frac{1}{(2n+2)!}\int_0^1 g^{(2n+2)}(x)[B_{2n+2}(x) - \beta_{2n+2}]\mathrm{d}x.
\end{aligned}$$

Aus Satz II.4.3 und dem Mittelwertsatz der Integralrechnung folgt, dass es ein $\eta \in (0,1)$ gibt mit

$$\begin{aligned}
&\int_0^1 g^{(2n+2)}(x)[B_{2n+2}(x) - \beta_{2n+2}]\mathrm{d}x \\
&\quad = g^{(2n+2)}(\eta) \int_0^1 B_{2n+2}(x) - \beta_{2n+2}\mathrm{d}x \\
&\quad = -\beta_{2n+2} g^{(2n+2)}(\eta). \qquad \square
\end{aligned}$$

Aus Satz II.4.4 folgt die angekündigte Fehlerentwicklung (II.4.1).

Satz II.4.5 (Fehlerentwicklung für die zusammengesetzte Trapezregel). *Sei* $f \in C^{2n+2}([0,1],\mathbb{R})$ *und* $h = \frac{1}{m}$, $m \in \mathbb{N}^*$. *Dann gibt es ein* $\eta(h) \in [0,1]$ *mit*

$$\begin{aligned}
T_h(f) = \int_0^1 f(x)\mathrm{d}x + \sum_{l=1}^{n} \frac{\beta_{2l}}{(2l)!} h^{2l}[f^{(2l-1)}(1) - f^{(2l-1)}(0)] \\
+ \frac{\beta_{2n+2}}{(2n+2)!} h^{2n+2} f^{(2n+2)}(\eta).
\end{aligned}$$

Beweis. Definiere für $0 \le i \le m-1$ Funktionen $g_i : [0,1] \to \mathbb{R}$ durch $g_i(t) = hf(ih+th)$. Dann folgt $\int_0^1 g_i(t)\mathrm{d}t = \int_{ih}^{(i+1)h} f(x)\mathrm{d}x$ und $g_i^{(k)}(t) = h^{k+1} f^{(k)}(ih+th)$ für $k \in \mathbb{N}^*$. Aus Satz II.4.4 folgt, dass es Zahlen $\eta_0, \dots, \eta_{m-1} \in (0,1)$ gibt mit

$$\begin{aligned}
T_h(f) &= \sum_{i=0}^{m-1} \frac{1}{2}[g_i(1) + g_i(0)] \\
&= \sum_{i=0}^{m-1} \left\{ \int_0^1 g_i(t)\mathrm{d}t + \sum_{l=1}^{n} \frac{\beta_{2l}}{(2l)!}[g_i^{(2l-1)}(1) - g_i^{(2l-1)}(0)] \right. \\
&\qquad \left. + \frac{\beta_{2n+2}}{(2n+2)!} g_i^{(2n+2)}(\eta_i) \right\} \\
&= \int_0^1 f(x)\mathrm{d}x + \sum_{l=1}^{n} \frac{\beta_{2l}}{(2l)!} h^{2l} \sum_{i=0}^{m-1} [f^{(2l-1)}((i+1)h) - f^{(2l-1)}(ih)] \\
&\quad + \frac{\beta_{2n+2}}{(2n+2)!} h^{2n+2} \sum_{i=0}^{m-1} h f^{(2n+2)}((i+\eta_i)h) \\
&= \int_0^1 f(x)\mathrm{d}x + \sum_{l=1}^{n} \frac{\beta_{2l}}{(2l)!} h^{2l}[f^{(2l-1)}(1) - f^{(2l-1)}(0)] \\
&\quad + \frac{\beta_{2n+2}}{(2n+2)!} h^{2n+2} \sum_{i=0}^{m-1} h f^{(2n+2)}((i+\eta_i)h).
\end{aligned}$$

Wegen

$$\min_{0\le x\le 1} f(x) \le \sum_{i=0}^{m-1} h f^{(2n+2)}((i+\eta_i)h) \le \max_{0\le x\le 1} f(x)$$

gibt es nach dem Zwischenwertsatz ein $\eta = \eta(h) \in [0,1]$ mit

$$\sum_{i=0}^{m-1} h f^{(2n+2)}((i+\eta_i)h) = f(\eta). \qquad \square$$

Das folgende Quadraturverfahren beruht auf der Fehlerentwicklung von Satz II.4.5 und konkretisiert die eingangs beschriebene Idee.

DEFINITION II.4.6 (Romberg-Verfahren; `romberg` in `Numerics`). Das *Romberg-Verfahren* ist definiert durch

$$T_{0,k} = T_{2^{-k}}(f)$$

für $0 \le k \le K$ und

$$T_{i+1,k} = \frac{[4^{i+1} T_{i,k+1} - T_{i,k}]}{[4^{i+1} - 1]}$$

für $0 \le i \le K-1$, $0 \le k \le K-i-1$.

BEMERKUNG II.4.7. *(1)* Das dem Romberg-Verfahren zugrunde liegende Prinzip nennt man *Extrapolation*. Es kann auf wesentlich allgemeinere als die hier betrachtete Situation angewendet werden. Es benötigt lediglich eine Familie L_n von Funktionalen, die ein gegebenes Funktional L approximieren und die eine asymptotische Fehlerentwicklung der Form

$$L_n(f) = L(f) + \sum_{\mu=1}^{M} c_\mu n^{-k_\mu} E_\mu(f)$$

mit $M \in \mathbb{N}^* \cup \{\infty\}$, $k_1 < k_2 < \dots$, $c_\mu \in \mathbb{R}$ zulassen.
(2) $T_{1,k}$ ist die zusammengesetzte Simpsonregel zur Schrittweite 2^{-k}.

Der folgende Satz gibt eine Fehlerabschätzung für das Romberg-Verfahren. Er beruht auf der Fehlerentwicklung von Satz II.4.5 und zeigt, dass die Quadraturformel $T_{i,k}$ die Ordnung $2i+2$ hat.

SATZ II.4.8 (Fehlerabschätzung für das Romberg-Verfahren). *Sei* $f \in C^{2n+2}([0,1],\mathbb{R})$. *Dann gilt für* $k \in \mathbb{N}$ *und* $0 \le i \le \min\{k-1, n-1\}$

$$T_{i,k} = \int_0^1 f(x)\mathrm{d}x + \sum_{l=i+1}^{n} c_l^{(i)} h_k^{2l} + R_{n,i}(h_k)$$

mit

$$c_l^{(i)} = c_l^{(i)}(f) \in \mathbb{R}, \quad R_{n,i}(h_k) = \mathrm{O}(h_k^{2n+2}), \quad h_k = 2^{-k}.$$

BEWEIS. Der Beweis erfolgt durch Induktion über i.
$\underline{i = 0:}$ Folgt aus Satz II.4.5 mit

$$c_l^{(0)} = \frac{\beta_{2l}}{(2l)!}[f^{(2l-1)}(1) - f^{(2l-1)}(0)],$$
$$R_{n,0} = \frac{\beta_{2n+2}}{(2n+2)!}h^{2n+2}f^{(2n+2)}(\eta(h_k)).$$

$\underline{i \to i+1:}$ Aus der Induktionsvoraussetzung und der Definition von $T_{i+1,k}$ folgt

$$\begin{aligned} T_{i+1,k} &= \frac{4^{i+1}T_{i,k+1} - T_{i,k}}{4^{i+1}-1} \\ &= \int_0^1 f(x)\mathrm{d}x + c_{i+1}^{(i)} \frac{\overbrace{4^{i+1}h_{k+1}^{2i+2} - h_k^{2i+2}}^{=0}}{4^{i+1}-1} \\ &\quad + \sum_{l=i+2}^{n} c_l^{(i)} \frac{4^{i+1}h_{k+1}^{2l} - h_k^{2l}}{4^{i+1}-1} \\ &\quad + \frac{4^{i+1}R_{n,i}(h_{k+1}) - R_{n,i}(h_k)}{4^{i+1}-1} \\ &= \int_0^1 f(x)\mathrm{d}x + \sum_{l=i+2}^{n} c_l^{(i+1)} h_k^{2l} + R_{n,i+1}(h_k) \end{aligned}$$

mit

$$c_l^{(i+1)} = c_l^{(i)} \frac{4^{i+1-l}-1}{4^{i+1}-1},$$
$$R_{n+i+1} = \frac{4^{i+1}R_{n,i}(h_{k+1}) - R_{n,i}(h_k)}{4^{i+1}-1}.$$

□

BEISPIEL II.4.9. Tabelle II.4.1 zeigt die Ergebnisse des Romberg-Verfahrens angewandt auf das Integral $\int_0^1 \frac{3}{2}\sqrt{x} = 1$.

TABELLE II.4.1. Romberg-Verfahren für $\int_0^1 \frac{3}{2}\sqrt{x} = 1$

k	$T_{0,k}$	$T_{1,k}$	$T_{2,k}$	$T_{3,k}$	$T_{4,k}$
0	0.750000	0.957107	0.986635	0.995411	0.998389
1	0.905330	0.984789	0.995274	0.998378	0.999431
2	0.964925	0.994619	0.998329	0.999426	0.999799
3	0.987195	0.998097	0.999409	0.999797	0.999929
4	0.995372	0.999327	0.999791	0.999928	
5	0.998338	0.999762	0.999926		
6	0.999406	0.999916			
7	0.999788				

II.5. Spezielle Integranden

In diesem Paragraphen gehen wir stichwortartig auf mögliche Modifikationen der bisher betrachteten Techniken ein, wenn der Integrand nicht genügend glatt oder der Integrationsbereich unbeschränkt ist.

1. Ist f auf den Teilintervallen $[a_i, a_{i+1}]$, $0 \le i \le m-1$, $a = a_0 < \ldots < a_m = b$ hinreichend oft differenzierbar, aber global nicht glatt, so zerlege man $\int_a^b f\mathrm{d}x$ in die Teilintegrale $\int_{a_i}^{a_{i+1}} f\mathrm{d}x$ und approximiere diese separat.

2. Besitzt f in einer der Intervallgrenzen eine Singularität, so hilft häufig eine geeignete Substitution weiter; z.B.

$$\int_0^b \frac{\cos x}{\sqrt{x}}\mathrm{d}x = \int_0^{\sqrt{b}} 2\cos t^2 \mathrm{d}t.$$

3. Eine andere Möglichkeit den Fall 2 zu behandeln, besteht in einer Aufspaltung des Integrals verbunden mit einer schnell konvergierenden Reihenentwicklung des Integranden; z.B.

$$\begin{aligned}\int_0^b \frac{\cos x}{\sqrt{x}}\mathrm{d}x &= \int_\varepsilon^b \frac{\cos x}{\sqrt{x}}\mathrm{d}x + \int_0^\varepsilon \frac{\cos x}{\sqrt{x}}\mathrm{d}x \\ &= \int_\varepsilon^b \frac{\cos x}{\sqrt{x}}\mathrm{d}x + \sum_{k=0}^{\infty}(-1)^k \frac{1}{(2k)!(2k+\frac{1}{2})}\varepsilon^{2k+\frac{1}{2}}\end{aligned}$$

wobei die Reihe für kleines ε sehr schnell konvergiert.

4. Eine andere Möglichkeit im Falle 2 ist die Subtraktion einer Funktion, die die gleiche Singularität aufweist und die exakt integrierbar ist; z.B.

$$\begin{aligned}\int_0^b \frac{\cos x}{\sqrt{x}}\mathrm{d}x &= \int_0^b \frac{\cos x - 1}{\sqrt{x}}\mathrm{d}x + \int_0^b \frac{1}{\sqrt{x}}\mathrm{d}x \\ &= \int_0^b \underbrace{\frac{\cos x - 1}{\sqrt{x}}}_{\in C^1}\mathrm{d}x + 2\sqrt{b}.\end{aligned}$$

5. Bei uneigentlichen Integralen der Form $\int_0^\infty f\mathrm{d}x$ oder $\int_{-\infty}^\infty f\mathrm{d}x$ hilft häufig die Verwendung einer Gaußschen Formel zur Gewichtsfunktion e^{-x} oder e^{-x^2}.

6. Durch geeignete Transformationen der Form $x = \frac{t}{1-t}$ oder $x = -\ln(t)$ oder $t = \frac{e^x-1}{e^x+1}$ können Integrale der Form 5 auf solche der Form $\int_a^b g(t)\mathrm{d}t$ zurückgeführt werden. Allerdings handelt man sich dabei häufig Singularitäten des transformierten Integranden ein.

7. Bei Integralen der Form 5 hilft häufig auch das Abschneiden des Intervalls. So ist z.B.

$$\int_0^\infty e^{-x^2}\mathrm{d}x = \int_0^a e^{-x^2}\mathrm{d}x + R_a$$

mit

$$0 < R_a = \int_a^\infty e^{-x^2}\mathrm{d}x < \int_a^\infty e^{-ax}\mathrm{d}x = \frac{1}{a}e^{-a^2}$$

und $0 < R_3 < 5 \cdot 10^{-5}$.

KAPITEL III

Nichtlineare Gleichungssysteme

In diesem Kapitel betrachten wir Verfahren zur Lösung nichtlinearer Gleichungssysteme:

- die Fixpunktiteration,
- das Newton-Verfahren,
- Quasi-Newton-Verfahren, insb. gedämpfte Newton-Verfahren.

III.1. Fixpunktiterationen

Im Folgenden ist stets $(X, \|\cdot\|)$ ein Banach-Raum und M eine abgeschlossene, nicht leere Teilmenge von X.

Wir betrachten *Fixpunktgleichungen* der Form $f(x) = x$. Deren Lösungen wollen wir durch *Fixpunktiterationen* der Form

$$x_{n+1} = f(x_n), \quad x_0 \in X \text{ gegeben}$$

näherungsweise berechnen. Hierzu benötigen wir den Begriff der Kontraktion.

Definition III.1.1 (Fixpunkt, Kontraktion). *(1)* Ein Punkt $x^* \in M$ heißt *Fixpunkt* der Abbildung $f : M \to X$, wenn gilt $f(x^*) = x^*$.
(2) Eine Abbildung $f : M \to X$ heißt *Kontraktion* (in M), wenn $f(M) \subset M$ ist und es ein $\kappa \in [0, 1)$ gibt mit

$$\|f(x) - f(y)\| \leq \kappa \|x - y\| \quad \forall x, y \in M.$$

κ heißt dann die *Kontraktionsrate* von f.

Bemerkung III.1.2 (Eigenschaften von Kontraktionen). *(1)* Eine Kontraktion ist stetig.
(2) Ist $U \supset M$ offen und konvex, $f \in C^1(U, X)$ mit $f(M) \subset M$ und

$$\kappa = \sup_{x \in U} \|Df(x)\|_{\mathcal{L}(X,X)} < 1,$$

so ist f eine Kontraktion.

Beweis. *ad (1):* Ist offensichtlich.
ad (2): Folgt aus

$$\begin{aligned} \|f(x) - f(y)\| &= \left\| \int_0^1 Df(x + t(y - x))(y - x) \mathrm{d}t \right\| \\ &\leq \|x - y\| \sup_{z \in U} \|Df(z)\|_{\mathcal{L}(X,X)} . \end{aligned}$$

□

Der folgende Satz ist zentral und sichert die eindeutige Lösbarkeit von Fixpunktgleichungen. Er ist konstruktiv und gibt eine Vorschrift zur näherungsweisen Lösung der Gleichung an. Ganz wichtig sind die Fehlerabschätzungen, die eine explizite Kontrolle über die erreichte Genauigkeit der Näherungslösung erlauben. Die a posteriori Abschätzung ist genauer. Dafür kann die rechte Seite der a priori Abschätzung schon nach der ersten Iteration ausgewertet werden und erlaubt damit bei vorgegebener Genauigkeit eine Vorababschätzung über die maximal erforderliche Zahl an Iterationen.

Satz III.1.3 (Banachscher Fixpunktsatz). *Sei $f : M \to X$ eine Kontraktion in M. Dann besitzt f genau einen Fixpunkt x^* in M. Für jedes $x_0 \in M$ konvergiert die Folge $(x_n)_{n\in\mathbb{N}}$ mit*

$$x_{n+1} = f(x_n) \quad \forall n \in \mathbb{N} \quad \text{(Fixpunktiteration)}$$

gegen x^*. Weiter gelten die Fehlerabschätzungen

$$\|x^* - x_n\| \leq \frac{\kappa^n}{1-\kappa} \|x_1 - x_0\| \qquad \text{(a priori Abschätzung)},$$
$$\|x^* - x_n\| \leq \frac{\kappa}{1-\kappa} \|x_n - x_{n-1}\| \quad \text{(a posteriori Abschätzung)}.$$

Beweis. Die Eindeutigkeit eines Fixpunktes folgt sofort aus der Kontraktionseigenschaft.
Sei $x_0 \in M$ beliebig. Wegen $f(M) \subset M$ ist die Folge $(x_n)_{n\in\mathbb{N}}$ wohl definiert. Aus der Kontraktionseigenschaft und der Dreiecksungleichung folgt für $n \in \mathbb{N}$ und $m \in \mathbb{N}^*$

$$\|x_{n+m+1} - x_{n+m}\| \leq \kappa \|x_{n+m} - x_{n+m-1}\| \leq \ldots \leq \kappa^m \|x_{n+1} - x_n\|$$

und

$$\begin{aligned}\|x_{n+m} - x_n\| &\leq \sum_{k=0}^{m-1} \|x_{n+k+1} - x_{n+k}\| \leq \sum_{k=0}^{m-1} \kappa^k \|x_{n+1} - x_n\| \\ &\leq \frac{1-\kappa^m}{1-\kappa} \|x_{n+1} - x_n\| \leq \frac{1}{1-\kappa} \|x_{n+1} - x_n\| \\ &\leq \frac{\kappa^n}{1-\kappa} \|x_1 - x_0\| .\end{aligned}$$

Also ist $(x_n)_{n\in\mathbb{N}}$ eine Cauchy-Folge und konvergiert gegen ein $x^* \in M$. Wegen der Stetigkeit von f ist x^* ein Fixpunkt.
Die Fehlerabschätzungen folgen durch Grenzübergang $m \to \infty$ in obiger Abschätzung. □

Beispiel III.1.4. Setze $X = C([0,1], \mathbb{R})$, $\|\cdot\| = \|\cdot\|_{C([0,1],\mathbb{R})}$, $M = \{x \in X : \|x\| \leq 1\}$ und

$$f(x)(t) = \int_0^t \cos(x(s)) \mathrm{d}s \quad \forall x \in X, t \in [0,1].$$

Offensichtlich ist $x \in X$ ein Fixpunkt von f genau dann, wenn x das Anfangswertproblem

$$\begin{aligned} \frac{d}{dt}x(t) &= \cos(x(t)) \quad \forall 0 \le t \le 1 \\ x(0) &= 0 \end{aligned}$$

löst. Wegen

$$|f(x)(t)| \le \int_0^t |\cos(x(s))| \, \mathrm{d}s \le t \le 1$$

ist $f(X) \subset M$. Wegen

$$\begin{aligned} &|f(x)(t) - f(y)(t)| \\ &\le \int_0^t |\cos(x(s)) - \cos(y(s))| \, \mathrm{d}s \\ &= \int_0^t \left| \int_0^1 \sin[x(s) + \tau(y(s) - x(s))][x(s) - y(s)] \mathrm{d}\tau \right| \mathrm{d}s \\ &\le \sin(1) \, \|x - y\| \\ &\le 0.85 \, \|x - y\| \end{aligned}$$

ist f eine Kontraktion. Also besitzt das Anfangswertproblem eine eindeutige Lösung.

Häufig verfügt man über eine Abschätzung für die Kontraktionsrate und einen Kandidaten für den Startwert der Fixpunktiteration, weiß aber nicht, wie man die Menge M wählen soll und ob sie von f in sich abgebildet wird. In dieser Situation hilft folgende lokale Variante des Banachschen Fixpunktsatzes weiter.

Satz III.1.5 (Lokale Variante des Banachschen Fixpunktsatzes). *Zu $f : X \to X$ gebe es ein $x_0 \in X$, ein $R \in \mathbb{R}_+^*$ und ein $\kappa \in [0, 1)$ mit*

$$\|f(x) - f(y)\| \le \kappa \, \|x - y\| \quad \forall x, y \in \overline{B(x_0, R)}$$

und

$$\|x_0 - f(x_0)\| \le (1 - \kappa)R.$$

Dann besitzt f einen eindeutigen Fixpunkt x^* in $\overline{B(x_0, R)}$. Die Fixpunktiteration konvergiert für jeden Startwert in $\overline{B(x_0, R)}$, und es gelten die Fehlerabschätzungen von Satz III.1.3.

Beweis. Sei $x \in \overline{B(x_0, R)}$. Dann folgt

$$\|x_0 - f(x)\| \le \|x_0 - f(x_0)\| + \|f(x_0) - f(x)\| \le (1 - \kappa)R + \kappa R = R.$$

Also ist f eine Kontraktion in $\overline{B(x_0, R)}$, und die Behauptung folgt aus Satz III.1.3. □

BEISPIEL III.1.6. Eine einfache Skizze zeigt, dass die Funktion $f(x) = e^{-x}$ einen eindeutigen Fixpunkt hat, der zwischen 0 und 1 liegt. Wir wählen $x_0 = 0.6$ und $R = 0.2$. Dann ist $\overline{B(x_0, R)} = [0.4, 0.8]$ und

$$|f'(x)| \leq e^{-0.4} \approx 0.6703$$

für alle $x \in [0.4, 0.8]$. Also ist $\kappa \leq 0.68$. Andererseits ist

$$|x_0 - f(x_0)| \approx 0.051188 \leq 0.06 < 0.064 = 0.32 \cdot 0.2 \leq (1 - \kappa)R.$$

Bei der praktischen Durchführung der Fixpunktiteration treten in der Regel Rundungsfehler auf. Diese kann man auch so interpretieren, dass man exakt, d.h. ohne Rundungsfehler, mit einer gestörten Funktion g an Stelle von f rechnet. Der folgende Satz zeigt, welche Konsequenzen eine solche Störung für die Genauigkeit der Näherungslösung der Fixpunktgleichung hat.

SATZ III.1.7 (Gestörte Fixpunktiteration). *Sei $f : M \to X$ eine Kontraktion in M mit Kontraktionsrate κ und $g : M \to M$ eine Störung von f, d.h. es gibt ein $\varepsilon > 0$ mit*

$$\|f(x) - g(x)\| \leq \varepsilon \quad \forall x \in M.$$

Dann gilt für den Fixpunkt x^* von f in M und jeden Fixpunkt y^* von g in M (sofern vorhanden)

$$\|x^* - y^*\| \leq \frac{\varepsilon}{1 - \kappa}.$$

Für jedes $y_0 \in M$ gelten für die Folge $(y_n)_{n \in \mathbb{N}}$ mit

$$y_{n+1} = g(y_n) \quad \forall n \in \mathbb{N}$$

die Abschätzungen

$$\|x^* - y_n\| \leq \frac{\varepsilon}{1 - \kappa} + \frac{\kappa^n}{1 - \kappa} \|y_1 - y_0\|,$$
$$\|x^* - y_n\| \leq \varepsilon \frac{1 + \kappa}{(1 - \kappa)^2} + \frac{\kappa}{1 - \kappa} \|y_n - y_{n-1}\|.$$

BEWEIS. Sei $y^* \in M$ ein Fixpunkt von g. Dann folgt

$$\begin{aligned} \|x^* - y^*\| &= \|f(x^*) - g(y^*)\| \\ &\leq \|f(x^*) - f(y^*)\| + \|f(y^*) - g(y^*)\| \\ &\leq \kappa \|x^* - y^*\| + \varepsilon \end{aligned}$$

und somit

$$\|x^* - y^*\| \leq \frac{\varepsilon}{1 - \kappa}.$$

Sei $y_0 \in M$ beliebig. Wegen $g(M) \subset M$ ist die Folge $(y_n)_{n \in \mathbb{N}}$ wohl definiert. Setze für alle $n \in \mathbb{N}$

$$x_0 = y_0, \quad x_{n+1} = f(x_n).$$

Wie oben folgt für alle $n \in \mathbb{N}$

$$\|x_n - y_n\| \leq \kappa \|x_{n-1} - y_{n-1}\| + \varepsilon$$

und daher mit Induktion

$$\|x_n - y_n\| \leq \varepsilon \frac{1-\kappa^n}{1-\kappa}.$$

Hieraus folgt mit den Abschätzungen aus Satz III.1.3

$$\begin{aligned}
\|x^* - y_n\| &\leq \|x_n - y_n\| + \|x^* - x_n\| \\
&\leq \varepsilon \frac{1-\kappa^n}{1-\kappa} + \frac{\kappa^n}{1-\kappa} \|x_1 - x_0\| \\
&\leq \varepsilon \frac{1-\kappa^n}{1-\kappa} + \frac{\kappa^n}{1-\kappa} \left\{ \underbrace{\|x_1 - y_1\|}_{\leq \varepsilon} + \|y_1 - y_0\| \right\} \\
&\leq \frac{\varepsilon}{1-\kappa} + \frac{\kappa^n}{1-\kappa} \|y_1 - y_0\|
\end{aligned}$$

und

$$\begin{aligned}
\|x^* - y_n\| &\leq \varepsilon \frac{1-\kappa^n}{1-\kappa} + \frac{\kappa}{1-\kappa} \|x_n - x_{n-1}\| \\
&\leq \varepsilon \frac{1-\kappa^n}{1-\kappa} \\
&\quad + \frac{\kappa}{1-\kappa} \{\|x_n - y_n\| + \|y_n - y_{n-1}\| + \|y_{n-1} - x_{n-1}\|\} \\
&\leq \frac{\varepsilon}{(1-\kappa)^2} \left[(1-\kappa)(1-\kappa^n) + \kappa(1-\kappa^n) + \kappa(1-\kappa^{n-1})\right] \\
&\quad + \frac{\kappa}{1-\kappa} \|y_n - y_{n-1}\| \\
&\leq \varepsilon \frac{1+\kappa}{(1-\kappa)^2} + \frac{\kappa}{1-\kappa} \|y_n - y_{n-1}\| . \qquad \square
\end{aligned}$$

III.2. Das Newton-Verfahren

Im Folgenden ist $(X, \|\cdot\|)$ ein Banach-Raum, U eine offene, nicht leere Teilmenge und $f \in C^1(U, X)$. Gesucht ist eine Nullstelle von f, d.h. ein $x^* \in U$ mit

$$f(x^*) = 0. \tag{III.2.1}$$

Das Newton-Verfahren ist durch folgende Überlegung motiviert: Angenommen wir verfügen über Näherung x_0 für x^*. In der Nähe von x_0 ist dann

$$f(x) \approx f(x_0) + Df(x_0)(x - x_0). \tag{III.2.2}$$

Daher ersetzen wir in Gleichung (III.2.1) f durch die rechte Seite von (III.2.2) und erhalten die lineare Bestimmungsgleichung

$$f(x_0) + Df(x_0)(x - x_0) = 0$$

mit der Lösung

$$x_1 = x_0 - Df(x_0)^{-1} f(x_0),$$

vorausgesetzt natürlich, die Ableitung $Df(x_0)$ ist invertierbar. Wegen (III.2.2) hoffen wir, dass x_1 eine bessere Näherung für x^* ist als x_0.

DEFINITION III.2.1 (Newton-Verfahren; `newton` in `Numerics`). Das *Newton-Verfahren* ist gegeben durch die Iterationsvorschrift $x_0 \in U$ und – so lange $Df(x_n)$ invertierbar ist –

$$x_{n+1} = x_n - Df(x_n)^{-1} f(x_n).$$

Algorithmus III.2.1 realisiert das Newton-Verfahren. Man beachte, dass $Df(x_n)^{-1} f(x_n)$ durch Lösen eines linearen Gleichungssystems bestimmt wird.

Algorithmus III.2.1 Newton-Verfahren

Gegeben: Funktion f, Startwert x, Toleranz ε, Maximalzahl an Iterationen N
Gesucht: Näherungslösung x mit $\|f(x)\| \leq \varepsilon$

1: $n \leftarrow 0$
2: **while** $\|f(x)\| > \varepsilon$ und $\|Df(x)\| > \varepsilon$ und $n \leq N$ **do**
3: Löse das lineare Gleichungssystem $Df(x)z = -f(x)$
4: $x \leftarrow x + z$, $n \leftarrow n + 1$
5: **end while**

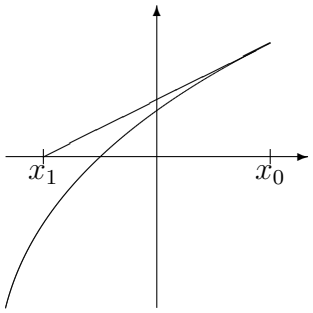

ABBILDUNG III.2.1. Geometrische Interpretation des Newton-Verfahrens

BEMERKUNG III.2.2 (Geometrische Interpretation und quadratische Konvergenz). *(1)* Ist $X = \mathbb{R}$, so ist x_{n+1} der Schnittpunkt der Tangente an f im Punkt x_n mit der x-Achse (vgl. Abbildung III.2.1). *(2)* Sei U konvex, $f \in C^2(U, X)$, $\sup_{x \in U} \|Df(x)^{-1}\|_{\mathcal{L}(X,X)} < \infty$ und x^* eine Nullstelle von f in U. Dann folgt für das Newton-Verfahren

$$\begin{aligned} x_{n+1} - x^* &= x_n - Df(x_n)^{-1} f(x_n) - x^* \\ &= Df(x_n)^{-1} \left[Df(x_n)(x_n - x^*) - f(x_n) \right] \\ &= Df(x_n)^{-1} \left[Df(x_n)(x_n - x^*) - f(x_n) + \underbrace{f(x^*)}_{=0} \right]. \end{aligned}$$

Aus dem Hauptsatz der Differential- und Integralrechnung ergibt sich

$$\begin{aligned}
&Df(x_n)(x_n - x^*) - f(x_n) + f(x^*) \\
&\quad = Df(x_n)(x_n - x^*) - \int_0^1 Df(x^* + t(x_n - x^*))(x_n - x^*)\mathrm{d}t \\
&\quad = \int_0^1 \left[Df(x_n) - Df(x^* + t(x_n - x^*))\right](x_n - x^*)\mathrm{d}t.
\end{aligned}$$

Abermalige Anwendung des Hauptsatzes der Differential- und Integralrechnung liefert

$$\begin{aligned}
&\int_0^1 \left[Df(x_n) - Df(x^* + t(x_n - x^*))\right](x_n - x^*)\mathrm{d}t \\
&\quad = \int_0^1 \int_0^1 (1-t)D^2 f(x^* + t(x_n - x^*) + s(1-t)(x_n - x^*)) \\
&\qquad\qquad [x_n - x^*, x_n - x^*]\mathrm{d}s\mathrm{d}t \\
&\quad = \int_0^1 \int_0^1 (1-t)D^2 f(x^* + (t + s - st)(x_n - x^*)) \\
&\qquad\qquad [x_n - x^*, x_n - x^*]\mathrm{d}s\mathrm{d}t
\end{aligned}$$

Da U konvex ist, ist $x^* + (t+s-st)(x_n - x^*) \in U$ für jedes $s, t \in [0,1]$. Daher ist

$$\begin{aligned}
&\left\| \int_0^1 \int_0^1 (1-t)D^2 f(x^* + (t + s - st)(x_n - x^*)) \right. \\
&\qquad\qquad \left. [x_n - x^*, x_n - x^*]\mathrm{d}s\mathrm{d}t \right\| \\
&\quad \leq \sup_{x \in U} \left\|D^2 f(x)\right\|_{\mathcal{L}^2(X)} \|x_n - x^*\|^2 \int_0^1 \int_0^1 (1-t)\mathrm{d}s\mathrm{d}t \\
&\quad = \frac{1}{2} \sup_{x \in U} \left\|D^2 f(x)\right\|_{\mathcal{L}^2(X)} \|x_n - x^*\|^2 .
\end{aligned}$$

Damit folgt insgesamt

$$\|x_{n+1} - x^*\| \leq \frac{1}{2} \sup_{x \in U} \left\|Df^{-1}(x)\right\|_{\mathcal{L}(X,X)} \sup_{x \in U} \left\|D^2 f(x)\right\|_{\mathcal{L}^2(X)} \|x_n - x^*\|^2 .$$

Das heißt, falls das Newton-Verfahren konvergiert, ist die Konvergenz *quadratisch*

$$\|x_{n+1} - x^*\| \leq c \|x_n - x^*\|^2 .$$

Die quadratische Konvergenz äußert sich in einer Verdoppelung der exakten Dezimalstellen mit jeder Iteration, so bald $c\|x_n - x^*\| \leq 1$ ist.

Das folgende Beispiel illustriert die quadratische Konvergenz des Newton-Verfahrens. Es zeigt aber auch, dass wir nicht erwarten können, dass das Newton-Verfahren für jeden Startwert konvergiert.

TABELLE III.2.1. Newton-Verfahren für $f(x) = x^2 - 2$

n	x_n	$f(x_n)$	$\lvert x_n - x^*\rvert$
0	2.0	2.0	0.5857864
1	1.5	0.25	0.0857864
2	$1.41\overline{6}$	0.00694	0.0024531
3	1.41421568627	$6 \cdot 10^{-6}$	$2 \cdot 10^{-6}$

BEISPIEL III.2.3. *(1)* Tabelle III.2.1 zeigt die Ergebnisse des Newton-Verfahrens angewandt auf die Funktion $f(x) = x^2 - 2$ mit Startwert $x_0 = 2$.
(2) Sei $U = X = \mathbb{R}$ und $f(x) = x^4 - 3x^2 - 2$. Dann lautet das Newton-Verfahren mit $x_0 = 1$,

$$\begin{aligned} x_1 &= 1 - \frac{-4}{-2} &&= -1, \\ x_2 &= -1 - \frac{-4}{2} &&= 1 = x_0. \end{aligned}$$

Man gerät in einen sogenannten *Kessel*.
(3) Sei $U = X = \mathbb{R}$ und $f(x) = \arctan x$. Wegen

$$\lim_{y\to\infty} \arctan(y) = \frac{\pi}{2}, \quad \lim_{y\to\infty} \frac{3y}{1+y^2} = 0$$

gibt es ein $R > 0$ mit

$$\arctan(y)(1 + y^2) \geq 3y$$

für alle $y \geq R$. Sei $x_0 \in \mathbb{R}$ mit $|x_0| \geq R$. Dann liefert das Newton-Verfahren

$$\begin{aligned} |x_1| &= \left|x_0 - (1 + x_0^2) \arctan(x_0)\right| \\ &\geq (1 + |x_0|^2) \arctan|x_0| - |x_0| \\ &\geq 2\,|x_0|\,. \end{aligned}$$

Mit Induktion folgt für $n \in \mathbb{N}^*$

$$|x_n| \geq 2^n\,|x_0|\,,$$

d.h., das Newton-Verfahren divergiert. Tabelle III.2.2 zeigt die Ergebnisse des Newton-Verfahrens mit Startwert $x_0 = 2$. Man vergleiche diese Ergebnisse mit denen aus Tabelle III.3.2 (S. 75) für das gedämpfte Newton-Verfahren.

Wir wollen zeigen, dass das Newton-Verfahren unter gewissen Voraussetzungen konvergiert. Dazu benötigen wir die folgenden technischen Hilfsergebnisse.

LEMMA III.2.4. *Sei $A \in \mathcal{L}(X, X)$ mit $\alpha = \|A\|_{\mathcal{L}(X,X)} < 1$. Dann ist $I - A \in \mathcal{GL}(X)$ und $\|(I - A)^{-1}\|_{\mathcal{L}(X,X)} \leq \frac{1}{1-\alpha}$.*

TABELLE III.2.2. Newton-Verfahren für $f(x) = \arctan(x)$

n	x_n	$f(x_n)$
0	2.0	1.107148
1	-3.535743	-1.295169
2	13.950959	1.499239
3	-279.344066	-1.567216
4	122016.998918	1.570788

BEWEIS. Wegen $\alpha < 1$ konvergiert die Reihe $\sum A^n$ in der Norm von $\mathcal{L}(X,X)$ gegen ein $B \in \mathcal{L}(X,X)$ mit $\|B\|_{\mathcal{L}(X,X)} \leq \frac{1}{1-\alpha}$. Wie man leicht nachrechnet, ist $B(I-A) = (I-A)B = I$. □

BEMERKUNG III.2.5. Lemma III.2.4 impliziert, dass $\mathcal{GL}(X)$ offen in $\mathcal{L}(X,X)$ und die Abbildung $\mathcal{GL}(X) \ni A \mapsto A^{-1}$ stetig ist.

LEMMA III.2.6. *Seien $U \subset X$ konvex und $f \in C^1(U,X)$. Es gebe ein $\gamma \in \mathbb{R}_+$ mit*

$$\|Df(x) - Df(y)\|_{\mathcal{L}(X,X)} \leq \gamma \|x-y\| \quad \forall x,y \in U.$$

Dann gilt für alle $x,y \in U$

$$\|f(x) - f(y) - Df(y)(x-y)\| \leq \frac{\gamma}{2}\|x-y\|^2.$$

BEWEIS. Seien $x,y \in U$ beliebig. Da U konvex ist, folgt

$$\begin{aligned}
&\|f(x) - f(y) - Df(y)(x-y)\| \\
&= \left\| \int_0^1 [Df(\underbrace{y + t(x-y)}_{\in U}) - Df(y)](x-y)\mathrm{d}t \right\| \\
&\leq \int_0^1 \|Df(y+t(x-y)) - Df(y)\|_{\mathcal{L}(X,X)} \|x-y\| \,\mathrm{d}t \\
&\leq \int_0^1 \gamma t \|x-y\|^2 \,\mathrm{d}t \\
&= \frac{1}{2}\gamma \|x-y\|^2.
\end{aligned}$$

□

BEMERKUNG III.2.7. Die Voraussetzungen von Lemma III.2.6 sind erfüllt, wenn $f \in C^2(U,X)$ und $\sup_{x\in U} \|D^2 f(x)\|_{\mathcal{L}^2(X)} < \infty$ ist.

Wir kommen nun zu dem angekündigten Konvergenzsatz für das Newton-Verfahren.

SATZ III.2.8 (3-Faktoren-Satz, Satz von Newton-Kantorovič). *Sei $U \subset X$ offen, $f \in C^1(U,X)$ und $x_0 \in U$ mit $Df(x_0) \in \mathcal{GL}(X)$. Definiere*

$$\alpha = \left\|Df(x_0)^{-1} f(x_0)\right\|, \quad \beta = \left\|Df(x_0)^{-1}\right\|_{\mathcal{L}(X,X)}, \quad R = \frac{3}{2}\alpha.$$

Weiter gelte

(1) $\overline{B(x_0,R)} \subset U$,
(2) $\|Df(x) - Df(y)\|_{\mathcal{L}(X,X)} \leq \gamma \|x-y\|$ *für alle* $x, y \in B(x_0,R)$,
(3) $h = \alpha\beta\gamma \leq \frac{1}{3}$.

Dann ist das Newton-Verfahren mit Startwert x_0 *durchführbar. Die damit gewonnene Folge* $(x_n)_{n\in\mathbb{N}}$ *ist in* $B(x_0,R)$ *enthalten und konvergiert gegen eine Nullstelle* x^* *von* f *in* $\overline{B(x_0,R)}$. *Für jedes* $n \in \mathbb{N}$ *gilt die Fehlerabschätzung*

$$\|x_n - x^*\| \leq \alpha \frac{h^{2^n-1}}{1-h^{2^n}}.$$

BEWEIS. *1. Schritt:* Für alle $x \in B(x_0,R)$ ist $Df(x) \in \mathcal{GL}(X)$ und $\|Df(x)^{-1}\|_{\mathcal{L}(X,X)} \leq 2\beta$.
Bew.: Folgt aus

$$Df(x) = Df(x_0)\left\{I - Df(x_0)^{-1}[Df(x_0) - Df(x)]\right\}$$

und

$$\left\|Df(x_0)^{-1}[Df(x) - Df(x_0)]\right\|_{\mathcal{L}(X,X)} \leq \beta\gamma \|x - x_0\| \leq \frac{3}{2}h \leq \frac{1}{2}$$

sowie Lemma III.2.4.
2. Schritt: Das Newton-Verfahren ist durchführbar, es ist $(x_n)_{n\in\mathbb{N}} \subset B(x_0,R)$ und für alle $n \in \mathbb{N}$ gilt

$$\|x_{n+1} - x_n\| \leq \alpha h^{2^n-1}. \tag{III.2.3}$$

Bew.: $\underline{n=0:}$ Ist offensichtlich.
$\underline{n \to n+1:}$ Wegen Schritt 1 ist x_{n+1} wohl definiert. Aus (III.2.3) für $n-1$ und Lemma III.2.6 folgt mit der Induktionsannahme

$$\begin{aligned}
\|x_{n+1} - x_n\| &= \left\|-Df(x_n)^{-1} f(x_n)\right\| \\
&\leq 2\beta \left\| f(x_n) \underbrace{- f(x_{n-1}) - Df(x_{n-1})(x_n - x_{n-1})}_{=0} \right\| \\
&\leq \gamma\beta \|x_n - x_{n-1}\|^2 \quad \text{(Induktionsannahme)} \\
&\leq \gamma\beta\alpha^2 h^{2(2^{n-1}-1)} \\
&= \alpha h^{2^n-1}.
\end{aligned}$$

Dies ist (III.2.3) für n. Aus (III.2.3) folgt schließlich

$$\|x_{n+1} - x_0\| \leq \sum_{j=0}^{n} \|x_{j+1} - x_j\| \leq \alpha \sum_{k=0}^{\infty} h^k \leq \frac{3}{2}\alpha = R.$$

3. Schritt: Die Folge $(x_n)_{n\in\mathbb{N}}$ konvergiert und es gilt die Fehlerabschätzung des Satzes.

Bew.: Aus (III.2.3) folgt für $n \in \mathbb{N}$, $m \in \mathbb{N}^*$

$$\begin{aligned}\|x_{n+m} - x_n\| &\le \alpha \sum_{j=0}^{m-1} h^{2^{n+j}-1} = \alpha h^{2^n-1} \sum_{j=0}^{m-1} h^{2^{n+j}-2^n} \\ &\le \alpha h^{2^n-1} \sum_{j=0}^{m-1} (h^{2^n})^j \quad \text{(wegen } 2^j - 1 \ge j\text{)} \\ &\le \alpha \frac{h^{2^n-1}}{1-h^{2^n}}.\end{aligned}$$

Also ist $(x_n)_{n\in\mathbb{N}}$ eine Cauchy-Folge und konvergiert gegen ein $x^* \in \overline{B(x_0, R)}$. Die Fehlerabschätzung folgt durch Grenzübergang $m \to \infty$.
4. Schritt: x^* ist eine Nullstelle von f.
Bew.: Für $n \in \mathbb{N}$ ist

$$\begin{aligned}\|Df(x_n)\|_{\mathcal{L}(X,X)} &\le \|Df(x_0)\|_{\mathcal{L}(X,X)} + \|Df(x_0) - Df(x_n)\|_{\mathcal{L}(X,X)} \\ &\le \|Df(x_0)\|_{\mathcal{L}(X,X)} + \gamma R \\ &= K.\end{aligned}$$

Hieraus folgt mit (III.2.3)

$$\|f(x_n)\| = \|-Df(x_n)(x_{n+1} - x_n)\| \le K\alpha h^{2^n-1} \underset{n\to\infty}{\longrightarrow} 0.$$

Wegen der Stetigkeit von f folgt hieraus die Behauptung. □

Bemerkung III.2.9 (Konvergenz bei einfachen Nullstellen, globale Konvergenz). *(1)* Ist $f \in C^2(U, X)$ und besitzt f eine Nullstelle x^* mit $Df(x^*) \in \mathcal{GL}(X)$, so sind die Voraussetzungen von Satz III.2.8 für alle $x_0 \in B(x^*, \varepsilon)$ mit hinreichend kleinem $\varepsilon > 0$ erfüllt.
(2) Unter geeigneten Voraussetzungen konvergiert das Newton-Verfahren global. Ist z.B. $X = \mathbb{R}$, U ein perfektes Intervall und $f \in C^2(U, \mathbb{R})$ strikt konvex bzw. strikt konkav mit einer Nullstelle x^* in U, so konvergiert das Newton-Verfahren für jeden Startwert $x_0 \in U$ mit $f(x_0) > 0$ bzw. $f(x_0) < 0$.

Beweis. *ad (1):* Setze

$$K = \left\|D^2 f(x^*)\right\|_{\mathcal{L}^2(X)}, \; L = \left\|Df(x^*)^{-1}\right\|_{\mathcal{L}(X,X)}, \; M = \|Df(x^*)\|_{\mathcal{L}(X,X)}.$$

Wegen der Offenheit von U, der zweimaligen stetigen Differenzierbarkeit von f und Bemerkung III.2.5 gibt es ein $\varepsilon_1 > 0$ mit $B(x^*, \varepsilon_1) \subset U$ und

$$\left\|D^2 f(x)\right\|_{\mathcal{L}^2(X)} \le 2K, \; \left\|Df^{-1}(x)\right\|_{\mathcal{L}(X,X)} \le 2L, \; \|Df(x)\|_{\mathcal{L}(X,X)} \le 2M$$

für alle $x \in B(x^*, \varepsilon_1)$. Wir wollen nun ein $\varepsilon \in (0, \varepsilon_1)$ so bestimmen, dass die Voraussetzungen von Satz III.2.8 für alle $x_0 \in B(x^*, \varepsilon)$ erfüllt sind. Seien dazu $0 < \varepsilon < \varepsilon_1$, $x_0 \in B(x^*, \varepsilon)$ und $x, y \in B(x_0, \varepsilon)$ beliebig. Aus der Dreiecksungleichung folgt

$$\|x - x^*\| \le \|x_0 - x^*\| + \|x - x_0\| < 2\varepsilon.$$

Also gewährleistet die Bedingung $\varepsilon \leq \frac{1}{2}\varepsilon_1$ die Inklusion $B(x_0, \varepsilon) \subset B(x^*, \varepsilon_1)$. Weiter folgt aus dem Hauptsatz der Differential- und Integralrechnung wegen $f(x^*) = 0$

$$\begin{aligned}\|f(x)\| = \|f(x) - f(x^*)\| &= \left\| \int_0^1 Df(x^* + t(x - x^*))(x - x^*)\mathrm{d}t \right\| \\ &\leq 2M\varepsilon.\end{aligned}$$

Also ist in Satz III.2.8 $\alpha \leq 4LM\varepsilon$, $\beta \leq 2L$ und $R \leq 6LM\varepsilon$. Schließlich folgt wiederum aus dem Hauptsatz der Differential- und Integralrechnung

$$\begin{aligned}\|Df(x) - Df(y)\|_{\mathcal{L}(X,X)} &= \left\| \int_0^1 D^2 f(y + t(x - y))(x - y)\mathrm{d}t \right\|_{\mathcal{L}(X,X)} \\ &\leq 2K \|x - y\| .\end{aligned}$$

Also ist in Satz III.2.8 $\gamma \leq 2K$ und $h \leq 16KL^2M\varepsilon$. Daher leistet $\varepsilon = \min\left\{\frac{1}{2}\varepsilon_1, \frac{\varepsilon_1}{6LM}, \frac{1}{48KL^2M}\right\}$ das Gewünschte.
ad (2): O.E. ist f strikt konvex, sonst betrachte $-f$. O.E. ist $x_0 > x^*$, sonst führe die Transformation $x \mapsto 2x^* - x$ aus. Wir betrachten den Fall $f(x_0) > 0$. Wegen $f(x^*) = 0$ ist die Steigung der Sekante durch die Punkte $(x^*, f(x^*)) = (x^*, 0)$ und $(x_0, f(x_0))$ positiv. Wegen des Mittelwertsatzes und der strikten Konvexität von f ist die Steigung kleiner als $f'(x_0)$. Also ist der Newtonschritt $x_0 \mapsto x_1$ durchführbar. Da x_1 der Schnittpunkt der Tangente in $(x_0, f(x_0))$ mit der x-Achse ist, folgt $x^* < x_1 < x_0$. Außerdem ist $f(x_1) > 0$, da sonst die Steigung der Sekante durch $(x_1, f(x_1))$ und $(x_0, f(x_0))$ größer wäre als $f'(x_0)$, im Widerspruch zum Mittelwertsatz und der strikten Konvexität von f. Also können wir obige Argumentation für x_0 auch auf x_1 anwenden. Insgesamt folgt, dass das Newton-Verfahren durchführbar ist und eine monoton fallende, nach unten durch x^* beschränkte Folge $(x_n)_{n\in\mathbb{N}}$ liefert. Diese konvergiert gegen ein $x^{**} \geq x^*$, das konstruktionsgemäß eine Nullstelle von f ist. Man beachte, dass $x^{**} \neq x^*$ durchaus möglich ist. □

III.3. Quasi-Newton-Verfahren

Bei der praktischen Durchführung des Newton-Verfahrens treten einige Schwierigkeiten auf:

- Die Berechnung der Ableitung $Df(x_n)$ ist aufwändig.
- Die Lösung des LGS $Df(x_n)z_n = -f(x_n)$ ist aufwändig.
- Das Verfahren ist nur lokal konvergent und besonders zu Beginn werden zu große Schritte ausgeführt.

Wir stellen im Folgenden einige Modifikationen des Newton-Verfahrens vor, die versuchen, diese Schwierigkeiten zu vermeiden.

Sei zunächst $X = \mathbb{R}$. Dann kann man $Df(x_n)$ durch den Differenzenquotienten $\frac{f(x_n)-f(x_{n-1})}{x_n-x_{n-1}}$ approximieren. Dies führt auf Algorithmus III.3.1.

Algorithmus III.3.1 Sekanten-Verfahren

Gegeben: Startwerte x, x, Toleranz ε, Maximalzahl an Iterationen N
Gesucht: Näherungslösung mit $\min\{|f(x)|, |f(y)|\} \leq \varepsilon$

1: $n \leftarrow 0$
2: **while** $\min\{|f(x)|, |f(y)|\} > \varepsilon$ und $|f(x) - f(y)| > \varepsilon$ und $n \leq N$ **do**
3: $\quad z \leftarrow \dfrac{f(x)y - f(y)x}{f(x) - f(y)}$
4: $\quad y \leftarrow x$, $x \leftarrow z$, $n \leftarrow n + 1$
5: **end while**
6: $x \leftarrow \operatorname{argmin}\{|f(x)|, |f(y)|\}$

Man kann dieses Verfahren so modifizieren, dass man eine Einschließung der Nullstelle erhält und im Nenner von Schritt 1 keine Auslöschung auftritt. Dies führt auf Algorithmus III.3.2.

Algorithmus III.3.2 Regula Falsi

Gegeben: Startwerte x, y mit $f(x)f(y) \leq 0$
Gesucht: Näherungslösung mit $\min\{|f(x)|, |f(y)|\} \leq \varepsilon$

1: $n \leftarrow 0$
2: **while** $\min\{|f(x)|, |f(y)|\} > \varepsilon$ und $|f(x) - f(y)| > \varepsilon$ und $n \leq N$ **do**
3: $\quad z \leftarrow \dfrac{f(x)y - f(y)x}{f(x) - f(y)}$
4: $\quad y \leftarrow \begin{cases} x & \text{falls } f(x)f(z) \leq 0, \\ y & \text{falls } f(y)f(z) \leq 0 \end{cases}$
5: $\quad x \leftarrow z$, $n \leftarrow n + 1$
6: **end while**
7: $x \leftarrow \operatorname{argmin}\{|f(x)|, |f(y)|\}$

Beispiel III.3.1. Tabelle III.3.1 gibt die Ergebnisse des Newton-Verfahrens, des Sekanten-Verfahrens und der Regula Falsi angewandt auf das Polynom $x^4 - 3x^2 - 2$ aus Beispiel III.2.3(2) (S. 68) wieder. Da bei diesem Beispiel die Werte y der Regula Falsi immer gleich dem Startwert $y = 2$ sind, geben wir für die Regula Falsi nur die x-Werte an.

Sei nun $X = \mathbb{R}^d$, $d \geq 2$. Algorithmus III.3.3 ist eine Variante des Newton-Verfahrens und reduziert den Aufwand zur Lösung des linearen Gleichungssystems mit einem der in Kapitel IV betrachteten Verfahren.

TABELLE III.3.1. Vergleich von Newton-Verfahren, Sekanten-Verfahren und Regula Falsi für $f(x) = x^4 - 3x^2 - 2$

n	Newton-Verfahren	Sekanten-Verfahren	Regula Falsi
0	2	1	1
1	1.9	2	1.66667
2	1.8874	1.66667	1.85562
3	1.88721	1.85562	1.88326
4		1.89735	1.88672
5		1.88682	1.88715
6		1.88720	1.88720
7		1.88721	1.88721

Algorithmus III.3.3 Modifiziertes Newton-Verfahren

Gegeben: Startwert x, Restartwert M, Toleranz ε, Maximalzahl für Iterationen N
Gesucht: Näherungslösung x mit $\|f(x)\| \leq \varepsilon$

```
1: n ← 0
2: while ‖f(x)‖ > ε und ‖Df(x)‖ > ε und n ≤ N do
3:     if n = 0 mod M then
4:         B = Df(x)
5:     end if
6:     Löse das lineare Gleichungssystem Bz = −f(x).
7:     x ← x + z, n ← n + 1
8: end while
```

Sei nun X ein Hilbert-Raum mit Skalarprodukt $(\cdot\,,\,\cdot)$ und zugehöriger Norm $\|\cdot\|$, $U \subset X$, $U \neq \emptyset$, offen und $f \in C^1(U, X)$. Dann ist jede Nullstelle von f ein Minimum der Funktion $F : U \to \mathbb{R}$ mit

$$F(x) = \|f(x)\|^2 = (f(x)\,,\, f(x))\,.$$

Daher kann man versuchen, die Funktion F in jedem Newtonschritt zu minimieren. Sei dazu $x \in U$ mit $f(x) \neq 0$, $Df(x) \in \mathcal{GL}(X)$ und $z = -Df(x)^{-1} f(x)$. Sei $\delta \in (0, 1]$ so, dass $B(x, \delta\,\|z\|) \subset U$ ist. Dann ist die Funktion $\varphi : (-\delta, \delta) \to \mathbb{R}$ mit

$$\varphi(t) = F(x + tz) = (f(x + tz)\,,\, f(x + tz))$$

stetig differenzierbar mit

$$\begin{aligned} \varphi(0) &= F(x) &&= \|f(x)\|^2 \\ \varphi'(0) &= 2\,(f(x)\,,\, Df(x)z) &&= -\,2\,\|f(x)\|^2\,. \end{aligned}$$

Daher gibt es eine in $t = 0$ stetige Funktion $r : (-\delta, \delta) \to \mathbb{R}$ mit $r(0) = 0$ und

$$\begin{aligned}\|f(x+tz)\|^2 = \varphi(t) &= \varphi(0) + t\varphi'(0) + tr(t)\\ &= (1-2t)\|f(x)\|^2 + tr(t).\end{aligned}$$

Also gibt es ein $\varepsilon \in (0, \delta]$ mit

$$\|f(x+tz)\|^2 \leq (1-t)\|f(x)\|^2 < \|f(x)\|^2$$

für alle $0 < t \leq \varepsilon$.
Dies führt auf Algorithmus III.3.4.

Algorithmus III.3.4 Gedämpftes Newton-Verfahren

Gegeben: Startwert x, Toleranz ε, Maximalzahl für Iterrationen N
Gesucht: Näherungslösung x mit $\|f(x)\| \leq \varepsilon$

1: $n \leftarrow 0$
2: **while** $\|f(x)\| > \varepsilon$ und $\|Df(x)\| > \varepsilon$ und $n \leq N$ **do**
3: Löse das lineare Gleichungssystem $Df(x)z = -f(x)$.
4: $t \leftarrow 1$
5: **while** $\|f(x+tz)\|^2 > (1 - \frac{t}{2})\|f(x)\|^2$ **do**
6: $t \leftarrow \frac{t}{2}$.
7: **end while**
8: $x \leftarrow x + tz$, $n \leftarrow n + 1$
9: **end while**

Bemerkung III.3.2. *(1)* Die Ideen der Algorithmen III.3.3 und III.3.4 können kombiniert werden.
(2) In der Nähe einer Nullstelle ist bei Algorithmus III.3.4 stets $t = 1$, so dass das Verfahren quadratisch konvergiert.

Beispiel III.3.3. Tabelle III.3.2 zeigt die Ergebnisse des gedämpften Newton-Verfahrens für die Funktion $f(x) = \arctan(x)$ und den Startwert $x = 2$. Man vergleiche diese Ergebnisse mit denjenigen aus Tabelle III.2.2 (S. 69) für das Newton-Verfahren ohne Dämpfung.

Tabelle III.3.2. Gedämpftes Newton-Verfahren für $\arctan(x)$

n	x_n	$f(x_n)$
0	2.000000	1.107148
1	-0.767871	-0.654841
2	0.273081	0.266581
3	-0.013380	-0.013379
4	0.000001	0.000001

KAPITEL IV

Lineare Gleichungssysteme

In diesem Kapitel betrachten wir direkte und iterative Verfahren zur Lösung linearer Gleichungssysteme. Die direkten Verfahren liefern bis auf Rundungsfehler nach endlich vielen Schritten die exakte Lösung. Sie beruhen alle im Prinzip auf dem Gaußschen Eliminationsverfahren. Die betrachteten iterativen Verfahren führen das lineare Gleichungssystem entweder auf ein Fixpunktproblem, auf das die Fixpunktiteration angewandt wird, oder auf ein Minimierungsproblem, auf das ein modifiziertes Gradienten-Verfahren angewandt wird, zurück.

IV.1. Der Gauß-Algorithmus

Im Folgenden ist $A \in \mathbb{R}^{n\times n}$ eine gegebene Matrix und $b \in \mathbb{R}^n$ ein gegebener Vektor. Gesucht ist die Lösung des linearen Gleichungssystems (LGS)

$$Ax = b. \tag{IV.1.1}$$

Die Idee des Gauß-Algorithmus ist folgende:
Für $i = 1, 2, \ldots, n-1$ subtrahiere man im i-ten Schritt für $j = i+1, \ldots, n$ nacheinander das A_{ji}/A_{ii}-fache der i-ten Gleichung von der j-ten Gleichung und eliminiere so die i-te Unbekannte aus der j-ten Gleichung.
Falls immer $A_{ii} \neq 0$ ist, erhält man so nach i-Schritten ein zu (IV.1.1) äquivalentes Gleichungssystem

$$A^{(i)}x = b^{(i)}$$

mit einer Matrix $A^{(i)}$ der Form

$$A^{(i)} = \begin{pmatrix} a_{11} & & \cdots & & \cdots & a_{1n} \\ & \ddots & & & & \\ & & a_{ii} & \cdots & & a_{in} \\ & & 0 & a_{i+1i+1} & \cdots & a_{i+1n} \\ 0 & & \vdots & \vdots & & \vdots \\ & & 0 & a_{ni+1} & \cdots & a_{nn} \end{pmatrix}.$$

Insbesondere ist $A^{(n)}$ eine *obere Dreiecksmatrix*, d.h.

$$A^{(n)} = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ & \ddots & \vdots \\ 0 & & a_{nn} \end{pmatrix}.$$

Das Gleichungssystem $A^{(n)}x = b^{(n)}$ kann dann durch *Rückwärtssubstitution* gelöst werden:

$$x_n = \frac{b_n^{(n)}}{A_{nn}^{(n)}},$$
$$x_i = \frac{1}{A_{ii}^{(n)}} \left[b_i^{(n)} - \sum_{j=i+1}^{n} A_{ij}^{(n)} x_j \right] \quad i = n-1, n-2, \ldots, 1.$$

Wie das Beispiel $A = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$ zeigt, ist der Gauß-Algorithmus in der soeben beschriebenen einfachen Form nicht für jede reguläre Matrix durchführbar. Selbst wenn er durchführbar ist, kann durch kleine Diagonalelemente $A_{ii}^{(i)}$ ein großer Rundungsfehler auftreten, der das Ergebnis total verfälscht. Um diese Schwierigkeiten zu umgehen, versucht man, in jedem Schritt ein betragsmäßig möglichst großes Element durch Zeilen- und/oder Spaltenvertauschungen in die entsprechende Diagonalposition zu bringen. Dieser Prozess heißt *Pivotisierung*, das entsprechende Element *Pivotelement*. Folgende *Pivot-Strategien* sind u.a. möglich:

- *Gesamtpivotisierung:* Suche im i-ten Schritt das betragsmäßig größte Element in der vollen Restmatrix $(A_{\mu\nu}^{(i)})_{i \le \mu,\nu \le n}$.
- *Spaltenpivotisierung:* Suche im i-ten Schritt das betragsmäßig größte Element in der ersten Spalte der Restmatrix $(A_{\mu i}^{(i)})_{i \le \mu \le n}$.

Offensichtlich gilt bei exakter Arithmetik:

A ist singulär.
$\iff$ Bei Gesamtpivotisierung ist für ein $1 \le i \le n$ das entsprechende Pivotelement gleich Null.
$\iff$ Bei Spaltenpivotisierung ist für ein $1 \le i \le n$ das entsprechende Pivotelement gleich Null.

Da die Gesamtpivotisierung aufwändiger ist und bei endlicher Arithmetik auch in der Regel keine besseren Ergebnisse liefert als die Spaltenpivotisierung, beschränkt man sich in der Praxis meistens auf letztere. Dies führt auf Algorithmus IV.1.1.

BEISPIEL IV.1.1. Betrachte das LGS $Ax = b$ mit

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 10 \end{pmatrix}, \quad b = \begin{pmatrix} 4 \\ 0 \\ 4 \end{pmatrix}.$$

Das Gaußsche Eliminationsverfahren liefert

$$\left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 4 & 5 & 6 & 0 \\ 7 & 8 & 10 & 4 \end{array}\right) \rightarrow \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -3 & -6 & -16 \\ 0 & -6 & -11 & -24 \end{array}\right) \rightarrow \left(\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -3 & -6 & -16 \\ 0 & 0 & 1 & 8 \end{array}\right)$$

Algorithmus IV.1.1 Gauß-Algorithmus mit Spaltenpivotisierung

Gegeben: Matrix A, rechte Seite b, Toleranz ε
Gesucht: x, Lösung des LGS $Ax = b$

for $i = 1, 2, \ldots, n-1$ **do** ▷ *Eliminationsteil*
  Finde ein $j_i \in \{i, \ldots, n\}$ mit $|A_{j_i i}| = \max\limits_{i \le k \le n} |A_{ki}|$.
  Falls $|A_{j_i i}| \le \varepsilon$ *stopp*, die Matrix ist singulär.
  Vertausche b_i und b_{j_i} und die Zeilen i und j_i von A.
  for $j = i+1, \ldots, n$ **do**
    for $k = i+1, \ldots, n$ **do**
      $A_{jk} \leftarrow A_{jk} - \dfrac{A_{ik} A_{ji}}{A_{ii}}$
    end for
    $b_j \leftarrow b_j - \dfrac{b_i A_{ji}}{A_{ii}}$
  end for
end for
Falls $|A_{nn}| \le \varepsilon$ *stopp*, die Matrix ist singulär. ▷ *Rücklösungsteil*
$x_n \leftarrow \dfrac{b_n}{A_{nn}}$
for $i = n-1, n-2, \ldots, 1$ **do**
  $x_i = \dfrac{1}{A_{ii}} \left[b_i - \sum\limits_{j=i+1}^{n} A_{ij} x_j \right]$
end for

und

$$\begin{aligned} x_3 &= 8 \\ x_2 &= -\frac{1}{3}\{-16 + 6 \cdot 8\} &&= -\frac{32}{3} \\ x_1 &= 4 - 2 \cdot (-\frac{32}{3}) - 3 \cdot 8 &&= \frac{4}{3}. \end{aligned}$$

Die eindeutige Lösung des LGS ist $(\frac{4}{3}, -\frac{32}{3}, 8)^t$.

Algorithmus IV.1.1 benötigt im i-ten Eliminationsschritt

$$\begin{aligned} (n-i)(n-i+1) &\quad \text{Additionen,} \\ (n-i)(n-i+1) &\quad \text{Multiplikationen,} \\ n+1-i &\quad \text{Divisionen} \end{aligned}$$

insgesamt also

$$\begin{aligned} \frac{1}{6}(n-1)n(2n-1) + \frac{1}{2}(n-1)n &\quad \text{Additionen,} \\ \frac{1}{6}(n-1)n(2n-1) + \frac{1}{2}(n-1)n &\quad \text{Multiplikationen,} \\ \frac{1}{2}(n-1)n &\quad \text{Divisionen,} \end{aligned}$$

wovon $\frac{1}{2}(n-1)n$ Additionen und Multiplikationen auf die Umformung der rechten Seite entfallen.
Algorithmus IV.1.1 benötigt im i-ten Rücklösungsschritt zur Berechnung von x_{n-i+1}

$$\begin{aligned} i-1 &\quad \text{Additionen},\\ i-1 &\quad \text{Multiplikationen},\\ 1 &\quad \text{Division} \end{aligned}$$

insgesamt also

$$\begin{aligned} \frac{1}{2}(n-1)n &\quad \text{Additionen},\\ \frac{1}{2}(n-1)n &\quad \text{Multiplikationen},\\ n &\quad \text{Divisionen}. \end{aligned}$$

IV.2. Dreieckszerlegungen

Die Überlegungen am Ende von Paragraph IV.1 zeigen, dass der Eliminationsteil des Gauß-Algorithmus wesentlich aufwändiger ist als der Rücklösungsteil und die Modifikation der rechten Seite. Muss man also mehrere Gleichungssysteme mit gleicher Matrix, aber verschiedenen rechten Seiten lösen, sollte man daher versuchen, den Rücklösungsteil und die Umformungen der rechten Seiten vom Eliminationsteil zu trennen und sich dazu die für die Umformungen der rechten Seite nötigen Größen zu merken.

Um diese Idee zu präzisieren, sei $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$. Bezeichne mit $A^{(i)}$ das Ergebnis nach dem i-ten Gaußschritt, $A^{(0)} = A$ und mit $\widetilde{P}^{(i)}$ die Matrix, die aus der Einheitsmatrix durch Vertauschen der i-ten und der j_i-ten Spalte entsteht. Dabei ist $j_i \in \{i, \dots, n\}$ die Größe aus Schritt (1a) von Algorithmus IV.1.1 (S. 79). Definiere weiter:

$$\begin{aligned} P^{(n-1)} &= \widetilde{P}^{(n-1)},\\ P^{(k)} &= P^{(k+1)}\widetilde{P}^{(k)}, && k = n-2, \dots, 1,\\ P &= P^{(1)},\\ \widetilde{L}^{(i)} &= \begin{pmatrix} 1 & & & & \\ & \ddots & & & 0 \\ & & 1 & & \\ & & -\frac{A^{(i-1)}_{i+1,i}}{A^{(i-1)}_{i,i}} & 1 & \\ 0 & & \vdots & & \ddots \\ & & -\frac{A^{(i-1)}_{n,i}}{A^{(i-1)}_{i,i}} & 0 & 1 \end{pmatrix}, && 1 \le i \le n-1,\\ L^{(n-1)} &= (\widetilde{L}^{(n-1)})^{-1},\\ L^{(k)} &= P^{(k+1)}(\widetilde{L}^{(k)})^{-1}P^{(k+1)}, && k = n-2, \dots, 1, \end{aligned}$$

$$L = L^{(1)} \dots L^{(n-1)},$$
$$R = A^{(n-1)}.$$

Dann ergibt sich aus Algorithmus IV.1.1 (S. 79)

$$\begin{aligned} R = A^{(n-1)} &= \widetilde{L}^{(n-1)} \widetilde{P}^{(n-1)} A^{(n-2)} \\ \Longrightarrow \qquad L^{(n-1)} R &= P^{(n-1)} A^{(n-2)} \\ &= P^{(n-1)} \widetilde{L}^{(n-2)} \widetilde{P}^{(n-2)} A^{(n-3)} \\ &= P^{(n-1)} \widetilde{L}^{(n-2)} \underbrace{P^{(n-1)} P^{(n-1)}}_{=I} \widetilde{P}^{(n-2)} A^{(n-3)} \\ &= (L^{(n-2)})^{-1} P^{(n-2)} A^{(n-3)} \\ \Longrightarrow L^{(n-2)} L^{(n-1)} R &= P^{(n-2)} A^{(n-3)} \end{aligned}$$

und durch Induktion

$$LR = PA. \tag{IV.2.1}$$

Weiter gilt:

- $\widetilde{L}^{(i)^{-1}}$ ergibt sich aus $\widetilde{L}^{(i)}$ durch Multiplikation der Elemente unterhalb der Diagonalen mit -1.
- $L^{(k)}$ ergibt sich aus $\widetilde{L}^{(k)^{-1}}$ durch Vertauschen der Elemente unterhalb der Diagonalen gemäß der durch $P^{(k+1)}$ beschriebenen Permutation.
- L ergibt sich durch Auffüllen des Unterdiagonalteils der Einheitsmatrix mit den entsprechenden Einträgen der $L^{(k)}, 1 \leq k \leq n-1$.

Wenn man diese Punkte berücksichtigt, ergibt sich aus dem Gauß-Algorithmus der Algorithmus IV.2.1 zur Berechnung der Matrizen L, R und P in (IV.2.1):

Bemerkung IV.2.1. Nach erfolgreicher Beendigung von Algorithmus IV.2.1 ist

$$\begin{aligned} L_{ij} &= A_{ij} \quad 1 \leq j < i \leq n, \\ R_{ij} &= A_{ij} \quad 1 \leq i \leq j \leq n, \\ P_{ij} &= \delta_{jp_i} \quad 1 \leq i, j \leq n. \end{aligned}$$

Falls (IV.2.1) gilt, kann das LGS $Ax = b$ wie folgt gelöst werden:

(1) berechne $z = Pb$,
(2) löse $Ly = z$,
(3) löse $Rx = y$.

Dies realisiert Algorithmus IV.2.2, wobei vorausgesetzt wird, dass L, R, P mit Algorithmus IV.2.1 berechnet wurden und wie dort beschrieben gespeichert sind.

Bemerkung IV.2.2. Bei der praktischen Durchführung von Algorithmus IV.2.2 kann man den Vektor z auf dem Speicherplatz des

Algorithmus IV.2.1 LR-Zerlegung mit Spaltenpivotisierung

Gegeben: Matrix A, Permutationsvektor p mit $p_i = i$ für $1 \leq i \leq n$, Toleranz ε

Gesucht: L, R, P, Zerlegung gemäß (IV.2.1) (A wird mit R und dem Teil von L unterhalb der Diagonalen überschrieben.)

for $i = 1, 2, \ldots, n-1$ **do**
  Finde $j_i \in \{i, \ldots, n\}$ mit $|A_{j_i i}| = \max\limits_{i \leq k \leq n} |A_{ki}|$.
  Falls $|A_{j_i i}| \leq \varepsilon$ *stopp*, die Matrix ist singulär.
  Vertausche p_i und p_{j_i} und die Zeilen i und j_i von A.
  for $j = i+1, \ldots, n$ **do**
    $A_{ji} \leftarrow A_{ji}/A_{ii}$
    for $k = i+1, \ldots, n$ **do**
      $A_{jk} \leftarrow A_{jk} - A_{ik}A_{ji}$
    end for
  end for
end for
Falls $|A_{nn}| \leq \varepsilon$ *stopp*, die Matrix ist singulär.

Algorithmus IV.2.2 Lösen bei bekannter LR-Zerlegung

Gegeben: Matrix A enthält L und R, Permutationsvektor p, rechte Seite b

Gesucht: x, Lösung von $Ax = b$ mit $PA = LR$

for $i = 1, \ldots, n$ **do**
  $z_i \leftarrow b_{p_i}$
end for
$y_1 \leftarrow z_1$
for $i = 2, \ldots, n$ **do**
  $$y_i \leftarrow z_i - \sum_{j=1}^{i-1} A_{ij} y_j$$
end for
$$x_n \leftarrow \frac{y_n}{A_{nn}}$$
for $i = n-1, n-2, \ldots, 1$ **do**
  $$x_i = \frac{1}{A_{ii}} \left[y_i - \sum_{j=i+1}^{n} A_{ij} x_j \right]$$
end for

Vektors x und den Vektor y auf dem Speicherplatz des Vektors b speichern.

Beispiel IV.2.3. Für die Matrix aus Beispiel IV.1.1 (S. 78) liefert Algorithmus IV.2.1 die LR-Zerlegung:

$$p = (1, 2, 3)$$

und

$$\begin{array}{ccccccccccc} 1 & 2 & 3 & & 1 & 2 & 3 & & 1 & 2 & 3 \\ 4 & 5 & 6 & \to & 4 & -3 & -6 & \to & 4 & -3 & -6 \\ 7 & 8 & 10 & & 7 & -6 & -11 & & 7 & 2 & 1 \end{array}$$

Wir erhalten $A = LR$ mit

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 7 & 2 & 1 \end{pmatrix}, \quad R = \begin{pmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & 0 & 1 \end{pmatrix}.$$

Für die rechte Seite $b = (4, 0, 4)^t$ ergibt sich $z = b$ und

$$y = \begin{pmatrix} 4 \\ -16 \\ 8 \end{pmatrix}, \quad x = \begin{pmatrix} \frac{4}{3} \\ -\frac{32}{3} \\ 8 \end{pmatrix}.$$

Ist A *symmetrisch positiv definit*, kurz *s.p.d.*, d.h. $A = A^t$ und $x^t Ax > 0$ für alle $x \in \mathbb{R}^n \setminus \{0\}$, kann man auf die Pivotisierung verzichten und die Berechnung der LR-Zerlegung vereinfachen. Dies beruht auf folgendem Ergebnis.

Satz IV.2.4 (Cholesky-Zerlegung). *Sei $A \in \mathbb{R}^{n\times n}$ s.p.d. Dann gibt es genau eine untere Dreiecksmatrix L mit Diagonalelementen 1 und genau eine Diagonalmatrix D mit positiven Diagonalelementen, so dass gilt $A = LDL^t$.*

Beweis. Wir beweisen die Behauptung durch Induktion über die Dimension n.
$\underline{n = 1:}$ Da 1×1 Matrizen reelle Zahlen sind, ist in diesem Fall die Behauptung offensichtlich richtig.
$\underline{n \to n+1:}$ Zerlege A, L, D gemäß

$$A = \begin{pmatrix} \widetilde{A} & a \\ a^t & \alpha \end{pmatrix}, \quad \widetilde{A} \in \mathbb{R}^{n\times n}, a \in \mathbb{R}^n, \alpha \in \mathbb{R},$$

$$L = \begin{pmatrix} \widetilde{L} & 0 \\ e^t & 1 \end{pmatrix}, \quad \widetilde{L} \in \mathbb{R}^{n\times n}, e \in \mathbb{R}^n,$$

$$D = \begin{pmatrix} \widetilde{D} & 0 \\ 0 & \delta \end{pmatrix}, \quad \widetilde{D} \in \mathbb{R}^{n\times n}, \delta \in \mathbb{R}_+^*.$$

Dabei ist $\widetilde{L}$ eine untere Dreiecksmatrix mit Diagonalelementen 1 und $\widetilde{D}$ eine Diagonalmatrix mit positiven Diagonalelementen.
Dann folgt

$$LDL^t = \begin{pmatrix} \widetilde{L}\widetilde{D}\widetilde{L}^t & \widetilde{L}\widetilde{D}e \\ e^t\widetilde{D}\widetilde{L}^t & e^t\widetilde{D}e + \delta \end{pmatrix}.$$

Gemäß Induktionsvoraussetzung sind $\widetilde{L}$ und $\widetilde{D}$ durch

$$\widetilde{L}\widetilde{D}\widetilde{L}^t = \widetilde{A}$$

eindeutig festgelegt. e ist dann die eindeutige Lösung von

$$\widetilde{L}\widetilde{D}e = a.$$

δ ergibt sich schließlich zu

$$\delta = \alpha - e^t \widetilde{D} e.$$

Wir müssen noch $\delta > 0$ zeigen. Definiere dazu

$$x = \begin{pmatrix} -\widetilde{A}^{-1}a \\ 1 \end{pmatrix} \in \mathbb{R}^{n+1} \setminus \{0\}.$$

Dann folgt aus der positiven Definitheit von A

$$\begin{aligned} 0 < x^t A x &= (-a^t \widetilde{A}^{-1}, 1) \begin{pmatrix} \widetilde{A} & a \\ a^t & \alpha \end{pmatrix} \begin{pmatrix} -\widetilde{A}^{-1}a \\ 1 \end{pmatrix} \\ &= \alpha - a^t \widetilde{A}^{-1} a \\ &= \alpha - a^t \widetilde{L}^{-T} \widetilde{D}^{-1} \widetilde{L}^{-1} a \\ &= \alpha - a^t \widetilde{L}^{-T} \widetilde{D}^{-1} \widetilde{D} \widetilde{D}^{-1} \widetilde{L}^{-1} a \\ &= \alpha - e^t \widetilde{D} e \\ &= \delta. \end{aligned}$$

□

Sei nun $1 \le i \le j \le n$. Dann folgt aus $A = LDL^t$ durch Vergleich der Elemente

$$\begin{aligned} A_{ij} &= \sum_{k,l=1}^{n} L_{ik} D_{kl} L_{lj}^t = \sum_{k=1}^{n} L_{ik} D_{kk} L_{jk} \\ &= \sum_{k=1}^{i-1} L_{ik} D_{kk} L_{jk} + D_{ii} L_{ji} \end{aligned}$$

und somit wegen $A_{ij} = A_{ji}$

$$\begin{aligned} D_{ii} &= A_{ii} - \sum_{k=1}^{i-1} L_{ik}^2 D_{kk}, \\ L_{ji} &= \frac{1}{D_{ii}} \left[A_{ji} - \sum_{k=1}^{i-1} L_{ik} D_{kk} L_{jk} \right], \quad \text{für } j > i. \end{aligned}$$

Dies führt auf Algorithmus IV.2.3 zur Berechnung von L und D. Dabei wird von der s.p.d. Matrix A nur der Teil unterhalb der Diagonalen gespeichert und durch D und den Unterdiagonalteil von L überschrieben.

Algorithmus IV.2.4 löst das LGS $Ax = b$ bei bekannter Cholesky-Zerlegung $A = LDL^t$.

Bemerkung IV.2.5. *(1)* Wie bei Algorithmus IV.2.2 kann z auf dem Speicherplatz für x und y auf demjenigen für b abgespeichert werden.
(2) Algorithmus IV.2.3 und IV.2.4 sind besonders einfach für s.p.d.

Algorithmus IV.2.3 Cholesky-Zerlegung

Gegeben: Matrix A (s.p.d., nur Teil unter der Diagonalen, Diagonale einschließlich)

Gesucht: L, D, Zerlegung $A = LDL^t$ mit $L_{ii} = 1$ (A wird durch D und den Teil von L unterhalb der Diagonalen überschrieben.)

1: **for** $j = 2, \ldots, n$ **do**
2: $\quad A_{j1} \leftarrow \dfrac{A_{j1}}{A_{11}}$
3: **end for**
4: **for** $i = 2, \ldots, n$ **do**
5: $\quad A_{ii} \leftarrow A_{ii} - \displaystyle\sum_{k=1}^{i-1} A_{ik}^2 A_{kk}$
6: $\quad$ **if** $i < n$ **then**
7: $\quad\quad$ **for** $j = i+1, \ldots, n$ **do**
8: $\quad\quad\quad A_{ji} \leftarrow \dfrac{1}{A_{ii}} \left[A_{ji} - \displaystyle\sum_{k=1}^{i-1} A_{ik} A_{kk} A_{jk} \right]$
9: $\quad\quad$ **end for**
10: $\quad$ **end if**
11: **end for**

Algorithmus IV.2.4 Lösung eines LGS bei bekannter Cholesky-Zerlegung

Gegeben: Matrix A (nur unterer Teil, enthält D und L), rechte Seite b

Gesucht: x, Lösung von $Ax = b$ mit $A = LDL^t$

1: $z_1 \leftarrow b_1$
2: **for** $i = 2, \ldots, n$ **do**
3: $\quad z_i \leftarrow b_i - \displaystyle\sum_{k=1}^{i-1} A_{ik} z_k$
4: **end for**
5: **for** $i = 1, \ldots, n$ **do**
6: $\quad y_i \leftarrow \dfrac{z_i}{A_{ii}}$
7: **end for**
8: $x_n \leftarrow y_n$
9: **for** $i = n-1, n-2, \ldots, 1$ **do**
10: $\quad x_i \leftarrow y_i - \displaystyle\sum_{k=i+1}^{n} A_{ki} x_k$
11: **end for**

Tridiagonalmatrizen, d.h. Matrizen mit $A_{ij} = 0$ falls $|i - j| > 1$. In diesem Fall treten in den Schritten 2 bzw. 1 und 3 nur die Terme mit

$k = i - 1$ oder $k = i + 1$ in Erscheinung. Daher ist in diesem Fall der Aufwand der Algorithmen $\mathrm{O}(n)$.

BEISPIEL IV.2.6. Wir betrachten die Matrix

$$A = \begin{pmatrix} 5 & -2 & 2 \\ -2 & 6 & -1 \\ 2 & -1 & 4 \end{pmatrix}.$$

Für $i = 1$ liefert Algorithmus IV.2.3

$$d_{11} = 5, \quad \ell_{21} = -\frac{2}{5}, \quad \ell_{31} = \frac{2}{5}.$$

Für $i = 2$ erhalten wir analog

$$d_{22} = 6 - \left(-\frac{2}{5}\right)^2 5 = \frac{26}{5}, \quad \ell_{32} = \frac{5}{26}\left[-1 - \left(-\frac{2}{5}\right)5\frac{2}{5}\right] = -\frac{1}{26}.$$

Für $i = 3$ ergibt sich schließlich

$$d_{33} = 4 - \left(\frac{2}{5}\right)^2 5 - \left(-\frac{1}{26}\right)^2 \frac{26}{5} = \frac{83}{26}.$$

Also ist

$$L = \begin{pmatrix} 1 & 0 & 0 \\ -\frac{2}{5} & 1 & 0 \\ \frac{2}{5} & -\frac{1}{26} & 1 \end{pmatrix}, \quad D = \begin{pmatrix} 5 & 0 & 0 \\ 0 & \frac{26}{5} & 0 \\ 0 & 0 & \frac{83}{26} \end{pmatrix}.$$

IV.3. Orthogonalisierungsverfahren

In diesem Paragraphen betrachten wir Verfahren, die eine gegebene Matrix A in der Form $A = Q^t R$ zerlegen mit einer oberen Dreiecksmatrix R und einer orthogonalen Matrix Q, d.h. $Q^t Q = QQ^t = I$. Das LGS $Ax = b$ kann dann in den zwei Schritten

(1) berechne $z = Qb$
(2) löse $Rx = z$

einfach gelöst werden. Im Vergleich zu den Dreieckszerlegungen aus Paragraph IV.2 haben diese Verfahren den Nachteil, etwa den doppelten bis dreifachen Aufwand zu benötigen. Dafür haben sie jedoch folgende Vorteile:

- Sie benötigen keine Pivotstrategien.
- Sie haben keine Schwierigkeiten mit singulären bzw. „fast singulären“ Matrizen.
- Sie sind auch für überbestimmte Gleichungssysteme anwendbar (siehe §IV.5).

Im Folgenden bezeichnet $(\cdot\,,\cdot)$ stets das euklidische Skalarprodukt auf $\mathbb{R}^n$ und $\|\cdot\|$ die euklidische Norm.

DEFINITION IV.3.1 (Householder-Transformation). Sei $u \in \mathbb{R}^n$ mit $\|u\| = 1$. Dann heißt $H(u) = I - 2uu^t$ die zu u gehörige *Householder-Transformation* oder *Householder-Matrix*.

Householder-Matrizen haben folgende offensichtlichen Eigenschaften.

BEMERKUNG IV.3.2 (Eigenschaften von Householder-Transformationen). Sei $u \in \mathbb{R}^n$ mit $\|u\| = 1$. Dann gilt:

(1) $H(u) = H(u)^t, H(u)H(u) = I$.
(2) Die Householder-Transformation zu u beschreibt geometrisch eine Spiegelung an der Hyperebene durch Null senkrecht zu u.
(3) $H(u)v = v - 2\,(u\,,\,v)\,u$ für alle $v \in \mathbb{R}^n$.

Der folgende Satz zeigt, dass jeder vom Nullvektor verschiedene Vektor durch eine geeignete Householder-Transformation auf ein Vielfaches des ersten Einheitsvektors abgebildet werden kann. Da Householder-Matrizen orthogonal und Householder-Transformationen damit längenerhaltend sind, ist der Faktor natürlich die Norm des Vektors.

SATZ IV.3.3 (Existenz von Householder-Transformationen). *Sei $v \in \mathbb{R}^n \setminus \{0\}$. Dann gibt es ein $\sigma \in \{-1, 1\}$ und ein $u \in \mathbb{R}^n$ mit $\|u\| = 1$ und $H(u)v = \sigma \|v\| e_1$. Es ist*

$$\sigma = -\operatorname{sgn}(v_1), \quad u = \frac{1}{c}(v - \sigma \|v\| e_1)$$

mit

$$c = \sqrt{2 \|v\| [\|v\| + |v_1|]}.$$

BEWEIS. Aus $H(u)v = \sigma \|v\| e_1$ und $\|u\| = 1$ folgt mit $c = 2\,(u\,,\,v)$

$$\sigma \|v\| e_1 = v - cu$$

und damit

$$c^2 = \|v - \sigma \|v\| e_1\|^2 = 2 \|v\|^2 - 2\sigma \|v\| v_1,$$
$$u = \frac{1}{c} [v - \sigma \|v\| e_1]\,.$$

Hieraus folgt die Behauptung. Die Wahl von σ gewährleistet dabei, dass bei der Berechnung von c keine Auslöschung auftritt. □

Satz IV.3.3 führt auf folgende Idee zur Berechnung von Q, R mit $A = Q^t R$:

- Setze $A^{(0)} = A$.
- Im i-ten Schritt, $1 \le i \le n-1$, sei $v^{(i)} \in \mathbb{R}^{n+1-i}$ die erste Spalte der Restmatrix $(A^{(i-1)}_{\mu\nu})_{i \le \mu,\nu \le n}$.
- Bestimme $u^{(i)} \in \mathbb{R}^{n-i+1}$ gemäß Satz IV.3.3 mit $v = v^{(i)}$.
- Setze

$$Q^{(i)} = \begin{pmatrix} I & 0 \\ 0 & H(u^{(i)}) \end{pmatrix}, \quad A^{(i)} = Q^{(i)} A^{(i-1)}.$$

Dann erhält man nach $n-1$ Schritten eine obere Dreiecksmatrix R. Es ist $Q = Q^{(n-1)} \cdot \ldots \cdot Q^{(1)}$.

Beachtet man, dass für $A^{(i)}$ nur das Produkt $H(u^{(i)})A^{(i-1)}$ berechnet werden muss, dass $H(u^{(i)})$ durch $u^{(i)}$ eindeutig bestimmt ist und dass $u^{(i)}$ auf den Speicherplätzen für $A_{\mu i}$, $i \le \mu \le n$, abgespeichert werden kann, erhält man Algorithmus IV.3.1.

Algorithmus IV.3.1 QR-Zerlegung

Gegeben: Matrix A, Toleranz ε

Gesucht: A, r, QR-Zerlegung von A, die untere Hälfte von A enthält die Vektoren der Householder-Matrizen, r enthält die Elemente R_{ii}

1: **for** $i = 1, \ldots, n-1$ **do**

2: $\quad \sigma \leftarrow -\operatorname{sgn}(A_{ii})$, $a \leftarrow \left\{ \sum_{j=i}^{n} A_{ji}^2 \right\}^{\frac{1}{2}}$, $c \leftarrow \{2a(a + |A_{ii}|)\}^{\frac{1}{2}}$

3: $\quad$ Falls $a < \varepsilon$ *stopp*, die Matrix ist singulär.

4: $\quad r_i \leftarrow \sigma a$, $A_{ii} \leftarrow \frac{1}{c}(A_{ii} - \sigma a)$

5: $\quad$ **for** $j = i+1, \ldots, n$ **do**

6: $\quad\quad A_{ji} \leftarrow \frac{A_{ji}}{c}$

7: $\quad$ **end for**

8: $\quad$ **for** $j = i+1, \ldots, n$ **do**

9: $\quad\quad s \leftarrow \sum_{k=i}^{n} A_{ki} A_{kj}$

10: $\quad\quad$ **for** $k = i, \ldots, n$ **do**

11: $\quad\quad\quad A_{kj} \leftarrow A_{kj} - 2sA_{ki}$

12: $\quad\quad$ **end for**

13: $\quad$ **end for**

14: **end for**

Algorithmus IV.3.2 löst das LGS $Ax = b$ bei bekannter QR-Zerlegung. Dabei wird die spezielle Gestalt $Q = Q^{(n-1)} \cdot \ldots \cdot Q^{(1)}$ ausgenützt.

Bemerkung IV.3.4. *(1)* Algorithmus IV.3.1 benötigt etwa doppelt so viele arithmetische Operationen wie Algorithmus IV.2.1 (S. 82), Algorithmus IV.3.2 etwa dreimal so viele wie Algorithmus IV.2.2 (S. 82). *(2)* Bei der Zerlegung $A = Q^t R$ kann man die orthogonale Matrix Q statt als Produkt von Spiegelungen auch als Produkt von Rotationen, so genannten *Givens-Rotationen*, darstellen [1, §3.9.1], [3, §4.9].

IV.4. Fehleranalyse

Wir suchen die Lösung x eines LGS $Ax = b$ und haben mit irgendeinem Verfahren, z.B. aus den Paragraphen IV.2 oder IV.3, eine Näherung $\widehat{x}$ für x berechnet. Wir fragen uns, wie groß ist der Fehler $x - \widehat{x}$? Da wir nur A, b, $\widehat{x}$ kennen, ist uns einzig das *Residuum* $r = A\widehat{x} - b$

Algorithmus IV.3.2 Lösen bei bekannter QR-Zerlegung

Gegeben: Matrix A, Vektor r (QR-Zerlegung gemäß Algorithmus IV.3.1), rechte Seite b
Gesucht: x, Lösung von $Ax = b$

1: **for** $i = 1, 2, \ldots, n-1$ **do**
2: $\quad s \leftarrow \sum_{k=i}^{n} A_{ki} b_k$
3: $\quad$ **for** $k = i, \ldots, n$ **do**
4: $\quad\quad b_k \leftarrow b_k - 2 s A_{ki}$
5: $\quad$ **end for**
6: **end for**
7: $x_n \leftarrow \dfrac{b_n}{A_{nn}}$
8: **for** $i = n-1, n-2, \ldots, 1$ **do**
9: $\quad x_i \leftarrow \dfrac{1}{r_i}\left(b_i - \sum_{k=i+1}^{n} A_{ik} x_k\right)$
10: **end for**

zugängig. Wir wollen daher den Fehler $x - \widehat{x}$ durch das Residuum r abschätzen. Dazu benötigen wir den Begriff der *Kondition*.

Im Folgenden bezeichnen $(X, \|\cdot\|_X)$ und $(Y, \|\cdot\|_Y)$ zwei normierte Vektorräume.

Definition IV.4.1 (Kondition). Sei $A : X \to Y$ eine lineare Abbildung. Dann heißt

$$\kappa(A) = \frac{\sup\limits_{x \in X, \|x\|_X = 1} \|Ax\|_Y}{\inf\limits_{x \in X, \|x\|_X = 1} \|Ax\|_Y}$$

die *Kondition* von A bzgl. $\|\cdot\|_X$ und $\|\cdot\|_Y$.

Die Kondition hat folgende Eigenschaften.

Bemerkung IV.4.2 (Eigenschaften der Kondition). *(1)* Die Kondition von A hängt von den verwendeten Normen ab.
(2) Es ist $\kappa(A) = \infty$, falls $A \notin \mathrm{Isom}(X, Y)$ ist.
(3) Es ist $\kappa(A) = \|A\|_{\mathcal{L}(X,Y)} \|A^{-1}\|_{\mathcal{L}(Y,X)}$, falls $A \in \mathrm{Isom}(X, Y)$ ist.
(4) Es ist $\kappa(A) = \frac{\lambda_{\max}(A^t A)^{\frac{1}{2}}}{\lambda_{\min}(A^t A)^{\frac{1}{2}}}$, falls $A \in \mathcal{GL}(\mathbb{R}^n)$ und $\|\cdot\|_X = \|\cdot\|_2$ ist. Dabei bezeichnet $\lambda_{\max}(B)$ bzw. $\lambda_{\min}(B)$ den maximalen bzw. minimalen Eigenwert der s.p.d. Matrix B.
(5) Es ist $\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$, falls $A \in \mathbb{R}^{n \times n}$ s.p.d. und $\|\cdot\|_X = \|\cdot\|_2$ ist.

Beweis. *ad (1):* Ist offensichtlich,
ad (2): Ist A nicht stetig, so ist der Zähler in Definition IV.4.1 gleich ∞. Ist $\dim(\ker A) > 0$ oder A^{-1} unstetig, so ist der Nenner in Definition

IV.4.1 gleich 0.
ad (3): Folgt aus der Definition der Operatornormen.
ad (4): Folgt aus (3).
ad (5): Folgt aus $\lambda_{\max / \min}(A^t A) = \lambda_{\max / \min}(A)^2$, falls A s.p.d. ist. □

Der folgende Satz zeigt, dass das Residuum ein Maß für den Fehler ist. Die Güte dieses Maßes hängt von der Kondition der Matrix A ab. Die Kondition misst somit die Empfindlichkeit des LGS gegen Störungen.

Satz IV.4.3 (Fehlerabschätzung für die Lösung eines LGS). *Seien $A \in \mathrm{Isom}(X, Y)$, $b \in Y$, $\widehat{x} \in X$ eine Näherung für $x = A^{-1}b$ und $r = A\widehat{x} - b$ das Residuum von $\widehat{x}$. Dann gilt*

$$\|A\|^{-1}_{\mathcal{L}(X,Y)} \|r\|_Y \leq \|x - \widehat{x}\|_X \leq \left\|A^{-1}\right\|_{\mathcal{L}(Y,X)} \|r\|_Y$$

und, falls $b \neq 0$ ist,

$$\kappa(A)^{-1} \frac{\|r\|_Y}{\|b\|_Y} \leq \frac{\|x - \widehat{x}\|_X}{\|x\|_X} \leq \kappa(A) \frac{\|r\|_Y}{\|b\|_Y}.$$

Beweis. Die Behauptung folgt aus Bemerkung IV.4.2 (3) und den Abschätzungen

$$\begin{aligned}
\|x - \widehat{x}\|_X &= \left\|A^{-1} r\right\|_X \leq \left\|A^{-1}\right\|_{\mathcal{L}(Y,X)} \|r\|_Y \,, \\
\|r\|_Y &= \|A(x - \widehat{x})\|_X \leq \|A\|_{\mathcal{L}(X,Y)} \|x - \widehat{x}\|_X \,, \\
\|b\|_Y &= \|Ax\|_Y \leq \|A\|_{\mathcal{L}(X,Y)} \|x\|_X \,, \\
\|x\|_X &= \left\|A^{-1} b\right\|_X \leq \left\|A^{-1}\right\|_{\mathcal{L}(Y,X)} \|b\|_Y \,.
\end{aligned}$$

□

Beispiel IV.4.4. Für das LGS aus Beispiel 2 (S. 7) mit der exakten Lösung $x = (1, -1)^t$ und der Näherungslösung $\widehat{x} = (0.278, 0)^t$ erhalten wir $r = (1.6 \cdot 10^{-4}, 1.86 \cdot 10^{-4})^t$ und für die euklidischen Normen $\|x\| = \sqrt{2}$, $\|b\| = 0.3340733$, $\|r\| = 2.45348 \cdot 10^{-4}$, $\|x - \widehat{x}\| = 1.2334034$ sowie

$$\frac{\|x - \widehat{x}\|}{\|x\|} = 0.8721479, \quad \frac{\|r\|}{\|b\|} = 7.34413 \cdot 10^{-4}.$$

Es ist

$$A^t A = \begin{pmatrix} 1.441969 & 1.040807 \\ 1.040807 & 0.75125 \end{pmatrix}$$

und

$$\lambda_{\min}(A^t A) = 5 \cdot 10^{-10}, \quad \lambda_{\max}(A^t A) = 2.1932189.$$

Damit ergibt sich bzgl. der euklidischen Norm die Kondition $\kappa(A) = 66230.187$.

Bemerkung IV.4.5. Die Abschätzungen von Satz IV.4.3 sind in dem Sinne scharf, dass für jede Wahl von $(X, \|\cdot\|_X)$, $(Y, \|\cdot\|_Y)$ und A Elemente x, $b = Ax$ und $\widehat{x}$ existieren, so dass eine der betreffenden Ungleichungen zu einer Gleichung wird.

IV.5. Lineare Ausgleichsprobleme

Seien $1 \leq m < n$, $b \in \mathbb{R}^n$ und $A \in \mathbb{R}^{n\times m}$. Dann hat das LGS $Ax = b$ i.a. keine Lösung. Stattdessen kann man versuchen, das Residuum $Ax - b$ in einer geeigneten Norm zu minimieren. Besonders einfach ist dies für die euklidische Norm auf dem $\mathbb{R}^n$. Wir betrachten daher im Folgenden das so genannte *lineare Ausgleichsproblem*

$$F(x) = \|Ax - b\|^2 = (Ax - b\,,\, Ax - b) \to \min \text{ in } \mathbb{R}^m. \tag{IV.5.1}$$

Beispiel IV.5.1 (Maximum-Likelihood-Schätzer). In der Statistik tritt oft das Problem auf, unbekannte Parameter $\vartheta_1, \ldots, \vartheta_m$ zu schätzen. Dazu werden $n > m$ Experimente mit den Ergebnissen

$$y_i = \varphi_i(\vartheta_1, \ldots, \vartheta_m) + z_i, \quad 1 \leq i \leq n$$

durchgeführt. Dabei sind die φ_i bekannte bzw. vermutete Funktionen der zu schätzenden Parameter und die z_i Messfehler. Oft trifft man die Modellannahme, dass diese Messfehler unabhängig und identisch normalverteilt sind mit Mittelwert 0. Dann ist der *Maximum-Likelihood-Schätzer* $(\widehat{\vartheta}_1, \ldots, \widehat{\vartheta}_m)$ der Wert, der die Funktion

$$(\vartheta_1, \ldots, \vartheta_m) \mapsto \sum_{i=1}^{n} \left(y_i - \varphi_i(\vartheta_1, \ldots, \vartheta_m)\right)^2$$

minimiert. Falls die Funktionen $\varphi_1, \ldots, \varphi_n$ linear sind, ist dies das lineare Ausgleichsproblem (IV.5.1) mit $b = (y_1, \ldots, y_n)^t$, $x = (\vartheta_1, \ldots, \vartheta_m)^t$ und $(Ax)_i = \varphi_i(\vartheta_1, \ldots, \vartheta_m)$.

Satz IV.5.2 (Eigenschaften des linearen Ausgleichsproblems). (1) *Problem* (IV.5.1) *besitzt mindestens eine Lösung.*
(2) *Für je zwei Lösungen* $x, y \in \mathbb{R}^m$ *von* (IV.5.1) *gilt* $x - y \in \ker(A)$.
(3) $x \in \mathbb{R}^m$ *löst* (IV.5.1) *genau dann, wenn* x *die so genannten* Normalengleichungen

$$A^tAx = A^tb \tag{IV.5.2}$$

löst.
(4) *Falls* $\operatorname{Rang}(A) = m$ *ist, besitzt* (IV.5.1) *und damit* (IV.5.2) *eine eindeutige Lösung.*

Beweis. *ad (1):* Sei $R(A) = \{y \in \mathbb{R}^n : \exists x \in \mathbb{R}^m : y = Ax\}$ das Bild von A. Dann ist $\mathbb{R}^n = R(A) \oplus R(A)^\perp$ und b besitzt eine eindeutige Zerlegung $b = u + v$ mit $u = Ax_0 \in R(A)$, $x_0 \in \mathbb{R}^m$ und $v \in R(A)^\perp$. Für jedes $x \in \mathbb{R}^m$ folgt dann wegen $Ax - Ax_0 \in R(A)$ aus dem Satz von Pythagoras

$$\|Ax - b\|^2 = \|Ax - Ax_0 - v\|^2 = \|Ax - Ax_0\|^2 + \|v\|^2 \geq \|v\|^2$$

mit Gleichheit genau dann, wenn $x \in x_0 + \ker(A)$ ist. Insbesondere ist x_0 eine Lösung von (IV.5.1).

ad (2): Folgt aus dem Beweis von (1).
ad (3): Offensichtlich ist

$$DF(x) = 2x^t A^t A - 2b^t A, \quad D^2F(x) = 2A^t A.$$

Da A^tA positiv semi-definit ist, folgt hieraus die Behauptung.
ad (4): Falls $\operatorname{Rang}(A) = m$ ist, ist A^tA s.p.d. und damit (IV.5.2) eindeutig lösbar. □

Wir betrachten zunächst den Fall $\operatorname{Rang}(A) = m$. Dann kann man Problem (IV.5.1) lösen, indem man eines der Verfahren der Paragraphen IV.1 – IV.3 auf die Normalengleichungen (IV.5.2) anwendet. Dies ist nicht empfehlenswert, da $\kappa(A^tA) \approx \kappa(A)^2$ ist. Viel effizienter ist folgende Vorgehensweise:
Wendet man Algorithmus IV.3.1 (S. 88) auf die Matrix A an, wobei man die Spaltenindizes bis m statt n laufen lässt und im Falle, dass in Schritt 3 $a < \varepsilon$ ist, direkt zu $i+1$ übergeht, so erhält man $n \times n$ orthogonale Matrizen $H^{(1)}, \ldots, H^{(m)}$ und eine invertierbare $m \times m$ obere Dreiecksmatrix R mit

$$H^{(m)} H^{(m-1)} \ldots H^{(1)} A = \begin{pmatrix} R \\ 0 \end{pmatrix}.$$

Setze $Q = H^{(m)} \cdot \ldots \cdot H^{(1)}$. Dann gilt für alle $x \in \mathbb{R}^m$

$$F(x) = \|Ax - b\|^2 = \|Q(Ax - b)\|^2 = \|Rx - u\|^2 + \|v\|^2,$$

wobei $Qb = \binom{u}{v}$ ist mit $u \in \mathbb{R}^m$, $v \in \mathbb{R}^{n-m}$. Offensichtlich ist $F(x)$ genau dann minimal, wenn $Rx = u$ ist.

Wir betrachten nun den Fall $\operatorname{Rang}(A) = k < m$. Dann können wir wie oben die Matrizen $H^{(1)}, \ldots, H^{(m)}$, Q und R bestimmen. Aber jetzt ist die Matrix R nicht mehr invertierbar, sondern hat den Rang k und die Struktur $R = \left(\begin{smallmatrix} S & T \\ 0 & 0 \end{smallmatrix}\right)$ mit einer invertierbaren $k \times k$ oberen Dreiecksmatrix S und einer $k \times (m-k)$ Matrix T. Setze nun $Qb = \binom{u}{v}$ mit $u \in \mathbb{R}^k$, $v \in \mathbb{R}^{n-k}$ und zerlege $x \in \mathbb{R}^m$ in der Form $x = \binom{x'}{x''}$ mit $x' \in \mathbb{R}^k$, $x'' \in \mathbb{R}^{m-k}$. Dann ist

$$F(x) = \|Ax - b\|^2 = \|Q(Ax - b)\|^2 = \|Sx' + Tx'' - u\|^2 + \|v\|^2$$

und $F(x)$ ist genau dann minimal, wenn $Sx' + Tx'' = u$ ist. Dieses LGS hat einen $(m-k)$-dimensionalen affinen Lösungsraum L, in dem es genau ein Element x mit minimaler Norm $\|x\|$ gibt.
Um also eine eindeutige Lösung zu erhalten, modifizieren wir Problem (IV.5.1) wie folgt:

(IV.5.3)
Finde ein Minimum von $\|Ax - b\|^2$
mit minimaler Norm $\|x\|^2$.

Offensichtlich sind im Fall $\operatorname{Rang}(A) = m$ die Probleme (IV.5.1) und (IV.5.3) äquivalent.

Zur Lösung von Problem (IV.5.3) gehen wir wie folgt vor:
Bezeichne mit $e^{(1)}, \ldots, e^{(m-k)}$ die Einheitsvektoren des $\mathbb{R}^{m-k}$ und setze

$$x^{(0)} = \begin{pmatrix} S^{-1}u \\ 0 \end{pmatrix},$$
$$x^{(1)} = \begin{pmatrix} -S^{-1}Te^{(1)} \\ e^{(1)} \end{pmatrix}, \ldots, x^{(m-k)} = \begin{pmatrix} -S^{-1}Te^{(m-k)} \\ e^{(m-k)} \end{pmatrix}.$$

Dann ist $L = x^{(0)} + \operatorname{span}\{x^{(1)}, \ldots, x^{(m-k)}\}$ und (IV.5.3) äquivalent zu

$$G(s) = \left\| x^{(0)} + \sum_{i=1}^{m-k} s_i x^{(i)} \right\|^2 \to \min \text{ in } \mathbb{R}^{m-k}. \tag{IV.5.4}$$

Offensichtlich hat Problem (IV.5.4) die gleiche Struktur wie (IV.5.1) mit b und A ersetzt durch $-x^{(0)}$ und $(x^{(1)}, \ldots, x^{(m-k)})$. Insbesondere hat die Matrix $(x^{(1)}, \ldots, x^{(m-k)})$ den maximalen Rang $m-k$, so dass wir (IV.5.4) wie oben beschrieben lösen können. Wegen der speziellen Struktur von (IV.5.4) ist es jetzt aber günstiger, die zugehörigen Normalengleichungen aufzustellen und mit dem Cholesky-Verfahren zu lösen.

Diese Überlegungen führen auf Algorithmus IV.5.1.

Bemerkung IV.5.3 (Pseudo-Inverse, Moore-Penrose-Inverse). Die Abbildung, die jedem Vektor $b \in \mathbb{R}^n$ die Lösung $x \in \mathbb{R}^m$ von (IV.5.3) zuordnet, ist linear und kann durch eine Matrix $A^+ \in \mathbb{R}^{m\times n}$ beschrieben werden, d.h. $x = A^+b$. Die Matrix A^+ heißt *Pseudo-Inverse* oder *Moore-Penrose-Inverse* von A [2, §3.3]. Für eine invertierbare quadratische Matrix A ist $A^+ = A^{-1}$. Für eine Rechtecksmatrix $A \in \mathbb{R}^{n\times m}$ mit $m < n$ und $\operatorname{Rang}(A) = m$ ist $A^+ = (A^tA)^{-1}A^t$. Die Spalten von A^+ können berechnet werden, indem man Algorithmus IV.5.1 sukzessive auf die Einheitsvektoren des $\mathbb{R}^n$ als rechter Seite b anwendet. Man beachte, dass dabei die orthogonalen Matrizen $H^{(1)}, \ldots, H^{(m)}$ und ggf. $x^{(1)}, \ldots, x^{(m-k)}$ nur einmal berechnet werden müssen. Man kann zeigen [2, Satz 3.16], dass A^+ durch die so genannten *Moore-Penrose-Axiome*

$$(A^+A)^t = A^+A, \quad (AA^+)^t = AA^+,$$
$$A^+AA^+ = A^+, \qquad AA^+A = A$$

eindeutig definiert ist. Die Pseudoinverse A^+ ist eng verknüpft mit der Singulärwertzerlegung von A, siehe Satz V.5.2 (S. 130).

IV.6. Stationäre Iterationsverfahren

Bei der numerischen Lösung partieller Differentialgleichungen treten große dünn besetzte Gleichungssysteme auf, d.h. Systeme mit sehr großen Matrizen, die nur sehr wenige von Null verschiedene Elemente haben. Für diese Systeme sind die bisher betrachteten direkten

Algorithmus IV.5.1 Lösen eines linearen Ausgleichsproblems mittels QR-Zerlegung

Gegeben: Matrix $A \in \mathbb{R}^{n\times m}$, Vektor $b \in \mathbb{R}^n$ mit $1 \le m \le n$
Gesucht: ein Minimum x von $\|Ax-b\|^2$ mit minimaler Norm $\|x\|^2$

1: Bestimme mittels einer entsprechend modifizierten Form von Algorithmus IV.3.1 (S. 88) m orthogonale Matrizen $H^{(1)}, \dots, H^{(m)} \in \mathbb{R}^{n\times n}$, so dass $H^{(m)} \cdot \ldots \cdot H^{(1)} A = \left(\begin{smallmatrix} R \\ 0 \end{smallmatrix}\right)$ ist mit einer oberen Dreiecksmatrix $R \in \mathbb{R}^{m\times m}$.
2: $k \leftarrow \operatorname{Rang}(R)$
3: Berechne mit einer entsprechend modifizierten Form von Algorithmus IV.3.1 $z = H^{(m)} \cdot \ldots \cdot H^{(1)} b$ und setze $u = (z_1, \dots, z_k)^t$.
4: Falls $k < m$ ist, gehe zu Schritt 5 andernfalls löse das LGS $Rx = u$; *stopp*.
5: Schreibe R in der Form $R = \left(\begin{smallmatrix} S & T \\ 0 & 0 \end{smallmatrix}\right)$ mit einer oberen Dreiecksmatrix $S \in \mathbb{R}^{k\times k}$ und $T \in \mathbb{R}^{k\times(m-k)}$, löse mit den Einheitsvektoren $e^{(1)}$, …, $e^{(m-k)}$ des $\mathbb{R}^{m-k}$ die LGS

$$Sw^{(0)} = u,\ Sw^{(1)} = Te^{(1)},\ \dots,\ Sw^{(m-k)} = Te^{(m-k)}.$$

6: $x^{(0)} \leftarrow \begin{pmatrix} w^{(0)} \\ 0 \end{pmatrix}$, $x^{(1)} \leftarrow \begin{pmatrix} -w^{(1)} \\ e^{(1)} \end{pmatrix}$, …, $x^{(m-k)} \leftarrow \begin{pmatrix} -w^{(m-k)} \\ e^{(m-k)} \end{pmatrix}$
7: Bestimme den Vektor $d \in \mathbb{R}^{m-k}$ und die symmetrisch positiv definite Matrix $C \in \mathbb{R}^{(m-k)\times(m-k)}$ mit den Einträgen

$$d_i = \left(w^{(0)}\,,\, w^{(i)}\right), \quad C_{ij} = \delta_{ij} + \left(w^{(i)}\,,\, w^{(j)}\right).$$

8: Löse das LGS $Cs = d$ mit dem Cholesky-Verfahren, Algorithmen IV.2.3 und IV.2.4 (S. 85).
9: $x \leftarrow x^{(0)} + \sum_{i=1}^{m-k} s_i x^{(i)}$

Lösungsverfahren zu aufwändig. Zudem liefern sie zwar bis auf Rundungsfehler die exakte Lösung des LGS, aber diese hat einen relativ großen Fehler im Vergleich zu der tatsächlich gesuchten Lösung der partiellen Differentialgleichung. Die Verfahren dieses und vor allem des nächsten Paragraphen bestimmen mit wesentlich geringerem Aufwand eine Näherungslösung des LGS, die im Vergleich zur gesuchten Lösung der partiellen Differentialgleichung ebenso genau ist wie die exakte Lösung des LGS. Die Beispiele IV.6.1 und IV.6.5 (S. 98) verdeutlichen diese Problematik an Hand eines sehr einfachen Modellproblems.

Beispiel IV.6.1. Sei $\Omega = (0,1)^d$, $d \in \{2,3\}$, der d-dimensionale Einheitswürfel. Zu gegebenem $f \in C(\Omega, \mathbb{R})$ wollen wir die Poissongleichung

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega, \\ u &= 0 \quad \text{auf } \partial\Omega \end{aligned} \tag{IV.6.1}$$

numerisch lösen. Wähle dazu ein $m \geq 2$, setze $h = \frac{1}{m}$ und betrachte das Gitter

$$\Omega_h = \left\{ \underline{i}h : \underline{i} \in (\mathbb{N}^*_{m-1})^d \right\}, \quad \overline{\Omega}_h = \left\{ \underline{i}h : \underline{i} \in (\mathbb{N}_m)^d \right\}.$$

Für $x_h \in \Omega_h$ bezeichne mit

$$N(x_h) = \{ y_h \in \overline{\Omega}_h : \|x_h - y_h\|_1 = h \}$$

die Nachbarn von x_h, wobei $\|\cdot\|_1$ die ℓ_1-Norm in $\mathbb{R}^d$ ist. Dann approximieren wir (IV.6.1) durch das LGS

$$2du_h(x_h) - \sum_{y_h \in N(x_h)} u_h(x_h) = h^2 f(x_h) \quad \forall x_h \in \Omega_h, \tag{IV.6.2}$$

wobei u_h für Gitterpunkte auf $\partial\Omega$ gleich Null gesetzt wird. Falls für die Lösung u von (IV.6.1) gilt $u \in C^4(\Omega, \mathbb{R})$, kann man zeigen, dass

$$\max_{x_h \in \Omega_h} |u(x_h) - u_h(x_h)| = \mathrm{O}(h^2)$$

ist. Das diskrete Problem (IV.6.2) ist ein LGS der Größe $n = (m-1)^d$. Die Matrix A dieses LGS ist s.p.d. und enthält pro Zeile höchstens $2d+1$ von Null verschiedene Elemente, d.h. sie ist dünn besetzt. Bezeichne mit

- E die Zahl der von Null verschiedenen Elemente von A,
- $B = \max\{|i-j| : A_{ij} \neq 0\}$ die Bandbreite von A,
- Z den Aufwand für eine Cholesky-Zerlegung von A und
- S den für die Cholesky-Zerlegung benötigten Speicherplatz.

Tabelle IV.6.1 gibt diese Zahlen in Abhängigkeit von d und h wieder. Sie zeigt, dass das Cholesky-Verfahren wegen seines Speicherbedarfes und Rechenaufwandes für die Lösung von (IV.6.2) ungeeignet ist.

TABELLE IV.6.1. Charakteristische Daten des LGS (IV.6.2)

d	h	n	E	B	S	Z
2	$\frac{1}{16}$	225	$1.1 \cdot 10^3$	15	$3.3 \cdot 10^3$	$7.6 \cdot 10^5$
	$\frac{1}{32}$	961	$4.8 \cdot 10^3$	31	$2.9 \cdot 10^4$	$2.8 \cdot 10^7$
	$\frac{1}{64}$	$3.9 \cdot 10^3$	$2.0 \cdot 10^4$	63	$2.5 \cdot 10^5$	$9.9 \cdot 10^8$
	$\frac{1}{128}$	$1.6 \cdot 10^4$	$8.0 \cdot 10^4$	127	$2.0 \cdot 10^6$	$3.3 \cdot 10^{10}$
3	$\frac{1}{16}$	3.310^3	$2.4 \cdot 10^4$	225	$7.6 \cdot 10^5$	$1.7 \cdot 10^8$
	$\frac{1}{32}$	3.010^4	$2.1 \cdot 10^5$	961	$2.8 \cdot 10^7$	$2.8 \cdot 10^{10}$
	$\frac{1}{64}$	$2.5 \cdot 10^5$	$1.8 \cdot 10^6$	3.910^3	$9.9 \cdot 10^8$	$3.9 \cdot 10^{12}$
	$\frac{1}{128}$	$2.0 \cdot 10^6$	$1.4 \cdot 10^7$	$1.6 \cdot 10^4$	$3.3 \cdot 10^{10}$	$5.3 \cdot 10^{14}$

Ausgangspunkt unserer Überlegungen ist ein LGS $Ax = b$ im $\mathbb{R}^n$ mit invertierbarer Matrix A. Anders als in den Paragraphen IV.1 – IV.3 soll es iterativ gelöst werden, indem wir es in eine äquivalente Fixpunktgleichung umformen und auf diese die Fixpunktiteration aus

§III.1 anwenden. Dazu betrachten wir eine beliebige invertierbare $n \times n$ Matrix M und formen das LGS wie folgt um:

$$\begin{aligned} & Ax = b \\ \iff \quad & M^{-1}Ax = M^{-1}b \\ \iff \quad & 0 = -M^{-1}Ax + M^{-1}b \\ \iff \quad & x = x - M^{-1}Ax + M^{-1}b. \end{aligned}$$

Also ist das LGS $Ax = b$ äquivalent zu der Fixpunktgleichung

$$x = F(x) = (I - M^{-1}A)x + M^{-1}b. \tag{IV.6.3}$$

Die entsprechende Fixpunktiteration lautet bei gegebenem Startwert $x_0 \in \mathbb{R}^n$

$$x_{i+1} = (I - M^{-1}A)x_i + M^{-1}b \quad i = 0, 1, \ldots \tag{IV.6.4}$$

Wegen des Banachschen Fixpunktsatzes, Satz III.1.3 (S. 62), konvergiert sie sicher dann gegen die eindeutige Lösung des LGS, wenn für eine Matrixnorm $\|\cdot\|_{\mathcal{L}}$ gilt

$$\left\|I - M^{-1}A\right\|_{\mathcal{L}} < 1. \tag{IV.6.5}$$

Da auf dem $\mathbb{R}^n$ alle Normen äquivalent sind, konvergiert die Fixpunktiteration dann in jeder Norm.
Wie das Beispiel

$$B = \begin{pmatrix} \frac{1}{2} & 1 \\ 0 & \frac{1}{2} \end{pmatrix}$$

zeigt, ist es bei gegebener Matrix B nicht einfach zu entscheiden, ob es eine Matrixnorm $\|\cdot\|_{\mathcal{L}}$ gibt mit $\|B\|_{\mathcal{L}} < 1$.
Damit stellen sich uns folgende Aufgaben:

- Bestimme invertierbare Matrizen M, so dass die Fixpunktiteration (IV.6.4) einfach durchzuführen ist und gegen die Lösung von $Ax = b$ konvergiert.
- Gebe ein möglichst einfaches und scharfes Kriterium an, das erlaubt, bei gegebenen Matrizen A und M zu entscheiden, ob (IV.6.5) für eine Matrixnorm $\|\cdot\|_{\mathcal{L}}$ erfüllt ist.

Im Folgenden bezeichnet $(\cdot\,,\cdot)$ stets das euklidische Skalarprodukt auf $\mathbb{C}^n$ und $\|\cdot\|_2$ die zugehörige Norm. Eine Matrix $A \in \mathbb{R}^{n\times n}$ zerlegen wir stets in der Form

$$A = D - L - R$$

mit

$$\begin{aligned} D_{ij} &= 0 \quad \text{falls } i \neq j, \\ L_{ij} &= 0 \quad \text{falls } j \geq i, \\ R_{ij} &= 0 \quad \text{falls } j \leq i. \end{aligned}$$

DEFINITION IV.6.2 (Stationäres Iterationsverfahren). Seien $A, M \in \mathcal{GL}(\mathbb{R}^n)$, $N = M - A$, und $b \in \mathbb{R}^n$. Eine Iterationsvorschrift der Form

$$\begin{aligned} &x_0 \text{ gegeben} \\ &x_{i+1} = M^{-1}Nx_i + M^{-1}b \quad i = 0, 1, \dots \end{aligned} \tag{IV.6.6}$$

nennen wir ein *stationäres Iterationsverfahren* zur Lösung des LGS $Ax = b$. Die Matrix

$$S = M^{-1}N = I - M^{-1}A$$

heißt *Iterationsmatrix* des Verfahrens.

BEMERKUNG IV.6.3. *(1)* Wegen $M^{-1} = (I - S)A^{-1}$ ist ein stationäres Iterationsverfahren eindeutig bestimmt durch die Angabe der Matrizen M oder S.
(2) In jedem Schritt eines stationären Iterationsverfahrens muss ein LGS der Form $Mz = c$ gelöst werden. Dieses LGS sollte leicht lösbar sein. Umgekehrt sollte M eine möglichst gute Approximation an A sein.
(3) Ist x^* die exakte Lösung von $Ax = b$ und $e_i = x^* - x_i$ der Fehler nach der i-ten Iteration von (IV.6.6), dann gilt für alle $i \in \mathbb{N}^*$

$$e_i = Se_{i-1}, \quad e_i = S^i e_0.$$

Das folgende Beispiel gibt die wichtigsten stationären Iterationsverfahren an.

BEISPIEL IV.6.4 (Funktionen `richardson`, `jacobi`, `gauss_seidel` in `Numerics`). *(1) Richardson-Relaxation*

$$S = I - \omega^{-1}A, \quad M = \omega I$$

mit $\omega \in \mathbb{R}_+^*$.
(2) Jacobi-Verfahren

$$S = S_J = I - D^{-1}A, \quad M = D$$

bzw. komponentenweise

$$x_{i+1,k} = \frac{1}{A_{kk}} \left\{ b_k - \sum_{l=1}^{k-1} A_{kl}x_{i,l} - \sum_{l=k+1}^{n} A_{kl}x_{i,l} \right\} \quad 1 \le k \le n, i \in \mathbb{N}.$$

(3) Jacobi-Relaxation

$$S = S_{J,\omega} = (1 - \omega)I + \omega S_J = I - \omega D^{-1}A, \quad M = \frac{1}{\omega}D$$

mit $\omega \in \mathbb{R}_+^*$.
(4) Gauß-Seidel-Verfahren

$$S = S_G = (D - L)^{-1}R, \quad M = D - L$$

bzw. komponentenweise

$$x_{i+1,k} = \frac{1}{A_{kk}} \left\{ b_k - \sum_{l=1}^{k-1} A_{kl}x_{i+1,l} - \sum_{l=k+1}^{n} A_{kl}x_{i,l} \right\} \quad 1 \le k \le n, i \in \mathbb{N}.$$

(5) Gauß-Seidel-Relaxation

$$S = S_{G,\omega} = I - \omega(D - \omega L)^{-1}A, \quad M = \frac{1}{\omega}(D - \omega L)$$

mit $\omega \in \mathbb{R}_+^*$ bzw. komponentenweise

$$x_{i+1,k} = (1-\omega)x_{i,k} + \omega\frac{1}{A_{kk}}\left\{b_k - \sum_{l=1}^{k-1} A_{kl}x_{i+1,l} - \sum_{l=k+1}^{n} A_{kl}x_{i,l}\right\}$$
$$1 \le k \le n, i \in \mathbb{N}.$$

Das folgende Beispiel vergleicht einige dieser Verfahren für das Modellproblem aus Beispiel IV.6.1. Es zeigt, dass diese Verfahren noch nicht effizient genug sind und dass die Verfahren des nächsten Paragraphen wesentlich bessere Ergebnisse liefern.

Beispiel IV.6.5. Wir greifen Beispiel IV.6.1 und das LGS (IV.6.2) auf. Tabelle IV.6.2 zeigt die Zahl der Iterationen, die das Gauß-Seidel-Verfahren (GS), das CG-Verfahren IV.7.2 (S. 110) und das PCG-Verfahren IV.7.3 (S. 113) mit SSOR-Vorkonditionierung IV.7.4 (S. 114) benötigen, um einen Anfangsfehler um den Faktor 10 zu reduzieren. Pro Iteration und Gitterpunkt benötigen das Gauß-Seidel-Verfahren, das CG-Verfahren und das PCG-Verfahren etwa $2d+1$, $2d+6$ bzw. $5d+8$ Operationen. Tabelle IV.6.3 stellt die Zahl der Operationen zusammen, die benötigt werden, um den Fehler um den Faktor 10 zu reduzieren. Beim Vergleich mit der Cholesky-Zerlegung, die ebenfalls angegeben ist, ist zu beachten, dass man in der Praxis (IV.6.2) nur mit einer Genauigkeit, die dem Diskretisierungsfehler von $O(h^2)$ entspricht, gelöst werden muss und dass bei geeigneter Vorgehensweise der Aufwand hierzu i.a. dem $2-3$-fachen der in der Tabelle angegebenen Zahlen entspricht. Beim Vergleich der Iterationszahlen ist zu beachten, dass sich die Konvergenzrate, d.h. die Reduktion des Fehlers pro Iteration, bei dem Gauß-Seidel-Verfahren, dem CG-Verfahren und dem PCG-Verfahren wie $\frac{\kappa(A)-1}{\kappa(A)+1} = 1 - O(h^2)$, $\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1} = 1 - O(h)$ bzw. $\frac{\sqrt{\kappa(C^{-1}A)}-1}{\sqrt{\kappa(C^{-1}A)}+1} = 1 - O(\sqrt{h})$ verhält. Dabei ist $\kappa(A) \approx h^{-2}$ die Kondition der Matrix von (IV.6.2), C die Vorkonditionierungsmatrix von Algorithmus IV.7.4 (S. 114) und $\kappa(C^{-1}A) \approx h^{-1}$. Daher wächst bei einer Halbierung der Gitterweite h die Zahl der benötigten Iterationen bei dem Gauß-Seidel-Verfahren, dem CG-Verfahren und dem PCG-Verfahren um den Faktor 4, 2 bzw. $\sqrt{2}$.

Als nächstes wollen wir ein relativ einfach nachprüfbares, notwendiges und hinreichendes Kriterium für die Konvergenz eines stationären Iterationsverfahrens herleiten. Dazu benötigen wir den Begriff des Spektralradius und einige seiner Eigenschaften.

TABELLE IV.6.2. Iterationen von Gauß-Seidel-, CG- und PCG-Verfahren

h	GS	CG	PCG
$\frac{1}{16}$	236	12	4
$\frac{1}{32}$	954	23	5
$\frac{1}{64}$	3820	47	7
$\frac{1}{128}$	15237	94	11

TABELLE IV.6.3. Rechenaufwand für Cholesky-, Gauß-Seidel-, CG- und PCG-Verfahren

d	h	Cholesky	GS	CG	PCG
2	$\frac{1}{16}$	$7.6 \cdot 10^5$	$2.7 \cdot 10^5$	$2.7 \cdot 10^4$	$1.6 \cdot 10^4$
	$\frac{1}{32}$	$2.8 \cdot 10^7$	$4.6 \cdot 10^6$	$2.2 \cdot 10^5$	$8.6 \cdot 10^4$
	$\frac{1}{64}$	$9.9 \cdot 10^8$	$7.6 \cdot 10^7$	$1.9 \cdot 10^6$	$5.0 \cdot 10^5$
	$\frac{1}{128}$	$3.3 \cdot 10^{10}$	$1.2 \cdot 10^9$	$1.5 \cdot 10^7$	$3.2 \cdot 10^6$
3	$\frac{1}{16}$	$1.7 \cdot 10^8$	$5.6 \cdot 10^6$	$4.9 \cdot 10^5$	$3.1 \cdot 10^5$
	$\frac{1}{32}$	$2.8 \cdot 10^{10}$	$2.0 \cdot 10^8$	$8.2 \cdot 10^6$	$3.4 \cdot 10^6$
	$\frac{1}{64}$	$3.9 \cdot 10^{12}$	$6.7 \cdot 10^9$	$1.4 \cdot 10^8$	$4.0 \cdot 10^7$
	$\frac{1}{128}$	$5.3 \cdot 10^{14}$	$2.2 \cdot 10^{11}$	$2.3 \cdot 10^9$	$5.2 \cdot 10^8$

DEFINITION IV.6.6 (Spektralradius). Sei $A \in \mathbb{C}^{n\times n}$. Dann heißt

$$\rho(A) = \max\{|\lambda| : \lambda \in \mathbb{C} \text{ ist ein Eigenwert von } A\}$$

der *Spektralradius* von A.

BEMERKUNG IV.6.7 (Eigenschaften des Spektralradius). *(1)* Der Spektralradius ist keine Norm, da z.B.

$$\rho\left(\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}\right) = 0$$

ist.
(2) Ist A symmetrisch, so ist $\rho(A) = \|A\|_{\mathcal{L}}$, wenn $\mathbb{R}^n$ mit der euklidischen Norm versehen wird.

SATZ IV.6.8 (Spektralradius und Operatornormen). (1) *Für jede Matrix $A \in \mathbb{C}^{n\times n}$ und jede Norm auf $\mathbb{C}^n$ gilt $\rho(A) \le \|A\|_{\mathcal{L}}$.*
(2) *Für jede Matrix $A \in \mathbb{C}^{n\times n}$ und jedes $\varepsilon \in \mathbb{R}_+^*$ gibt es eine Norm $\|\cdot\|_{A,\varepsilon}$ auf $\mathbb{C}^n$, so dass für die zugehörige Operatornorm $\|\cdot\|_{A,\varepsilon}$ gilt*

$$\|A\|_{A,\varepsilon} \le \rho(A) + \varepsilon.$$

(3) *Für jede Matrix $A \in \mathbb{C}^{n\times n}$ und jede Norm auf $\mathbb{C}^n$ gilt*

$$\rho(A) = \lim_{k\to\infty} \left\|A^k\right\|_{\mathcal{L}}^{\frac{1}{k}}.$$

Beweis. *ad (1):* Seien $A \in \mathbb{C}^{n\times n}$, $\|\cdot\|$ eine Norm auf $\mathbb{C}^n$, λ ein Eigenwert von A mit $|\lambda| = \rho(A)$ und x ein Eigenvektor von A zu λ. Dann gilt

$$\rho(A) = |\lambda| = \frac{\|Ax\|}{\|x\|} \leq \|A\|_{\mathcal{L}}.$$

ad (2): Sei $A \in \mathbb{C}^{n\times n}$, dann gibt es ein $U \in \mathcal{GL}(\mathbb{C}^n)$ mit

$$U^{-1}AU = \begin{pmatrix} \lambda_1 & \alpha_{12} & \alpha_{1n} \\ & \ddots & \vdots \\ & & \lambda_n \end{pmatrix}.$$

Zu $\varepsilon \in \mathbb{R}_+^*$ gibt es ein $\delta \in (0,1]$ mit

$$\max_{1\leq i\leq n-1} \sum_{j=i+1}^{n} \delta^{j-i} |\alpha_{ij}| \leq \delta \max_{1\leq i\leq n-1} \sum_{j=i+1}^{n} |\alpha_{ij}| < \varepsilon.$$

Setze

$$D_\delta = \operatorname{diag}(1, \delta, \delta^2, \ldots, \delta^{n-1}).$$

Dann folgt

$$(UD_\delta)^{-1}A(UD_\delta) = \begin{pmatrix} \lambda_1 & \delta\alpha_{12} & \delta^2\alpha_{13} & \ldots & \delta^{n-1}\alpha_{1n} \\ & \lambda_2 & \delta\alpha_{23} & \ldots & \delta^{n-2}\alpha_{2n} \\ & & \ddots & & \vdots \\ & & & & \lambda_n \end{pmatrix}$$

und

$$\begin{aligned} \left\|(UD_\delta)^{-1}A(UD_\delta)\right\|_\infty &= \max_{1\leq i\leq n} \left\{ |\lambda_i| + \sum_{j=i+1}^{n} \delta^{j-i} |\alpha_{ij}| \right\} \\ &\leq \rho(A) + \varepsilon. \end{aligned}$$

Definiere

$$\|x\|_{A,\varepsilon} = \left\|(UD_\delta)^{-1}x\right\|_\infty,$$

dann folgt

$$\|A\|_{A,\varepsilon} = \left\|(UD_\delta)^{-1}A(UD_\delta)\right\|_\infty \leq \rho(A) + \varepsilon.$$

ad (3): Aus (1) folgt für alle $k \in \mathbb{N}^*$

$$\rho(A)^k = \rho(A^k) \leq \left\|A^k\right\|_{\mathcal{L}}$$

und damit

$$\rho(A) \leq \liminf_{k\to\infty} \left\|A^k\right\|_{\mathcal{L}}^{\frac{1}{k}}.$$

Sei $\varepsilon \in \mathbb{R}_+^*$ beliebig und $\|\cdot\|_{A,\varepsilon}$ wie in Teil (2). Dann gibt es ein $C_\varepsilon \in \mathbb{R}_+^*$ mit

$$\|B\|_{\mathcal{L}} \leq C_\varepsilon \|B\|_{A,\varepsilon}$$

für alle $B \in \mathbb{C}^{n\times n}$. Damit folgt

$$\left\|A^k\right\|_{\mathcal{L}}^{\frac{1}{k}} \leq C_\varepsilon^{\frac{1}{k}} \left\|A^k\right\|_{A,\varepsilon}^{\frac{1}{k}} \leq C_\varepsilon^{\frac{1}{k}} \|A\|_{A,\varepsilon} \leq C_\varepsilon^{\frac{1}{k}} \{\rho(A) + \varepsilon\}$$

und daher

$$\limsup_{k\to\infty} \left\|A^k\right\|_{\mathcal{L}}^{\frac{1}{k}} \leq \rho(A) + \varepsilon.$$

Da ε beliebig war, folgt hieraus die Behauptung. □

Damit können wir nun das angekündigte Konvergenzkriterium angeben.

Satz IV.6.9 (Konvergenzkriterium für stationäre Iterationsverfahren). *Das stationäre Iterationsverfahren* (IV.6.6) *konvergiert genau dann für jede rechte Seite b und jeden Startwert x_0 gegen die Lösung des LGS $Ax = b$, wenn für die Iterationsmatrix S gilt $\rho(S) < 1$.*

Beweis. „$\Longrightarrow$“: Sei λ ein Eigenwert von S mit $|\lambda| = \rho(S)$ und e_0 ein zugehöriger Eigenvektor. Dann folgt aus Bemerkung IV.6.3 (3)

$$|\lambda|^i \|e_0\| = \left\|S^i e_0\right\| \xrightarrow[i\to\infty]{} 0.$$

Also ist $|\lambda| = \rho(S) < 1$.
„$\Longleftarrow$“: Folgt aus unseren Überlegungen zu Beginn dieses Paragraphen und Satz IV.6.8 (2) mit $\varepsilon = \frac{1}{2}(1 - \rho(S))$. □

Der Spektralradius der Iterationsmatrix erlaubt auch einen Effizienzvergleich verschiedener Verfahren.

Bemerkung IV.6.10. Sei $\|\cdot\|$ eine beliebige Norm auf $\mathbb{R}^n$. Wegen Bemerkung IV.6.3 (3) wird im Laufe von k-Iterationen von (IV.6.6) der Fehler durchschnittlich pro Iteration um den Faktor

$$R_k(e_0) = \left[\frac{\|e_k\|}{\|e_0\|}\right]^{\frac{1}{k}} = \left[\frac{\left\|S^k e_0\right\|}{\|e_0\|}\right]^{\frac{1}{k}}$$

reduziert. Die schlechtest mögliche Rate ist dann

$$\sup_{e_0\neq 0} R_k(e_0) = \left\|S^k\right\|_{\mathcal{L}}^{\frac{1}{k}}.$$

Daher wird wegen Satz IV.6.8 (3) der Fehler asymptotisch pro Iteration um den Faktor $\rho(S)$ reduziert. Diese Zahl ist unabhängig von der gewählten Norm. Sind S_1, S_2 die Iterationsmatrizen von zwei stationären Iterationsverfahren, wird man daher bei vergleichbarem Aufwand das Verfahren vorziehen, dessen Iterationsmatrix den kleineren Spektralradius hat.

Als nächstes geben wir einige Konvergenzkriterien für die Verfahren aus Beispiel IV.6.4 an.

SATZ IV.6.11 (Konvergenz der Richardson-Relaxation). *Sei $A \in \mathbb{R}^{n\times n}$ s.p.d. Dann konvergiert die Richardson-Relaxation für jeden Wert $\omega > \frac{1}{2}\rho(A)$.*

BEWEIS. Seien $0 < \lambda_1 \leq \ldots \leq \lambda_n$ die Eigenwerte von A. Dann sind $1 > 1 - \omega^{-1}\lambda_1 \geq \ldots \geq 1 - \omega^{-1}\lambda_n$ die Eigenwerte von $I - \omega^{-1}A$. Die Bedingung $\omega > \frac{1}{2}\rho(A)$ garantiert daher $\rho(I - \omega^{-1}A) < 1$. □

DEFINITION IV.6.12 (Zeilensummenkriterium). *(1)* $A \in \mathbb{R}^{n\times n}$ erfüllt das *starke Zeilensummenkriterium*, wenn für alle $1 \leq i \leq n$ gilt

$$|A_{ii}| > \sum_{\substack{j=1\\ j\neq i}}^{n} |A_{ij}| \,.$$

(2) $A \in \mathbb{R}^{n\times n}$ erfüllt das *schwache Zeilensummenkriterium*, wenn für alle $1 \leq i \leq n$ gilt

$$|A_{ii}| \geq \sum_{\substack{j=1\\ j\neq i}}^{n} |A_{ij}|$$

und wenn für ein $i_0 \in \{1, \ldots, n\}$ in obiger Ungleichung das „>“-Zeichen steht.
(3) $A \in \mathbb{R}^{n\times n}$ heißt *reduzibel*, wenn es zwei nicht leere Teilmengen N_1, N_2 von $\mathbb{N}_n^*$ gibt mit

(1) $N_1 \cap N_2 = \emptyset$,
(2) $N_1 \cup N_2 = \mathbb{N}_n^*$,
(3) $A_{ij} = 0$ für alle $i \in N_1$, $j \in N_2$.

Ist A nicht reduzibel, so heißt A *irreduzibel*.

BEMERKUNG IV.6.13. Ist A reduzibel, so gibt es eine Permutationsmatrix P, so dass $P^{-1}AP$ die Form $\left(\begin{smallmatrix} E & F \\ 0 & G \end{smallmatrix}\right)$ hat. Das LGS $Ax = b$ zerfällt dann in zwei kleinere Gleichungssysteme.

SATZ IV.6.14 (Konvergenz des Jacobi-Verfahrens). (1) *A erfülle das starke Zeilensummenkriterium. Dann ist $\rho(S_J) < 1$.*
(2) *A erfülle das schwache Zeilensummenkriterium und sei irreduzibel. Dann ist $\rho(S_J) < 1$.*

BEWEIS. Sei $\|\cdot\|$ die Maximumnorm und $\|\cdot\|_{\mathcal{L}}$ die zugehörige Operatornorm, d.h. die Zeilensummennorm.
ad (1): Aus Beispiel IV.6.4 (2), Satz IV.6.8 (1) und Definition IV.6.12 (1) folgt

$$\rho(S_J) \leq \|S_J\|_{\mathcal{L}} < 1.$$

ad (2): Wie oben folgt

$$\rho(S_J) \leq \|S_J\|_{\mathcal{L}} \leq 1.$$

Angenommen, es gibt einen Eigenwert λ von S_J mit $|\lambda| = 1$. Sei x ein zugehöriger Eigenvektor mit $\|x\|_\infty = 1$. Definiere

$$N_1 = \{i \in \mathbb{N}_n^* : |x_i| = 1\}, \quad N_2 = \{i \in \mathbb{N}_n^* : |x_i| < 1\}.$$

Offensichtlich gilt

$$N_1 \neq \emptyset, \quad N_1 \cup N_2 = \mathbb{N}_n^*, \quad N_1 \cap N_2 = \emptyset.$$

Für $1 \leq i \leq n$ gilt wegen Definition IV.6.12 (2)

$$|x_i| = |\lambda x_i| = \left| \frac{1}{A_{ii}} \left\{ \sum_{\substack{j=1 \\ j\neq i}}^{n} A_{ij} x_j \right\} \right| \leq \frac{1}{|A_{ii}|} \left\{ \sum_{\substack{j=1 \\ j\neq i}}^{n} |A_{ij}|\, |x_j| \right\} \leq 1$$

und daher für alle $i \in N_1$

$$|A_{ii}| = \sum_{\substack{j=1 \\ j\neq i}}^{n} |A_{ij}|\,.$$

Aus Definition IV.6.12 (2) folgt daher $N_2 \neq \emptyset$. Da A irreduzibel ist, gibt es ein $i_0 \in N_1$ und ein $j_0 \in N_2$ mit $A_{i_0 j_0} \neq 0$. Wegen $0 \leq |x_{j_0}| < 1$ gilt dann aber in obiger Ungleichungskette für $i = i_0$ das „$<$“-Zeichen. Dies ist ein Widerspruch. □

Satz IV.6.15 (Konvergenz des Gauß-Seidel-Verfahrens). (1) *A erfülle das starke Zeilensummenkriterium. Dann ist $\rho(S_G) < 1$.*
(2) *A sei s.p.d. Dann ist $\rho(S_G) < 1$.*

Beweis. *ad (1):* Wegen Satz IV.6.14 (1) ist $\kappa = \|S_J\|_{\mathcal{L}} < 1$, wenn $\mathbb{R}^n$ mit der Maximumsnorm $\|\cdot\|_\infty$ versehen wird. Sei $y \in \mathbb{R}^n$ und $z = S_G y$. Wir zeigen durch Induktion über die Komponente i, dass für alle $1 \leq i \leq n$ gilt $|z_i| \leq \kappa \|y\|_\infty$. Hieraus folgt dann die Behauptung mit Satz IV.6.8 (1).

$\underline{i = 1:}$

$$|z_1| = \left| \frac{1}{A_{11}} \sum_{j=2}^{n} A_{1j} y_j \right| = |(S_J y)_1| \leq \kappa \|y\|_\infty\,.$$

$\underline{i \to i+1:}$

$$\begin{aligned}
|z_{i+1}| &= \left| \frac{1}{A_{i+1,i+1}} \left\{ \sum_{j=1}^{i} A_{i+1,j} z_j + \sum_{j=i+2}^{n} A_{i+1,j} y_j \right\} \right| \\
&\leq \frac{1}{|A_{i+1,i+1}|} \left\{ \sum_{j=1}^{i} |A_{i+1,j}|\, \kappa \|y\|_\infty + \sum_{j=i+2}^{n} |A_{i+1,j}|\, \|y\|_\infty \right\} \\
&\leq \|y\|_\infty \frac{1}{|A_{i+1,i+1}|} \sum_{\substack{j=1 \\ j\neq i+1}}^{n} |A_{i+1,j}| \\
&\leq \kappa \|y\|_\infty\,.
\end{aligned}$$

ad (2): Da A s.p.d. ist, folgt

$$\begin{aligned} S_G &= (D-L)^{-1}R \\ &= (D-L)^{-1}L^t \\ &= D^{-\frac{1}{2}}(I - D^{-\frac{1}{2}}LD^{-\frac{1}{2}})^{-1}D^{-\frac{1}{2}}L^t \\ &= D^{-\frac{1}{2}}(I-L_1)^{-1}L_1^t D^{\frac{1}{2}} \end{aligned}$$

mit

$$L_1 = D^{-\frac{1}{2}}LD^{-\frac{1}{2}}.$$

Also hat S_G die gleichen Eigenwerte wie

$$S = (I-L_1)^{-1}L_1^t.$$

Sei λ ein Eigenwert von S und x ein zugehöriger Eigenvektor mit $\|x\|_2 = 1$. Dann folgt

$$\begin{aligned} & & \lambda x &= (I-L_1)^{-1}L_1^t x \\ &\Longrightarrow & \lambda x - \lambda L_1 x &= L_1^t x \\ &\Longrightarrow & \lambda - \lambda\,(x\,,\,L_1 x) &= \big(x\,,\,L_1^t x\big) \\ &\Longrightarrow & \lambda &= \frac{(x\,,\,L_1^t x)}{1-(x\,,\,L_1 x)}. \end{aligned}$$

Setze $(x\,,\,L_1 x) = a + ib$ mit $a, b \in \mathbb{R}$. Wegen $(x\,,\,L_1^t x) = \overline{(x\,,\,L_1 x)}$ folgt

$$|\lambda|^2 = \frac{a^2+b^2}{a^2+b^2+1-2a}.$$

Mit A ist auch $D^{-\frac{1}{2}}AD^{-\frac{1}{2}} = I - L_1 - L_1^t$ positiv definit. Damit folgt

$$0 < \big(x\,,\,(I-L_1-L_1^t)x\big) = 1 - [a+ib] - [a-ib] = 1-2a.$$

Also ist $|\lambda| < 1$. Damit ist die Behauptung bewiesen. □

Man kann den Vergleich zwischen Jacobi- und Gauß-Seidel-Verfahren noch wesentlich verschärfen.

Satz IV.6.16 (Stein-Rosenberg). *Die Elemente der Matrizen $D^{-1}L$ und $D^{-1}R$ seien alle nicht-negativ. Dann gilt einer der folgenden Fälle:*

(1) $\rho(S_G) = \rho(S_J) = 0$,
(2) $0 < \rho(S_G) < \rho(S_J) < 1$,
(3) $\rho(S_G) = \rho(S_J) = 1$,
(4) $1 < \rho(S_J) < \rho(S_G)$.

Beweis. [6, Thm. 3.3]. □

Zum Abschluss betrachten wir noch die Relaxationsverfahren von Jacobi und Gauß-Seidel.

SATZ IV.6.17 (Konvergenz der Jacobi-Relaxation). *Sei $S_{J,\omega}$ die Iterationsmatrix der Jacobi-Relaxation. $S_J = S_{J,1}$ habe lauter reelle Eigenwerte und es gelte $\rho(S_J) < 1$. Dann ist $\rho(S_{J,w})$ optimal für*

$$\omega_{op} = \frac{2}{2 - \lambda_{\max}(S_J) - \lambda_{\min}(S_J)}.$$

Es ist

$$\rho(S_{J,w_{op}}) = \frac{\lambda_{\max}(S_J) - \lambda_{\min}(S_J)}{2 - \lambda_{\max}(S_J) - \lambda_{\min}(S_J)}.$$

BEWEIS. Seien $-1 < \lambda_1 \leq \ldots \leq \lambda_n < 1$ die Eigenwerte von S_J. Dann hat $S_{J,\omega}$ wegen Beispiel IV.6.4 (3) die Eigenwerte

$$(1-\omega) + \omega\lambda_1 \leq \ldots \leq (1-\omega) + \omega\lambda_n.$$

Also ist

$$\rho(S_{J,\omega}) = \max\left\{\left|(1-\omega) + \omega\lambda_1\right|, \left|(1-\omega) + \omega\lambda_n\right|\right\}.$$

Dieser Ausdruck ist genau dann minimal, wenn gilt

$$1 - \omega + \omega\lambda_1 = -[1 - \omega + \omega\lambda_n] \quad \Longleftrightarrow \quad \omega = \frac{2}{2 - \lambda_n - \lambda_1}. \qquad \square$$

SATZ IV.6.18 (Konvergenz der Gauß-Seidel-Relaxation). *Die Matrix A sei s.p.d. Dann konvergiert das Gauß-Seidel-Relaxationsverfahren für alle $\omega \in (0, 2)$.*

BEWEIS. Wir benutzen die gleichen Notationen wie in Beweis von Satz IV.6.15 (2). Dann folgt aus Beispiel IV.6.4 (5)

$$\begin{aligned} S_{G,\omega} &= (D - \omega L)^{-1}\{(1-\omega)D + \omega L^t\} \\ &= D^{-\frac{1}{2}}(I - \omega L_1)^{-1}[(1-\omega)I + \omega L_1^t]D^{\frac{1}{2}}. \end{aligned}$$

Also hat $S_{G,\omega}$ die gleichen Eigenwerte wie

$$S = (I - \omega L_1)^{-1}[(1-\omega)I + \omega L_1^t].$$

Sei λ ein Eigenwert von S und x ein zugehöriger Eigenvektor mit $\|x\|_2 = 1$. Dann folgt mit $(x\,,\, L_1 x) = a + ib$ mit $a, b \in \mathbb{R}$

$$\lambda = \frac{(1-\omega) + \omega a - i\omega b}{1 - \omega a - i\omega b}.$$

Da gemäß dem Beweis von Satz IV.6.15 (2) $1 - 2a > 0$ ist, folgt hieraus für alle $\omega \in (0, 2)$

$$|\lambda|^2 = \frac{(1 - \omega + \omega a)^2 + \omega^2 b^2}{(1 - \omega + \omega a)^2 + \omega^2 b^2 + \omega(2-\omega)(1-2a)} < 1. \qquad \square$$

IV.7. Das konjugierte Gradienten-Verfahren

Wir wollen wieder ein lineares Gleichungssystem $Ax = b$ im $\mathbb{R}^n$ mit invertierbarer Matrix A iterativ lösen. Im Gegensatz zum vorigen Paragraphen fordern wir jetzt zusätzlich, dass A s.p.d ist. Wir zeigen, dass unter dieser Voraussetzung die Lösung des LGS äquivalent ist zur Minimierung einer geeigneten quadratischen Funktion auf $\mathbb{R}^n$. Die Minimierung führen wir zunächst mit Hilfe des Gradienten-Verfahrens durch. Es stellt sich aber heraus, dass dieser Ansatz nicht effizient genug ist. Die unbefriedigende Konvergenzgeschwindigkeit des Gradienten-Verfahrens liegt an der zunehmenden Parallelität der Suchrichtungen. Daher modifizieren wir das Verfahren in einem zweiten Schritt so, dass die Suchrichtungen in einem geeigneten Sinn orthogonal sind. Dies führt auf das konjugierte Gradienten-Verfahren. Schließlich verbessern wir das Verfahren in einem dritten Schritt weiter, indem wir das LGS vor Anwendung des Minimierungsprozesses geschickt umformulieren. Diese Vorgehensweise heißt Vorkonditionierung. Insgesamt erhalten wir ein Verfahren, das zu den leistungsfähigsten Methoden zur Lösung großer linearer Gleichungssysteme gehört.

Im Folgenden ist stets $A \in \mathbb{R}^{n\times n}$ s.p.d.; $(\cdot\,,\,\cdot)$ bezeichnet das euklidische Skalarprodukt und $\|\cdot\|$ die zugehörige Norm. Da A s.p.d. ist, wird durch

$$\|x\|_A = (x\,,\,Ax)^{\frac{1}{2}} \quad \forall x \in \mathbb{R}^n$$

eine Norm, die so genannte *Energienorm* (zu A) definiert.

Der folgende Satz präzisiert die angekündigte Äquivalenz des LGS zu einem Minimierungsproblem und stellt zudem ein technisches Hilfsergebnis bereit.

Satz IV.7.1 (Minimierungsproblem). (1) *$x^* \in \mathbb{R}^n$ ist genau dann Lösung des LGS $Ax = b$, wenn x^* die Funktion $J : \mathbb{R}^n \to \mathbb{R}$ mit*

$$J(x) = \frac{1}{2}\,(x\,,\,Ax) - (b\,,\,x)$$

minimiert.
(2) *Seien $x, y \in \mathbb{R}^n$, $y \neq 0$. Dann nimmt die Funktion $\varphi : \mathbb{R} \to \mathbb{R}$ mit*

$$\varphi(t) = J(x + ty)$$

ihr Minimum an der Stelle

$$t^* = \frac{(y\,,\,b - Ax)}{(y\,,\,Ay)}$$

an.

Beweis. *ad (1):* Folgt aus

$$DJ(x) = x^t A - b^t, \quad D^2 J(x) = A$$

und A s.p.d.
ad (2): Folgt aus

$$\varphi(t) = \frac{1}{2}t^2 \,(y\,,\, Ay) - t\,(y\,,\, b - Ax) + J(x). \qquad \square$$

Da bekanntlich der negative Gradient die Richtung des steilsten Abstieges angibt, ergibt sich aus Satz IV.7.1 Algorithmus IV.7.1 zur Lösung des LGS $Ax = b$.

Algorithmus IV.7.1 Gradienten-Verfahren

Gegeben: Matrix, A, rechte Seite b, Startnäherung x, Toleranz ε, Maximalzahl für Iterationen N
Gesucht: Näherungslösung x mit $\|Ax - b\| \leq \varepsilon$

1: $r \leftarrow b - Ax$, $n \leftarrow 0$
2: **while** $\|r\| > \varepsilon$ und $n \leq N$ **do**
3: $\quad s \leftarrow Ar$, $\alpha \leftarrow \dfrac{(r\,,\, r)}{(r\,,\, s)}$, $x \leftarrow x + \alpha r$, $r \leftarrow r - \alpha s$, $n \leftarrow n + 1$
4: **end while**

Bemerkung IV.7.2. Algorithmus IV.7.1 benötigt $3n$ Speicherplätze.

Der folgende Satz beweist die Konvergenz von Algorithmus IV.7.1. Ein Vergleich mit dem Beweis von Satz IV.6.11 (S. 102) zeigt zudem, dass die Konvergenzrate des Gradienten-Verfahrens die gleiche ist wie diejenige des Richardson-Verfahrens mit optimaler Wahl des Relaxationsparameters ω. Daher ist das Gradienten-Verfahren für die Praxis nicht effizient genug.

Satz IV.7.3 (Konvergenz des Gradienten-Verfahrens). *Seien x^* die Lösung des LGS $Ax = b$, x_i die i-te Iterierte von Algorithmus IV.7.1 und $e_i = x^* - x_i$ der Fehler nach i Iterationen. Dann gilt für alle $i \in \mathbb{N}$*

$$\|e_i\|_A \leq \left[\frac{\kappa_2(A) - 1}{\kappa_2(A) + 1}\right]^i \|e_0\|_A\,,$$

wobei $\kappa_2(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$ die Kondition von A bzgl. der euklidischen Norm ist.

Beweis. Wegen

$$r_i = b - Ax_i = Ae_i$$

und

$$\|e_i\|_A^2 = (b\,,\, x^*) + 2J(x_i)$$

folgt aus Satz IV.7.1 (2)

$$\|e_1\|_A = \min_{t\in\mathbb{R}} \|e_0 - tAe_0\|_A\,.$$

Seien $0 < \lambda_1 \leq \ldots \leq \lambda_n$ die Eigenwerte von A und $z_1, \ldots, z_n$ die zugehörigen Eigenvektoren mit $(z_i\,,\, z_j) = \delta_{ij}$. Sei

$$e_0 = \sum_{i=1}^{n} c_i z_i.$$

Dann folgt

$$\|e_0\|_A = \left\{ \sum_{i=1}^{n} c_i^2 \lambda_i \right\}^{\frac{1}{2}}$$

und für alle $t \in \mathbb{R}$

$$\|e_0 - tAe_0\|_A = \left\{ \sum_{i=1}^{n} c_i^2 (1 - t\lambda_i)^2 \lambda_i \right\}^{\frac{1}{2}}.$$

Also ist

$$\begin{aligned} \|e_1\|_A &= \min_{t\in\mathbb{R}} \left\{ \sum_{i=1}^{n} c_i^2 (1 - t\lambda_i)^2 \lambda_i \right\}^{\frac{1}{2}} \\ &\leq \|e_0\|_A \min_{t\in\mathbb{R}} \max_{1\leq i\leq n} |1 - t\lambda_i| \\ &= \|e_0\|_A \left[1 - \frac{2}{\lambda_1 + \lambda_n} \lambda_1 \right] \\ &= \frac{\kappa_2(A) - 1}{\kappa_2(A) + 1} \|e_0\|_A . \end{aligned}$$

Da x_0 und damit e_0 beliebig waren, folgt hieraus die Behauptung. □

In der Praxis konvergiert Algorithmus IV.7.1 anfangs recht gut. Danach wird die Konvergenzgeschwindigkeit zunehmend langsamer und nähert sich dem theoretischen Wert aus Satz IV.7.3. Eine genauere Analyse zeigt, dass dies an der zunehmenden Parallelität der Vektoren r_k liegt. Um dies zu vermeiden, versucht man, in Richtung paarweise A-orthogonaler Vektoren, die den gleichen Raum erzeugen wie die r_k, zu minimieren. Die folgenden beiden Hilfssätze zeigen, welche Konsequenzen dies hat, und dass es möglich ist, Vektoren mit den gewünschten Eigenschaften zu konstruieren.

LEMMA IV.7.4. *Seien $d_0, \ldots, d_{n-1} \in \mathbb{R}^n$ paarweise A-orthogonal, d.h., $(d_i\,,\, Ad_j) = 0$ für $i \neq j$, und alle von Null verschieden. Für $x_0 \in \mathbb{R}^n$ und $0 \leq k \leq n-1$ sei*

$$r_k = b - Ax_k, \quad \alpha_k = \frac{(r_k\,,\, d_k)}{(d_k\,,\, Ad_k)}, \quad x_{k+1} = x_k + \alpha_k d_k.$$

Dann gilt:

(1) $(r_k\,,\, d_j) = 0$ *für alle* $0 \leq j < k \leq n-1$.
(2) x_k *minimiert* J *in* $x_0 + \operatorname{span}\{d_0, \ldots, d_{k-1}\}$.
(3) *Es gibt ein* $m \leq n$ *mit* $Ax_m = b$.

BEWEIS. *ad (1):* Wir beweisen die Behauptung durch Induktion über k.
$\underline{k = 1:}$

$$(r_1\,,\,d_0) = (b - Ax_0 - \alpha_0 Ad_0\,,\,d_0) = (r_0\,,\,d_0) - \alpha_0\,(d_0\,,\,Ad_0) = 0.$$

$\underline{k \to k+1:}$ Sei $j < k$. Dann folgt

$$(r_{k+1}\,,\,d_j) = (r_k - \alpha_k Ad_k\,,\,d_j) = \underbrace{(r_k\,,\,d_j)}_{=0 \text{ I.v.}} - \alpha_k \underbrace{(d_k\,,\,Ad_j)}_{=0 \text{ orth.}} = 0.$$

Aus der Definition von α_k folgt

$$(r_{k+1}\,,\,d_k) = (r_k\,,\,d_k) - \alpha_k\,(d_k\,,\,Ad_k) = 0.$$

ad (2): Sei $x \in x_0 + \operatorname{span}\{d_0, \ldots, d_{k-1}\}$ und $y = x - x_k$. Wegen $y \in \operatorname{span}\{d_0, \ldots, d_{k-1}\}$ und (1) ist $(r_k\,,\,y) = 0$. Hieraus folgt

$$J(x) = J(x_k + y) = J(x_k) - (r_k\,,\,y) + \frac{1}{2}\,(y\,,\,Ay) = J(x_k) + \frac{1}{2}\,(y\,,\,Ay)$$

und damit die Behauptung.
ad (3): Folgt aus (2) und $x_0 + \operatorname{span}\{d_0, \ldots, d_{n-1}\} = \mathbb{R}^n$. □

LEMMA IV.7.5. *Sei $x_0 \in \mathbb{R}^n$ und $d_0 = r_0 = b - Ax_0$. Falls $r_k \neq 0$ ist, definiere*

$$\begin{aligned}
\alpha_k &= \frac{(r_k\,,\,d_k)}{(d_k\,,\,Ad_k)},\\
x_{k+1} &= x_k + \alpha_k d_k,\\
r_{k+1} &= b - Ax_{k+1} = r_k - \alpha_k Ad_k,\\
\beta_k &= -\frac{(r_{k+1}\,,\,Ad_k)}{(d_k\,,\,Ad_k)},\\
d_{k+1} &= r_{k+1} + \beta_k d_k.
\end{aligned}$$

Sei l_0 der kleinste Index mit $r_{l_0} = 0$. Dann gilt für $0 \leq k < l_0$:

(1) $\operatorname{span}\{r_0, \ldots, r_k\} = \operatorname{span}\{r_0, Ar_0, \ldots, A^k r_0\}$,
(2) $\operatorname{span}\{d_0, \ldots, d_k\} = \operatorname{span}\{r_0, \ldots, A^k r_0\}$,
(3) $(d_k\,,\,Ad_j) = 0$ *für alle* $0 \leq j \leq k-1$,
(4) $\alpha_k = \frac{(r_k\,,\,r_k)}{(d_k\,,\,Ad_k)}$,
(5) $\beta_k = \frac{(r_{k+1}\,,\,r_{k+1})}{(r_k\,,\,r_k)}$.

BEWEIS. *ad (1) – (3):* Wir beweisen die Behauptungen simultan durch Induktion über k.
$\underline{k = 0:}$ Ist offensichtlich.
$\underline{k \to k+1:}$ Nach Voraussetzung ist $\operatorname{span}\{r_0, \ldots, r_k\} = \operatorname{span}\{d_0, \ldots, d_k\}$ und $r_i \neq 0$ für $0 \leq i \leq k$. Also ist $d_k \neq 0$ und damit α_k wohldefiniert. Wegen

$$Ad_k \in \operatorname{span}\{Ar_0, \ldots, Ar_k\} = \operatorname{span}\{Ar_0, \ldots, A^{k+1} r_0\}$$

folgt hieraus (1).
(2) folgt aus der Induktionsvoraussetzung, der Definition von d_{k+1} und (1).
Aus der Induktionsvoraussetzung und Lemma IV.7.4 folgt $(r_k\,,\,d_j)=0$ für $0\le j\le k-1$. Wie in Beweis von Lemma IV.7.4 folgt hieraus $(r_{k+1}\,,\,d_j)=0$ für $0\le j\le k$. Damit folgt für $0\le j\le k-1$ wegen $Ad_j\in\operatorname{span}\{d_0,\ldots,d_k\}$

$$(d_{k+1}\,,\,Ad_j)=(r_{k+1}\,,\,Ad_j)+\beta_k\underbrace{(d_k\,,\,Ad_j)}_{=0\text{ I.v.}}=0.$$

Aus der Definition von β_k folgt schließlich

$$(d_{k+1}\,,\,Ad_k)=(r_{k+1}\,,\,Ad_k)+\beta_k\,(d_k\,,\,Ad_k)=0.$$

ad (4): Aus (1) – (3) und Lemma IV.7.4 folgt

$$(r_k\,,\,r_k)=(d_k-\beta_{k-1}d_{k-1}\,,\,r_k)=(d_k\,,\,r_k)-\beta_{k-1}\underbrace{(d_{k-1}\,,\,r_k)}_{=0}=(d_k\,,\,r_k)\,.$$

ad (5): Wegen $r_k\in\operatorname{span}\{d_0,\ldots,d_k\}$ und $(r_{k+1}\,,\,d_j)=0$ für $0\le j\le k$ folgt

$$(r_{k+1}\,,\,r_{k+1})=(r_{k+1}\,,\,r_{k+1}-r_k)=-\alpha_k\,(r_{k+1}\,,\,Ad_k)$$

und damit wegen (4)

$$\beta_k=-\frac{(r_{k+1}\,,\,Ad_k)}{(d_k\,,\,Ad_k)}=\frac{(r_{k+1}\,,\,r_{k+1})}{\alpha_k\,(d_k\,,\,Ad_k)}=\frac{(r_{k+1}\,,\,r_{k+1})}{(r_k\,,\,r_k)}.$$ □

Aus Lemma IV.7.5 ergibt sich Algorithmus IV.7.2.

Algorithmus IV.7.2 Konjugiertes Gradienten-Verfahren, CG-Verfahren

---

Gegeben: Matrix, A, rechte Seite b, Startnäherung x, Toleranz ε, Maximalzahl für Iterationen N
Gesucht: Näherungslösung x mit $\|Ax-b\|\le\varepsilon$

1: $r\leftarrow b-Ax$, $d\leftarrow r$, $\gamma\leftarrow(r\,,\,r)$, $n\leftarrow 0$
2: **while** $\gamma>\varepsilon^2$ und $n\le N$ **do**
3: $\quad s\leftarrow Ad$, $\alpha\leftarrow\dfrac{\gamma}{(d\,,\,s)}$, $x\leftarrow x+\alpha d$, $r\leftarrow r-\alpha s$
4: $\quad\beta\leftarrow\frac{(r\,,r)}{\gamma}$, $\gamma\leftarrow(r\,,\,r)$, $d\leftarrow r+\beta d$, $n\leftarrow n+1$
5: **end while**

Bemerkung IV.7.6. Im Vergleich zu Algorithmus IV.7.1 benötigt Algorithmus IV.7.2 einen zusätzlichen Hilfsvektor zur Speicherung von d.

Der erste Teil des folgenden Satzes zeigt, dass das CG-Verfahren bei exakter Arithmetik so gar ein direktes Verfahren ist, d.h. in endlich vielen Schritten die exakte Lösung des LGS liefert. Der Aufwand ist dann etwa vergleichbar zu demjenigen des Gaußschen Eliminationsverfahrens

aus §IV.1. Dieses Ergebnis hat aber nur theoretische Bedeutung. Für die Praxis viel wichtiger ist die Abschätzung der Konvergenzrate des CG-Verfahrens im zweiten Teil des Satzes. Ein Vergleich mit der entsprechenden Abschätzung aus Satz IV.7.3 zeigt, dass das CG-Verfahren wesentlich schneller konvergiert als das Gradienten-Verfahren.

SATZ IV.7.7 (Konvergenz des CG-Verfahrens). (1) *Algorithmus IV.7.2 bricht bei exakter Arithmetik für jedes $\varepsilon \geq 0$ nach spätestens n Schritten ab. Im Falle $\varepsilon = 0$ liefert er dann die exakte Lösung des LGS $Ax = b$.*
(2) *Sei x^* die exakte Lösung von $Ax = b$ und $e_i = x^* - x_i$ der Fehler der i-ten Iterierten von Algorithmus IV.7.2. Dann gilt*

$$\|e_i\|_A \leq 2 \left[\frac{\sqrt{\kappa_2(a)} - 1}{\sqrt{\kappa_2(A)} + 1} \right]^i \|e_0\|_A ,$$

wobei $\kappa_2(A)$ die Kondition von A bzgl. der euklidischen Norm ist.

BEWEIS. *ad (1):* Folgt aus Lemma IV.7.4 (3) und Lemma IV.7.5.
ad (2): Aus Lemma IV.7.4 (2) und Lemma IV.7.5 folgt für $i \in \mathbb{N}^*$

$$\begin{aligned} \|e_i\|_A &= \min_{v \in \operatorname{span}\{r_0, \ldots, A^{i-1} r_0\}} \|e_0 + v\|_A \\ &= \min_{v \in \operatorname{span}\{Ae_0, \ldots, A^i e_0\}} \|e_0 + v\|_A \\ &= \min_{\substack{p \in \mathbb{P}_i \\ p(0)=1}} \|p(A) e_0\|_A . \end{aligned}$$

Mit den gleichen Notationen wie in Beweis von Satz IV.7.3 folgt hieraus

$$\begin{aligned} \|e_i\|_A &= \min_{\substack{p \in \mathbb{P}_i \\ p(0)=1}} \left| \sum_{j=1}^{n} c_j^2 p(\lambda_j)^2 \lambda_j \right|^{\frac{1}{2}} \\ &\leq \|e_0\|_A \min_{\substack{p \in \mathbb{P}_i \\ p(0)=1}} \max_{1 \leq j \leq n} |p(\lambda_j)| \\ &\leq \|e_0\|_A \min_{\substack{p \in \mathbb{P}_i \\ p(0)=1}} \max_{\lambda_1 \leq x \leq \lambda_n} |p(x)| . \end{aligned}$$

Mit Hilfe eines allgemeinen Satzes der Approximationstheorie, den wir hier nicht beweisen können, folgt

$$\min_{\substack{p \in \mathbb{P}_i \\ p(0)=1}} \max_{\lambda_1 \leq x \leq \lambda_n} |p(x)| = \left| T_i \left(\frac{\lambda_n + \lambda_1}{\lambda_n - \lambda_1} \right) \right|^{-1}$$

mit

$$T_i(x) = \frac{1}{2} \left\{ \left[x + \sqrt{x^2 - 1} \right]^i + \left[x - \sqrt{x^2 - 1} \right]^i \right\} .$$

Mit $\kappa = \kappa_2(A) = \frac{\lambda_n}{\lambda_1}$ folgt hieraus

$$\begin{aligned}\frac{\|e_i\|_A}{\|e_0\|_A} &\le \left|T_i\left(\frac{\kappa+1}{\kappa-1}\right)\right|^{-1} \\ &\le 2\left\{\left[\frac{\kappa+1}{\kappa-1}+\sqrt{(\frac{\kappa+1}{\kappa-1})^2-1}\right]^i \right. \\ &\qquad \left. +\left[\frac{\kappa+1}{\kappa-1}-\sqrt{(\frac{\kappa+1}{\kappa-1})^2-1}\right]^i\right\}^{-1}.\end{aligned}$$

Wegen

$$\frac{\kappa+1}{\kappa-1} \pm \sqrt{\left(\frac{\kappa+1}{\kappa-1}\right)^2-1} = \frac{\kappa+1\pm 2\sqrt{\kappa}}{\kappa-1} = \frac{(\sqrt{\kappa}\pm 1)^2}{(\sqrt{\kappa}+1)(\sqrt{\kappa}-1)}$$

liefert dies

$$\frac{\|e_i\|_A}{\|e_0\|_A} \le 2\left\{\left[\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right]^i+\left[\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right]^i\right\}^{-1} \le 2\left[\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right]^i. \qquad \square$$

Um die Konvergenz von Algorithmus IV.7.2 zu verbessern, betrachtet man in der Praxis *vorkonditionierte CG-Verfahren* kurz *PCG-Verfahren*. Sei dazu $H \in \mathcal{GL}(\mathbb{R}^n)$ und $C = HH^t$. Dann ist C s.p.d. Wir nehmen an, dass C eine gute Approximation an A ist und dass ein LGS der Form $Cz = d$ für beliebiges d leicht lösbar ist. Setze

$$\widehat{A} = H^{-1}AH^{-t}, \quad \widehat{b} = H^{-1}b.$$

Dann ist $\widehat{A}$ s.p.d., und x^* ist eine Lösung von $Ax = b$ genau dann, wenn $\widehat{x}^* = H^t x^*$ eine Lösung von $\widehat{A}\widehat{x} = \widehat{b}$ ist. Da $\widehat{A}$ die gleichen Eigenwerte hat wie $C^{-1}A$, können wir hoffen, dass bei geschickter Wahl von C gilt $\kappa_2(\widehat{A}) \ll \kappa_2(A)$. Dann konvergiert Algorithmus IV.7.2 für das LGS $\widehat{A}\widehat{x} = \widehat{b}$ wesentlicher schneller als für das ursprüngliche LGS.

Da wir aber mit den ursprünglichen Größen rechnen wollen, gehen wir wie folgt vor:
Wähle einen Startwert $x_0 \in \mathbb{R}^n$ und wende Algorithmus IV.7.2 auf das LGS $\widehat{A}\widehat{x} = \widehat{b}$ mit Startwert $\widehat{x}_0 = H^t x_0$ an. Die resultierenden Größen werden mit $\widehat{\ }$ gekennzeichnet. Setze

$$r_i = H\widehat{r}_i, \quad d_i = H^{-t}\widehat{d}_i, \quad z_i = C^{-1}r_i.$$

Dann liefert eine leichte Rechnung die folgenden Rekursionsformeln

$$\begin{aligned}\widehat{\alpha}_i &= \frac{(r_i\,,\,z_i)}{(d_i\,,\,Ad_i)}, \qquad x_{i+1} = x_i + \widehat{\alpha}_i d_i, \qquad r_{i+1} = r_i - \widehat{\alpha}_i A d_i, \\ \widehat{\beta}_i &= \frac{(r_{i+1}\,,\,z_{i+1})}{(r_i\,,\,z_i)}, \quad d_{i+1} = z_{i+1} + \widehat{\beta}_i d_i.\end{aligned}$$

Also liefert Algorithmus IV.7.2 angewandt auf $\widehat{A}\widehat{x} = \widehat{b}$ Algorithmus IV.7.3 zur Lösung des LGS $Ax = b$.

Algorithmus IV.7.3 Vorkonditioniertes konjugiertes Gradienten-Verfahren, PCG-Verfahren

Gegeben: Matrix, A, rechte Seite b, Startnäherung x, Toleranz ε, Vorkonditionierungsmatrix C, Maximalzahl für Iterationen N
Gesucht: Näherungslösung x mit $\|Ax - b\| \leq \varepsilon$

1: $r \leftarrow b - Ax$, $z \leftarrow C^{-1}r$, $d \leftarrow z$, $\gamma \leftarrow (r\,,\,z)$, $n \leftarrow 0$
2: **while** $\gamma > \varepsilon^2$ und $n \leq N$ **do**
3: $\quad s \leftarrow Ad$, $\alpha \leftarrow \dfrac{\gamma}{(d\,,\,s)}$, $x \leftarrow x + \alpha d$, $r \leftarrow r - \alpha s$
4: $\quad z \leftarrow C^{-1}r$, $\beta \leftarrow \frac{(r\,,\,z)}{\gamma}$, $\gamma \leftarrow (r\,,\,z)$, $d \leftarrow z + \beta d$, $n \leftarrow n + 1$
5: **end while**

Bemerkung IV.7.8. Im Vergleich zu Algorithmus IV.7.2 benötigt Algorithmus IV.7.3 einen zusätzlichen Speicherplatz für z.

Der folgende Satz beweist die Konvergenz von Algorithmus IV.7.3.

Satz IV.7.9 (Konvergenz des PCG-Verfahrens). *Sei x^* die exakte Lösung von $Ax = b$ und $e = x^* - x_i$ der Fehler der i-ten Iterierten von Algorithmus IV.7.3. Dann gilt*

$$\|e_i\|_A \leq 2 \left[\frac{\sqrt{\kappa_2(\widehat{A})} - 1}{\sqrt{\kappa_2(\widehat{A})} + 1} \right]^i \|e_0\|_A$$

mit

$$\kappa_2(\widehat{A}) = \frac{\lambda_{\max}(\widehat{A})}{\lambda_{\min}(\widehat{A})} = \frac{\lambda_{\max}(C^{-1}A)}{\lambda_{\min}(C^{-1}A)}.$$

Beweis. Konstruktionsgemäß gilt

$$\begin{aligned} \|\widehat{x}^* - \widehat{x}_i\|_{\widehat{A}}^2 &= \left(H^t(x^* - x_i)\,,\, H^{-1}AH^{-t}H^t(x^* - x_i)\right) \\ &= (x^* - x_i\,,\, A(x^* - x_i)) \\ &= \|x^* - x_i\|_A^2\,. \end{aligned}$$

Damit folgt die Fehlerabschätzung aus Satz IV.7.7 (2) angewandt auf $\widehat{A}$. Die Formel für $\kappa_2(\widehat{A})$ folgt aus der Ähnlichkeit von $\widehat{A} = H^{-1}AH^{-t}$ zu $H^{-t}H^{-1}A = C^{-1}A$ (vgl. §V.1). □

Bemerkung IV.7.10. Man beachte, dass Algorithmus IV.7.3 und Satz IV.7.9 nur die s.p.d. Matrix C, nicht aber die Matrix H benötigen.

Satz IV.7.9 zeigt, dass das PCG-Verfahren bei geschickter Wahl der Vorkonditionierungsmatrix C wesentlich schneller konvergiert als das CG-Verfahren. Andererseits muss dafür in jeder Iteration ein LGS der

$Cz = d$ gelöst werden. Eine gute Vorkonditionierungsmatrix C muss daher die beiden widerstrebenden Bedingungen erfüllen:

- Gleichungssysteme der Form $Cz = d$ sind leicht lösbar,
- $\frac{\lambda_{\max}(C^{-1}A)}{\lambda_{\min}(C^{-1}A)} \approx 1$.

Verschiedene Wahlen von C sind möglich. Eine gebräuchliche, sehr einfache Wahl ist die SSOR-Vorkonditionierung. Dabei ist für ein $\omega \in (0, 2)$

$$C = \frac{1}{\omega(2-\omega)}(D - \omega L)D^{-1}(D - \omega L^t).$$

Praktisch wird sie wie folgt durchgeführt.

Algorithmus IV.7.4 SSOR-Vorkonditionierung

Gegeben: Matrix A, Vektor r, Relaxationsparameter $\omega \in (0, 2)$
Gesucht: $z = C^{-1}r$

1: $z \leftarrow 0$
2: **for** $i = 1, \ldots, n$ **do**
3: $\quad z_i \leftarrow z_i + \frac{\omega}{A_{ii}} \left\{ r_i - \sum_{j=1}^{n} A_{ij} z_j \right\}$
4: **end for**
5: **for** $i = n, n-1, \ldots, 1$ **do**
6: $\quad z_i \leftarrow z_i + \frac{\omega}{A_{ii}} \left\{ r_i - \sum_{j=1}^{n} A_{ij} z_j \right\}$
7: **end for**

KAPITEL V

Eigenwertprobleme

In diesem Kapitel betrachten wir Verfahren zur numerischen Bestimmung von Eigenwerten und Eigenvektoren. Zunächst stellen wir einige bekannte theoretische Ergebnisse zusammen. Danach betrachten wir eine Ähnlichkeitstransformation auf so genannte Hessenberg-Form. Sie reduziert den Aufwand der in den folgenden Abschnitten betrachteten Verfahren zur Berechnung des größten bzw. kleinsten Eigenwertes oder aller Eigenwerte und Eigenvektoren. Abschließend gehen wir noch kurz auf die Singulärwertzerlegung rechteckiger Matrizen ein.

V.1. Allgemeine Ergebnisse

Im Folgenden bezeichnet $(\cdot\,,\,\cdot)$ das euklidische Skalarprodukt auf $\mathbb{C}^n$ und $\|\cdot\|$ die zugehörige Norm.

Ist $A \in \mathbb{C}^{n\times m}$, so bezeichnet $A^* \in C^{m\times n}$ die zu A *adjungierte Matrix* mit

$$(A^*)_{ij} = \overline{A_{ji}} \quad 1 \le i \le m, 1 \le j \le n.$$

Ist $A = A^*$, so heißt A *hermitisch*. Ist $A \in \mathbb{R}^{n\times n}$ hermitisch, so ist A *symmetrisch*. Offensichtlich gilt für alle $x, y \in \mathbb{C}^n$, $A \in \mathbb{C}^{n\times n}$

$$(x\,,\,Ay) = (A^*x\,,\,y)\,.$$

Sofern nicht anders vermerkt, ist im Folgenden $A \in \mathbb{C}^{n\times n}$. Das *charakteristische Polynom* χ_A von A ist definiert durch

$$\chi_A(\lambda) = \det(\lambda I - A).$$

Es hat genau n komplexe Nullstellen $\lambda_1, \ldots, \lambda_n$. Diese heißen die *Eigenwerte (EW)* von A. Die Menge

$$\sigma(A) = \{\lambda \in \mathbb{C} : \lambda \text{ ist EW von } A\}$$

ist das *Spektrum* von A. Zu jedem $\lambda \in \sigma(A)$ gibt es mindestens einen *Rechtseigenvektor* $x \in \mathbb{C}^n \setminus \{0\}$ mit

$$Ax = \lambda x$$

und mindestens einen *Linkseigenvektor* $y \in \mathbb{C}^n \setminus \{0\}$ mit

$$y^*A = \lambda y^*.$$

Statt Rechtseigenvektor sagen wir kurz *Eigenvektor (EV)*. Ein Eigenvektor x mit $\|x\| = 1$ heißt *normiert*.

Ist $A \in \mathbb{R}^{n\times n}$ und $\lambda \in \mathbb{C} \setminus \mathbb{R}$ ein EW, so ist auch $\overline{\lambda}$ ein EW. Ist A zusätzlich hermitisch, so sind alle EW reell und Links- und Rechtseigenvektoren sind gleich. Es gibt dann eine Orthonormalbasis des $\mathbb{R}^n$ aus Eigenvektoren.

Zwei Matrizen $A, B \in \mathbb{C}^{n\times n}$ heißen *ähnlich*, wenn es ein $X \in \mathcal{G}L(\mathbb{C}^n)$ gibt mit

$$B = XAX^{-1}.$$

Die Transformation $A \mapsto XAX^{-1}$ heißt *Ähnlichkeitstransformation*. Ähnliche Matrizen haben das gleiche Spektrum. Ist $\lambda \in \sigma(A)$ und x ein zugehöriger EV, so ist Xx ein EV von XAX^{-1} zu λ.

Das folgende Beispiel zeigt, dass das Eigenwertproblem numerisch schlecht konditioniert ist.

Beispiel V.1.1 (Schlecht konditioniertes Eigenwertproblem). Seien

$$A = \begin{pmatrix} 0 & 1 & & \\ & \ddots & \ddots & 0 \\ & & \ddots & 1 \\ 0 & & & 0 \end{pmatrix} \in \mathbb{R}^{10\times 10},$$

$$B = \begin{pmatrix} 0 & 1 & & \\ & \ddots & \ddots & 0 \\ & & \ddots & 1 \\ 0 & & & 0 \\ 10^{-10} & 0 & & \end{pmatrix} \in \mathbb{R}^{10\times 10}.$$

Dann ist

$$\det(\lambda I - A) = \lambda^{10}, \quad \det(\lambda I - B) = \lambda^{10} - 10^{-10}$$

und

$$\sigma(A) = \{0\}, \quad \sigma(B) = \{-10^{-1}, 10^{-1}\}.$$

Eine Störung von 10^{-10} in den Koeffizienten der Matrix bewirkt also eine Störung von 10^{-1} in den Eigenwerten.

Die folgenden beiden Sätze erlauben unter geeigneten Voraussetzungen eine Abschätzung der Kondition des Eigenwertproblems. In Verbindung mit Beispiel V.1.1 zeigen sie, dass mehrfache Eigenwerte besonders empfindlich auf Störungen reagieren.

Satz V.1.2 (Verhalten von Eigenwerten unter Störungen). *Seien $A, B \in \mathbb{C}^{n\times n}$, $X \in \mathcal{GL}(\mathbb{C}^n)$ mit*

$$XAX^{-1} = D = \operatorname{diag}(d_1, \ldots, d_n) \in \mathbb{C}^{n\times n}$$

und $\mu \in \sigma(A + B)$. Dann gilt

$$\min_{\lambda\in\sigma(A)} |\lambda - \mu| \leq \kappa(X) \, \|B\|_{\mathcal{L}(\mathbb{C}^n,\mathbb{C}^n)},$$

wobei $\kappa(X)$ die Kondition von X bzgl. $\|\cdot\|$ ist.

BEWEIS. Ohne Einschränkung ist $\mu \notin \sigma(A)$. Dann ist $D - \mu I$ regulär und $(D-\mu I)^{-1}X(A+B-\mu I)X^{-1}$ singulär. Betrachte ein $z \in \ker((D-\mu I)^{-1}X(A+B-\mu I)X^{-1})$ mit $\|z\| = 1$. Dann ist

$$(D-\mu I)^{-1}X(A-\mu I)X^{-1}z = (D-\mu I)^{-1}(XAX^{-1}-\mu I)z = z$$

und

$$\begin{aligned}
&\left\|(D-\mu I)^{-1}XBX^{-1}\right\|_{\mathcal{L}} \\
&\quad \geq \left\|(D-\mu I)^{-1}XBX^{-1}z\right\| \\
&\quad = \left\|(D-\mu I)^{-1}XBX^{-1}z + (D-\mu I)^{-1}X(A-\mu I)X^{-1}z - z\right\| \\
&\quad = \left\|(D-\mu I)^{-1}X(A+B-\mu I)X^{-1}z - z\right\| \\
&\quad = \|z\| \\
&\quad = 1.
\end{aligned}$$

Andererseits gilt

$$\begin{aligned}
\left\|(D-\mu I)^{-1}XBX^{-1}\right\|_{\mathcal{L}} &\leq \left\|(D-\mu I)^{-1}\right\|_{\mathcal{L}} \kappa(X) \|B\|_{\mathcal{L}} \\
&= \frac{1}{\min\limits_{\lambda\in\sigma(A)} |\lambda-\mu|} \kappa(X) \|B\|_{\mathcal{L}} . \qquad \square
\end{aligned}$$

SATZ V.1.3 (Verhalten von einfachen Eigenwerten unter Störungen). *Seien $A, B \in \mathbb{C}^{n\times n}$, λ_0 ein einfacher Eigenwert von A und x_0, y_0 normierte Rechts- bzw. Linkseigenvektoren von A zu λ_0. Dann besitzt $A + \varepsilon B$ für hinreichend kleines ε einen Eigenwert $\lambda(\varepsilon)$ mit*

$$\lambda(\varepsilon) = \lambda_0 + \varepsilon \frac{(y_0\,,\, Bx_0)}{(y_0\,,\, x_0)} + \mathrm{O}(\varepsilon^2).$$

BEWEIS. Definiere $\varphi \in C^\infty(\mathbb{C}\times\mathbb{R}, \mathbb{C})$ durch

$$\varphi(\lambda, \varepsilon) = \det(\lambda I - A - \varepsilon B).$$

Da λ_0 ein einfacher Eigenwert von A ist, gilt

$$\varphi(\lambda_0, 0) = 0 \quad \text{und} \quad \frac{\partial}{\partial\lambda}\varphi(\lambda_0, 0) \neq 0.$$

Aus dem Satz über implizite Funktionen folgt, dass es ein $\varepsilon_0 \in \mathbb{R}_+^*$ und Abbildungen $\lambda \in C^\infty((-\varepsilon_0, \varepsilon_0), \mathbb{C})$ und $x \in C^1((-\varepsilon_0, \varepsilon_0), \mathbb{C}^n)$ gibt mit

$$\begin{aligned}
&\lambda(\varepsilon) \in \sigma(A + \varepsilon B), \\
&x(\varepsilon) \text{ ist Eigenvektor von } A + \varepsilon B \text{ zu } \lambda(\varepsilon), \\
&\lambda(0) = \lambda_0, \\
&x(0) = x_0.
\end{aligned}$$

Aus

$$(A + \varepsilon B)x(\varepsilon) = \lambda(\varepsilon)x(\varepsilon)$$

folgt durch Differentiation bzgl. des Parameters ε

$$(A + \varepsilon B)\dot{x}(\varepsilon) + Bx(\varepsilon) = \dot{\lambda}(\varepsilon)x(\varepsilon) + \lambda(\varepsilon)\dot{x}(\varepsilon)$$

und

$$A\dot{x}(0) + Bx_0 = \dot{\lambda}(0)x_0 + \lambda_0\dot{x}(0)$$

sowie

$$\begin{aligned}(y_0\,,\,A\dot{x}(0)) + (y_0\,,\,Bx_0) &= \dot{\lambda}(0)\,(y_0\,,\,x_0) + \lambda_0\,(y_0\,,\,\dot{x}(0)) \\ &= \dot{\lambda}(0)\,(y_0\,,\,x_0) + (y_0\,,\,A\dot{x}(0))\,,\end{aligned}$$

da y_0 ein Linkseigenvektor zu λ_0 ist. Hieraus folgt die Behauptung wegen $\lambda(\varepsilon) = \lambda_0 + \varepsilon\dot{\lambda}(0) + \mathrm{O}(\varepsilon^2)$. □

Bemerkung V.1.4 (Verhalten von mehrfachen Eigenwerten unter Störungen). Die Bezeichnungen seien wie in Satz V.1.3. Ist λ_0 ein p-facher Eigenwert mit $p > 1$, so verhält sich $\lambda(\varepsilon)$ i.a. wie $\lambda_0 + \mathrm{O}(\varepsilon^{\frac{1}{p}})$.

V.2. Reduktion auf Normalform

In diesem Paragraphen betrachten wir die so genannte Hessenberg-Form von Matrizen und zeigen wie Matrizen mit Householder-Transformationen in ähnliche Matrizen in Hessenberg-Form transformiert werden können. Hintergrund hierfür ist die Tatsache, dass die Algorithmen der Paragraphen V.3 und V.4 zur Berechnung von Eigenwerten und Eigenvektoren für Matrizen in Hessenberg-Form einen wesentlich geringeren Aufwand erfordern als für allgemeine Matrizen. Daher ist es häufig sinnvoll, vor Anwendung der Verfahren der Paragraphen V.3 und V.4 eine gegeben Matrix auf Hessenberg-Form zu transformieren.

Im Folgenden sind alle Matrizen, sofern nicht extra vermerkt, reell; $(\cdot\,,\,\cdot)$ und $\|\cdot\|$ bezeichnen das euklidische Skalarprodukt und die euklidische Norm auf $\mathbb{R}^n$.

Definition V.2.1 (Hessenberg-Form). Eine Matrix $A \in \mathbb{R}^{n\times n}$ hat *Hessenberg-Form*, wenn für alle $1 \le j < j+2 \le i \le n$ gilt $A_{ij} = 0$, d.h. alle Elemente von A unterhalb der ersten Nebendiagonalen verschwinden.

Bemerkung V.2.2 (Eigenschaften der Hessenberg-Form). *(1)* A sei symmetrisch und habe Hessenberg-Form. Dann ist A tridiagonal. *(2)* Hat A Hessenberg-Form, so kann die Q-R-Zerlegung von A aus Paragraph IV.3 in $\mathrm{O}(n^2)$ statt $\mathrm{O}(n^3)$ Operationen für allgemeine Matrizen berechnet werden.

Wir wollen nun eine gegebene Matrix A mittels Householder-Transformationen in eine ähnliche Matrix mit Hessenberg-Form transformieren. Sei dazu $1 \le k \le n-2$. Wir nehmen an, dass A die Form hat

$$A = \begin{pmatrix} B & C \\ D & E \end{pmatrix}$$

mit

$$B \in \mathbb{R}^{k\times k} \text{ hat Hessenberg-Form}$$
$$C \in \mathbb{R}^{k\times(n-k)}$$
$$D \in \mathbb{R}^{(n-k)\times k} \text{ und } D_{ij} = 0 \quad \forall 1 \le i \le n-k, 1 \le j \le k-1$$
$$E \in \mathbb{R}^{(n-k)\times(n-k)}.$$

Sei d die k-te Spalte von D. Dann gibt es gemäß Satz IV.3.3 (S. 87) eine Householder-Matrix $H_k \in \mathbb{R}^{(n-k)\times(n-k)}$ mit

$$H_k d = \sigma \|d\| e_1.$$

Setze

$$Q_k = \begin{pmatrix} I_k & 0 \\ 0 & H_k \end{pmatrix},$$

wobei $I_k \in \mathbb{R}^{k\times k}$ die $k \times k$ Einheitsmatrix bezeichnet. Dann folgt

$$Q_k A Q_k^{-1} = Q_k A Q_k = \begin{pmatrix} B & CH_k \\ H_k D & H_k E H_k \end{pmatrix},$$

so dass $Q_k A Q_k^{-1}$ die obigen Bedingungen für $k+1$ erfüllt.

Wenn wir berücksichtigen, dass eine beliebige Matrix $A \in \mathbb{R}^{n\times n}$ die obigen Voraussetzungen mit $k = 1$ erfüllt, erhalten wir so orthogonale Matrizen $Q_1, \ldots, Q_{n-2}$, derart dass

$$Q_{n-2}Q_{n-3}\ldots Q_1 A Q_1 \ldots Q_{n-2} = Q_{n-2}Q_{n-3}\ldots Q_1 A (Q_{n-2}\ldots Q_1)^{-1}$$

zu A ähnlich ist und Hessenberg Form hat.

Algorithmus V.2.1 realisiert diese Überlegungen. Dabei werden die Vektoren der Householder-Matrizen in einer Hilfsmatrix Q gespeichert. Wenn man später nur an den Eigenwerten und nicht an den Eigenvektoren der Ausgangsmatrix interessiert ist, braucht man diese Hilfsmatrix nicht abzuspeichern.

V.3. Berechnung eines Eigenwertes

In diesem Abschnitt betrachten wir Verfahren zur Berechnung des betragsmäßig größten oder kleinsten Eigenwertes. Alle Matrizen und Vektoren sind dabei reell; $(\cdot,\cdot)$ und $\|\cdot\|$ bezeichnen das euklidische Skalarprodukt und die euklidische Norm auf $\mathbb{R}^n$.

Zur Beschreibung der Grundidee des ersten Verfahrens nehmen wir an, dass die Matrix $A \in \mathbb{R}^{n\times n}$ n linear unabhängige Eigenvektoren $z_1, \ldots, z_n$ mit zugehörigen Eigenwerten $\mu_1, \ldots, \mu_n$ besitzt und dass die Eigenwerte die Beziehung

$$|\mu_n| \le \ldots \le |\mu_2| < |\mu_1|$$

erfüllen. Sei

$$x = \sum_{k=1}^{n} \alpha_k z_k$$

Algorithmus V.2.1 Ähnlichkeitstransformation mittels Householder-Matrizen auf Hessenberg-Form

Gegeben: Matrix A (wird mit ähnlicher Matrix überschrieben), Hilfsmatrix Q (speichert die Transformationen), Toleranz ε
Gesucht: Transformierte Matrix A

1: **for** $i = 1, 2, \ldots, n-2$ **do**
2: $\quad s \leftarrow \left\{ \sum_{j=i+1}^{n} A_{ji}^2 \right\}^{\frac{1}{2}}, \sigma \leftarrow -\operatorname{sgn}(A_{i+1,i}), c \leftarrow \left\{ 2s \left(s + |A_{i+1,i}| \right) \right\}^{\frac{1}{2}}$
3: $\quad$ Falls $s \leq \varepsilon$ ist, gehe zu 2 mit $i+1$ statt i über.
4: $\quad Q_{i+1,i} \leftarrow \dfrac{(A_{i+1,i} - \sigma s)}{c}$
5: $\quad$ **for** $j = i+2, \ldots, n$ **do**
6: $\quad\quad Q_{ji} \leftarrow \dfrac{A_{ji}}{c}$.
7: $\quad$ **end for**
8: $\quad A_{i+1,i} \leftarrow \sigma s$
9: $\quad$ **for** $j = i+2, \ldots, n$ **do**
10: $\quad\quad A_{ji} \leftarrow 0$
11: $\quad$ **end for**
12: $\quad$ **for** $j = 1, \ldots, i$ **do**
13: $\quad\quad s \leftarrow 2 \sum_{k=i+1}^{n} Q_{ki} A_{jk}$
14: $\quad\quad$ **for** $k = i+1, \ldots, n$ **do**
15: $\quad\quad\quad A_{jk} \leftarrow A_{jk} - s Q_{ki}$
16: $\quad\quad$ **end for**
17: $\quad$ **end for**
18: $\quad$ **for** $j = i+1, \ldots, n$ **do**
19: $\quad\quad s \leftarrow 2 \sum_{k=i+1}^{n} Q_{ki} A_{kj}$
20: $\quad\quad$ **for** $k = i+1, \ldots, n$ **do**
21: $\quad\quad\quad A_{kj} \leftarrow A_{kj} - s Q_{ki}$
22: $\quad\quad$ **end for**
23: $\quad$ **end for**
24: $\quad$ **for** $j = i+1, \ldots, n$ **do**
25: $\quad\quad s \leftarrow 2 \sum_{k=i+1}^{n} Q_{ki} A_{jk}$
26: $\quad\quad$ **for** $k = i+1, \ldots, n$ **do**
27: $\quad\quad\quad A_{jk} \leftarrow A_{jk} - s Q_{ki}$
28: $\quad\quad$ **end for**
29: $\quad$ **end for**
30: **end for**

ein Vektor mit $\alpha_1 \neq 0$. Für $m \in \mathbb{N}$ ist dann

$$A^m x = \sum_{k=1}^{n} \alpha_k \mu_k^m z_k = \mu_1^m \sum_{k=1}^{n} \alpha_k \left(\frac{\mu_k}{\mu_1}\right)^m z_k.$$

Wegen

$$\left\| \sum_{k=1}^{n} \alpha_k \left(\frac{\mu_k}{\mu_1}\right)^m z_k \right\| \underset{m\to\infty}{\longrightarrow} |\alpha_1| \, \|z_1\|$$

gilt

$$\frac{\|A^{m+1}x\|}{\|A^m x\|} \underset{m\to\infty}{\longrightarrow} |\mu_1| \, .$$

Daher können wir hoffen, durch sukzessive Multiplikation mit A und anschließende Reskalierung den größten Eigenwert von A berechnen zu können.

Diese Idee wir durch Algorithmus V.3.1 konkretisiert.

Algorithmus V.3.1 Potenzmethode zur Berechnung des betragsmäßig größten Eigenwertes

---

Gegeben: Matrix A, Startnäherung x für einen Eigenvektor mit $\|x\| = 1$, Toleranz ε, Maximalzahl für Iterationen N

Gesucht: Näherung λ für betragsmäßig größten Eigenwert und x für zugehörigen Eigenvektor

1: $d \leftarrow \infty$, $\lambda \leftarrow \infty$, $n \leftarrow 0$, $u \leftarrow Ax$
2: **while** $d > \varepsilon$ und $n \leq N$ und $\|u\| > \varepsilon$ **do**
3: $\quad \sigma \leftarrow \operatorname{sgn}((u\,,\,x))$, $d \leftarrow \lambda$, $\lambda \leftarrow \sigma\,\|u\|$, $d \leftarrow |\lambda - d|$
4: $\quad x \leftarrow \dfrac{\sigma u}{\|u\|}$, $u \leftarrow Ax$, $n \leftarrow n + 1$
5: **end while**

Der folgende Satz beweist die Konvergenz der Potenzmethode und konkretisiert unsere anfängliche heuristische Überlegung.

Satz V.3.1 (Konvergenz der Potenzmethode). *$A \in \mathbb{R}^{n\times n}$ habe n linear unabhängige Eigenvektoren $z_1, \ldots, z_n \in \mathbb{R}^n$ mit zugehörigen Eigenwerten $\mu_1, \ldots, \mu_n \in \mathbb{R}$. Es gelte*

$$0 \leq |\mu_n| \leq \ldots \leq |\mu_{p+1}| < |\mu_p| = \ldots = |\mu_1|$$

und

$$\mu_p = \ldots = \mu_1.$$

Ferner sei in der Darstellung

$$x_0 = \sum_{i=1}^{n} \alpha_i z_i$$

mindestens ein α_i mit $1 \leq i \leq p$ ungleich Null. Dann gibt es einen Eigenvektor x^* von A zum Eigenwert μ_1 mit

$$\|x_m - x^*\| = \mathrm{O}(q^m), \quad |\lambda_m - \mu_1| = \mathrm{O}(q^m),$$

wobei $q = \frac{|\mu_{p+1}|}{|\mu_p|}$ *ist.*

BEWEIS. Sei $\sigma_0 = 1$ und

$$\eta_m = \prod_{i=0}^{m} \sigma_i.$$

Mittels Induktion zeigt man leicht, dass

$$u_{m+1} = \eta_m \frac{1}{\|A^m x_0\|} A^{m+1} x_0$$

ist. Weiter ist

$$A^m x_0 = \sum_{i=1}^{n} \alpha_i \mu_i^m z_i = \mu_1^m \{\widehat{x} + y_m\}$$

mit

$$\widehat{x} = \sum_{i=1}^{p} \alpha_i z_i, \quad y_m = \sum_{i=p+1}^{n} \alpha_i \left(\frac{\mu_i}{\mu_1}\right)^m z_i$$

und $\|y_m\| = \mathrm{O}(q^m)$. Damit folgt

$$\begin{aligned}
\sigma_{m+1} &= \operatorname{sgn}\left[(u_{m+1}\,,\, x_m)\right] = \operatorname{sgn}\left[(u_{m+1}\,,\, \sigma_m u_m)\right] \\
&= \operatorname{sgn}\left[\eta_m^2 \left(A^{m+1} x_0\,,\, A^m x_0\right)\right] \\
&= \operatorname{sgn}\left[\mu_1^{2m+1} \left(\widehat{x} + y_{m+1}\,,\, \widehat{x} + y_m\right)\right].
\end{aligned}$$

Also gibt es ein $m_0 \in \mathbb{N}$ mit $\sigma_m = \operatorname{sgn}(\mu_1)$ für alle $m \geq m_0$. Setze $\eta = \eta_{m_0}$. Dann folgt für all $m \geq m_0$

$$\operatorname{sgn}(\lambda_{m+1}) = \operatorname{sgn}(\sigma_{m+1}) = \operatorname{sgn}(\mu_1)$$

und

$$\begin{aligned}
|\lambda_{m+1}| = \|u_{m+1}\| &= \frac{\|A^{m+1} x_0\|}{\|A^m x_0\|} = |\mu_1| \, \frac{\|\widehat{x} + y_{m+1}\|}{\|\widehat{x} + y_m\|} \\
&= |\mu_1| \left[1 + \mathrm{O}(q^m)\right].
\end{aligned}$$

Das beweist die behauptete Konvergenz der λ_m. Definiere

$$x^* = \eta \left[\operatorname{sgn}(\mu_1)\right]^{m_0} \frac{1}{\|\widehat{x}\|} \widehat{x}.$$

Dann ist x^* ein Eigenvektor von A zu μ_1, und es gilt

$$\begin{aligned}
x_{m+1} &= \sigma_{m+1} \frac{1}{\|u_{m+1}\|} u_{m+1} \\
&= \eta_{m+1} \frac{1}{\|A^{m+1} x_0\|} A^{m+1} x_0 \\
&= (\operatorname{sgn}(\mu_1))^{m+1-m_0} \, \eta \left[\frac{\mu_1}{|\mu_1|}\right]^{m+1} \frac{1}{\|\widehat{x} + y_{m+1}\|} \left[\widehat{x} + y_{m+1}\right] \\
&= \eta \left[\operatorname{sgn}(\mu_1)\right]^{m_0} \frac{1}{\|\widehat{x} + y_{m+1}\|} \left[\widehat{x} + y_{m+1}\right]
\end{aligned}$$

und daher

$$\|x_{m+1} - x^*\| = \mathrm{O}(q^m). \qquad \square$$

BEMERKUNG V.3.2. *(1)* Die Voraussetzung $\alpha_i \neq 0$ für ein $1 \leq i \leq p$ ist in der Praxis aufgrund von Rundungsfehlern meistens erfüllt.
(2) Ist $|\mu_1| = \ldots = |\mu_p|$ aber $\mu_1 \neq \ldots \neq \mu_p$, so konvergiert $|\lambda_m|$ immer noch gegen $|\mu_1|$.

Falls A symmetrisch ist, kann man Algorithmus V.3.1 wesentlich beschleunigen. Die Grundidee lässt sich wie folgt beschreiben:
Wenn A symmetrisch ist, sind die Eigenvektoren paarweise orthogonal und können zusätzlich normiert werden. Dann gilt für alle $m, m' \in \mathbb{N}$

$$\begin{aligned}\left(A^m x\,,\, A^{m'} x\right) &= \left(\sum_{k=1}^{n} \alpha_k \mu_k^m z_k\,,\, \sum_{l=1}^{n} \alpha_l \mu_l^{m'} z_l\right) = \sum_{k=1}^{n} \alpha_k^2 \mu_k^{m+m'} \\ &= \mu_1^{m+m'} \sum_{k=1}^{n} \alpha_k^2 \left(\frac{\mu_k}{\mu_1}\right)^{m+m'}.\end{aligned}$$

Insbesondere folgt

$$\frac{(A^m x\,,\, A^{m+1} x)}{(A^m x\,,\, A^m x)} = \mu_1 \left[1 + O\left(\left(\frac{\mu_2}{\mu_1}\right)^{2m}\right)\right].$$

Daher sollten die Quotienten $\frac{\left(A^m x\,,\, A^{m+1} x\right)}{(A^m x\,,\, A^m x)}$ doppelt so schnell gegen den Eigenwert μ_1 konvergieren wie die Quotienten $\frac{\|A^{m+1} x\|}{\|A^m x\|}$ der Potenzmethode.

Diese Überlegungen werden durch Algorithmus V.3.2 konkretisiert.

Algorithmus V.3.2 Rayleigh-Quotienten-Iteration für die Bestimmung des betragsmäßig größten Eigenwertes einer symmetrischen Matrix

---

Gegeben: Matrix A, Startnäherung x für einen Eigenvektor mit $\|x\| = 1$, Toleranz ε, Maximalzahl für Iterationen N
Gesucht: Näherung λ_m für betragsmäßig größten Eigenwert

1: $d \leftarrow \infty$, $\lambda \leftarrow \infty$, $n \leftarrow 0$, $u \leftarrow Ax$
2: **while** $d > \varepsilon$ und $n \leq N$ und $\|u\| > \varepsilon$ **do**
3: $\quad x \leftarrow \dfrac{u}{\|u\|}$, $d \leftarrow \lambda$, $\lambda \leftarrow (x\,,\, u)$, $d \leftarrow |d - \lambda|$
4: $\quad u \leftarrow Ax$, $n \leftarrow n + 1$
5: **end while**

Der folgende Satz beweist die Konvergenz von Algorithmus V.3.2 und zeigt, dass er doppelt so schnell konvergiert wie Algorithmus V.3.1.

SATZ V.3.3 (Konvergenz der Rayleigh-Quotienten-Iteration). *Die Voraussetzungen und Bezeichnungen seien wie in Satz V.3.1. Zusätzlich*

sei A symmetrisch und die Eigenvektoren orthonormiert, d.h. $(z_i\,,\,z_j) = \delta_{ij}$ für alle $1 \le i,j \le n$. Dann gilt $|\lambda_m - \mu_1| = \mathrm{O}(q^{2m})$, wobei q wie in Satz V.3.1 ist.

Beweis. Mittels Induktion zeigt man leicht, dass

$$u_{m+1} = \frac{1}{\|A^m x_0\|} A^{m+1} x_0$$

ist. Weiter ist

$$A^m x_0 = \sum_{i=1}^{n} \alpha_i \mu_i^m z_i$$

und wegen der Orthonormierung der Eigenvektoren

$$\|A^m x_0\| = \left\{ \sum_{i=1}^{n} \alpha_i^2 \mu_i^{2m} \right\}^{\frac{1}{2}} = |\mu_1|^m \left\{ \underbrace{\sum_{i=1}^{p} \alpha_i^2}_{=a^2} + \sum_{i=p+1}^{n} \alpha_i^2 \left(\frac{\mu_i}{\mu_1} \right)^{2m} \right\}^{\frac{1}{2}}$$

$$= |\mu_1|^m \left\{ a^2 + O\left(q^{2m}\right) \right\}^{\frac{1}{2}}.$$

Damit folgt

$$\begin{aligned}
\lambda_{m+1} &= (x_m\,,\,u_{m+1}) = \frac{1}{\|u_m\|} (u_m\,,\,u_{m+1}) \\
&= \frac{1}{\|A^m x_0\|^2} \left(A^m x_0\,,\,A^{m+1} x_0 \right) \\
&= \frac{1}{\mu_1^{2m} \left[a^2 + O\left(q^{2m}\right) \right]} \sum_{i=1}^{n} \alpha_i^2 \mu_i^{2m+1} \\
&= \mu_1 \frac{[a^2 + \mathrm{O}(q^{2m+1})]}{[a^2 + \mathrm{O}(q^{2m})]} \\
&= \mu_1 [1 + \mathrm{O}(q^{2m})].
\end{aligned}$$

□

Bemerkung V.3.4 (Konvergenzbeschleunigung). Man kann die Kovergenz der Algorithmen V.3.1 und V.3.2 beschleunigen, indem man das Δ^2-*Verfahren von Aitken* [2, Aufgabe 4.4] auf die Folge $(\lambda_m)_{m\in\mathbb{N}}$ anwendet oder eine *Spektralverschiebung* durchführt, d.h. statt A die Matrix $A - \alpha I$ mit geeignetem $\alpha \in \mathbb{R}$ betrachtet.

Wendet man die Algorithmen V.3.1 und V.3.2 auf A^{-1} an, und formt die Ergebnisse geeignet um, so erhält man Verfahren zur Berechnung des betragsmäßig kleinsten Eigenwertes.

Der folgende Satz beweist die Konvergenz von Algorithmus V.3.3.

Satz V.3.5 (Konvergenz der inversen Iteration von Wielandt). *Die Matrix A habe n linear unabhängige Eigenvektoren $z_1, \ldots, z_n \in \mathbb{R}^n$ mit zugehörigen Eigenwerten $\mu_1, \ldots, \mu_n \in \mathbb{R}$. Es gelte*

$$0 < |\mu_n| = \ldots = |\mu_{n-r}| < |\mu_{n-r-1}| \le \ldots \le |\mu_1|$$

Algorithmus V.3.3 Inverse Iteration von Wielandt zur Berechnung des betragsmäßig kleinsten Eigenwertes

Gegeben: Matrix A, Startnäherung x für einen Eigenvektor mit $\|x\| = 1$, Toleranz ε, Maximalzahl für Iterationen N

Gesucht: Näherung ρ für betragsmäßig kleinsten Eigenwert und x für zugehörigen Eigenvektor

1: $d \leftarrow \infty$, $\rho \leftarrow \infty$, $n \leftarrow 0$
2: Löse das LGS $Au = x$
3: **while** $d > \varepsilon$ und $\|u\| > \varepsilon$ und $n \leq N$ **do**
4: $\quad \sigma \leftarrow \operatorname{sgn}((u\,,\,x))$, $d \leftarrow \rho$, $\rho \leftarrow \dfrac{\sigma}{\|u\|}$
5: $\quad d \leftarrow |d - \rho|$, $x \leftarrow \dfrac{\sigma u}{\|u\|}$, $n \leftarrow n + 1$
6: $\quad$ Löse das LGS $Au = x$
7: **end while**

und

$$\mu_n = \ldots = \mu_{n-r}.$$

Zudem sei in der Darstellung

$$x_0 = \sum_{i=1}^{n} \alpha_i z_i$$

mindestens ein α_i mit $n - r \leq i \leq n$ ungleich Null. Dann gibt es einen Eigenvektor y^* von A zum Eigenwert μ_n mit

$$\|x_m - y^*\| = \mathrm{O}(q^m), \quad |\rho_m - \mu_n| = \mathrm{O}(q^m)$$

mit $q = \frac{|\mu_n|}{|\mu_{n-r-1}|}$.

Beweis. Setze $\lambda_{m+1} = \frac{1}{\rho_{m+1}}$. Dann ist Algorithmus V.3.3 identisch mit Algorithmus V.3.1 angewandt auf A^{-1}. Damit folgt die Behauptung aus Satz V.3.1 angewandt auf A^{-1}. □

Sei nun A symmetrisch. Dann kann Algorithmus V.3.2 auf A^{-1} angewandt werden.

Der folgende Satz beweist die Konvergenz von Algorithmus V.3.4.

Satz V.3.6 (Konvergenz der inversen Rayleigh-Quotienten-Iteration). *Die Voraussetzungen und Bezeichnungen seien wie in Satz V.3.5. Zudem seien A symmetrisch und die Eigenvektoren orthonormiert. Dann gilt $|\rho_m - \mu_n| = \mathrm{O}(q^{2m})$, wobei q wie in Satz V.3.5 ist.*

Beweis. Setze $\lambda_{m+1} = \frac{1}{\rho_{m+1}}$. Dann ist Algorithmus V.3.4 identisch mit Algorithmus V.3.2 angewandt auf A^{-1}. Damit folgt die Behauptung aus Satz V.3.3 angewandt auf A^{-1}. □

Algorithmus V.3.4 Inverse Rayleigh-Quotienten-Iteration für die Bestimmung des betragsmäßig kleinsten Eigenwertes einer symmetrischen Matrix

Gegeben: Matrix A, Startnäherung x für einen Eigenvektor mit $\|x\| = 1$, Toleranz ε, Maximalzahl für Iterationen N
Gesucht: Näherung ρ für betragsmäßig kleinsten Eigenwert

1: $d \leftarrow \infty$, $\rho \leftarrow \infty$, $n \leftarrow 0$
2: Löse das LGS $Au = x$
3: **while** $d > \varepsilon$ und $\|u\| > \varepsilon$ und $n \leq N$ **do**
4: $\quad x \leftarrow \dfrac{u}{\|u\|}$, $d \leftarrow \rho$, $\rho \leftarrow (x\,,\,u)^{-1}$
5: $\quad d \leftarrow |d - \rho|$, $n \leftarrow n + 1$
6: $\quad$ Löse das LGS $Au = x$
7: **end while**

Zum Abschluss gehen wir noch kurz auf die Berechnung von Eigenvektoren bei bekannter Näherung für den entsprechenden Eigenwert ein.

Bemerkung V.3.7 (Berechnung eines Eigenvektors bei gegebener Näherung für einen Eigenwert). Mit einem geeigneten Shift kann Algorithmus V.3.3 auch zur Berechnung eines Eigenvektors eingesetzt werden. Genauer nehmen wir an, dass wir eine Näherung $\widetilde{\mu}$ für einen Eigenwert μ^* der Matrix A bestimmt haben und dass wir einen zugehörigen Eigenvektor $\widetilde{y}$ bestimmen wollen. Wir setzen voraus, dass $\widetilde{\mu}$ dichter an μ^* liegt als an jedem anderen Eigenwert von A, d.h. $|\widetilde{\mu} - \mu^*| < |\widetilde{\mu} - \mu|$ für alle $\mu \in \sigma(A) \setminus \mu^*$. Dann ist $\mu^* - \widetilde{\mu}$ der betragsmäßig kleinste Eigenwert der Matrix $A - \widetilde{\mu}I$ und jeder andere Eigenwert dieser Matrix hat einen echt größeren Betrag. Außerdem ist jeder Eigenvektor von $A - \widetilde{\mu}I$ zum Eigenwert $\mu^* - \widetilde{\mu}$ ein Eigenvektor von A zum Eigenwert μ^*. Daher können wir Algorithmus V.3.3 auf die Matrix $A - \widetilde{\mu}I$ anwenden und erhalten eine Folge von Vektoren x_m, die gegen einen Eigenvektor von A zum Eigenwert μ^* konvergiert. Im allgemeinen liefern schon wenige Iterationen eine hinreichend genaue Approximation für einen solchen Eigenvektor. Man beachte, dass in jeder Iteration des Verfahrens ein lineares Gleichungssystem der Form $(A - \widetilde{\mu}I)u_{m+1} = x_m$ gelöst werden muss.

V.4. Berechnung aller Eigenwerte

In diesem Abschnitt bezeichnet $A \in \mathbb{R}^{n\times n}$ stets eine Matrix mit Hessenberg-Form. Die Hessenberg-Form ist für Algorithmus V.4.1 und Satz V.4.3 nicht erforderlich, reduziert aber erheblich den Rechenaufwand für Algorithmus V.4.1.

Algorithmus V.4.1 QR-Algorithmus ohne Shift

Gegeben: Matrix A, Toleranz ε, Maximalzahl für Iterationen N
Gesucht: Näherungen für die Eigenwerte von A

1: $d \leftarrow \infty$, $n \leftarrow 0$
2: **while** $d > \varepsilon$ und $n \leq N$ **do**
3: Bestimme die QR-Zerlegung $A = QR$ von A.
4: $B \leftarrow A$, $A \leftarrow RQ$, $d \leftarrow \|B - A\|$, $n \leftarrow n + 1$
5: **end while**

LEMMA V.4.1 (Eigenschaften der QR-Zerlegung). *Sei $A = QR$ mit einer orthogonalen Matrix Q und einer oberen Dreiecksmatrix R. Definiere $A' = RQ$. Dann sind A und A' ähnlich. Falls A Hessenberg-Form hat, gilt gleiches für A'.*

BEWEIS. Es ist $A' = RQ = Q^tQRQ = Q^tAQ$. Falls A Hessenberg-Form hat, liefert Algorithmus IV.3.1 (S. 88) eine Matrix Q der Form $Q = Q_1 \cdot \ldots \cdot Q_{n-1}$ mit

$$Q_i = \begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & & \alpha & \beta & & \\ & & & \gamma & \delta & & \\ & & & & & 1 & \\ & & & & & & \ddots \\ & & & & & & & 1 \end{pmatrix},$$

wobei Q_i nur in der i-ten und $(i+1)$-ten Zeile und Spalte von der Einheitsmatrix abweicht. Dann hat RQ_1 die Gestalt

$$RQ_1 = \begin{pmatrix} ** & & & * \\ ** & & & \\ & \ddots & & \\ & & \ddots & \\ 0 & & & * \end{pmatrix}$$

und sukzessive folgt

$$RQ_1 \cdot \ldots \cdot Q_i = \begin{pmatrix} * & & & & & \\ * & \ddots & & & * & \\ & \ddots & \ddots & & & \\ & & & * & * & \\ & & & * & * & \\ & 0 & & & \ddots & \\ & & & & & * \end{pmatrix}. \qquad \square$$

Aus Lemma V.4.1 folgt, dass die Matrizen A_i alle zu A ähnlich sind und Hessenberg-Form haben. Das folgende Lemma stellt ein technisches Hilfsergebnis für den Konvergenzbeweis von Algorithmus V.4.1 bereit.

LEMMA V.4.2. *Die Matrizen A_i, R_i, Q_i seien wie in Algorithmus V.4.1. Definiere $P_i = Q_1 \cdot \ldots \cdot Q_i$ und $U_i = R_i \cdot \ldots \cdot R_1$. Dann gilt*

(1) $A_{i+1} = Q_i^t A_i Q_i$,
(2) $A_{i+1} = P_i^t A P_i$.
(3) $A^i = P_i U_i$.

BEWEIS. *ad (1):* Folgt aus der Definition von A_{i+1} und Lemma V.4.1.
ad (2): Folgt aus (1) durch Induktion.
ad (3): Aus (2) folgt $P_iA_{i+1} = AP_i$ und damit

$$P_iU_i = P_{i-1}Q_iR_iU_{i-1} = P_{i-1}A_iU_{i-1} = AP_{i-1}U_{i-1}. \qquad \square$$

Der folgende Satz beweist die Konvergenz des QR-Algorithmus.

SATZ V.4.3 (Konvergenz des QR-Algorithmus). *Die Matrix A erfülle folgende Voraussetzungen:*

(1) $\sigma(A) = \{\lambda_1, \ldots, \lambda_n\}$ *mit* $0 < |\lambda_n| < \ldots < |\lambda_1|$.
(2) *Die Matrix Y mit $A = Y^{-1}DY$ und $D = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ besitzt eine LR-Zerlegung $Y = L_YR_Y$.*

Dann gilt für die Matrizen A_i, Q_i, R_i aus Algorithmus V.4.1

$$\lim_{i\to\infty} Q_i = I \quad \textit{und} \quad \lim_{i\to\infty} R_i = \lim_{i\to\infty} A_i = \begin{pmatrix} \lambda_1 & & * \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}.$$

BEWEIS. Sei $Q_XR_X = Y^{-1}$ die QR-Zerlegung von Y^{-1}. Dann gilt

$$A^i = Y^{-1}D^iY = Q_XR_XD^iL_YR_Y = Q_XR_X(D^iL_YD^{-i})D^iR_Y.$$

Eine leichte Rechnung liefert

$$\left(D^iL_YD^{-i}\right)_{kl} = \left(\frac{\lambda_k}{\lambda_l}\right)^i (L_Y)_{kl}.$$

Also ist $D^iL_YD^{-i} = I + E_i$ mit $\lim\limits_{i\to\infty} E_i = 0$. Sei $F_i = R_XE_iR_X^{-1}$. Dann folgt $\lim\limits_{i\to\infty} F_i = 0$ und

$$A^i = Q_XR_X(I + E_i)D^iR_Y = Q_X(I + F_i)R_XD^iR_Y.$$

Sei $\widetilde{Q}_i\widetilde{R}_i = I + F_i$ die QR-Zerlegung von $I + F_i$ gemäß Algorithmus IV.3.1 (S. 88). Dann gilt $\lim\limits_{i\to\infty} \widetilde{Q}_i = \lim\limits_{i\to\infty} \widetilde{R}_i = I$. Aus Lemma V.4.2 folgt

$$P_iU_i = A^i = Q_X\widetilde{Q}_i\widetilde{R}_iR_XD^iR_Y.$$

Wegen der Eindeutigkeit[1] der QR-Zerlegung gilt daher $P_i = Q_X\widetilde{Q}_i$ und $U_i = \widetilde{R}_iR_XD^iR_Y$. Hieraus folgt

$$Q_i = P_{i-1}^tP_i = \widetilde{Q}_{i-1}^tQ_X^tQ_X\widetilde{Q}_i = \widetilde{Q}_{i-1}^t\widetilde{Q}_i \xrightarrow[i\to\infty]{} I$$

und

$$R_i = U_iU_{i-1}^{-1} = \widetilde{R}_iR_XD^iR_YR_Y^{-1}D^{-i+1}R_X^{-1}\widetilde{R}_{i-1}^{-1} = \widetilde{R}_iR_XDR_X^{-1}\widetilde{R}_{i-1}^{-1}$$

$$\xrightarrow[i\to\infty]{} R_XDR_X^{-1} = \begin{pmatrix} \lambda_1 & & * \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}$$

[1] Wir setzen hierbei voraus, dass das Vorzeichen der Diagonalelemente der oberen Dreiecksmatrix festgelegt ist. Ohne diese Annahme ist Beweis technisch etwas aufwändiger.

sowie

$$A_i = Q_i R_i \xrightarrow[i\to\infty]{} \begin{pmatrix} \lambda_1 & & * \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}. \qquad \square$$

BEMERKUNG V.4.4. *(1)* Algorithmus V.4.1 konvergiert am schnellsten für λ_1, am zweit schnellsten für λ_2 usw.
(2) Auf die Voraussetzung $|\lambda_n| > 0$ kann verzichtet werden.
(3) Auf die Voraussetzung (2) kann verzichtet werden. Die Eigenwerte erscheinen dann nicht mehr der Größe nach geordnet.
(4) Falls σ eine Näherung für den Eigenwert λ_k ist, kann man den zugehörigen Eigenvektor entweder mit den Algorithmen IV.3.1 (S. 88) und IV.3.2 (S. 89) angewandt auf das homogene LGS $(\sigma I - A)x = 0$ oder mit Algorithmus V.3.3 (S. 125) angewandt auf $\sigma I - A$ berechnen (vgl. Bemerkung V.3.7 (S. 126)).
(5) Falls A tridiagonal ist, sind alle A_i tridiagonal. Jeder Schritt von Algorithmus V.4.1 erfordert dann $O(n)$ Operationen.
(6) Die Konvergenz von Algorithmus V.4.1 kann durch *Shift-Techniken* beschleunigt werden. Dazu definiert man im k-ten Schritt mit geeignetem Shift $s_k \in \mathbb{R}$ die Matrizen Q_k, R_k durch $A_k - s_k I = Q_k R_k$ und setzt $A_{k+1} = R_k Q_k + s_k I$. Dann gilt wieder $A_{k+1} = Q_k^t A_k Q_k$. Es gibt verschiedene Strategien zur Wahl von s_k. Eine bewährte ist $s_k = (A_k)_{n,n-1}$.
(7) Durch Zusammenfassen von je zwei aufeinander folgenden QR-Schritten und geeignete Wahl von Shifts kann man mit Algorithmus V.4.1 auch Paare konjugiert komplexer Eigenwerte mit reeller Arithmetik berechnen.

V.5. Singulärwertzerlegung

Die *Singulärwerte* einer Matrix und die zugehörige *Singulärwertzerlegung* verallgemeinern die Begriffe „Eigenwert“ und „Eigenvektor“ von quadratischen auf allgemeine rechteckige Matrizen. Im Folgenden bezeichnen $\|\cdot\|$ und $|||\cdot|||$ die Euklidische Norm und die zugehörige Matrixnorm.

SATZ V.5.1 (Singulärwertzerlegung, Singulärwerte). *Zu jeder Matrix $A \in \mathbb{R}^{n\times m}$ gibt es orthogonale Matrizen $U \in \mathbb{R}^{n\times n}$ und $V \in \mathbb{R}^{m\times m}$, so dass*

$$U^t A V = \Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_k) \tag{V.5.1}$$

ist mit $k = \min\{m, n\}$ und $\sigma_1 \geq \ldots \geq \sigma_k \geq 0$. Die Darstellung (V.5.1) heißt Singulärwertzerlegung *von A, die Zahlen $\sigma_1, \ldots, \sigma_k$ heißen* Singulärwerte *von A.*

BEWEIS. Für $A = 0$ ist nichts zu zeigen. Sei also

$$\sigma = |||A||| = \max_{x\in\mathbb{R}^m\setminus\{0\}} \frac{\|Ax\|}{\|x\|} > 0$$

und $v \in \mathbb{R}^m$ ein Vektor mit $\|v\| = 1$, für den das Maximum angenommen wird. Setze $u = \frac{1}{\sigma}Av$. Dann ist $\|u\| = 1$. Die Vektoren u und v können zu orthonormalen Basen $\{u, \widetilde{u}_2, \ldots, \widetilde{u}_n\}$ und $\{v, \widetilde{v}_2, \ldots, \widetilde{v}_m\}$ des $\mathbb{R}^n$ bzw. $\mathbb{R}^m$ ergänzt werden. Wir fassen diese Vektoren als Spalten entsprechender orthogonaler Matrizen $P \in \mathbb{R}^{n\times n}$ und $Q \in \mathbb{R}^{m\times m}$ auf. Wegen $\widetilde{u}_i^t Av = \sigma \widetilde{u}_i^t u = 0$ ist

$$P^t AQ = \begin{pmatrix} \sigma & w^t \\ 0 & B \end{pmatrix}$$

mit $w \in \mathbb{R}^{m-1}$. Aus

$$\left|\left|\left| P^t AQ \right|\right|\right| = |||A||| = \sigma$$

und

$$\left\| P^t AQ \begin{pmatrix} \sigma \\ w \end{pmatrix} \right\| = \left\| \begin{pmatrix} \sigma^2 + w^t w \\ Bw \end{pmatrix} \right\| \geq \sigma^2 + w^t w = \left\| \begin{pmatrix} \sigma \\ w \end{pmatrix} \right\|^2$$

folgt

$$\sigma = \left|\left|\left| P^t AQ \right|\right|\right| \geq \frac{\left\| P^t AQ \begin{pmatrix} \sigma \\ w \end{pmatrix} \right\|}{\left\| \begin{pmatrix} \sigma \\ w \end{pmatrix} \right\|} \geq \sqrt{\sigma^2 + w^t w}.$$

Also ist $w = 0$. Damit folgt die Behauptung des Satzes durch Induktion. □

Die Singulärwertzerlegung von A ist eng verknüpft mit der Pseudoinversen A^+ von A aus Bemerkung IV.5.3 (S. 93).

Satz V.5.2 (Pseudoinverse und Singulärwertzerlegung). *Sei $U^t AV = \Sigma$ eine Singulärwertzerlegung von $A \in \mathbb{R}^{n\times m}$ wie in Satz V.5.1 und $\ell \leq k = \min\{m, n\}$ der größte Index mit $\sigma_\ell > 0$. Dann gilt für die Pseudoinverse von A*

$$A^+ = V\Sigma^+ U^t \quad \textit{mit} \quad \Sigma^+ = \operatorname{diag}\left(\frac{1}{\sigma_1}, \ldots, \frac{1}{\sigma_\ell}, 0, \ldots, 0\right) \in \mathbb{R}^{m\times n}.$$

Beweis. Man rechnet leicht nach, dass $V\Sigma^+ U^t$ die Moore-Penrose-Axiome von Bemerkung IV.5.3 (S. 93) erfüllt. □

Der folgende Satz fasst einige wichtige Eigenschaften der Singulärwertzerlegung zusammen.

Satz V.5.3 (Eigenschaften der Singulärwertzerlegung). *Sei $U^t AV = \Sigma$ eine Singulärwertzerlegung von $A \in \mathbb{R}^{n\times m}$ wie in Satz V.5.1. Bezeichne mit u_i und v_i die Spalten von U und V und mit $\ell \leq k = \min\{m, n\}$ den größten Index mit $\sigma_\ell > 0$. Dann gilt:*

(1) $Av_i = \sigma_i u_i$, $A^t u_i = \sigma_i v_i$ *für* $1 \leq i \leq k = \min\{m, n\}$.
(2) $\operatorname{Rang}(A) = \ell$.
(3) $\mathrm{R}(A) = \operatorname{span}\{u_1, \ldots, u_\ell\}$, $\ker(A) = \operatorname{span}\{v_{\ell+1}, \ldots, v_m\}$.
(4) $|||A||| = \sigma_1$.

(5) $\kappa(A) = |||A||| \, |||A^+||| = \frac{\sigma_1}{\sigma_\ell}$, *wobei* $\kappa(A)$ *die Kondition von* A *bzgl. der Euklidischen Norm* $\|\cdot\|$ *gemäß Definition IV.4.1 (S. 89) ist.*

(6) *Die strikt positiven Singulärwerte von* A *sind die Wurzeln der strikt positiven Eigenwerte von* A^tA.

BEWEIS. Die Eigenschaften (1) – (4) folgen direkt aus Satz V.5.1. Eigenschaft (5) folgt aus Satz V.5.2, Bemerkung IV.5.3 (S. 93) und Definition IV.4.1 (S. 89). Eigenschaft (6) schließlich folgt aus der Identität

$$A^tA = \left(U\Sigma V^t\right)^t \left(U\Sigma V^t\right) = V\Sigma^tU^tU\Sigma V^t = V\Sigma^t\Sigma V^t = V\Sigma^t\Sigma V^{-1}. \quad \square$$

Satz V.5.3 (6) legt es nahe, die Singulärwerte von A mit dem QR-Algorithmus V.4.1 (S. 127) angewandt auf A^tA zu berechnen. Hierbei kann der Aufwand wesentlich reduziert werden, indem die Matrix A zuerst durch geeignete Householder-Transformationen auf so genannte *Bidiagonalform* gebracht wird, da dann der QR-Algorithmus nur auf eine Tridiagonalmatrix angewandt werden muss.
Zur Beschreibung dieser Transformation nehmen wir o.E. an, dass $n > m$ ist. Gemäß Satz IV.3.3 (S. 87) gibt es eine Householder-Matrix $P_1 \in \mathbb{R}^{n\times n}$, die die erste Spalte von A auf ein Vielfaches des ersten Einheitsvektors transformiert. Ebenso gibt es eine Householder-Matrix $Q_1 \in \mathbb{R}^{(m-1)\times(m-1)}$, die den Anteil rechts von der Diagonalen des ersten Zeilenvektors von P_1A auf ein Vielfaches des ersten Einheitsvektors transformiert. Daher hat $P_1AQ_1^t$ die Struktur

$$P_1AQ_1^t = \begin{pmatrix} * & * & 0 & \dots & 0 \\ 0 & * & * & \dots & * \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ 0 & * & * & \dots & * \end{pmatrix}.$$

Indem wir diese Vorgehensweise sukzessive auf die $\begin{pmatrix} * & * & \dots & * \\ \vdots & & & \vdots \\ \vdots & & & \vdots \\ * & * & \dots & * \end{pmatrix}$-Blöcke anwenden, erhalten wir $(m-1)$ Housholder-Matrizen P_i und $(m-2)$ Householder-Matrizen Q_i, so dass mit $P = P_{m-1} \cdot \ldots \cdot P_1$ und $Q = Q_{m-2} \cdot \ldots \cdot Q_1$ gilt

$$PAQ^t = \begin{pmatrix} B \\ 0 \end{pmatrix} \quad \text{mit} \quad B = \begin{pmatrix} * & * & 0 & \dots & 0 \\ 0 & * & * & \dots & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & * & * \\ 0 & & & 0 & * \end{pmatrix}.$$

Wegen

$$B^tB = \left(PAQ^t\right)^t \left(PAQ^t\right) = QA^tP^tPAQ^t = QA^tAQ = Q^{-1}A^tAQ$$

haben B^tB und A^tA die gleichen Eigenwerte. Außerdem ist B^tB eine Tridiagonalmatrix.

Index

Literaturverzeichnis

[1] W. Dahmen and A. Reusken, *Numerik für Ingenieure und Naturwissenschaftler*, Springer, Berlin, 2006.

[2] P. Deuflhard and A. Hohmann, *Numerische Mathematik. 1*, 4th ed., de Gruyter Lehrbuch, Walter de Gruyter & Co., Berlin, 2008.

[3] R. W. Freund and R. H. W. Hoppe, *Stoer/Bulirsch: Numerische Mathematik 1*, 10th ed., Springer, Berlin, 2007.

[4] H. R. Schwarz and N. Köckler, *Numerische Mathematik*, 7th ed., Vieweg &Teubner, Wiesbaden, 2009.

[5] L. R. Scott, *Numerical Analysis*, Princeton University Press, Princeton, 2011.

[6] R. S. Varga, *Matrix Iterative Analysis*, Springer Series in Computational Mathematics, vol. 27, Springer, Berlin, 2000.