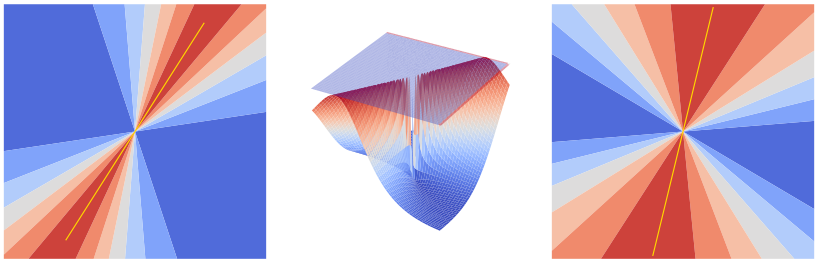
Skript zur Vorlesung

Einführung in die Numerik

Wintersemester 2022/2023



Prof. Dr. Martin Burger,
Dr. Daniel Tenbrinck, Tim Roith
Department Mathematik, AMMN

Version vom **8. Februar 2023**

Friedrich-Alexander-Universität
Naturwissenschaftliche Fakultät

Department
MATHEMATIK

Inhaltsverzeichnis

1 Direkte Lösung linearer Gleichungssysteme **1**
- 1.1 Gauss-Eliminationsverfahren . . . 2
- 1.2 LR-Zerlegung . . . 6
- 1.3 Pivotisierung mittels Permutationen . . . 9
- 1.4 Cholesky-Verfahren . . . 14

2 Grundlagen des numerischen Rechnens **19**
- 2.1 Digitale Zahldarstellung im Computer . . . 19
  - 2.1.1 Festkommazahlen . . . 20
  - 2.1.2 Gleitkommazahlen . . . 20
  - 2.1.3 Normalisierung . . . 21
  - 2.1.4 IEEE 754 Standard . . . 22
- 2.2 Rundungsfehler und Gleitkommaarithmetik . . . 23
- 2.3 Mathematische Grundlagen . . . 27
  - 2.3.1 Vektornormen . . . 27
  - 2.3.2 Matrixnormen . . . 28
  - 2.3.3 Konditionszahl einer Matrix . . . 33
  - 2.3.4 Neumannsche Reihe . . . 33
  - 2.3.5 Singulärwertzerlegung . . . 35
- 2.4 Fehlerabschätzung bei der Lösung linearer Gleichungssysteme . . . 38
- 2.5 Kondition eines Problems und Stabilität von Verfahren . . . 40
  - 2.5.1 Kondition und Verstärkungsfaktoren . . . 40
  - 2.5.2 Stabilität numerischer Verfahren . . . 43

3 Über- und unterbestimmte lineare Gleichungssysteme **49**
- 3.1 Gaußsche Normalengleichungen und Ausgleichsprobleme . . . 51
  - 3.1.1 Lineare Ausgleichsprobleme . . . 51
  - 3.1.2 Gaußsche Normalengleichung . . . 51
  - 3.1.3 Normalengleichung und SVD . . . 56
- 3.2 QR-Zerlegung für überbestimmte Systeme . . . 58
  - 3.2.1 Die QR-Zerlegung . . . 59
  - 3.2.2 QR-Verfahren mit Reflexionen und Rotationen . . . 60
- 3.3 Die Minimum-Norm-Lösung . . . 64
- 3.4 Die Moore–Penrose Inverse . . . 66

4 Iterative Lösungsverfahren für (nicht-)lineare Gleichungssysteme **69**
- 4.1 Banachscher Fixpunktsatz . . . 69

4.2 Das Newton-Verfahren . . . 73
4.2.1 Lokale Regularität der Jacobi-Matrix . . . 74
4.2.2 Konvergenz . . . 75
4.3 Konvergenzgeschwindigkeit . . . 77
4.4 Fixpunktiterationen ohne Ableitungen . . . 80
4.5 Gauss–Newton Verfahren . . . 81
4.6 Gesamt- und Einzelschrittverfahren . . . 83
4.6.1 Gesamtschrittverfahren . . . 84
4.6.2 Einzelschrittverfahren . . . 86

5 Interpolation **89**
5.1 Interpolationsformel nach Lagrange . . . 92
5.2 Vandermonde-Matrix . . . 93
5.3 Interpolationsformel nach Newton . . . 95
5.3.1 Dividierte Differenzen . . . 98
5.3.2 Dämpfung von Rundungsfehlern . . . 99
5.4 Stabilität und Fehlerabschätzung der Polynominterpolation . . . 101
5.4.1 Fehlerabschätzung . . . 101
5.4.2 Stabilität unter Störungen . . . 104
5.5 Spline-Interpolation . . . 105
5.5.1 Lineare Splinefunktionen . . . 105
5.5.2 Kubische Splinefunktionen . . . 105
5.5.3 Variationsprinzip für Splinefunktionen . . . 106
5.6 Trigonometrische Interpolation . . . 111
5.6.1 Schnelle Fouriertransformation . . . 113

6 Numerische Integration **116**
6.1 Interpolatorische Quadraturformeln . . . 118
6.1.1 Integrationsformeln nach Newton-Cotes . . . 118
6.1.2 Numerischer Integrationsfehler . . . 122
6.1.3 Stückweise interpolatorische Quadratur . . . 126
6.2 Numerische Integration mit unterschiedlichen Schrittweiten . . . 128
6.2.1 Euler-Maclaurinsche Summenformel . . . 128
6.2.2 Romberg-Verfahren . . . 137
6.3 Gauss Quadratur . . . 141

7 Eigenwertprobleme **149**
7.1 Potenzmethode und inverse Iteration . . . 150
7.2 Der Satz von Gerschgorin . . . 154
7.3 Die QR-Iteration . . . 156

Abbildungsverzeichnis

2.1 Visualisierung für Beispiel 2.30. 38
2.2 Visualisierung für Beispiel 2.33. 42
2.3 Verhalten des Heron-Verfahrens aus Beispiel 2.38 für verschiedene $d \in \mathbb{R}_0^+$ und unterschiedliche Anfangswerte. 44
2.4 Visualisierung für Beispiel 2.42. 46

3.1 Visualisierung der Daten in Beispiel 3.3. 50
3.2 Visualisierung des Ausgleichsproblems für Beispiele 3.3 und 3.5. 52
3.3 Visualsierung für Beispiel 3.15. 59
3.4 Visualisierung für Beispiel 3.20. 61
3.5 Visualisierung einer Spiegelung des Vektors a auf e_1. 62
3.6 Visualisierung für Beispiel 3.23. 64

4.1 Geometrische Illustration des Newton-Verfahrens im Eindimensionalen durch Annäherung mittels Tangenten. 75

5.1 Skalierung einer Computergrafik und eines digitalen Fotos durch Interpolation. 90

6.1 Illustration zur numerischen Approximation des bestimmten Integrals einer Funktion f mit Hilfe der Mittelpunktsregel. 117
6.2 Visualisierung der Bernoulli-Polynome B_n vom Grad 1 bis 6 auf dem Intervall $[0, 1]$ [Poly] 130

7.1 Die ersten vier Iterationen der Potenzmethode aus Beispiel 7.7 in blau. In gelb sind die Eigenvektoren skalirt mit den Eigenwerten dargestellt. Im rechten Plot ist das verhalten des Rayleigh-Quotienten abgebildet, wobei der eigentliche Wert des größten Eigenwerts in hell-rot dargestellt ist. 153
7.2 Die inverse Iteration für die Matrix aus Beispiel 7.7 mit $\lambda = -1$. 154
7.3 Die Gerschgorin-Kreise für die Matrix aus Beispiel 7.7. 156

Nomenklatur

Allgemein

$\lvert\cdot\rvert$	Die Betragsfunktion.
$\lVert\cdot\rVert$	Eine Norm.
∇	Der Gradient.
Δ	Der Laplace Operator.

Mengen und Körper

$\mathbb{N}$	Die Menge der natürlichen Zahlen.
$\mathbb{Z}$	Die Menge der ganzen Zahlen.
$\mathbb{R}$	Der Körper der reellen Zahlen.
$\mathbb{C}$	Der Körper der komplexen Zahlen.
$\mathbb{K}$	Ein allgemeiner Körper.

Matrizen und lineare Operatoren

$\mathbb{K}^{m\times n}$	Menge der $m \times n$ Matrizen im Körper $\mathbb{K}$.
I	Die Einheitsmatrix, die Dimensionen und der Körper ergeben sich jeweils aus dem Kontext.
^{T}	Für eine Matrix $A \in \mathbb{R}^{m\times n}$ bezeichnet A^T ihre Transponierte.
*	Für eine Matrix $A \in \mathbb{K}^{m\times n}$ bezeichnet A^* ihre Adjungierte. D.h. für $\mathbb{K} = \mathbb{R}$ bezeichnet A^* die Transponierte, für $\mathbb{K} = \mathbb{C}$ bezeichnet A^* die transponiert-konjugierte Matrix.
$\mathcal{N}$	Der Kern (oder Nullspace) $\mathcal{N}(A)$ einer linearen Abbildung A.
$\mathcal{R}$	Das Bild (oder der Range) $\mathcal{R}(A)$ einer linearen Abbildung A.
rank	Der Rang $\operatorname{rank}(A)$ einer Matrix $A \in \mathbb{K}^{m\times n}$.
$^+$	Für eine Matrix A bezeichnet A^+ die Moore–Penrose Inverse.

Kapitel 1

Direkte Lösung linearer Gleichungssysteme

In diesem Kapitel beschäftigen wir uns mit der direkten Lösung von linearen Gleichungssystemen (abgekürzt als LGS) der Form

$$Ax = b, \tag{1.1}$$

wobei $A \in \mathbb{R}^{n \times m}$ eine gegebene reelle $n \times m$-Matrix, $b \in \mathbb{R}^n$ ein gegebener n-dimensionaler Vektor und $x \in \mathbb{R}^n$ der unbekannte m-dimensionale Lösungsvektor ist. Diese werden wir immer in folgender Form indizieren:

$$A = \begin{pmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \dots & a_{n,n} \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}.$$

Für unsere Betrachtungen nehmen wir in diesem Kapitel vorerst an, dass $m = n$ gilt, d.h., es handelt sich bei der Matrix A um eine quadratische Matrix.

Wir suchen nun einen Vektor $x \in \mathbb{R}^n$, der die Gleichung (1.1) löst. Wir suchen also nach denjenigen Koeffizienten, mit denen sich der Vektor b als Linearkombination aus Spaltenvektoren der Matrix A darstellen lässt. Diese Koeffizienten entsprechen den Einträgen des Vektors x. In Kapitel 3 werden wir uns mit dem komplizierteren Fall von überbestimmten und unterbestimmten LGS beschäftigen. Aus der linearen Algebra ist bekannt, dass es genau dann eine eindeutige Lösung $x \in \mathbb{R}^n$ für die Gleichung (1.1) gibt, falls die Determinante $\det(A) \neq 0$ ist, wie folgendes Theorem besagt.

THEOREM 1.1.

Das LGS (1.1) ist **eindeutig lösbar**, wenn A nicht singulär ist, d.h., wenn die Determinante der Matrix A ungleich Null ist.
Allgemeiner ist das LGS (1.1) genau dann **lösbar**, wenn der Vektor b linear abhängig von den Spaltenvektoren der Matrix A ist.

Wir sehen für den Fall $\det(A) \neq 0$, dass die Spalten von A linear unabhängig sind, und damit eine Basis des $\mathbb{R}^n$, deshalb ist b immer als Linearkombination darstellbar. Für den Fall $\det(A) = 0$ lässt sich bemerken, dass das Problem einen Lösungsvektor $x \in \mathbb{R}^n$ für (1.1) zu finden *schlecht gestellt* ist, da hier keine eindeutige Lösung existiert.

Leider lässt sich der Matrix A in der Regel nicht direkt ansehen, ob sie singulär ist oder nicht, d.h., ob $\det(A) = 0$ gilt oder nicht. Würde man die Determinante von A mit der in der linearen Algebra geläufigen Leibniz-Formel

$$\det(A) \;=\; \sum_{\sigma \in S_n} \left(\operatorname{sgn}(\sigma) \prod_{i=1}^{n} a_{i,\sigma(i)} \right) \tag{1.2}$$

berechnen, so stellt man fest, dass der Rechenaufwand für wachsende $n \in \mathbb{N}$ sehr schnell ansteigt, was an der großen Zahl an kombinatorischen Möglichkeiten liegt, die es gibt um die Einträge von A mit Hilfe von Permutationen σ der symmetrischen Gruppe S_n zu permutieren. In diesem Zusammenhang bezeichnet $\operatorname{sgn}(\sigma)$ das Signum der Permutation σ, welches $+1$ für gerade und -1 für ungerade Permutationen ist. Auf Grund der Anzahl der möglichen Permutation wird klar, dass man für gegebenes $n \in \mathbb{N}$ genau $n!$ Summanden in (1.2) aufsummieren muss, um die Determinante zu prüfen.

Um im Allgemeinen Aussagen über den Rechenaufwand eines Algorithmus machen zu können, benötigen wir folgende Definition.

DEFINITION 1.2: Rechenaufwand und Landau-Notation.

Um den **Rechenaufwand** eines Algorithmus anzugeben, zählt man die benötigte Anzahl an Additionen, Multiplikationen und Vergleichen. Angegeben wird der Rechenaufwand häufig in *Floating Point Operations* (abgekürzt als FLOPs), bei welchen historisch bedingt eine Addition und eine Multiplikation zu einer Rechenoperation zusammengefasst werden. Für moderne Mikroprozessorarchitekturen ist dies natürlich eine sehr starke Vereinfachung. Weitere Informationen zur Berechnung von FLOPS finden sich unter [FLOPS].

Um den Rechenaufwand eines Algorithmus in eine Klasse von Funktionen einzuordnen verwendet man häufig die **Landau-Notation** wie folgt. Seien $f, g \colon \mathbb{R} \to \mathbb{R}$ zwei reelle Funktionen. Wir schreiben $f \in \mathcal{O}(g)$, d.h., g ist eine asymptotisch obere Schranke für f, wenn gilt:

$$\exists C > 0, \exists x_0 > 0, \forall x > x_0 \colon |f(x)| \leq C|g(x)|.$$

BEISPIEL 1.3: Elementsuche.

Die Anzahl der Vergleiche, die man beim Suchen eines Werts in einer unsortierten Menge aus n Elementen benötigt liegt in $\mathcal{O}(n)$, da man jedes Element einzeln überprüfen muss. Ist die Menge bereits sortiert, so liegt die Anzahl der benötigten Vergleichsoperationen mit dem Intervallhalbierungsverfahren in $\mathcal{O}(\log n)$.

Für große n ist immer nur die führende Ordnung beim Aufwand interessant, deshalb ist es nicht so wichtig die weiteren Ordnungen genau zu bestimmen. So werden wir für ein Verfahren, dass $2n^2 + 4n + 3$ Operationen benötigt, als Aufwand typischerweise $2n^2 + \mathcal{O}(n)$ angeben.

1.1 Gauss-Eliminationsverfahren

Bevor wir den allgemeinen Fall eines LGS wie in Gleichung (1.1) betrachten, wollen wir zwei Spezialfälle diskutieren, in denen sich die Determinante von A direkt ablesen lässt. Dies bedeutet wiederum, dass man in diesen Fällen direkt eine Aussage über die Lösbarkeit des LGS

machen kann. Der einfachste zu betrachtende Fall ist, wenn die Matrix A eine **Diagonalmatrix** ist, wenn also gilt $a_{i,j} = 0$ für alle $i \neq j$:

$$A = \begin{pmatrix} a_{1,1} & & 0 \\ & \ddots & \\ 0 & & a_{n,n} \end{pmatrix} =: D.$$

In diesem Fall weiß man, dass

$$\det(A) = \prod_{i=1}^{n} a_{i,i} \tag{1.3}$$

gilt und somit $\det(A) \neq 0 \Leftrightarrow a_{i,i} \neq 0$ für alle $i = 1, \ldots, n$, d.h., das LGS in Gleichung (1.1) ist genau dann lösbar, wenn alle Diagonaleinträge der Matrix D ungleich Null sind. Weiterhin lässt sich eine Lösung $x \in \mathbb{R}^n$ in diesem Fall sehr einfach bestimmen, nämlich:

$$x_i = \frac{b_i}{a_{i,i}}, \quad \forall i = 1, \ldots, n. \tag{1.4}$$

Da man zur Bestimmung des Lösungsvektors x in (1.4) nur n Divisionen benötigt, liegt der Rechenaufwand bei $\mathcal{O}(n)$.

Widmen wir uns nun dem zweiten Fall unserer Diskussion, nämlich wenn die Matrix A die Gestalt einer **rechten, obere Dreiecksmatrix** besitzt, d.h. wenn $a_{ij} = 0$ für alle $i > j$ gilt:

$$A = \begin{pmatrix} a_{1,1} & \ldots & a_{1,n} \\ 0 & \ddots & \vdots \\ 0 & 0 & a_{n,n} \end{pmatrix} =: R.$$

Wie im Fall der Diagonalmatrix lässt sich die Determinante hier analog als Produkt der Diagonaleinträge nach Gleichung (1.3) berechnen und es gelten die gleichen Aussagen zur Existenz einer Lösung. Die konkrete Berechnung des Lösungsvektors $x \in \mathbb{R}^n$ stellt sich jedoch, verglichen mit dem ersten Spezialfall, als etwas schwieriger heraus. Betrachten wir das ursprüngliche LGS in Gleichung (1.1) für den Fall einer rechten, oberen Dreiecksmatrix $A = R$ so ergeben sich die folgenden gestaffelten Gleichungen

$$\begin{aligned} a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n-1}x_{n-1} + a_{1,n}x_n &= b_1 \\ a_{2,2}x_2 + \ldots + a_{2,n-1}x_{n-1} + a_{2,n}x_n &= b_2 \\ &\vdots \\ a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n &= b_{n-1} \\ a_{n,n}x_n &= b_n \end{aligned} \tag{1.5}$$

Für den Fall, dass A nicht singulär ist, d.h., dass alle Diagonalelemente der Matrix $A = R$ ungleich Null sind, lässt sich direkt ein Verfahren zur Bestimmung der unbekannten Lösung $x \in \mathbb{R}^n$ durch **Rückwärtseinsetzen** angeben. Hierzu beginnt man mit dem letzten Element x_n des Vektors x und setzt die Lösung sukzessive in die gestaffelten Gleichungen (1.5) ein. Hieraus ergibt sich das folgende Lösungsverfahren.

ALGORITHMUS 1.4: Rückwärtseinsetzen.

$$\begin{aligned} x_n &= b_n/a_{n,n} \\ x_{n-1} &= (b_{n-1} - a_{n-1,n}x_n)/a_{n-1,n-1} \\ &\vdots \\ x_1 &= (b_1 - a_{1,2}x_2 - \ldots - a_{1,n}x_n)/a_{1,1} \end{aligned} \tag{1.6}$$

BEMERKUNG 1.5. In der ersten Gleichung von (1.6) benötigt man eine einzige Division zur Bestimmung von x_n. Mit jeder weiteren Gleichung kommt eine Subtraktion und eine Multiplikation hinzu (wir fassen diese zwei Schritte nach 1 zu einem FLOP zusammen). Dementsprechend lässt sich der Rechenaufwand für das Rückwärtseinsetzen mit

$$1 + \sum_{i=2}^{n} i \;=\; \sum_{i=1}^{n} i \;=\; \frac{n(n+1)}{2} = \frac{n^2}{2} + \frac{n}{2},$$

also in $\mathcal{O}(n^2)$ bzw. $\frac{1}{2}n^2 + \mathcal{O}(n)$ angeben. △

Sei A nun eine *beliebige* nichtsinguläre Matrix. Wir versuchen im Folgenden mit Hilfe des **Gauss'schen Eliminationsverfahren** durch geschickte Rechenoperationen eine Rückführung auf den oben diskutierten Fall der rechten, oberen Dreiecksmatrix zu erhalten. Betrachten wir hierzu zunächst wieder die gestaffelten Gleichungen für den allgemeinen Fall

$$\begin{aligned} a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n-1}x_{n-1} + a_{1,n}x_n &= b_1 \\ &\vdots \\ a_{i,1}x_1 + a_{i,2}x_2 + \ldots + a_{i,n-1}x_{n-1} + a_{i,n}x_n &= b_i \\ &\vdots \\ a_{n,1}x_n + a_{n,2}x_2 + \ldots + a_{n,n-1}x_{n-1} + a_{n,n}x_n &= b_n \end{aligned} \tag{1.7}$$

Die Idee beim Eliminationsverfahren ist es im $j-$ten Schritt die unbekannte Variable x_j aus allen darunter liegenden Gleichungen zu entfernen, sodass nur Nullen unterhalb des j-ten Diagonaleintrags bleiben und so sukzessive eine rechte, obere Dreiecksmatrix entsteht. Obwohl schon aus der linearen Algebra bekannt, geben wir im Folgenden den Algorithmus des Gauss-Eliminationsverfahrens ausführlich an.

ALGORITHMUS 1.6: Gauss-Eliminationsverfahren.

Wir nehmen zunächst an, dass die Matrix A nicht singulär ist. Man beginnt häufig **im ersten Schritt** mit der Elimination der Variablen x_1 aus den Gleichungen $i = 2, \ldots, n$ in (1.7). Hierzu bestimmen wir für jede Gleichung den passenden Vorfaktor, der nötig ist, damit eine Differenz mit der ersten Gleichung zu einer Null unterhalb der Diagonalen führt, d.h., wir setzen für $i = 2, \ldots, n$

$$l_{i,1} \;=\; \frac{a_{i,1}}{a_{1,1}}. \tag{1.8}$$

An dieser Stelle wollen wir annehmen, dass $a_{1,1} \neq 0$ ist. Diesen Vorfaktor multiplizieren wir mit der ersten Gleichung und subtrahieren diese von der i-ten Gleichung. Im Ergeb-

nis verschwindet nun die unbekannte Variable x_1, da wir die Vorfaktoren $l_{i,1}$ in (1.9) entsprechend gewählt haben. Wir erhalten somit die folgenden neuen Gleichungen für $i = 2, \ldots, n$:

$$0 \cdot x_1 + (a_{i,2} - l_{i,1}a_{1,2})x_2 + \ldots + (a_{i,n} - l_{i,1}a_{1,n})x_n \;=\; b_i - l_{i,1}b_1.$$

Wir können das verbleibende System von n Gleichungen nun wie folgt schreiben

$$\begin{aligned}
a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n-1}x_{n-1} + a_{1,n}x_n &= b_1 \\
&\vdots \\
a^{(2)}_{2,2}x_2 + \ldots + a^{(2)}_{2,n-1}x_{n-1} + a^{(2)}_{2,n}x_n &= b^{(2)}_2 \\
&\vdots \\
a^{(2)}_{i,2}x_2 + \ldots + a^{(2)}_{i,n-1}x_{n-1} + a^{(2)}_{i,n}x_n &= b^{(2)}_i \\
&\vdots \\
a^{(2)}_{n,2}x_2 + \ldots + a^{(2)}_{n,n-1}x_{n-1} + a^{(2)}_{n,n}x_n &= b^{(2)}_n
\end{aligned}$$

mit

$$a^{(2)}_{i,k} \coloneqq a^{(1)}_{i,k} - l_{i,1}a^{(1)}_{1,k}, \quad b^{(2)}_i \coloneqq b^{(1)}_i - l_{i,1}b^{(1)}_1 \quad \text{für} \quad i,k = 2, \ldots, n.$$

Hierbei haben wir implizit $a^{(1)}_{i,k} = a_{i,k}$ und $b^{(1)}_i = b_i$ gesetzt.
Dieses Vorgehen lässt sich nun für den ***j*-ten Schritt** des Verfahrens für $j = 2, \ldots, n-1$ wie folgt verallgemeinern. Wir setzen nun für die relevanten Gleichungen $i = j+1, \ldots, n$ die Vorfaktoren als

$$l_{i,j} \;\coloneqq\; \frac{a^{(j)}_{i,j}}{a^{(j)}_{j,j}}. \tag{1.9}$$

Hierbei nehmen wir wiederum an, dass $a^{(j)}_{j,j} \neq 0$ für $j = 2, \ldots, n$ gilt. Diesen Vorfaktor multipliziert man wieder mit der j-ten Gleichung und subtrahiert diese von der i-ten Gleichung für $i = j + 1, \ldots, n$. Im Ergebnis verschwindet nun die unbekannte Variable x_j. Man erhält damit die folgenden neuen Gleichungen für $i = j + 1, \ldots, n$:

$$0 \cdot x_1 + \ldots + 0 \cdot x_j + (a^{(j)}_{i,j+1} - l_{i,j}a^{(j)}_{j,j+1})x_{j+1} + \ldots + (a^{(j)}_{i,n} - l_{i,j}a^{(j)}_{j,n})x_n = b^{(j)}_i - l_{i,j}b^{(j)}_j.$$

Wir können dieses System von $(n - j)$ Gleichungen nun wie folgt schreiben

$$\begin{aligned}
a^{(j+1)}_{j+1,j+1}x_{j+1} + \ldots + a^{(j+1)}_{j+1,n-1}x_{n-1} + a^{(j+1)}_{j+1,n}x_n &= b^{(j+1)}_{j+1} \\
&\vdots \\
a^{(j+1)}_{n,j+1}x_{j+1} + \ldots + a^{(j+1)}_{n,n-1}x_{n-1} + a^{(j+1)}_{n,n}x_n &= b^{(j+1)}_n
\end{aligned}$$

mit

$$a^{(j+1)}_{i,k} \coloneqq a^{(j)}_{i,k} - l_{i,j-1}a^{(j)}_{j-1,k}, \quad b^{(j+1)}_i \coloneqq b^{(j)}_i - l_{i,j-1}b^{(j)}_1 \quad \text{für} \quad i,k = j+1, \ldots, n.$$

Nach n Schritten des oben beschriebenen Verfahrens hat man die ursprüngliche Matrix A in eine obere, rechte Dreiecksform überführt und man erhält **nach dem letzten Schritt** in der letzten Zeile eine Gleichung mit nur einer unbekannten Variable x_n wie folgt

$$a_{n,n}^{(n)} x_n \;=\; b_n^{(n)}$$

mit der oben beschriebenen Notation.
Nun lässt sich der unbekannte Vektor $x \in \mathbb{R}^n$ durch Rückwärtseinsetzen berechnen, wie in Algorithmus 1.6 beschrieben.

BEMERKUNG 1.7. Der Rechenaufwand des Gauss-Eliminationsverfahrens in Algorithmus 1.6 liegt in $\mathcal{O}(n^3)$ bzw. genauer $\frac{1}{3}n^3 + \mathcal{O}(n^2)$. △

BEISPIEL 1.8.

Sei

$$A \;=\; \begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \\ 3 & 3 & 1 \end{pmatrix}, \quad b \;=\; \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix}.$$

Das Gauss-Eliminationsverfahren in Algorithmus 1.6 liefert einem die rechte, obere Dreiecksmatrix $R := A^{(n)} \in \mathbb{R}^{3\times 3}$ und den modifizierten Vektor $b^{(n)} \in \mathbb{R}^3$ mit

$$R \;=\; \begin{pmatrix} 1 & 2 & 3 \\ 0 & -1 & -2 \\ 0 & 0 & -2 \end{pmatrix}, \quad b^{(n)} \;=\; \begin{pmatrix} 2 \\ 0 \\ -6 \end{pmatrix}.$$

Durch Rückwärtseinsetzen mit Algorithmus 1.6 erhält man den Lösungsvektor $x = (5, -6, 3)^T$.

1.2 LR-Zerlegung

Wir betrachten im Folgenden das in Kapitel 1.1 hergeleitete Gauss-Eliminationsverfahren in einer alternativen, für uns interessanten Form. Hierfür benötigen wir vorab eine Definition von rechten, oberen und linken, unteren Dreiecksmatrizen, wie sie bereits bei der Gauss-Elimination auftauchten.

DEFINITION 1.9: Dreiecksmatrizen und Normierung.

Wir nennen eine $n \times n$ Matrix L **linke, untere Dreiecksmatrix**, wenn $l_{i,j} = 0$ für alle $i < j$ gilt. Analog definieren wir eine $n\times n$ Matrix R als **rechte, obere Dreiecksmatrix**, wenn $r_{i,j} = 0$ für alle $i > j$ gilt. Wir nennen eine Dreiecksmatrix **normiert**, falls auf der Hauptdiagonalen nur Einsen stehen.

THEOREM 1.10.

Die invertierbaren linken, unteren (und auch rechten, oberen) Matrizen bilden eine Gruppe bezüglich der Multiplikation.

Wir definieren im Folgenden eine nützliche Menge von linken, unteren Dreiecksmatrizen, die spezielle Eigenschaften haben.

DEFINITION 1.11: Elementarmatrizen.

Sei $L_j \in \mathbb{R}^{n\times n}$ eine linke, untere Dreiecksmatrix. Wir nennen L_j eine **Elementarmatrix** (oder auch **Frobenius-Matrix**), wenn sie nur in der j-ten Spalte von der Einheitsmatrix $I_n \in \mathbb{R}^{n\times n}$ abweicht und folgende Gestalt hat:

$$L_j = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{j+1,j} & & \\ & & \vdots & \ddots & \\ & & -l_{n,j} & & 1 \end{pmatrix} \tag{1.10}$$

LEMMA 1.12: Eigenschaften von Elementarmatrizen.

Für Elementarmatrizen $L_j \in \mathbb{R}^{n\times n}$ der Gestalt (1.10) lassen sich folgende nützliche Eigenschaften zeigen.

(i) Bei Anwendung von L_j auf einen beliebigen Vektor $a \in \mathbb{R}^n$ werden nur die Einträge a_i von $i = j+1, \ldots, n$ des Vektors a verändert, d.h.,

$$L_j \begin{pmatrix} a_1 \\ \vdots \\ a_j \\ a_{j+1} \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} a_1 \\ \vdots \\ a_j \\ a_{j+1} - l_{j+1,j}a_j \\ \vdots \\ a_n - l_{n,j}a_j \end{pmatrix}$$

(ii) Bei Anwendung von L_j auf eine beliebige Matrix $A \in \mathbb{R}^{n\times n}$ erhält man das Resultat L_jA, indem man das $l_{i,j}$-fache der j-ten Zeile von A von der i-ten Zeile von A subtrahiert für $i = j+1, \ldots, n$. Das entspricht genau dem j-ten Schritt des Gauss-Eliminationsverfahrens in Algorithmus 1.6.

(iii) Aus Eigenschaft (i) folgt direkt, dass man die Inverse L_j^{-1} von L_j durch Umkehrung des Vorzeichens der Einträge $l_{i,j}$ für $i = j+1, \ldots, n$ erhält.

(iv) Das Produkt zweier Elementarmatrizen L_j, L_k für $j < k$ lässt sich durch Überlagern der beiden Matrizen berechnen, d.h.,

$$L_j L_k = \begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & -l_{j+1,j} & \ddots & & & \\ & & \vdots & & 1 & & \\ & & \vdots & & -l_{k+1,k} & \ddots & \\ & & \vdots & & \vdots & & \ddots \\ & & -l_{n,j} & & -l_{n,k} & & 1 \end{pmatrix}.$$

Wir wollen nun eine Zerlegung einer gegebenen Matrix $A \in \mathbb{R}^{n\times n}$ definieren.

DEFINITION 1.13: LR-Zerlegung.

Wir definieren die **LR-Zerlegung** einer Matrix $A \in \mathbb{R}^{n\times n}$ als ein Produkt einer linken, unteren Dreiecksmatrix $L \in \mathbb{R}^{n\times n}$ und einer rechten, oberen Dreiecksmatrix $R \in \mathbb{R}^{n\times n}$, die, wenn sie existiert, folgende Gestalt hat

$$A = LR. \tag{1.11}$$

Nehmen wir nun an, dass die Matrix A weiterhin nicht singulär ist und eine LR-Zerlegung von A existiert. Mit Hilfe der Eigenschaften von Elementarmatrizen aus Lemma 1.12 und dem Zusammenhang zum Gauss-Eliminationsverfahren in Algorithmus 1.6 können wir nun schreiben:

$$L^{-1}Ax = L_{n-1}\dots L_1 Ax = Rx = y = L_{n-1}\dots L_1 b = L^{-1}b, \tag{1.12}$$

wobei jede der Elementarmatrizen L_j für $j = 1,\dots,n-1$ den j-ten Schritt des Gauss-Eliminationsverfahrens durchführt. Das bedeutet, dass wir die Matrix A in eine rechte, obere Dreiecksmatrix R überführen können, indem wir sukzessive Elementarmatrizen L_j mit Einträgen $l_{i,j}$ wie in (1.9) definiert von links multiplizieren für $j = 1,\dots,n-1$. Aus der ursprünglichen rechten Seite $b \in \mathbb{R}^n$ des LGS in (1.1) wird durch diese Operationen die neue rechte Seite $y \in \mathbb{R}^n$.

Sei nun für eine Matrix $A \in \mathbb{R}^{n\times n}$ eine LR-Zerlegung $A = LR$ gegeben. Dann lässt sich das LGS $Ax = b$ äquivalent schreiben als das Lösen der beiden folgenden Probleme:

$$Ly = b \quad \text{und} \quad Rx = y, \qquad \text{für } x, y, b \in \mathbb{R}^n. \tag{1.13}$$

Das heißt, man bestimmt zuerst die rechte Seite y durch Vorwärtseinsetzen von $Ly = b$ und löst dann $Rx = y$ durch Rückwärtseinsetzen nach Algorithmus 1.6. Der Rechenaufwand beträgt nach Bemerkung 1.1 in diesem Fall $n^2 + \mathcal{O}(n)$ FLOPS.

BEMERKUNG 1.14. Für eine invertierbare $n \times n$ Matrix A lässt sich die LR-Zerlegung auch zur Berechnung der Inversen A^{-1} von A nutzen, denn es gilt:

$$A = LR \quad \Rightarrow \quad A^{-1} = R^{-1}L^{-1},$$

wobei die Inversen L^{-1} und R^{-1} sehr einfach berechnet werden können (siehe Übungsaufgaben zur Vorlesung!). △

BEMERKUNG 1.15. Wir haben oben angenommen, dass bereits eine LR-Zerlegung von der Matrix A bekannt ist. Die Aufwand zur Bestimmung einer LR-Zerlegung hat sich jedoch nicht verringert und liegt weiterhin in $\mathcal{O}(n^3)$. Dennoch ist obige Betrachtung in Fällen interessant in denen man $m \in \mathbb{N}, m > 1$ LGS der Form

$$Ax_k = b_k, \quad k = 1, \ldots, m$$

lösen muss, d.h., wenn die Matrix A in allen m Problemen gleich ist. In diesen Fällen muss nur eine LR-Zerlegung der Matrix A bestimmt werden, welche dann zur Lösung aller m Probleme effizient genutzt werden kann. △

1.3 Pivotisierung mittels Permutationen

Bisher haben wir immer angenommen, dass $a_{j,j}^{(j)} \neq 0$ auf der Hauptdiagonalen gilt im j-ten Schritt des Gauss-Eliminationsverfahrens für $j = 1, \ldots, n-1$. Falls nun durch eine der vorherigen Operationen des Verfahrens eine Null auf der Hauptdiagonalen entsteht, so wird der Algorithmus abgebrochen, da man bei der Berechnung der Elemente $l_{i,j} = a_{i,j}^{(j)} / a_{j,j}^{(j)}$ nicht durch Null teilen kann. Das folgende Beispiel illustriert, was passiert wenn diese Bedingung verletzt ist.

BEISPIEL 1.16.

Wir suchen eine Lösung $x \in \mathbb{R}^2$ des LGS $Ax = b$ für eine 2×2 Matrix

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix},$$

und eine beliebige rechte Seite $b \in \mathbb{R}^2$. Man sieht ein, dass A invertierbar ist mit

$$A^{-1} = \begin{pmatrix} -1 & 1 \\ 1 & 0 \end{pmatrix}.$$

Dennoch scheitert das Gauss-Elimininationsverfahren in Algorithmus 1.6 bereits im ersten Schritt (da wir nicht durch Null teilen können) und es existiert keine LR-Zerlegung $A = LR$ mit L linker, unterer und R rechter, oberer Dreieckmatrix. Der Grund hierfür ist das Matrixelement $a_{11} = 0$ auf der Hauptdiagonalen.

Um das Problem in Beispiel 1.16 zu lösen führen wir im Folgenden das Konzept der **Pivotisierung** der Matrix A ein.

DEFINITION 1.17: Pivotelement.

Sei $j \in \{1, \ldots, n-1\}$, dann bezeichnen wir das Hauptdiagonalelement $a_{j,j}^{(j)}$ der Matrix $A^{(j)}$ im j-ten Schritt des Gauss-Eliminationsverfahrens als **Pivotelement**.

BEMERKUNG 1.18. Für die Gauss-Elimination ist nicht nur der Fall eines Pivotelements $a_{j,j}^{(j)} = 0$ problematisch, sondern auch der Fall eines *sehr kleinen Pivotelements* $0 < a_{j,j}^{(j)} < < 1$, da man bei der Berechnung der Elemente $l_{i,j}$ in (1.9) große Zwischenergebnisse erhält. Wir werden in Kapitel 2 näher analysieren, warum solche Fälle zu großen Ungenauigkeiten beim numerischen Rechnen führen. Aus diesem Grund lohnt es sich häufig den im folgenden beschriebenen Algorithmus der Spaltenpivotsuche in jedem Schritt durchzuführen. △

Die Idee der Pivotisierung ist recht einfach und soll im folgenden Algorithmus beschrieben werden.

ALGORITHMUS 1.19: Spaltenpivotsuche.

Wir nehmen zunächst an, dass wir uns im j-ten Schritt des Gauss-Eliminationsverfahrens in Algorithmus 1.6 befinden und für das aktuelle Pivotelement gelte $a_{j,j}^{(j)} = 0$. Das LGS hat in diesem Schritt dann die folgende Form:

$$A^{(j)}x = \begin{pmatrix} a_{1,1}^{(1)} & * & * & \ldots & * \\ 0 & \ddots & * & \ldots & * \\ 0 & 0 & a_{j,j}^{(j)} = 0 & \ldots & a_{j,n}^{(j)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & a_{n,j}^{(j)} & \ldots & a_{n,n}^{(j)} \end{pmatrix} x = \begin{pmatrix} b_1^{(1)} \\ \vdots \\ b_j^{(j)} \\ \vdots \\ b_n^{(j)} \end{pmatrix}.$$

Bei der sogenannten **Spaltenpivotsuche** versucht man nun durch Zeilenvertauschungen zwischen den Zeilen j bis n ein günstiges Pivotelement zu finden. Hierbei wird in der Regel das *betragsmäßig größte Element* $a_{i,j}^{(j)}$, mit $i > j$ ausgewählt, so dass man im j-ten Schritt des Gauss-Eliminationsverfahrens nun das folgende äquivalente Problem löst:

$$\tilde{A}^{(j)}x = \begin{pmatrix} a_{1,1}^{(1)} & * & * & \ldots & * \\ 0 & \ddots & * & \ldots & * \\ 0 & 0 & a_{i,j}^{(j)} \neq 0 & \ldots & a_{i,n}^{(j)} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & a_{j,j}^{(j)} = 0 & \ldots & a_{j,n}^{(j)} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & a_{n,j}^{(j)} & \ldots & a_{n,n}^{(j)} \end{pmatrix} x = \begin{pmatrix} b_1^{(1)} \\ \vdots \\ b_i^{(j)} \\ \vdots \\ b_j^{(j)} \\ \vdots \\ b_n^{(j)} \end{pmatrix}.$$

Hierbei erhält man die Matrix $\tilde{A}^{(j)}$ aus $A^{(j)}$ durch Vertauschung der j-ten und i-ten Zeile für $i > j$. Dieses Vorgehen wird im Allgemeinen auch **Pivotisierung** genannt.

BEMERKUNG 1.20. Der Rechenaufwand für die Spaltenpivotsuche im j-ten Schritt des Gauss-Eliminationsvefahrens liegt in $\mathcal{O}(n)$ und ist somit vernachlässigbar gegenüber dem Rechenaufwand des gesamten Verfahrens. △

THEOREM 1.21.

Ist die Matrix A invertierbar, so lässt sich in jedem Schritt des Gauss-Eliminationsverfahrens eine Spaltenpivotsuche durchführen, so dass man ein Pivotelement $a_{i,j}^{(j)}, i \geq j$ erhält, das nicht Null ist.

Beweis. Nehmen wir an, dass im j-ten Schritt des Gauss-Eliminationsverfahrens kein Pivotelement $a_{i,j}^{(j)} \neq 0, i > j$ existiert. Das bedeutet, dass die erste Spalte der Matrix $(a_{k,k}^{(j)})_{k=j,\dots,n}$ nur aus Nullen besteht und somit nicht vollen Rang hat. Daraus folgt mit Argumenten der linearen Algebra, dass $\det((a_{k,k}^{(j)})_{k=j,\dots,n}) = 0$ gilt. Andererseits gilt jedoch auch:

$$0 \neq \det(A) = \prod_{i=1}^{j-1} a_{i,i} \cdot \det((a_{k,k}^{(j)})_{k=j,\dots,n}).$$

Dies ist ein klarer Widerspruch zur Annahme, woraus die Behauptung folgt. □

Die gewünschte Zeilenvertauschung bei der Pivotisierung im j-ten Schritt des Gauss-Eliminationsverfahrens lässt sich formal auch durch die Multiplikation mit einer $n \times n$ Permutationsmatrix P_σ beschreiben, wobei $\sigma \in S_n$ die entsprechende Permutation aus der symmetrischen Gruppe ist.

DEFINITION 1.22: Permutationsmatrix.

Sei $\sigma \in S_n$ eine Permutation der Zahlen 1 bis n mit $\sigma(\{1,\dots,n\}) = \{i_1,\dots,i_n\}$ und sei e_j der j-te Einheitsvektor. Dann lässt sich mit Hilfe von σ eine **Permutationsmatrix** P_σ definieren mit

$$P = \begin{pmatrix} e_{\sigma(1)}^T \\ \vdots \\ e_{\sigma(n)}^T \end{pmatrix} = \begin{pmatrix} e_{i_1}^T \\ \vdots \\ e_{i_n}^T \end{pmatrix}.$$

Das heißt es handelt sich um eine Permutation der Einheitsmatrix, bei der die Zeilen gemäß σ vertauscht wurden.

Für die oben eingeführte Permutationsmatrix lassen sich folgende mathematische Eigenschaften beobachten.

LEMMA 1.23: Eigenschaften von Permutationsmatrizen.

Sei $\sigma \in S_n$ eine Permutation und sei $P_\sigma \in \mathbb{R}^{n\times n}$ die zugehörige Permutationsmatrix. Dann gelten die folgenden Rechenregeln:

(i) Die Anwendung auf einen Vektor $b \in \mathbb{R}^n$ permutiert die Elemente gemäß $\sigma \in S_n$, d.h.,

$$P_\sigma \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} = \begin{pmatrix} b_{i_1} \\ \vdots \\ b_{i_n} \end{pmatrix}.$$

(ii) Die Multiplikation $P_\sigma A$ von links an eine $n \times n$ Matrix A permutiert deren Zeilen gemäß $\sigma \in S_n$.

(iii) Die Multiplikation AP_σ^T von rechts an eine $n \times n$ Matrix A permutiert deren Spalten gemäß $\sigma \in S_n$.

(iv) Jede Permutationsmatrix P_σ ist invertierbar und es gilt $P_\sigma^{-1} = P_\sigma^T$. Da insbesondere gilt $P_\sigma P_\sigma^T = P_\sigma^T P_\sigma = I_n$ folgt, dass P_σ *unitär* ist.

Das folgende Beispiel soll die Gestalt und Wirkung von Permutationsmatrizen für einen dreidimensionalen Vektor verdeutlichen.

BEISPIEL 1.24: Permutationsmatrix.

Sei $\sigma\{1,2,3\} = \{2,3,1\}$, dann lässt sich die zugehörige Permutationsmatrix P_σ schreiben als

$$P_\sigma = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix},$$

und bei Anwendung auf einen Vektor $b \in \mathbb{R}^3$ gilt

$$P_\sigma b = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} b_2 \\ b_3 \\ b_1 \end{pmatrix}.$$

Mit Hilfe der oben eingeführten Elementar- und Permutationsmatrizen lässt sich nun das Gauss-Eliminationsverfahren mit Pivotisierung für eine $n \times n$ Matrix A schreiben als

$$L_{n-1}P_{n-1}\dots L_2P_2L_1P_1Ax = Rx = y = L_{n-1}P_{n_1}\dots L_2P_2L_1P_1b. \tag{1.14}$$

Um diese Form der rekursiven Linksmultiplikation mit Elementar- und Permutationsmatrizen in eine geschlossene Form einer LR-Zerlegung zu überführen benötigen wir noch das folgende Lemma.

LEMMA 1.25: Wirkung auf Elementarmatrizen.

Sei $j \in \{1,\dots,n-1\}$, $L_j \in \mathbb{R}^{n\times n}$ eine Elementarmatrix und sei $P_\sigma \in \mathbb{R}^{n\times n}$ eine Permutationsmatrix, die nur Zeilen mit Index $i > j$ vertauscht, d.h., $\sigma(\{1,\dots,j\}) = \{1,\dots,j\}$. Dann gilt

$$P_\sigma L_j P_\sigma^T = L'_j,$$

wobei L'_j die gleiche Form wie L_j hat, jedoch mit gemäß der Permutation σ vertauschten Elementen $l'_{k,j} = l_{i_k,j}$. Aus der Eigenschaft, dass P_σ unitär ist, folgt folgende Vertauschungsrelation:

$$P_\sigma L_j = L'_j P_\sigma. \tag{1.15}$$

Folgendes Beispiel soll die Wirkung der Permutationsmatrizen P und P^T auf eine Elementarmatrix L, wie in Lemma 1.25 beschrieben, näher verdeutlichen.

BEISPIEL 1.26.

Sei

$$P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} = P^T \quad \text{und} \quad L = \begin{pmatrix} 1 & 0 & 0 \\ -l_{2,1} & 1 & 0 \\ -l_{3,1} & 0 & 1 \end{pmatrix},$$

dann gilt:

$$\begin{aligned} PLP^T &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ -l_{2,1} & 1 & 0 \\ -l_{3,1} & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & 0 \\ -l_{3,1} & 0 & 1 \\ -l_{2,1} & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & 0 \\ -l_{3,1} & 1 & 0 \\ -l_{2,1} & 0 & 1 \end{pmatrix}. \end{aligned}$$

Durch die in Lemma 1.25 beschriebene Vertauschungsrelation in (1.15) lässt sich nun die Linksmultiplikation des LGS $Ax = b$ in (1.14) umschreiben. Ohne Beschränkung der Allgemeinheit betrachten wir hierfür den Fall $n = 4$:

$$\begin{aligned} L_3P_3L_2P_2L_1P_1 &= L_3P_3L_2L_1'P_2P_1 \\ &= L_3L_2'P_3L_1'P_2P_1 \\ &= L_3L_2'L_1''P_3P_2P_1 = L^{-1}P. \end{aligned}$$

Man erhält nun die Matrix L der LR-Zerlegung von A durch Invertierung als

$$L = (L_3L_2'L_1'')^{-1} = (L_1'')^{-1}(L_2')^{-1}(L_3)^{-1}.$$

Wir können nun eine LR-Zerlegung mit Pivotisierung für beliebige $A \in \mathbb{R}^{n\times n}$ angeben:

$$PA = LR. \tag{1.16}$$

THEOREM 1.27.

Sei A eine reguläre $n \times n$ Matrix. Dann existiert eine **eindeutige LR-Zerlegung** der Form $PA = LR$, wobei L normierte, linke, untere Dreiecksmatrix und P eine Permutationsmatrix ist.

BEMERKUNG 1.28. Ist die $n \times n$ Matrix A singulär, so existiert dennoch eine Zerlegung $PA = LR$, die jedoch nicht eindeutig ist, da die linke, untere Dreiecksmatrix L nicht notwendigerweise normiert ist. In diesem Fall existiert ein $j \in \{1, \ldots, n-1\}$, so dass im j-ten Schritt des Gauss-Eliminationsverfahrens kein Pivotelement $a_{i,j}^{(j)} \neq 0$ für $i \geq j$ gefunden werden kann. Dann überspringt man den Eliminationsschritt, d.h., man setzt $L_j = P_j = I_n$. Ein Beispiel die

Zerlegung einer singulären Matrix $A \in \mathbb{R}^{2\times 2}$ ist:

$$PA = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} = LR.$$

△

1.4 Cholesky-Verfahren

In vielen Anwendungsgebieten der numerischen Mathematik tauchen symmetrische Matrizen auf, die häufig sogar positiv definit sind, wie zum Beispiel *Korrelationsmatrizen* bei statistischen Berechnungen, die *Normalengleichungen* bei linearen Approximationsproblemen oder *Steifigkeitsmatrizen* bei der Methode der finiten Elemente. Für diese spezielle Klasse von Matrizen lassen sich beim Lösen von linearen Gleichungssystemen der Form $Ax = b$ ihre besonderen Eigenschaften für effizientere Algorithmen ausnutzen.

Zur Herleitung des Cholesky-Verfahrens im Folgenden benötigen wir zuerst einige grundlegende Definitionen.

DEFINITION 1.29: Hermitesche Matrizen.

Sei A eine $n \times n$ Matrix. Wir nennen A **hermitesch**, falls die Matrix ihrer Adjungierten entspricht, d.h., es gilt $A = A^*$.
Wir bezeichnen hierbei mit A^* diejenige Matrix, die durch Spiegelung der Einträge von A an der Hauptdiagonalen mit anschließender komplexen Konjugation entsteht, d.h., für alle $i \neq j$ gilt $a_{i,j} = \overline{a_{j,i}}$.

LEMMA 1.30: Eigenschaften hermitescher Matrizen.

Für hermitsche Matrizen lassen sich folgende Eigenschaften zeigen:

(i) Jede hermitesche Matrix A mit rein reellen Einträgen $a_{i,j}, 1 \leq i, j \leq n$ ist symmetrisch.

(ii) Jede hermitesche Matrix A besitzt n reelle Eigenwerte und die zugehörigen Eigenvektoren bilden ein Orthogonalsystem des $\mathbb{R}^n$.

(iii) Jede hermitesche Matrix A ist selbstadjungiert und es lässt sich mit dem komplexen Standardskalarprodukt zeigen, dass für alle $x, y \in \mathbb{C}^n$ gilt:

$$\langle Ax, y\rangle = (Ax)^* y = x^* A^* y = x^* Ay = \langle x, Ay\rangle.$$

Nach der Einführung von hermiteschen bzw. symmetrischen Matrizen und ihren Eigenschaften können wir Aussagen über ihre Definitheit machen.

DEFINITION 1.31: Definitheit von Matrizen.

Sei A eine hermitesche Matrix.

(i) Falls gilt $\langle x, Ax\rangle > 0$ bzw. $\langle x, Ax\rangle \geq 0$ für alle $x \in \mathbb{C}^n, x \neq 0$, so nennen wir die Matrix A **positiv definit** bzw. **positiv semi-definit**.

(ii) Analog lassen sich die Begriffe **negativ definit** bzw. **negativ semi-definit** für den Fall $\langle x, Ax\rangle < 0$ bzw. $\langle x, Ax\rangle \leq 0$ definieren.

(iii) Ist A weder positiv noch negativ semi-definit so nennen wir A **indefinit**.

LEMMA 1.32.

Sei A eine positiv definite Matrix und sei $m \in \mathbb{N}$ beliebig mit $1 \leq m < n$. Dann sind alle Untermatrizen $A_{i_1,\ldots,i_m} \in \mathbb{C}^{(n-m)\times(n-m)}$, die durch Streichung der Spalten und Zeilen von A mit Indizes $\{i_1, \ldots, i_m\}$ auch positiv definit.

Beweis. Wir definieren zunächst $\overline{A} := A_{i_1,\ldots,i_m} \in \mathbb{C}^{(n-m)\times(n-m)}$ als diejenige Untermatrix, die durch Streichung der Spalten und Zeilen von A mit Indizes $\{i_1, \ldots, i_m\}$ entsteht. Sei nun $\overline{x} \in \mathbb{C}^{n-m}$ ein beliebiger Vektor mit $\overline{x} \neq 0$. Wir müssen zeigen, dass dann schon gilt $\langle \overline{x}, \overline{A}\overline{x}\rangle > 0$.

Betrachten wir nun die Einbettung $\Phi\colon \mathbb{C}^{n-m} \to \mathbb{C}^n$, die jeden Vektor $\overline{x} \in \mathbb{C}^{n-m}$ auf den eingebetteten Vektor $x \in \mathbb{C}^n$ abbildet, d.h., auf denjenigen Vektor x dessen Einträge mit denen von $\overline{x}$ übereinstimmen und zusätzlich Nullen als Einträge für die Indizes $i_1, \ldots, i_m$ besitzt. Aus der positiven Definitheit von A folgt dann aber schon:

$$\langle \overline{x}, \overline{A}\overline{x}\rangle = \langle \Phi(\overline{x}), A\Phi(\overline{x})\rangle = \langle x, Ax\rangle > 0.$$

Da der Vektor $\overline{x} \in \mathbb{C}^{n-m}$ und die Untermatrix $\overline{A}$ beliebig gewählt waren, folgt hieraus schon die Aussage. □

Aus Lemma 1.32 lässt sich direkt folgendes Korollar ableiten.

KOROLLAR 1.33.

Sei A eine positiv definite Matrix. Dann sind alle Diagonalelemente von A positiv, d.h., es gilt $a_{i,i} > 0$ für $i = 1, \ldots, n$.

BEMERKUNG 1.34. Man kann zeigen, dass jede positiv definite Matrix die schöne Eigenschaft hat nur reelle, positive Eigenwerte zu besitzen. △

Sei nun A eine positiv definite $n \times n$ Matrix, für welche eine LR-Zerlegung der Form $A = LR$ existiert, wobei L eine normierte, linke, untere Dreiecksmatrix ist. Wir wissen nun auf Grund der Eigenschaften von hermiteschen Matrizen, dass folgender Zusammenhang gelten muss:

$$LR \;=\; A \;=\; A^* \;=\; (LR)^* \;=\; R^*L^*.$$

Man sieht also, dass R^*L^* ebenfalls eine LR-Zerlegung von A ist. Sei nun D die Matrix, die sich aus der Hauptdiagonalen von R ergibt. Dann sieht man ein, dass

$$A \;=\; R^*(D^*)^{-1}D^*L^*$$

ebenfalls eine LR-Zerlegung von A darstellt, die sich von $A = R^*L^*$ nur durch die Normierung der Diagonalelemente der linken Seite unterscheidet. Da A als positiv definite Matrix regulär ist, folgt aber aus der Eindeutigkeit der LR-Zerlegung in Satz 1.27 schon, dass $L = R^*(D^*)^{-1}$ und $R = D^*L^*$ ist und wir somit eine eindeutige Darstellung

$$A \;=\; LD^*L^*$$

erhalten. Da die Matrix A positiv definit ist, müssen die Elemente von D nach Bemerkung 1.34 reell und positiv sein. Wir können also $D* = D$ setzen. Definieren wir nun die Matrix $\sqrt{D}$ als diejenige Matrix, die sich durch Wurzelziehen aus den Diagonalelementen von D ergibt, so können wir $\tilde{L} := L\sqrt{D}$ definieren und es ergibt sich die sogenannte **Cholesky-Zerlegung** der Matrix A als

$$A = LDL^* = L\sqrt{D}\sqrt{D}L^* = \tilde{L}\tilde{L}^* \tag{1.17}$$

Man beachte, dass die linke, untere Dreiecksmatrix $\tilde{L}$ durch die Multiplikation mit $\sqrt{D}$ im Allgemeinen nicht mehr normiert ist. Das Ergebnis lässt sich in folgendem Satz zusammenfassen.

THEOREM 1.35.

Sei A eine positiv definite Matrix. Dann gibt es genau eine linke, untere Dreiecksmatrix $\tilde{L}$ mit positiven Diagonalelementen, so dass eine eindeutige **Cholesky-Zerlegung** von A der folgenden Form existiert:

$$A = \tilde{L}\tilde{L}^*.$$

Der Beweis von Satz 1.35 ist konstruktiv und liefert zugleich den Algorithmus zur Berechnung der Cholesky-Zerlegung. Ohne Beschränkung der Allgemeinheit wollen wir diesen für den Fall von reellen Matrizen $A \in \mathbb{R}^{n\times n}$ im Folgenden angeben.

ALGORITHMUS 1.36: Cholesky-Verfahren.

Zur Berechnung der Cholesky-Zerlegung einer symmetrischen Matrix $A = LL^T$ betrachten wir die Gleichung zuerst elementweise:

$$a_{i,j} = \sum_{k=1}^{n} l_{i,k}l_{k,j}^T = \sum_{k=1}^{j} l_{i,k}l_{k,j}^T = \sum_{k=1}^{j} l_{i,k}l_{j,k}, \quad 1 \le j \le i \le n. \tag{1.18}$$

Hierbei bezeichnet $l_{k,j}^T$ ein Element der Matrix L^T. Auf Grund der Symmetrie von A besteht das *nichtlineare Gleichungssystem* (1.18) nur aus $n(n+1)/2$ Gleichungen. Da L eine linke, untere Dreiecksmatrix ist, gibt es genauso viele Unbekannte $l_{i,j}$ für $1 \le j \le i \le n$.
Dieses Gleichungssystem lässt sich wie folgt rekursiv auflösen. Wir versuchen zunächst die **erste Spalte** von L zu berechnen indem wir $j = 1$ setzen. Da alle Summanden bis auf den ersten Null sind, ergeben sich die folgenden n Gleichungen:

$$l_{i,1}l_{1,1} = a_{i,1}, \quad i = 1, \dots, n.$$

Insbesondere ergibt sich für den Index $i = 1$ die Lösung

$$l_{1,1} = \sqrt{a_{1,1}}. \tag{1.19}$$

Hierbei haben wir ausgenutzt, dass $l_{i,i} > 0$ für $i = 1, \dots, n$ und die Diagonalelemente von A immer positiv sind nach Korollar 1.33. Die weiteren Elemente der ersten Spalte von L kann man durch Einsetzen von (1.19) lösen als

$$l_{i,1} = a_{i,1}/l_{1,1} = a_{i,1}/\sqrt{a_{1,1}}, \quad i = 2, \dots, n.$$

Um die **zweite Spalte** von L zu berechnen setzen wir wiederum $j = 2$ und erhalten die Gleichungen (1.18)

$$l_{i,1}l_{2,1} + l_{i,2}l_{2,2} = a_{i,2}, \quad i = 2, \dots, n.$$

Für das Diagonalelement $l_{2,2}$ erhalten wir im Fall $i = 2$ die Lösung

$$l_{2,2} = \sqrt{a_{2,2} - l_{2,1}^2}.$$

Hierbei haben wir angenommen, dass der Radikand positiv ist. Die weiteren Elemente der zweiten Spalte von L für $3 \leq i \leq n$ kann man nun durch Einsetzen berechnen:

$$l_{i,2} = (a_{i,2} - l_{i,1}l_{2,1})/l_{2,2}, \quad i = 3, \ldots, n.$$

Wir wollen dieses Prinzip nun für den **j-ten Schritt** des Cholesky-Verfahrens verallgemeinern. Nehmen wir also an, dass die Spalten 1 bis $j-1$ der Matrix L bereits bestimmt wurden. Dann lauten die Gleichungen (1.18) für die j-te Spalte von L:

$$l_{i,1}l_{j,1} + \ldots + l_{i,j-1}l_{j,j-1} + l_{i,j}l_{j,j} = a_{i,j}, \quad i = j, \ldots, n.$$

Für $i = j$ ergibt sich für das Diagonalelement $l_{i,i}$ der Matrix L

$$l_{j,j} = \sqrt{a_{j,j} - l_{j,1}^2 - \ldots - l_{j,j-1}^2}. \tag{1.20}$$

Auch hier nehmen wir an, dass der Radikand positiv ist. Die weiteren Elemente für $i = j+1, \ldots, n$ lassen sich dann durch Einsetzen berechnen als

$$l_{i,j} = (a_{i,j} - l_{i,1}l_{j,1} - \ldots - l_{i,j-1}l_{j,j-1})/l_{j,j}, \quad i = j+1, \ldots, n.$$

Wir sehen also, dass wir mithilfe des Cholesky-Verfahrens alle unbekannten Elemente von L berechnen können.

Wir haben in (1.20) von Algorithmus 1.36 angenommen, dass der Radikand immer positiv ist. Folgendes Lemma zeigt, dass diese Annahme gerechtfertigt ist.

LEMMA 1.37.

Sei A eine positiv definite Matrix und $j \in \{1, \ldots, n\}$ beliebig. Dann gilt für den j-ten Schritt des Cholesky-Verfahrens in Algorithmus 1.36, dass der Radikand in (1.20) positiv ist, d.h., es gilt

$$a_{j,j} > \sum_{i=1}^{j-1} l_{j,i}^2.$$

Beweis. Wir beweisen das Lemma durch vollständige Induktion. Der **Induktionsanfang** für $j = 1$ ist richtig, da $a_{1,1} > 0$ nach Lemma 1.32 gilt.

Wir nehmen nun an, dass die Behauptung für die ersten $j-1$ Schritte gilt. Dann lassen sich die ersten $j-1$ Spalten der linken, unteren Dreiecksmatrix L mit dem Cholesky-Verfahren berechnen. Wir bezeichnen diese Spalten der Matrix L mit der $n \times (j-1)$ Matrix L_j und es

gilt

$$L_j L_j^T = \begin{pmatrix} a_{1,1} & & & & & \\ \vdots & \ddots & & & * & \\ a_{j-1,1} & \dots & a_{j-1,j-1} & & & \\ a_{j,1} & \dots & a_{j,j-1} & x_{j,j} & & \\ \vdots & & & \vdots & \ddots & \\ a_{n,1} & \dots & a_{n,j-1} & x_{n,j} & \dots & x_{n,n} \end{pmatrix} \tag{1.21}$$

Es reicht die linke, untere Hälfte des Produkts $L_j L_j^T$ zu betrachten, da sich die rechte Hälfte aus der Symmetrie von A ergibt. Man kann ausrechnen, dass das Element $x_{j,j}$ die folgende Gestalt hat

$$x_{j,j} = \sum_{k=1}^{j} l_{j,k} l_{j,k} = l_{j,1}^2 + \ldots + l_{j,j-1}^2.$$

Wenn wir nun zeigen können, dass $x_{j,j} < a_{j,j}$ gilt, so haben wir den **Induktionsschritt** vollzogen, denn dann ist der Radikand in (1.20) positiv.

Nehmen wir nun an, dass das Gegenteil gilt, d.h., $x_{j,j} \geq a_{j,j}$. Da jede Untermatrix einer positiv definiten Matrix auch positiv definit ist nach Lemma 1.32, wäre die Matrix $C \in \mathbb{R}^{j \times j}$ mit

$$C := \begin{pmatrix} a_{1,1} & & & \\ \vdots & \ddots & & \\ a_{j-1,1} & \dots & a_{j-1,j-1} & \\ a_{j,1} & \dots & a_{j,j-1} & x_{j,j} \end{pmatrix}$$

auf Grund der Annahme $x_{j,j} > a_{j,j}$ positiv definit. Dies sieht man durch folgendes Argument ein. Sei $\delta := x_{j,j} - a_{j,j}$, dann gilt $\delta > 0$ nach Voraussetzung. Definieren wir nun eine Matrix $B \in \mathbb{R}^{j \times j}$, die überall Nullen enthält außer dem Eintrag $b_{j,j} = \delta$ auf der Hauptdiagonalen. Es ist offensichtlich, dass die Matrix B positiv semi-definit ist und es gilt $C = A + B$. Dann gilt für alle Vektoren $x \in \mathbb{R}^j / \{\vec{0}\}$:

$$\langle x, Cx \rangle = \langle x, (A+B)x \rangle = \underbrace{\langle x, Ax \rangle}_{>0} + \underbrace{\langle x, Bx \rangle}_{\geq 0} > 0.$$

Die Matrix C ist also auch positiv definit und somit hätte die Matrix auf der rechten Seite von (1.21) mindestens den Rang j, während das Produkt $L_j L_j^T$ aber maximal den Rang $j-1$ haben kann. Dieser Widerspruch zeigt, dass $x_{j,j} < a_{j,j}$ gelten muss. □

BEMERKUNG 1.38. Der Rechenaufwand zur Berechnung der Cholesky-Zerlegung in Algorithmus 1.36 liegt in $\frac{1}{6}n^3 + \mathcal{O}(n^2)$ und verhält sich dadurch asymptotisch doppelt so schnell wie das Gauss-Eliminationsverfahren in Algorithmus 1.6. Dies liegt vor allem daran, dass man beim Cholesky-Verfahren die Symmetrie der Matrix A geschickt ausnutzt. △

Kapitel 2

Grundlagen des numerischen Rechnens

Eines der Hauptziele der numerischen Mathematik ist es die Genauigkeit bei der maschinellen Berechnung eines mathematischen Problems am Computer zu analysieren und zu beurteilen. In der Regel wird jede numerische Berechnung durch drei mögliche *Fehlerquellen* gestört:

1. Fehler in den Eingabedaten
2. Rundungsfehler bei der Zahldarstellung
3. Approximationsfehler durch die Annäherung eines mathematischen Problems

Fehler in den Eingabedaten entstehen typischerweise durch Fehler bei der physikalischen Messung dieser Daten, z.B., durch die Beschränkung des Werteraums der Messapparatur oder durch kleine stochastische Störungen wie elektronisches Rauschen.

Rundungsfehler treten natürlicherweise bei der Darstellung einer reellen Zahl mit Hilfe eines Datentyps im Computer auf, der nur eine fixe, endliche Anzahl von Stellen speichern kann.

Approximationsfehler hingegen treten immer dann auf, wenn ein mathematisches Problem numerisch angenähert wird. Beispiele hierfür sind die Approximation von Integralen durch Summen oder die Annäherung einer Ableitung durch Differenzenquotienten. Man spricht in diesem Zusammenhang häufig von der Diskretisierung eines Problems und ihrer Genauigkeit.

2.1 Digitale Zahldarstellung im Computer

Wir beschäftigen uns zunächst mit der Frage, wie moderne Digitalrechner, wie zum Beispiel jeder handelsübliche PC, eine reelle Zahl $x \in \mathbb{R}$ intern speichern und verarbeiten. Es ist klar, dass durch die endliche Anzahl der integrierten Schaltungen in einem Computerspeicher eine Zahl nur endlich viele Stellen besitzen kann. Dadurch ist es unmöglich eine beliebige irrationale Zahl numerisch *exakt* zu speichern, so dass durch die Endlichkeit des Computerspeichers zwangsläufig **Rundungsfehler** auftreten müssen.

DEFINITION 2.1: Maschinenzahlen.

Wir bezeichnen die Menge aller in einem Digitalrechner darstellbaren Zahlen $A \subset \mathbb{Q}$ als **Maschinenzahlen**. Die Menge der Maschinenzahlen wird durch das gewählte Format der Zahldarstellung definiert.

Um die Rundungsfehler durch die digitale Zahldarstellung im Folgenden näher zu charakterisieren, muss man verstehen, dass jede Zahl in einem Digitalrechner eine Darstellung aus 0en und 1en, den sogenannten *Bits*, besitzt. Die Anzahl $n \in \mathbb{N}$ der Dualstellen für die Repräsentation einer Zahl wird auch *Wortlänge* genannt. Sie wird typischerweise durch die Prozessorarchitektur vorgegeben. Aktuelle PCs verwenden eine Wortlänge von $n = 64$ Bit. Wir nehmen im folgenden eine Rechnerarchitektur an, die ein Speicherwort in der *„Little Endian"*-Konvention speichert, d.h., dass das am wenigsten signifikante Byte (1 Byte $\hat{=}$ 8 Bit) am Ende des Wortes steht.

2.1.1 Festkommazahlen

Betrachtet man die sogenannte **Festkommadarstellung** einer Zahl in einem Digitalrechner, so teilt man die vorliegende Wortlänge in $n = n_1 + n_2$ Dualstellen auf und nimmt implizit an, dass der Dezimalpunkt zwischen dem n_1-ten und $(n_1 + 1)$-ten Bit liegt. Nehmen wir zur Vereinfachung im Folgenden an, dass wir nur positive Zahlen darstellen wollen, dann können wir auf die Darstellung eines Vorzeichens verzichten. Damit kann man Maschinenzahlen $x \in A$ der folgenden Form darstellen:

$$x = \sum_{i=0}^{n_1-1} a_i 2^i + \sum_{i=1}^{n_2} b_i 2^{-i} = \underbrace{a_{n_1-1}2^{n_1-1} + \ldots + a_1 2^1 + a_0 2^0}_{n_1 \text{ Summanden}} + \underbrace{b_1 2^{-1} + \ldots b_{n_2} 2^{-n_2}}_{n_2 \text{ Summanden}}$$

Die Koeffizienten a_i, b_i stellen die Bits eines Speicherwortes der Länge n dar und können dementsprechend nur die Werte 0 oder 1 annehmen.

BEISPIEL 2.2.

Sei die Wortlänge eines Digitalrechners $n = 4$. Verwenden wir also die Festkommadarstellung mit $n_1 = 3$ und $n_2 = 1$, so würde das Speicherwort $\boxed{101}\boxed{1}$ die Zahl 5.5 darstellen, denn es lässt sich ausrechnen:

$$\boxed{1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0} + \boxed{1 \cdot 2^{-1}} = 4 + 0 + 1 + 0.5 = 5.5$$

2.1.2 Gleitkommazahlen

Heutzutage wird die oben beschriebene Festkommadarstellung von Zahlen kaum noch in Digitalrechnern verwendet, da sie zu unflexibel ist und über eine zu geringe Mächtigkeit bei der Zahldarstellung verfügt. Aus diesem Grund hat man die **Gleitkommadarstellung** von Zahlen in Digitalrechnern eingeführt. Man verwendet typischerweise die normalisierte Gleitkommadarstellung, für die es bereits seit 1985 den IEEE 754 Standard gibt, welchen wir in Abschnitt 2.1.4 näher betrachten werden. Dieser ist auf fast allen modernen Digitalrechnern heutzutage implementiert. Hierbei wird die Stelle des Dezimalpunktes nicht fest vorgegeben, sondern kann in Abhängigkeit der zu darstellenden Zahl *flexibel* gewählt werden. Der zu Grunde liegende Gedanke bei dieser Darstellung ist es, dass große Zahlen häufig wenig Genauigkeit

bei den Nachkommastellen benötigen, wohingegen sehr kleine Zahlen eine hohe Genauigkeit bei den Nachkommastellen brauchen.

Wir betrachten im Folgenden die digitale Zahldarstellung mit Hilfe von Gleitkommazahlen.

DEFINITION 2.3: Gleitkommazahlen.

Die Menge aller Maschinenzahlen $A \subset \mathbb{Q}$ der Form

$$x \;=\; m \cdot b^e, \tag{2.22}$$

für $x \in A$ stellt die Menge der exakt darstellbaren **Gleitkommazahlen** in Abhängigkeit der gewählten *Mantisse* $m \in \mathbb{Q}$, der *Basis* $b \in \mathbb{N}$ und des *Exponenten* $e \in \mathbb{Z}$ dar.

Typischerweise wird die *Basis* b in Digitalrechnern als $b = 2$ angenommen, wohingegen Menschen natürlicherweise mit einer Zehnerpotenz, d.h. $b = 10$ als Basis rechnen.

Für eine Gleitkommazahl $x \in A$ wird die *Mantisse* $m \in \mathbb{Q}$ durch $p \in \mathbb{N}$ Stellen mit $p < n$ festgelegt und bestimmt die Genauigkeit bei der Approximation einer Zahl durch x. Für die Speicherung einer Maschinenzahl im Digitalrechner wird angenommen, dass die Mantisse eine positive Ganzzahl repräsentiert und die entsprechenden Bits werden daher als solche interpretiert.

Der *Exponent* erhält bei der Zahldarstellung $r \in \mathbb{N}$ Stellen mit $r < n$ und ist maßgeblich für die Definition des Wertebereichs von Maschinenzahlen und der genauen Stelle des Kommas bei einer Zahldarstellung verantwortlich. Für die Speicherung einer Maschinenzahl im Digitalrechner wird angenommen, dass der Exponent eine positive oder negative Ganzzahl repräsentiert und die entsprechenden Bits werden daher als solche interpretiert. Hierbei verwendet man typischerweise jedoch nicht die Darstellung des Exponenten im Zweierkomplement [2-Kompl], sondern führt einen festen Biaswert ein (mehr dazu in Abschnitt 2.1.4 zum IEEE 754 Standard).

Das erste Bit einer Gleitkommazahl wird für das Vorzeichen reserviert ($\boxed{0}$ ist positiv und $\boxed{1}$ ist negativ), während die restlichen Bit zwischen Mantissenlänge und Exponentenlänge aufgeteilt werden, d.h., es gilt insgesamt

$$n \;=\; 1 + p + r.$$

2.1.3 Normalisierung

Im Allgemeinen ist die Darstellung einer Zahl durch eine Gleitkommazahl nicht eindeutig, wie folgendes Beispiel zeigt.

BEISPIEL 2.4: Uneindeutigkeit von Gleitkommazahlen.

Wir betrachten im Folgenden unterschiedliche Zahldarstellungen einer Zahl jeweils im Dezimal- sowie im Dualsystem.

(i) Die Darstellung der Zahl $x = 3.5$ lässt sich in der Dezimaldarstellung bezüglich der Basis 10 auch schreiben als $x = 35 \cdot 10^{-1}$ oder $x = 0.35 \cdot 10^{1}$. Die Darstellung ist also nicht eindeutig, da das Komma an beliebige Stelle verschoben werden kann durch die Änderung des Exponenten.

(ii) Für ein Speicherwort der Länge $n = 8$ Bit mit Basis 2, Mantissenlänge $p = 5$ und Exponentenlänge $r = 2$ gibt es für die Zahl 14 die Gleitpunktdarstellungen

$$\boxed{0}\underbrace{\boxed{00}}_{r}\underbrace{\boxed{01110}}_{p} = +2^{(0\cdot 2^1+0\cdot 2^0)}\cdot(0\cdot 2^4+1\cdot 2^3+1\cdot 2^2+1\cdot 2^1+0\cdot 2^0) = 1\cdot 14 = 14$$

und

$$\boxed{0}\boxed{01}\boxed{00111} = +2^{(0\cdot 2^1+1\cdot 2^0)}\cdot(0\cdot 2^4+0\cdot 2^3+1\cdot 2^2+1\cdot 2^1+1\cdot 2^0) = 2\cdot 7 = 14.$$

Für dieses Beispiel haben wir die Bits für den Exponenten und für die Mantisse jeweils als eine positive Ganzzahl interpretiert.

Um die Eindeutigkeit der Darstellung einer Zahl in einem Digitalrechner sicher zu stellen, muss man die Gleitkommazahl normieren.

DEFINITION 2.5.

Wir nennen die Gleitkommadarstellung einer Zahl **normalisiert**, falls für die Mantisse eine feste Bedingung $a_1 \leq m < a_2$ für $0 \leq a_1 < a_2$ gilt.
Typische Normalisierungsbedingungen für Gleitkommazahlen sind $b^{-1} \leq m < 1$ oder $1 \leq m < b$.

Der Vorteil der Normalisierung besteht darin, dass man die Ziffer vor dem Komma nicht explizit speichern muss, da man davon ausgehen kann, dass diese für alle normalisierte Gleitkommazahlen gleich ist. Man spricht auch in diesem Zusammenhang vom *„Hidden Bit“*.

2.1.4 IEEE 754 Standard

Es wird klar, dass man bei der Aufteilung der $(n-1)$ Bit eines Maschinenworts der Länge n einen **Kompromiss** zwischen der Approximationsgenauigkeit einer Zahl (definiert durch p) und der Größe des Wertebereichs (definiert durch r) finden muss. Aus diesem Grund gibt es verschiedene vorgeschlagene IEEE-Standards zur Implementierung und Interpretation von Gleitkommazahlen in Digitalrechnern, z.B. *Half-, Single-* und *Double-Precision* Gleitkommazahlen. Wir wollen uns im Folgenden ein Beispiel zur Speicherung einer Gleitkommazahl im *Single Precision* Format nach dem IEEE 754 Standard [IEEE754] betrachten.

BEISPIEL 2.6.

Wir wollen die Zahl $x = 18.4$ in einem Digitalrechner nach dem IEEE 754 Standard für *Single Precision* Gleitkommazahlen zur Basis $b = 2$ darstellen, d.h., wir haben eine Wortlänge von $n = 32$, Mantissenlänge $p = 23$ und Exponentenlänge $r = 8$. Der IEEE 754 Standard benutzt zur Speicherung des Exponenten einen festen *Bias* von $B = 127$, der auf den Exponenten aufaddiert wird und dadurch immer eine vorzeichenfreie, positive Zahl $E = e + B$ entsteht. Dies hat Vorteile beim Vergleich von Exponenten verschiedener Gleitkommazahlen.
Im ersten Schritt nähern wir die Zahl 18.4 als **Festkommazahl** an wie in Abschnitt 2.1.1 beschrieben. Hierzu stellen wir zuerst die Zahl 18 vor dem Komma im Binärsystem dar

als:

$$18 \;=\; 16+2 \;=\; 1\cdot 2^4+0\cdot 2^3+0\cdot 2^2+1\cdot 2^1+0\cdot 2^0 \;=\; \boxed{10010}.$$

Versuchen wir nun den Rest 0.4 nach dem Komma darzustellen:

$$\begin{aligned} 0.4 \;&= 0\cdot 2^{-1}+1\cdot 2^{-2}+1\cdot 2^{-3}+0\cdot 2^{-4}+0\cdot 2^{-5}+1\cdot 2^{-6}+1\cdot 2^{-7}+\ldots \\ &= \boxed{0110011\ldots} \end{aligned}$$

sehen wir, dass die Zahl 0.4 eine periodische Struktur in der Binärdarstellung besitzt. Da man nur eine endliche Wortlänge von $n = 32$ hat, wird die Zahl inexakt dargestellt. Es ergibt sich also folgende Festkommadarstellung der Zahl 18.4 im Dualsystem:

$$18.4 \;=\; \boxed{10010{,}0110011\ldots}$$

Für die **Normalisierung** der Gleitkommazahl im IEEE 754 Format müssen wir die Bedingung $1 \leq m < 2$ erfüllen. Dies kann erreicht werden indem wir den Exponenten $e = 4$ wählen, d.h., wir multiplizieren die obige Zahldarstellung mit dem Faktor 2^4 und erhalten somit die Darstellung:

$$\boxed{1{,}00100110011\ldots}\cdot 2^4.$$

Wir stellen den Exponenten nun mit Hilfe des **Bias** von $B = 127$ dar, d.h.,

$$E \;=\; e+B \;=\; 4+127 \;=\; \boxed{00000100}+\boxed{01111111} \;=\; \boxed{10000011} \;=\; 131.$$

Da wir normalisierte Gleitkommazahlen verwenden müssen wir die führende 1 vor dem Komma nicht explizit speichern. Aus den obigen Berechnungen lässt sich die Zahl 18.4 nun als folgende 32 Bit Gleitkommazahl approximieren:

$$18.4 \;\approx\; \underbrace{\boxed{0}}_{\text{Vorz.}}\;\underbrace{\boxed{10000011}}_{\text{8 Bit Exponent}}\;\underbrace{\boxed{00100110011001100110011}}_{\text{23 Bit Mantisse}}$$

BEMERKUNG 2.7. Bei der Rückkonvertierung der in Beispiel 2.6 berechneten Gleitkommazahldarstellung in eine Dezimalzahl erhält man auf Grund der Endlichkeit der Mantisse und der Periodizität der Dualdarstellung von 0.4 das inexakte Ergebnis

$$18.4 \;\approx\; \boxed{0\,|\,10000011\,|\,00100110011001100110011} \;=\; 18.39999961853$$

△

Wir erkennen also, dass $x = 18.4 \notin A$ keine Maschinenzahl in Bezug auf den IEEE 754 Standard ist und daher nicht exakt dargestellt werden kann. Es kommt automatisch zu Rundungsfehlern.

2.2 Rundungsfehler und Gleitkommaarithmetik

Wie wir schon im vorigen Abschnitt gesehen haben ist die Menge aller Maschinenzahlen $A \subset \mathbb{Q}$ endlich. Im Folgenden wollen wir uns also mit der Frage beschäftigen, wie sich eine nicht

exakt darstellbare Zahl $x \notin A$ durch eine Maschinenzahl $\tilde{x} \in A$ approximieren lässt. Wie wir sehen werden taucht dieses Problem nicht nur bei der *Speicherung* der Zahl x auf, sondern bei allen *mathematischen Berechnungen* im Digitalrechner. Eine vernünftige Bedingung an eine Approximation der Zahl x durch eine Maschinenzahl $\tilde{x} = \mathrm{rd}(x) \in A$ ist gegeben durch:

$$|x - \tilde{x}| \;=\; |x - \mathrm{rd}(x)| \;\leq\; |x - y|, \quad \forall y \in A. \tag{2.23}$$

Man erhält solch eine Approximation $\mathrm{rd}(x)$ im Digitalrechner durch **Rundung**.

BEISPIEL 2.8.

Für die Basis $b = 10$ und eine Mantissenlänge von $p = 4$ erhalten wir folgenden Approximationen:

(i) $x = 1.97689743 \quad \Rightarrow \quad \mathrm{rd}(x) = 1.9769$

(ii) $x = 1.374651 \quad \Rightarrow \quad \mathrm{rd}(x) = 1.3747$

(iii) $x = 1.07743846282 \quad \Rightarrow \quad \mathrm{rd}(x) = 1.0774$

Falls $\mathrm{rd}(x) \in A$ eine Maschinenzahl ist, so erfüllt diese Approximation die gewünschte Eigenschaft (2.23) für alle $y \in A$. Jedoch gibt es auch Fälle in denen die für uns intuitiv gewählte Rundungsoperation nicht zwangsläufig eine Maschinenzahl liefert, wie wir im folgenden Beispiel sehen können.

BEISPIEL 2.9.

Wir betrachten folgende Beispiele für den Fall der Basis $b = 10$ mit einer Mantissenlänge von $p = 4$ und einer Exponentenlänge von $r = 2$:

(i) $\mathrm{rd}(1.31794 \cdot 10^{145}) \;=\; 1.3179 \cdot 10^{145} \notin A$

(ii) $\mathrm{rd}(1.368589 \cdot 10^{99}) \;=\; 0.13686 \cdot 10^{100} \notin A$

Im ersten Fall handelt es sich um keine Maschinenzahl da der Exponent 145 die Länge 3 hat und nur eine Exponentenlänge von $r = 2$ vorliegt. Im zweiten Fall kommt es zu einem *Exponentenüberlauf* durch die in diesem Beispiel gewählte Normalisierung der Gleitkommazahl durch $1/10 \leq m < 1$.

Um den Fehler einer numerischen Operation anzugeben gibt es zwei gängige Fehlermaße, die in folgender Definition erklärt werden.

DEFINITION 2.10: Absoluter und relativer Fehler.

Sei x ein exakter Wert und $\tilde{x}$ eine Approximation des exakten Wertes x, d.h. es gilt $\tilde{x} \approx x$. Dann definieren wir die folgenden Fehlergrößen:

(i) $\Delta_x := |\tilde{x} - x|$ bezeichnet den **absoluten Fehler**.

(ii) $\delta_x := \frac{\Delta_x}{|x|} = \frac{|\tilde{x}-x|}{|x|}$ bezeichnet für $x \neq 0$ den **relativen Fehler**.

Für den Fall, dass für eine Zahl $x \in \mathbb{R}$ die Rundungsoperation eine Maschinenzahl $\mathrm{rd}(x) \in A$ liefert, wollen wir den *relativen Fehler* dieser Approximation für die Basis $b = 10$ betrachten.

Wir setzen also

$$\frac{|x - \mathrm{rd}(x)|}{|x|} = \frac{|m - \tilde{m}| \cdot 10^e}{|m| \cdot 10^e} \leq \frac{5 \cdot 10^{-(p+1)}}{|m|} = \frac{5 \cdot 10^{-p}}{|m| \cdot 10^1} \leq 5 \cdot 10^{-p} \tag{2.24}$$

Wir haben hierbei ausgenutzt, dass sich die Zahl x als normalisierte Gleitkommazahl darstellen lässt, bei der gilt, dass $1/10 \leq m < 1$ gilt. Der relative Fehler durch die Rundung wird also (wie zu erwarten) maßgeblich durch die Mantissenlänge p bestimmt. Basierend auf dieser Abschätzung können wir eine Kennzahl für die Genauigkeit eines Digitalrechners einführen.

DEFINITION 2.11: Maschinengenauigkeit.

Wir definieren die **Maschinengenauigkeit** eines Digitalrechners mit Mantissenlänge $p \in \mathbb{N}$ als

$$\mathrm{eps}_{10} \coloneqq 5 \cdot 10^{-p}$$

bezüglich der Basis $b = 10$. Analog lässt sich die Maschinengenauigkeit bezüglich der Basis $b = 2$ definieren als

$$\mathrm{eps}_2 \coloneqq 2^{-p}.$$

Kontextabhängig verwenden wir auch die einfach die Bezeichnung eps für die Maschinengenauigkeit wenn die Wahl der Basis klar ist.

Lassen wir einmal die Probleme mit einem Exponentenüber- bzw. Exponentenunterlauf außer Acht, so lässt sich mit Hilfe der Maschinengenauigkeit aus dem relativen Fehler durch die Rundungsoperation in Gleichung (2.24) folgendes ableiten:

$$\mathrm{rd}(x) = x \cdot (1 + \epsilon) \quad \text{mit} \quad |\epsilon| \leq \mathrm{eps}\,. \tag{2.25}$$

Da das Resultat arithmetischer Operationen auf Maschinenzahlen $x, y \in A$ wie $x + y, x - y, x \cdot y$, und x/y keine Maschinenzahl sein muss, so kann man erwarten, dass diese Operationen nicht exakt im Digitalrechner realisiert werden. Stattdessen werden Ersatzoperationen $+^*, -^*, \cdot^*, /^* \colon A \times A \to A$ implementiert, die eine vernünftige Gleitpunktarithmetik realisieren. Diese arithmetischen Gleitpunktoperationen lassen sich mit Hilfe der Rundungsoperation in (2.25) wie folgt angeben:

$$\begin{aligned}
(x +^* y) &\coloneqq \mathrm{rd}(x + y) = (x + y) \cdot (1 + \varepsilon_1) \\
(x -^* y) &\coloneqq \mathrm{rd}(x - y) = (x - y) \cdot (1 + \varepsilon_2) \\
(x \cdot^* y) &\coloneqq \mathrm{rd}(x \cdot y) = (x \cdot y) \cdot (1 + \varepsilon_3) \\
(x /^* y) &\coloneqq \mathrm{rd}(x / y) = (x / y) \cdot (1 + \varepsilon_4)
\end{aligned}$$

für $x, y \in A$ und $|\varepsilon_i| \leq \mathrm{eps}$ für $i = 1, \ldots, 4$.

BEMERKUNG 2.12. Die üblichen Gesetzmäßigkeiten der arithmetischen Operationen in $\mathbb{R}$ gelten im Allgemeinen nicht für die arithmetischen Gleitpunktoperationen. So ist zum Beispiel

$$x +^* y = x \quad \text{falls } |y| < \frac{\mathrm{eps}}{b}|x|.$$

Der Grund ist, dass bei einer Angleichung des Exponenten von der Gleitkommarepräsentation von y an den Exponenten von x die gesamte Mantisse von y ausgelöscht wird, da für die Basis

$b = 2$ beispielsweise gilt:

$$|y| < \frac{\text{eps}}{b}|x| \quad \Rightarrow \quad |y| \cdot 2^{r+1} < |x|.$$

Aus dieser Beobachtung lässt sich die Maschinengenauigkeit auch als die kleinste positive Maschinenzahl $\text{eps} \in A$ definieren, für die gilt:

$$\text{eps} \;:=\; \min\{y \in A \,|\, 1 +^* y > 1\}.$$

△

Folgendes Beispiel zeigt, dass bekannte algebraische Gesetzmäßigkeiten des Körpers der reellen Zahlen für die Menge der Maschinenzahlen $A \subset \mathbb{R}$ bezüglich der arithmetischen Gleitpunktoperationen im Allgemeinen nicht gelten, wie z.B. das Assoziativgesetz. Hier tritt insbesondere ein als **Auslöschung** bekanntes Phänomen auf, welches zu gravierenden numerischen Fehlern führen kann.

BEISPIEL 2.13: Auslöschung.

Wir betrachten drei Maschinenzahlen $a, b, c \in A$ in der normalisierten Gleitpunktdarstellung bezüglich der Basis $b = 10$ mit einer Mantissenlänge von $p = 8$ und einer Exponentenlänge von $r = 2$. Dann gilt für

$$\begin{aligned} a &:= 0.23371258 \cdot 10^{-4}, \\ b &:= 0.33678429 \cdot 10^{2}, \\ c &:= -0.33677811 \cdot 10^{2} \end{aligned}$$

bei unterschiedlicher Reihenfolge der Additionsoperation $+^* \colon A \times A \to A$ folgendes:

$$\begin{aligned} a +^* (b +^* c) &= 0.23371258 \cdot 10^{-4} +^* 0.61800000 \cdot 10^{-3} \\ &= 0.64137126 \cdot 10^{-3} \\ (a +^* b) +^* c &= 0.33678452 \cdot 10^{2} -^* 0.33677811 \cdot 10^{2} \\ &= 0.64100000 \cdot 10^{-3} \end{aligned}$$

Bei der ersten Berechnung traten glücklicherweise keine Rundungsfehler bei der Operation $(b +^* c)$ auf. Bei der zweiten Berechnung entstehen bei der Operation $(a +^* b)$ jedoch wegen der beschränkten Mantissenlänge signifikante Rundungsfehler, die dann durch die anschließende Auslöschungsoperation zu einem stark veränderten Ergebnis führt. Das exakte Ergebnis in $\mathbb{R}$ ist gegeben durch

$$a + b + c \;=\; 0.641371258 \cdot 10^{-3}.$$

Auslöschung tritt immer dann auf, wenn Maschinenzahlen mit dem gleichen Vorzeichen voneinander subtrahiert werden, die den gleichen Exponenten aufweisen und in den ersten Stellen der Mantisse übereinstimmen.

2.3 Mathematische Grundlagen

In diesem Abschnitt werden wir einige mathematische Grundlagen wiederholen, die in späteren Kapiteln nützlich sein werden, wie zum Beispiel bei der Fehleranalyse von linearen Gleichungssystemen. Hierbei wollen wir insbesondere verstehen wie sich Fehler in den Eingangsdaten von linearen Gleichungssystemen der Form $Ax = b$ auswirken, d.h. bei Störungen des Vektors b und der Matrix A. Dazu müssen wir die Fehler in vernünftiger Weise messen und benötigen daher also geeignete Normen.

2.3.1 Vektornormen

Da die Definition einer Vektornorm eines beliebigen Vektorraums V für alle folgenden Konzepte eine wichtige Rolle spielt wollen wir diese grundlegende Definition im Folgenden zunächst wiederholen.

DEFINITION 2.14: Vektornorm.

Sei V ein beliebiger Vektorraum über dem Körper $\mathbb{K}$ der reellen oder komplexen Zahlen. Wir nennen eine Abbildung $||\cdot|||$ von V in die nichtnegativen reellen Zahlen mit

$$\begin{aligned} ||\cdot||\colon V &\to \mathbb{R}_0^+, \\ x &\mapsto ||x||, \end{aligned}$$

eine **(Vektor-)Norm** auf V, wenn für alle Vektoren $x, y \in V$ und alle Skalare $\alpha \in \mathbb{K}$ die folgenden Axiome erfüllt sind:

(i) $||x|| = 0 \;\Rightarrow\; x = 0,$ (*Definitheit*)

(ii) $||\alpha x|| = |\alpha| \cdot ||x||,$ (*absolute Homogenität*)

(iii) $||x + y|| \leq ||x|| + ||y||.$ (*Subadditivität*)

Aus der Linearen Algebra kennen wir die obigen Eigenschaften einer Norm auf einem Vektorraum bereits und auch einige gängige Beispiele, die wir im folgenden wiederholen werden.

BEISPIEL 2.15: Vektornormen in $\mathbb{R}^n$.

Betrachten wir Vektornormen auf dem endlichdimensionalen Vektorraum $V := \mathbb{R}^n$. Eine interessante Klasse von Vektornormen auf V ist durch die **p-Normen** gegeben für $1 \leq p < \infty$ mit

$$\|x\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}.$$

Ein Spezialfall ist die *Euklidische Norm* für $p = 2$.
Darüber hinaus verwendet man auch häufig die sogenannte **Maximumsnorm** für $p = \infty$ mit

$$\|x\|_\infty := \max_{1 \leq i \leq n} |x_i|.$$

Unter den obigen Vektornormen hat die Euklidische Norm eine spezielle Rolle, da sie aus einem Skalarprodukt entsteht, nämlich

$$\langle x, y\rangle \;=\; \sum_{i=1}^{n} x_i y_i.$$

Ein Skalarprodukt, d.h. eine bilineare Abbildung mit $\langle x, x\rangle > 0$ für $x \neq 0$ induziert immer eine Norm durch

$$\|x\| \;:=\; \sqrt{\langle x, x\rangle} \;=\; \sqrt{\sum_{i=1}^{n} x_i^2}.$$

Dies vereinfacht viele Rechnungen im Vergleich zu allgemeinen Normen, typischerweise rechnet man dann mit dem Quadrat der Norm und verwendet die Eigenschaften das Skalarprodukts um einfache Relationen wie den Satz des Pythagoras zu erhalten. Man spricht in solchen Fällen von einem *Hilbertraum*. Der $\mathbb{R}^n$ als normierter Raum hat die selben Elemente und topologisch äquivalente Normen, jedoch ist die Geometrie in der Interpretation eines Hilbertraums deutlich einfacher.

Ein grundlegendes Resultat ist die **topologische Äquivalenz** aller Normen in einem endlichdimensionalen Raum. Um also Konvergenzresultate zu beweisen ist es egal welche Norm wir verwenden. Wollen wir hingegen *quantitative Abschätzungen*, so besteht - insbesondere für den am meisten relevanten Fall von großer Dimension $n \in \mathbb{R}^n$ - ein potentiell großer Unterschied zwischen den verschiedenen Normen. So gilt etwa die Abschätzung

$$\|x\|_\infty \leq \|x\|_1 \leq n\|x\|_\infty.$$

Im schlimmsten Fall ist also die 1-Norm n-mal so groß wie die Maximumsnorm.

BEMERKUNG 2.16 (Wahl der Vektornorm). In der Praxis ist es wichtig, welche Fehler man abschätzen möchte bzw. in den Eingabedaten abschätzen kann. Will man etwa garantieren, dass jeder Eintrag bis auf einen maximalen Fehler korrekt ist, muss man eine Abschätzung in der Maximumsnorm verwenden, während es für den mittleren Fehler reicht die 1-Norm zu verwenden. Ähnliches gilt bei Eingabedaten, wenn man nur eine Schranke für jeden gemessenen Eintrag hat. In diesem Fall ist dies natürlich eine Schranke an die Maximumsnorm. Bei stochastischen Fehlern, die häufig auftreten, kommt man zu anderen Schranken. Ein Beispiel ist ein Modell der Form $\tilde{x}_i = x_i + \epsilon_i$, mit Fehlern ϵ_i die gleichverteilt, unabhängig sind und Mittelwert Null haben. Dann hat man üblicherweise eine Schranke an die Varianz, die bei großem n gut der empirischen Varianz, also dem Quadrat der Euklidischen Norm, entspricht. △

2.3.2 Matrixnormen

Für eine Matrix kann man beliebige Normen definieren, etwa die Frobenius-Norm als Verallgemeinerung der Euklidischen Norm mit

$$\|A\|_F = \sqrt{\sum_{i,j} A_{ij}^2}.$$

Allerdings ist dabei nicht klar, wie der mathematische Zusammenhang zwischen Matrix- und Vektornorm aussieht. Insbesondere hätte man für einfache Abschätzungen gern eine Eigenschaft der Form

$$\|Ax\| \;\leq\; \|A\| \cdot \|x\|,$$

wie sie auch im eindimensionalen für den Betrag gelten. Deshalb ist es natürlich die Norm einer Matrix (oder auch die Norm linearer Operatoren, die im endlichdimensionalen isomorph zu den Matrizen sind) über folgende Relation in einem endlichdimensionalen Vektorraum V zu definieren

$$\|A\| \;:=\; \sup_{x\neq 0} \frac{\|Ax\|}{\|x\|},$$

denn dann gilt schon automatisch

$$\frac{||Ay||}{||y||} \;\leq\; \sup_{x\neq 0} \frac{\|Ax\|}{\|x\|} \;=:\; ||A|| \quad\Rightarrow\quad ||Ay|| \;\leq\; ||A|| \cdot ||y|| \quad \forall y \in V.$$

Wir müssen zunächst klären, dass wir damit eine Norm definiert haben und dass das Supremum auch endlich ist. Es ist auch zu beachten, dass in obiger Definition eigentlich zwei Normen vorkommen können: ist $A \in \mathbb{R}^{m\times n}$ dann haben wir eine Norm in $\mathbb{R}^n$ für x und eine Norm in $\mathbb{R}^m$ für Ax. Selbst im Fall $m = n$ müssen diese nicht die Gleichen sein. Die folgende Definition deckt diesen allgemeinen Fall ab.

DEFINITION 2.17: Induzierte Operatornorm.

Seien V und W normierte Vektorräume mit den Normen $\|\cdot\|_V$ bzw. $\|\cdot\|_W$. Dann ist die **induzierte Operatornorm** eines linearen Operators $A : V \to W$ definiert durch

$$\|A\| \;:=\; \sup_{x\neq 0} \frac{\|Ax\|_W}{\|x\|_V}. \tag{2.26}$$

Das folgende Beispiel zeigt häufig verwendete Matrixnormen.

BEISPIEL 2.18: Matrixnormen.

Bei Matrixnormen betrachtet man vor allem jene, die von p-Normen im $\mathbb{R}^n$ bzw. $\mathbb{R}^m$ induziert sind, d.h.

$$\|A\|_{p,q} \;:=\; \sup_{x\neq 0} \frac{\|Ax\|_p}{\|x\|_q}.$$

Besonders wichtig ist der Fall $p = q$. Hier schreibt man einfach

$$\|A\|_p := \sup_{x\neq 0} \frac{\|Ax\|_p}{\|x\|_p}.$$

Hierbei sind wieder $p = 1, 2, \infty$ die kanonischen Fälle. Wir werden in Beispiel 2.21 und Beispiel 2.22 sehen, dass man diese Matrixnormen in diesen Fällen auch einfacher charakterisieren kann.

Das folgende Theorem besagt, dass die induzierte Operatornorm für endlichdimensionale Vektorräume eine Vektornorm für Matrizen ist.

THEOREM 2.19: Operatornorm für endlichdim. Vektorräume.

Seien V und W endlichdimensionale Vektorräume. Dann definiert Gleichung (2.26) eine Vektornorm auf dem Raum der linearen Operatoren von V nach W, der isomorph zum Raum der Matrizen $\mathbb{R}^{m\times n}$ ist, wobei $m = \dim(W) \in \mathbb{N}$ und $n = \dim(V) \in \mathbb{N}$. Darüber hinaus gilt

$$\|Ax\|_W \;\leq\; \|A\| \cdot \|x\|_V. \tag{2.27}$$

Beweis. Zunächst zeigen wir, dass das Supremum endlich ist, dazu verwenden wir, dass

$$\frac{\|Ax\|_W}{\|x\|_V} \;=\; \|Ay\|_W$$

gilt mit $y := \frac{x}{\|x\|_V}$, wobei $\|y\|_V = 1$. Also ist

$$\sup_{x\neq 0} \frac{\|Ax\|_W}{\|x\|_V} \;=\; \sup_{\|y\|_V=1} \|Ay\|_W.$$

Nun suchen wir ein Supremum der stetigen Funktion $y \mapsto \|Ay\|_W$ über der kompakten Einheitssphäre $\{y \mid \|y\|_V = 1\}$ in V. Damit ist dieses endlich und wird sogar als Maximum angenommen.

Nun müssen wir noch die drei Eigenschaften einer Norm nachweisen:

1. Per Definition ist $\|A\|$ nichtnegativ und wir sehen $\|0\| = 0$. Ist nun umgekehrt $\|A\| = 0$, so folgt sofort
$$\sup_{x\neq 0} \frac{\|Ax\|_W}{\|x\|_V} \;=\; 0$$
und damit
$$0 \;\leq\; \frac{\|Ax\|_W}{\|x\|_V} \;\leq\; 0$$
für alle $x \in V$. Also folgt $Ax = 0$ für alle x und somit muss schon $A = 0$ gelten.

2. Um *absolute Homogenität* zu erhalten benutzen wir für $\alpha \in \mathbb{R}$
$$\|\alpha A\| = \sup_{x\neq 0} \frac{\|\alpha Ax\|_W}{\|x\|_V} = \sup_{x\neq 0} |\alpha| \; \frac{\|Ax\|_W}{\|x\|_V} = |\alpha| \; \sup_{x\neq 0} \frac{\|Ax\|_W}{\|x\|_V} = |\alpha| \; \|A\|.$$

3. Für die *Dreiecksungleichung* benutzen wir einfach die Dreiecksungleichung der Vektornorm in W
$$\begin{aligned}\|A + B\| \;&=\; \sup_{x\neq 0} \frac{\|(A+B)x\|_W}{\|x\|_V} \;\leq\; \sup_{x\neq 0} \frac{\|Ax\|_W}{\|x\|_V} + \frac{\|Bx\|_W}{\|x\|_V} \\ &\leq\; \sup_{x\neq 0} \frac{\|Ax\|_W}{\|x\|_V} + \sup_{y\neq 0} \frac{\|By\|_W}{\|y\|_V} \\ &=\; \|A\| + \|B\|.\end{aligned}$$

Gleichung (2.27) folgt direkt aus der Definition über das Supremum. □

Eine weitere interessante Eigenschaft einer induzierten Matrixnorm ist die **Submultiplikativität**.

LEMMA 2.20: Submultiplikativität.

Seien $A : V \to W$ und $B : U \to V$ lineare Operatoren. Dann gilt für die induzierten Operatornormen

$$\|AB\| \leq \|A\| \cdot \|B\|.$$

Beweis. Dies sieht man leicht aus der Identität

$$\frac{\|ABx\|_W}{\|x\|_U} = \frac{\|ABx\|_W}{\|Bx\|_V} \frac{\|Bx\|_V}{\|x\|_U}$$

falls $Bx \neq 0$. Damit folgt sofort die Abschätzung für das Supremum mit

$$\|AB\| = \sup_{x \neq 0} \frac{\|ABx\|_W}{\|Bx\|_V} \frac{\|Bx\|_V}{\|x\|_U} \leq \sup_{y \neq 0} \frac{\|Ay\|_W}{\|y\|_V} \sup_{x \neq 0} \frac{\|Bx\|_V}{\|x\|_U} = \|A\| \cdot \|B\|.$$

□

Wir erwähnen noch eine natürliche Eigenschaft der induzierten Matrixnorm. Die Norm der Einheitsmatrix in $\mathbb{R}^{n \times n}$ ist für eine induzierte Matrixnorm mit $V = W$ immer gleich 1.

Zum Abschluss machen wir noch einige Beispiele, die Matrixnormen genauer charakterisieren. Wir beginnen mit der sogenannten Zeilensummennorm.

BEISPIEL 2.21: Zeilensummennorm.

Wir beginnen mit dem Fall $p = q = \infty$. Es gilt per Definition

$$\|A\|_\infty = \sup_{x \neq 0} \frac{\max_j |\sum_k A_{jk} x_k|}{\max_i |x_i|}.$$

Wir beginnen mit einer einfachen Abschätzung nach oben. Es gilt für jedes $j = 1, \ldots, n$

$$\left| \sum_k A_{jk} x_k \right| \leq \sum_k |A_{jk}| \cdot |x_k| \leq \sum_k |A_{jk}| \cdot \max_i |x_i|.$$

Daraus folgt

$$\|A\|_\infty = \sup_{x \neq 0} \frac{\max_j |\sum_k A_{jk} x_k|}{\max_i |x_i|} \leq \sup_{x \neq 0} \frac{\max_j \sum_k |A_{jk}| \cdot \max_i |x_i|}{\max_i |x_i|} = \max_j \sum_k |A_{jk}|.$$

Nun wollen wir zeigen, dass sogar Gleichheit gilt. Dazu betrachten wir ein spezielles x. Sei $\ell \in \{1, \ldots, n\}$ so, dass

$$\sum_k |A_{\ell k}| = \max_j \sum_k |A_{jk}|.$$

Dann wählen wir $\overline{x}_k := \operatorname{sgn}(A_{\ell k})$ und erhalten

$$\|A\|_\infty = \sup_{x \neq 0} \frac{\max_j |\sum_k A_{jk} x_k|}{\max_i |x_i|} \geq \frac{\max_j |\sum_k A_{jk} \overline{x}_k|}{\max_i |\overline{x}_i|}.$$

Da offensichtlich aus der Wahl von $\overline{x}$ folgt $\max_i |\overline{x}_i| = 1$, gilt

$$\|A\|_\infty \geq \max_j |\sum_k A_{jk} \overline{x}_k| \geq |\sum_k A_{\ell k} \overline{x}_k| = \sum_k |A_{\ell k}| \max_j \sum_k |A_{jk}|.$$

Wir haben also gezeigt, dass die folgende Gleichung gilt:

$$\|A\|_\infty \;=\; \max_j \sum_k |A_{jk}|$$

Dies ist deutlich einfacher zu berechnen als die ursprüngliche Definition über das Supremum. Man nennt diese Norm aus offensichtlichen Gründen auch **Zeilensummennorm**.

Darüber hinaus wollen wir die wichtige Spektralnorm im Folgenden näher charakterisieren.

BEISPIEL 2.22: Spektralnorm.

Nun betrachten wir die von der *Euklidischen Norm* induzierte Matrixnorm, d.h. $p = q = 2$. Wie immer im Fall der Euklidischen Norm ist es einfacher mit dem Quadrat der Norm zu rechnen, wir betrachten also

$$\frac{\|Ax\|_2^2}{\|x\|^2} \;=\; \frac{\langle Ax, Ax\rangle}{\langle x, x\rangle} \;=\; \frac{x^T A^T A x}{x^T x}.$$

Die Matrix $A^T A$ ist symmetrisch und positiv semidefinit, also existiert eine Diagonalisierung der Form $A^T A = QDQ^T$ mit einer orthogonalen Matrix Q und einer Diagonalmatrix D aus nichtnegativen Eigenwerten von $A^T A$. Es gilt also

$$x^T A^T A x \;=\; x^T Q D Q^T x, \qquad x^T x \;=\; x^T Q Q^T x.$$

Definieren wir nun einen Vektor $y = Q^T x$, so folgt sofort

$$\|A\|_2^2 \;=\; \sup_{x\neq 0} \frac{\|Ax\|_2^2}{\|x\|^2} \;=\; \sup_{y\neq 0} \frac{y^T D y}{y^T y}. \tag{2.28}$$

Ist nun $d \in \mathbb{R}_0^+$ der größte Eintrag in D, so folgt $y^T D y \leq d y^T y$ und eingesetzt in Gleichung (2.28) somit auch $\|A\|_2^2 \leq d$.
Andererseits können wir $\overline{y}$ als einen Einheitsvektor wählen, der den Eintrag 1 in der selben Zeile hat wie der Eintrag d in der Matrix D. Also ist

$$\|A\|_2^2 \;=\; \sup_{y\neq 0} \frac{y^T D y}{y^T y} \;\geq\; \frac{\overline{y}^T D \overline{y}}{\overline{y}^T \overline{y}} \;=\; d.$$

Also haben wir gezeigt, dass wir die sogenannte **Spektralnorm** einer Matrix berechnen können als

$$\|A\|_2 \;=\; \sqrt{\lambda_{\max}(A^T A)},$$

wobei $\lambda_{\max}(A^T A) =: d$ den größten Eigenwert von $A^T A$ bezeichnet.

Eine wichtige Größe im Zusammenhang mit Matrixnormen ist auch der **Spektralradius** einer Matrix

$$\rho(A) \;:=\; \max\{|\lambda| \mid \lambda \text{ Eigenwert von } A\}.$$

Es gilt offensichtlich für jede Matrixnorm

$$\|A\| \geq \frac{\|Ax_\lambda\|}{\|x_\lambda\|} = \frac{\|\lambda x_\lambda\|}{\|x_\lambda\|} = |\lambda| \cdot \frac{\|x_\lambda\|}{\|x_\lambda\|} = |\lambda|,$$

wobei x_λ der zugehörige Eigenvektor zu λ ist. Damit ist der Spektralradius eine untere Schranke für jede Matrixnorm. Ist A eine symmetrische Matrix, so gilt sogar nach Beispiel 2.22 $\rho(A) = \|A\|_2 = \sqrt{\lambda_{\max}(A^T A)}$, da der betragsmäßig größte Eigenwert von $A^T A = A^2$ gleich dem Quadrat des betragsmäßig größten Eigenwerts von A ist.

BEMERKUNG 2.23. Allgemeiner gilt, dass für jede Matrix A und jedes $\epsilon > 0$ eine Matrixnorm existiert, so dass $\|A\| < \rho(A) + \epsilon$ ist. △

2.3.3 Konditionszahl einer Matrix

Eine wichtige Kennzahl für eine invertierbare Matrix ist die sogenannte Konditionszahl, die wir im Folgenden definieren werden.

DEFINITION 2.24: Konditionszahl.

Sei $A \in \mathbb{R}^{n\times n}$ eine invertierbare Matrix und $\|\cdot\|$ eine Matrixnorm. Dann ist die **(relative) Konditionszahl** der Matrix A definiert als:

$$\kappa(A) := \|A\| \cdot \|A^{-1}\|.$$

Aus obiger Definition wird klar, dass die Konditionszahl $\kappa(A)$ einer Matrix A direkt von der gewählten Matrixnorm abhängt. Für p-Normen schreiben wir deshalb auch

$$\kappa_p(A) := \|A\|_p \cdot \|A^{-1}\|_p.$$

Unabhängig von der Matrixnorm gilt immer $\kappa(I) = 1$ und wegen der Submultiplikativität für induzierte Matrixnormen aus Lemma 2.20 erhalten wir

$$\kappa(A) = \|A\| \cdot \|A^{-1}\| \geq \|AA^{-1}\| = \|I\| = 1.$$

Wir sehen also, dass die Konditionszahl immer größer oder gleich eins ist und bei der Einheitsmatrix ihr Minimum annimmt.

Wie wir später sehen werden ist die Konditionszahl ein entscheidendes **Stabilitätsmaß** für die Lösung linearer Gleichungssysteme.

2.3.4 Neumannsche Reihe

Betrachten wir Störungen einer regulären Matrix, so stellt sich immer die Frage, ob die gestörte Matrix dann noch regulär ist. Ist $A \in \mathbb{R}^{n\times n}$ eine reguläre Matrix und $\Delta_A := \tilde{A} - A \in \mathbb{R}^{n\times n}$ eine Störung von A, so gilt offensichtlich

$$\tilde{A} = A + \Delta_A = A\,(I + A^{-1}\Delta_A).$$

Da wir A als regulär angenommen haben ist die Matrix $\tilde{A}$ also genau dann regulär, wenn die Matrix $I + A^{-1}\Delta_A = I - B$ mit $B := -A^{-1}\Delta_A$ regulär ist. Diese lässt sich auch als eine Störung

der Einheitsmatrix auffassen. Das heißt wir müssen diese Frage nur für die Einheitsmatrix beantworten und somit verstehen unter welcher Bedingung an die Störung $B \in \mathbb{R}^{n\times n}$ wir garantieren können, dass $I - B$ noch regulär ist.

Für ein besseres Verständnis betrachten wir zunächst den eindimensionalen Fall für $n = 1$. Man sieht ein, dass die Zahl $(1 - b) \in \mathbb{R}$ invertierbar ist (also ungleich Null sein muss), wenn $b \neq 1$ gilt. Darüberhinaus wissen wir aus der Analysis, dass für die stärkere Bedingung $|b| < 1$ gilt, dass die folgende **geometrische Reihe** absolut konvergiert mit

$$\sum_{k=0}^{\infty} b^k \;=\; \frac{1}{1-b} \;=\; (1-b)^{-1}.$$

Diese Beobachtung motiviert eine analoge Aussage im höherdimensionalen Matrix-Fall, was uns zum Konzept der sogenannten Neumannschen Reihe führt.

THEOREM 2.25: Konvergenz der Neumannschen Reihe.

Sei $B \in \mathbb{R}^{n\times n}$ mit $\|B\| < 1$. Dann konvergiert die **Neumannsche Reihe**

$$S_n \;\coloneqq\; \sum_{k=0}^{n} B^k$$

für $n \to \infty$ absolut und es gilt $\lim_{n\to\infty} S_n = (I - B)^{-1}$.
Insbesondere ist $I - B$ somit invertierbar und es gilt außerdem die Abschätzung

$$\left\|(I-B)^{-1}\right\| \;\leq\; \frac{1}{1-\|B\|}.$$

Beweis. Wir zeigen zunächst, dass $(S_n)_{n\in\mathbb{N}}$ eine *Cauchy-Folge* ist. Es gilt für $m > n$ wegen der Dreiecksungleichung

$$\|S_m - S_n\| \;=\; \|\sum_{k=n+1}^{m} B^k\| \;\leq\; \sum_{k=n+1}^{m} \|B^k\|.$$

Auf Grund der Submultiplikativität der Matrixnorm folgern wir direkt $\|B^k\| \leq \|B\|^k$ und somit erhalten wir insgesamt

$$\|S_m - S_n\| \;\leq\; \sum_{k=n+1}^{m} \|B\|^k \;=\; \|B\|^{n+1} \cdot \sum_{k=0}^{m-(n+1)} \|B\|^k \;\leq\; \|B\|^{n+1} \sum_{k=0}^{\infty} \underbrace{\|B\|}_{<1}^{k} \;=\; \frac{\|B\|^{n+1}}{1-\|B\|}.$$

Da $\|B\| < 1$ ist, konvergiert die rechte Seite für $n \to \infty$ unabhängig von m absolut gegen Null, d.h. S_n ist eine Cauchy-Folge.

Betrachten wir nun den Grenzwert der Folge $(S_n)_{n\in\mathbb{N}}$. Es gilt folgende nützliche Identität

$$S_n(I-B) \;=\; S_n - \sum_{k=0}^{n} B^{k+1} \;=\; I - B^{n+1} = (I-B)S_n.$$

Wir nutzen nun aus, dass B^{n+1} gegen die Nullmatrix konvergiert und verwenden die obige Gleichung um zu zeigen, dass sowohl eine Links- und eine Rechtsinverse zur Matrix $(I - B)$

existieren und dass diese übereinstimmen:

$$S_\infty(I-B) \;=\; \lim_{n\to\infty}\sum_{k=0}^{n} B^k(I-B) \;=\; \lim_{n\to\infty}\sum_{k=0}^{n} B^k - \sum_{k=1}^{n+1} B^k \;=\; \lim_{n\to\infty} I - B^{n+1}$$
$$= I$$
$$= \lim_{n\to\infty} I - B^{n+1} \;=\; \lim_{n\to\infty}\sum_{k=0}^{n} B^k - \sum_{k=1}^{n+1} B^k \;=\; (I-B)\lim_{n\to\infty}\sum_{k=0}^{n} B^k \;=\; (I-B)S_\infty.$$

Daraus folgt also schon direkt, dass

$$S_\infty \;=\; \lim_{n\to\infty} S_n \;=\; (I-B)^{-1}.$$

Zuletzt zeigen wir noch die Abschätzung des Theorems. Wegen der Voraussetzung $||B|| < 1$ und

$$\|S_n\| \;=\; \|\sum_{k=0}^{n} B^k\| \;\le\; \sum_{k=0}^{n}\|B^k\| \;\le\; \sum_{k=0}^{\infty}\|B^k\| \;\le\; \sum_{k=0}^{\infty}\|B\|^k \;=\; \frac{1}{1-\|B\|}$$

erhalten wir die Abschätzung

$$\left\|(I-B)^{-1}\right\| \;=\; ||S_\infty|| \;=\; \left\|\lim_{n\to\infty} S_n\right\| \;\le\; \frac{1}{1-||B||}.$$

□

2.3.5 Singulärwertzerlegung

Eine weitere nützliche Technik bei der Fehleranalyse linearer Probleme ist die Singulärwertzerlegung einer Matrix. Wir erinnern zunächst an die aus der linearen Algebra bekannte Diagonalisierung symmetrischer Matrizen.

THEOREM 2.26: Diagonalisierung.

Sei $B \in \mathbb{R}^{n\times n}$ eine symmetrische Matrix, d.h. $B = B^T$. Dann existiert eine orthogonale Matrix Q, deren Spalten aus den Eigenvektoren von B bestehen und eine Diagonalmatrix D, deren Einträge die Eigenwerte von B sind, so dass gilt:

$$B \;=\; QDQ^T. \tag{2.29}$$

Eine andere Version die Diagonalisierung zu schreiben bezieht sich auf das Matrix-Vektorprodukt, denn es gilt

$$Bx \;=\; QDQ^Tx \;=\; QD\begin{pmatrix}\langle u_1, x\rangle\\ \vdots \\ \langle u_n, x\rangle\end{pmatrix} \;=\; Q\begin{pmatrix}\lambda_1\langle u_1, x\rangle\\ \vdots \\ \lambda_n\langle u_n, x\rangle\end{pmatrix} \;=\; \sum_{j=1}^{n}\lambda_j\langle u_j, x\rangle u_j.$$

wobei λ_j die Eigenwerte und u_j die zugehörigen Eigenvektoren der Matrix B sind für $j = 1,\ldots,n$.

Für allgemeinere $n \times n$ Matrizen gilt im Allgemeinen kein solches Resultat, da die Eigenvektoren im Allgemeinen keine Orthogonalbasis bilden. Für *nichtquadratische Matrizen* sind

Eigenwertprobleme gar nicht erst definiert. Deshalb verallgemeinert man bei der **Singulärwertzerlegung** einer Matrix $A \in \mathbb{R}^{m\times n}$ die Definition auf Singulärvektoren $u_i \in \mathbb{R}^n$ und $v_i \in \mathbb{R}^m$. Um diese zu konstruieren betrachten wir die beiden symmetrischen, positiv semidefiniten Matrizen

$$B_1 \coloneqq A^T A \in \mathbb{R}^{n\times n}, \qquad B_2 \coloneqq AA^T \in \mathbb{R}^{m\times m}.$$

Diese sind nach Theorem 2.26 diagonalisierbar, d.h. es gibt eine Orthonormalbasis aus Eigenvektoren, so dass man orthogonale Matrizen $U \in \mathbb{R}^{n\times n}$, $V \in \mathbb{R}^{m\times m}$ und Diagonalmatrizen $D_1 \in \mathbb{R}^{n\times n}$, $D_2 \in \mathbb{R}^{m\times m}$ erhält, so dass folgendes gilt

$$B_1 = UD_1U^T, \qquad B_2 = VD_2V^T.$$

BEMERKUNG 2.27. Zwischen den Eigenwerten der beiden Matrizen B_1 und B_2 besteht ein interessanter Zusammenhang. Sei $\lambda_i \neq 0$ ein beliebiger Eigenwert von B_1 mit Eigenvektor u_i, dann gilt wegen der Eigenwertgleichung

$$B_2Au_i \;=\; AA^TAu_i \;=\; AB_1u_i \;=\; \lambda_iAu_i.$$

Dementsprechend ist λ_i auch Eigenwert von B_2 mit Eigenvektor $v_i \coloneqq Au_i$. Umgekehrt kann man auch zeigen, dass jeder Eigenwert von B_2 auch Eigenwert von B_1 ist. Wir beachten weiter, dass für die Euklidische Norm dieses Eigenvektors v_i gilt:

$$\|v_i\|_2 \;=\; \|Au_i\|_2 \;=\; \sqrt{\langle Au_i, Au_i\rangle} \;=\; \sqrt{u_i^TA^TAu_i} \;=\; \sqrt{u_i^TB_1^Tu_i} \;=\; \sqrt{\lambda_iu_i^Tu_i} \;=\; \sqrt{\lambda_i}.$$

△

Aus obiger Beobachtung können wir nun sinnvoll die Singulärwertzerlegung einer nichtquadratischen Matrix $A \in \mathbb{R}^{m\times n}$ definieren.

DEFINITION 2.28: Singulärwertzerlegung.

Sei $A \in \mathbb{R}^{m\times n}$ eine reelle Matrix mit Rang $r \in \mathbb{N}$ mit $r \leq \min\{m, n\}$. Dann bezeichnen wir ein Matrixprodukt der Gestalt

$$A \;=\; U\Sigma V^T,$$

als **Singulärwertzerlegung (SVD)** von A, wenn $U \in \mathbb{R}^{m\times m}$ und $V \in \mathbb{R}^{n\times n}$ orthogonale Matrizen sind und $\Sigma \in \mathbb{R}^{m\times n}$ eine Diagonalmatrix ist, deren erste r Einträge positive, reelle Werte $\sigma_1, \ldots, \sigma_r$ enthält und sonst nur aus Nullen besteht.

BEMERKUNG 2.29. Für $A \in \mathbb{R}^{m\times n}, m > n$ hat die Singulärwertzerlegung von A mit $A = U\Sigma V^T$ dann folgende Form,

$$A = \begin{pmatrix} U_{11} & \dots & U_{1m} \\ \vdots & & \vdots \\ U_{m1} & \dots & U_{mm} \end{pmatrix} \begin{pmatrix} \sigma_1 & & & & \\ & \ddots & & 0 & \\ & & \sigma_r & & \\ & & & \ddots & \\ & & & & 0 \\ 0 & & & \dots & 0 \\ \vdots & & & \dots & \vdots \\ 0 & & & & 0 \end{pmatrix} \begin{pmatrix} V_{11} & \dots & V_{n1} \\ \vdots & & \vdots \\ V_{1n} & \dots & V_{nn} \end{pmatrix}.$$

Wir nennen die positiven Diagonaleinträge $\sigma_i, i = 1, \dots, r$ **Singulärwerte** der Matrix A und es lässt sich leicht zeigen, dass $\sigma_i = \sqrt{\lambda_i}$ gilt, wobei λ_i Eigenwert von $A^T A$ ist für $i = 1, \dots, r$. Die Spaltenvektoren der Matrizen U und V nennen wir **links-** bzw. **rechts-Singulärvektoren**.△

Eine andere Version die SVD zu schreiben bezieht sich auf das Matrix-Vektorprodukt, denn es gilt

$$Ax = \sum_{j=0}^{n} \sigma_j \langle u_j x \rangle v_j.$$

Wegen der Eigenschaften der orthogonalen Matrizen U und V können wir hieraus sofort folgern, dass die folgenden Gleichungen für die Singulärvektoren gelten

$$A^T u_i = \sigma_i v_i, \qquad Av_i = \sigma_i u_i.$$

BEISPIEL 2.30: Scherung.

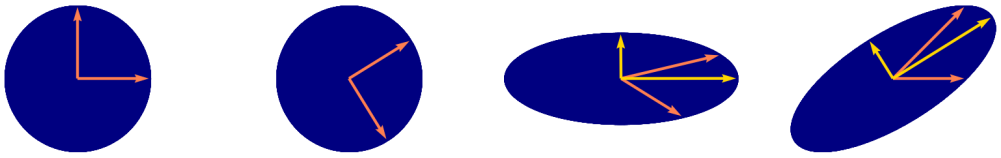
Wir betrachten die Scherungsmatrix

$$M = \begin{pmatrix} 1 & m \\ 0 & 1 \end{pmatrix}$$

zu einem Parameter $m \in \mathbb{R}$. Wir betrachten im Folgenden Punkte im Einheitskreis $x \in B_1(0)$ und dazu die Singulärwertzerlegung $M = U\Sigma V^T$ von M.
In Abb. 2.1 bilden wir die Effekte der einzelnen Teilmatrizen für $m = 1$ angewendet auf $x \in B_1(0)$ ab. In diesem Fall gilt für die Matrix Σ der Singulärwerte von M

$$\Sigma = \begin{pmatrix} \phi & 0 \\ 0 & 1/\phi \end{pmatrix}$$

wobei $\phi = (1 + \sqrt{5})/2$ der goldene Schnitt ist. Diesen Wert erhält man hier nur speziell für $m = 1$. Die orangen Pfeile zeigen am Anfang auf $e_1 = (1, 0)$ und $e_2 = (0, 1)$. Die goldenen Pfeile in (c) zeigen auf die maximale Ausdehnung in x und y Richtung und haben die jeweils die Längen $\sigma_1 = \Phi$ und $\sigma_2 = 1/\Phi$.



(a) $B_1(0)$. (b) $V^T B_1(0)$. (c) $\Sigma\, V^T B_1(0)$. (d) $U\, \Sigma\, V^T B_1(0)$.

Abbildung 2.1: Visualisierung für Beispiel 2.30.

2.4 Fehlerabschätzung bei der Lösung linearer Gleichungssysteme

Im Folgenden wollen wir verstehen, wie sich Fehler in den Eingabedaten auf die echte Lösung x linearer Gleichungssysteme der Form $Ax = b$ mit $A \in \mathbb{R}^{n\times n}$ und $b, x \in \mathbb{R}^n$ auswirken. Wir betrachten $\tilde{A}\tilde{x} = \tilde{b}$ als ein gestörtes lineares Gleichungssystem von $Ax = b$ und wollen eine Abschätzung für die Abweichung $\tilde{x} - x$ herleiten. Zu den Störungen betrachten wir jeweils

- $\Delta_x := \tilde{x} - x \in \mathbb{R}^n$ die Störung in der Lösung x,
- $\Delta_b := \tilde{b} - b \in \mathbb{R}^n$ die Störung in b,
- $\Delta_A := \tilde{A} - A \in \mathbb{R}^{n\times n}$ die Störung in A.

Dabei ist einerseits der **absolute Fehler**

$$\delta_x^{\text{abs}} := \|\tilde{x} - x\| = \|\Delta_x\| \in \mathbb{R}_0^+$$

interessant, jedoch sucht man meist aber eine Abschätzung für den **relativen Fehler**

$$\delta_x^{\text{rel}} := \frac{\|\Delta_x\|}{\|x\|} = \frac{\delta_x^{\text{abs}}}{\|x\|} \in \mathbb{R}_0^+.$$

THEOREM 2.31: Fehlerabschätzung für das gestörte LGS.

Sei $A \in \mathbb{R}^{n\times n}$ eine reguläre Matrix und $\Delta_A \in \mathbb{R}^{n\times n}$ eine Störung von A, so dass $||A^{-1}\Delta_A|| < 1$ gilt und die gestörte Matrix $\tilde{A} \in \mathbb{R}^{n\times n}$ noch regulär ist.
Betrachten wir das exakte und das **gestörte lineare Gleichungssystem**

$$Ax = b, \qquad \tilde{A}\tilde{x} = \tilde{b},$$

für $x, \tilde{x}, b, \tilde{b} \in \mathbb{R}^n$, so gilt für den **absoluten Fehler**,

$$\delta_x^{\text{abs}} \leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\| \cdot \|\Delta_A\|}(\|\Delta_A\| \cdot \|x\| + \|\Delta_b\|)$$

und für den **relativen Fehler**,

$$\delta_x^{\text{rel}} \leq \frac{\kappa(A)}{1 - \kappa(A) \cdot \frac{\|\Delta_A\|}{\|A\|}} \left(\frac{\|\Delta_A\|}{\|A\|} + \frac{\|\Delta_b\|}{\|b\|}\right). \tag{2.30}$$

Beweis. Es gilt zunächst

$$\tilde{A}(x-\tilde{x}) \;=\; \tilde{A}x-\tilde{A}\tilde{x} \;=\; \tilde{A}x-Ax+Ax-\tilde{b} \;=\; \Delta_A x-\Delta_b.$$

Da die gestörte Matrix $\tilde{A}\in\mathbb{R}^{n\times n}$ regulär ist erhalten wir auf Grund der Submultiplikativität der Matrixnorm

$$\|x-\tilde{x}\| \;=\; \|\tilde{A}^{-1}(\Delta_A x-\Delta_b)\| \;\leq\; \|\tilde{A}^{-1}\|\cdot\|\Delta_A x-\Delta_b\|. \tag{2.31}$$

Um $\|\tilde{A}^{-1}\|$ abzuschätzen, betrachten wir zunächst die folgende Identität,

$$\|(I-A^{-1}(A-\tilde{A}))^{-1}A^{-1}\| \;=\; \|(I-I+A^{-1}\tilde{A})^{-1}A^{-1}\| \;=\; \|\tilde{A}^{-1}\|.$$

Durch Anwendung des Satzes über die Neumannsche Reihe in Theorem 2.25 und wegen der Submultiplikativität der Matrixnorm erhalten wir somit die folgende Abschätzung

$$\begin{aligned}\|\tilde{A}^{-1}\| &\leq \|(I+A^{-1}\Delta_A)^{-1}\|\cdot\|A^{-1}\| \\ &\leq \frac{\|A^{-1}\|}{1-\|A^{-1}\Delta_A\|} \;\leq\; \frac{\|A^{-1}\|}{1-\|A^{-1}\|\cdot\|\Delta_A\|}.\end{aligned}$$

Eingesetzt in Gleichung (2.31) folgt nun insgesamt die Abschätzung für den **absoluten Fehler**,

$$\delta_x^{\text{abs}} \;=\; \|x-\tilde{x}\| \;\leq\; \frac{\|A^{-1}\|}{1-\|A^{-1}\|\cdot\|\Delta_A\|}(\|\Delta_A\|\cdot\|x\|+\|\Delta_b\|).$$

Wir dividieren nun durch die Norm von x und erweitern außerdem den Bruch um $\|A\|$ und erhalten

$$\begin{aligned}\frac{\|x-\tilde{x}\|}{\|x\|} &\leq \frac{\|A^{-1}\|}{1-\|A^{-1}\|\cdot\|\Delta_A\|}\cdot\frac{\|\Delta_A\|\cdot\|x\|+\|\Delta_b\|}{\|x\|}\cdot\frac{\|A\|}{\|A\|} \\ &= \frac{\|A\|\cdot\|A^{-1}\|}{1-\|A^{-1}\|\cdot\|\Delta_A\|}\left(\frac{\|\Delta_A\|}{\|A\|}+\frac{\|\Delta_b\|}{\|A\|\cdot\|x\|}\right).\end{aligned}$$

Mit $\|b\|=\|Ax\|\leq\|A\|\cdot\|x\|$ folgern wir nun die zentrale Abschätzung des **relativen Fehlers** in x durch den relativen Fehler in A und b, denn es gilt

$$\delta_x^{\text{rel}} \;=\; \frac{\|x-\tilde{x}\|}{\|x\|} \;\leq\; \frac{\kappa(A)}{1-\kappa(A)\cdot\frac{\|\Delta_A\|}{\|A\|}}\left(\frac{\|\Delta_A\|}{\|A\|}+\frac{\|\Delta_b\|}{\|b\|}\right).$$

□

Wir sehen, dass die Konditionszahl $\kappa(A)$ der Matrix A die entscheidende Konstante für die Abschätzung des relativen Fehlers ist. Ist die *Konditionszahl nahe der Eins* und der relative Fehler in der Matrix nicht zu groß, dann wird der relative Fehler in den Daten nur gering verstärkt.

Ist hingegen die *Konditionszahl groß*, so ist zunächst der erlaubte relative Fehler in der Matrix, der Regularität erhält, sehr klein (siehe auch der Nenner in der Abschätzung). Darüber hinaus wird auch die Fehlerverstärkung im Zähler signifikant. Dies gilt auch, wenn *keine Störung der Matrix A* vorliegt, denn in diesem Fall erhalten wir aus Gleichung (2.30) die kompakte Abschätzung:

$$\frac{\|x-\tilde{x}\|}{\|x\|} \;\leq\; \kappa(A)\cdot\frac{\|\Delta_b\|}{\|b\|}.$$

2.5 Kondition eines Problems und Stabilität von Verfahren

Bei der Fehleranalyse numerischer Verfahren betrachtet man typischerweise zwei verschiedene Fehlerquellen:

- die **Kondition eines mathematisches Problems**, d.h. wir betrachten die exakte Lösung eines Problems und untersuchen, wie sich Störungen auf diese Lösung auswirken,
- die **Stabilität eines numerischen Verfahrens**, d.h. wir approximieren bewusst die Lösung eines mathematisches Problem numerisch und untersuchen ob dieses Verfahren gegen die echte Lösung konvergiert.

In den folgenden beiden Abschnitten wollen wir die wichtigsten Konzepte dieser beiden Fehleranalysen einführen.

2.5.1 Kondition und Verstärkungsfaktoren

Im Folgenden erweitern wir die Definition der Konditionszahl einer Matrix in Definition 2.24 auf die Kondition allgemeiner mathematischer Probleme. Abstrakt können wir die Lösung eines mathematischen Problems meist als Funktion zwischen (normierten) Vektorräumen interpretieren, die gegebene Daten d auf eine Lösung x abbildet, d.h. $x = f(d)$. Wir interessieren uns im Folgenden insbesondere für die Änderungen des Ergebnisses $f(d)$ bei Störungen von d.

DEFINITION 2.32: Kondition eines mathematischen Problems.

Seien V, W zwei normierte Vektorräume, $f\colon V \to W$ eine Funktion und sei $\delta > 0$ die maximale Größe relevanter Störungen. Dann definieren wir die **Kondition des mathematischen Problems** $x = f(d)$ mit Daten d und Lösung x als den größtmöglichen relativen Fehler bei Störungen der Größe δ als

$$\kappa(f; d) \coloneqq \sup_{\|\Delta_d\| \leq \delta} \frac{\|f(d + \Delta_d) - f(d)\|}{\|f(d)\|} \frac{\|d\|}{\|\Delta_d\|}.$$

Im Folgenden wollen wir die obige Definition der Kondition eines mathematischen Problems auf den schon bekannten Fall der Lösung eines linearen Gleichungssystems anwenden.

BEISPIEL 2.33: Lösung eines linearen Gleichungssystems.

Wir betrachten die Lösung des linearen Gleichungssystems $Ax = b$, also nach der Notation in Definition 2.32 setzen wir $x = A^{-1}b = f(d)$ mit den gegebenen Daten $b = d$. Dann gilt

$$\begin{aligned} \kappa(A; b) &= \sup_{\|\Delta_d\| \leq \delta} \frac{\|A^{-1}(b + \Delta_b) - A^{-1}b\| \cdot \|b\|}{\|A^{-1}b\| \|\Delta_b\|} \\ &= \sup_{\|\Delta_b\| \leq \delta} \frac{\|A^{-1}\Delta_d\|}{\|A^{-1}b\|} \frac{\|b\|}{\|\Delta_d\|} = \|A^{-1}\| \frac{\|b\|}{\|A^{-1}b\|}. \end{aligned}$$

Die Zahl κ misst also die lokale Kondition bei der Lösung mit Störung von b. Wollen wir ein globales Maß, so können wir noch den schlechtest möglichen Fall betrachten, d.h.

das Supremum des Fehlers oben. Dann gilt

$$\sup_b \kappa(A;b) = \sup_b \|A^{-1}\| \cdot \frac{\|b\|}{\|A^{-1}b\|} \;=\; \|A^{-1}\| \sup_x \frac{\|Ax\|}{\|x\|} \;=\; \|A^{-1}\|\ \|A\| = \kappa(A).$$

In diesem Sinne ist unsere Definition also auch konsistent mit der Konditionszahl einer Matrix aus Definition 2.24. Wir erkennen aber auch, dass die Matrix-Konditionszahl $\kappa(A)$ deutlich gröber ist, als die Problem-Kondition $\kappa(A;b)$. Dazu plotten wir in Abb. 2.2 die beiden Größen für die Matrix

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 2 \end{pmatrix}$$

über $b \in [0,1]^2$. Wir erkennen, das der Wert $\kappa(A;b)$ meist deutlich unter $\kappa(A)$ liegt. In (a) und (b) plotten wir $\frac{\|b\|}{\|A^{-1}b\|}$ über b und die gelbe Linie im Konturplot zeigt die Richtung des ersten Links-Singulärvektoren von A^{-1} an. Wir erkennen dass der Wert von $\kappa(A;b)$ gerade für diese Vektoren maximal wird. In (c) plotten wir $\frac{\|Ax\|}{\|x\|}$ über x und die gelbe Linie im Konturplot zeigt die Richtung des ersten Links-Singulärvektoren von A^{-1} an

BEMERKUNG 2.34 (Zusammenhang SVD und Spektralnorm). Es sei $A \in \mathbb{R}^{m,n}$ eine Matrix mit SVD $A = U\Sigma V^T$ mit $r \leq \min(m,n)$ absteigend geordneten Singulärwerten. Wir sehen, dass

$$\begin{aligned} \|Ax\|_2 = \langle Ax, Ax\rangle = \langle A^T Ax, x\rangle = \langle V\Sigma_n V^T x, x\rangle = \langle \Sigma_n^2 V^T x, V^T x\rangle \\ = \sum_{i=1}^{r} \sigma_i^2 \langle v_i, x\rangle^2 \end{aligned}$$

und da $\sigma_1 \geq \ldots \sigma_r$ folgt dann

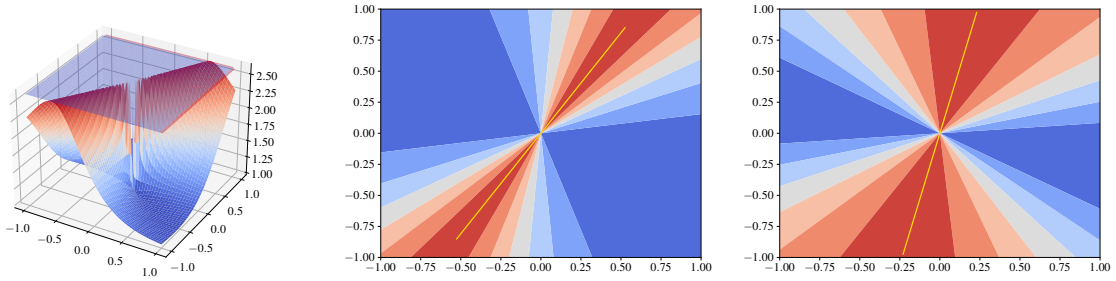
$$\|A\|_2 = \sup_{\|x\|_2=1} \|Ax\|_2 = \sigma_1 \langle v_1, v_1\rangle = \sigma_1.$$

Dies bedeutet einerseits, dass die Spektralnorm durch dn größten Singulärwert gegeben ist und andererseits, dass das Supremum in der Spektralnorm gerade für den Vektor v_1 angenommen wird. △

Da wir im Allgemeinen von kleinen Störungen Δ_d ausgehen, können wir die Kondition eines mathematischen Problems auch lokal durch *Taylor-Approximation* beschreiben. Ist $f : \mathbb{R}^n \to \mathbb{R}^m$ stetig differenzierbar, so gilt bekanntermaßen

$$f(d+\Delta_d) - f(d) \;=\; f'(\Delta_d)d + o(\delta).$$

Unter Vernachlässigung des Terms höherer Ordnung erhalten wir die sogenannten Verstärkungsfaktoren.



(a) Surface plot $\frac{\|b\|}{\|A^{-1}b\|}$. (b) Contour plot $\frac{\|b\|}{\|A^{-1}b\|}$. (c) Contour plot $\frac{\|Ax\|}{\|x\|}$.

Abbildung 2.2: Visualisierung für Beispiel 2.33.

DEFINITION 2.35: Verstärkungsfaktor.

Es sei $f : \mathbb{R}^n \to \mathbb{R}^m$ eine stetig differenzierbare Funktion, dann definieren wir den **Verstärkungsfaktor** von f für Daten d durch,

$$V(f;d) \coloneqq \sup_{\|\Delta_d\| \leq \delta} \frac{\|f'(d)\Delta_d\|}{\|f(d)\|} \cdot \frac{\|d\|}{\|\delta d\|}.$$

Damit berechnen wir nichts anderes als die Norm der Jacobi-Matrix $f'(d)$ und erhalten somit schließlich

$$V(f;d) = \frac{\|f'(d)\| \cdot \|d\|}{\|f(d)\|}. \tag{2.32}$$

BEMERKUNG 2.36. Für lineare Problem $f(b) = A^{-1}b$ mit einer Matrix $A \in \mathbb{R}^{n,n}$ gilt $f'(d) = A^{-1}$ und somit

$$V(f;b) = \frac{\|A^{-1}\| \cdot \|b\|}{\|A^{-1}b\|} = \kappa(f;b).$$

Dies passt zur Intuition, dass der Verstärkungsfaktor eine Approximation erster Ordnung an die Kondition eines Problems ist. △

Im folgenden Beispiel wollen wir die Rolle der Verstärkungsfaktoren am Beispiel quadratischer Gleichungen untersuchen.

BEISPIEL 2.37: Quadratische Gleichung.

Als Beispiel betrachten wir die Lösung quadratischer Gleichungen der Form

$$x^2 + px + q = 0$$

als Funktion $x = f(d)$ der gegeben Daten $d = (p, q)$. Nehmen wir die positive Wurzel im Fall $p^2 > 4q$, so ist

$$x = f(p,q) = -\frac{p}{2} + \sqrt{\frac{p^2}{4} - q}.$$

Wir berechnen für die Verstärkungsfaktoren zunächst die partiellen Ableitungen und erhalten

$$f'(p,q) = \left(-\frac{1}{2} + \frac{p}{2\sqrt{\frac{p^2}{4} - q}}, -\frac{1}{\sqrt{\frac{p^2}{4} - q}} \right).$$

Damit lassen sich die Verstärkungsfaktoren in Gleichung (2.32) berechnen als

$$V = \frac{\left\| \left(-\frac{1}{2} + \frac{p}{2\sqrt{\frac{p^2}{4} - q}}, -\frac{1}{\sqrt{\frac{p^2}{4} - q}} \right) \right\| \cdot \|(p,q)\|}{\| -\frac{p}{2} + \sqrt{\frac{p^2}{4} - q}\|}.$$

Wir sehen, dass wir typischerweise einen großen Verstärkungsfaktor erhalten, wenn $q \to \frac{p^2}{4}$ gilt, d.h. wenn wir nahe einer *doppelten Nullstelle* sind.

2.5.2 Stabilität numerischer Verfahren

Bisher haben wir nur das Verhalten des Problems, z.B. der exakten Lösung linearer Gleichungssysteme, unter Störungen betrachtet, aber nicht das Verhalten bei einem speziellen Verfahren mit Rundungsfehler, bzw. einem approximativen Schema wie sie in der Numerik häufig vorkommen. Zu eine Problem $x = f(d)$ mit $f : \mathbb{R}^n \to \mathbb{R}^m$ betrachten wir nun eine Folge von Funktionen

$$f_k : \mathbb{R}^n \to \mathbb{R}^m, k \in \mathbb{N}$$

als sogenanntes numerisches Verfahren, welche f approximieren soll. Wir können z.B. eine Folge von Rundungsfehlern betrachten, die gegen Null konvergieren, oder eine Iteration, die eine Lösung approximiert, wie im folgenden Beispiel.

BEISPIEL 2.38: Heron-Verfahren.

Wir betrachten das mathematische Problem

$$f(d) := \sqrt{d},$$

für $d \in \mathbb{R}_0^+$, wir wollen also die Wurzel berechnen. Das aus der Antike bekannte **Heron-Verfahren** zur Approximation der Wurzel ist durch folgendes Iterationsverfahren gegeben,

$$x_{k+1} = \frac{1}{2}\left(x_k + \frac{d}{x_k} \right),$$

mit gegebenem Anfangswert $x_0 \in \mathbb{R}_0^+$. Hier ist dann das numerische Verfahren bestimmt durch

$$f_{k+1}(d) = \frac{1}{2}\left(x_k + \frac{d}{x_k} \right).$$

Für den Anfangswert kann beispielsweise $x_0 = 1$ verwendet werden, oder eine leicht bessere Schätzung der Wurzel $x_0 = (d+1)/2$. In Abb. 2.3 visualisieren wir jeweils 10 Iteration für verschiedene d und unterschiedliche Anfangswerte.

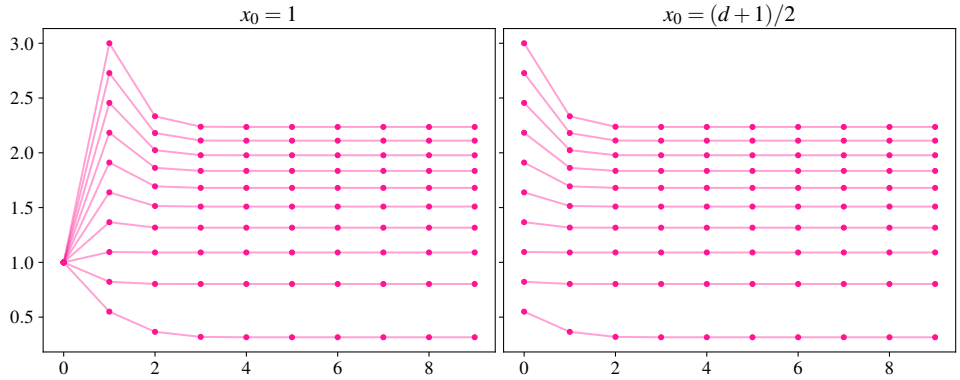


Abbildung 2.3: Verhalten des Heron-Verfahrens aus Beispiel 2.38 für verschiedene $d \in \mathbb{R}_0^+$ und unterschiedliche Anfangswerte.

Dieses Beispiel motiviert auch, dass wir werden im Folgenden $D \subset \mathbb{R}^n$ als Menge der zulässigen Daten betrachten. D.h. wir wollen die zulässigne Daten eventuell einschränken.

Eine wichtige Eigenschaft für numerische Verfahren ist die sogenannte Konsistenz. Sie besagt im wesentlichen, dass das Verfahren für die echten Daten auch gegen die echte Lösung konvergiert.

DEFINITION 2.39: Konsistenz.

Für $f : D \to \mathbb{R}^m$, heißt das numerische Verfahren $f_k : D \to \mathbb{R}^m$ **konsistent** zu f, falls

$$\lim_{k\to\infty} f_k(d) = f(d)$$

für alle $d \in D$.

Wir können anhand des Heron-Verfahrens den Begriff der Konsistenz an einem Beispiel betrachten.

BEISPIEL 2.40: Konsistenz des Heron-Verfahrens.

Wir zeigen zunächst, dass das Heron-Verfahren konvergiert, dazu sehen wir, dass für $k \in \mathbb{N}$ gilt,

$$x_{k+1}^2 = \frac{1}{4}\left(x_k + \frac{d}{x_k}\right)^2 = \frac{1}{4}\left(\left(x_k - \frac{d}{x_k}\right)^2 + 4d\right) \geq d$$

und somit $|x_k| \geq \sqrt{d}$ für alle $k \geq 1$. Wir nehmen nun an, dass $x_0 > 0$ gilt, dann zeigen wir mithilfe von Induktion, dass $x_k \geq \sqrt{d}$ für $k \geq 1$ gilt, x_k ist in diesem Fall also von unten beschränkt.

Weiterhin folgt für $k \geq 1$, dann dass

$$x_{k+1} = \frac{1}{2}\left(x_k + \frac{d}{x_k}\right) \leq \frac{1}{2}\left(x_k + \frac{x_k^2}{x_k}\right) = x_k.$$

Somit ist die Folge für $k \geq 1$ monoton fallend und beschränkt nach unten und somit konvergent. Sei nun $x^* := \lim_{k\to\infty} x_k$ der Grenzwert, dann erfüllt dieser die Fixpunktgleichung

$$x^* = \frac{1}{2}\left(x^* + \frac{d}{x^*}\right) \Leftrightarrow 2(x^*)^2 = (x^*)^2 + d \Leftrightarrow (x^*)^2 = d$$

und somit löst x^* das Problem. Weiterhin gilt $x^* = \lim_{k\to\infty} x_k \geq \sqrt{d}$ und somit gilt $x^* = \sqrt{d}$.
Für den Fall $x_0 < 0$ zeigt man analog, dass x_k eine monoton steigende Folge ist die durch $x_k \leq -\sqrt{d}$ nach oben beschränkt ist. Hier gilt dann $x^* = -\sqrt{d}$.

Aus den vorherigen Abschnitten, wissen wir bereits, dass es relevant ist Störungen in den Daten d zu betrachten. Für numerische Verfahren wollen wir bei Störungen sicherstellen, dass Konsistenz gegeben ist, sofern auch die Störung gegen null konvergiert. Dies führt auf die Definition der Konvergenz.

DEFINITION 2.41: Konvergenz.

Für $f : D \to \mathbb{R}^m$, heißt das numerische Verfahren $f_k : D \to \mathbb{R}^m$ **konvergent** zum Problem $x = f(d)$, falls

$$\lim_{k\to\infty} f_k(d_k) = f(d)$$

für alle Folgen von Daten mit $\lim_{k\to\infty} d_k \to d$.

BEISPIEL 2.42: Verrauschtes Heron-Verfahren.

Wir führen nun das Heron-Verfahren mit verrauschten Daten für $d = 2$ durch, d.h., in jedem Schritt haben wir

$$x_{k+1} = f_{k+1}(d_{k+1}) = \frac{1}{2}\left(x_k + \frac{d_{k+1}}{x_k}\right)$$

wobei wir $d_{k+1} := d + \sigma_k$ setzen wobei $\sigma_k \sim \mathcal{N}(0, \delta_k)$ eine normalverteilte Größe mit Varianz δ_k ist. Für unser Experiment wählen wir eine Nullfolge $\delta_k \to 0$ s.d. die Folge im Erwartungswert konvergiert, d.h. $\mathbb{E}(d_k) \to d$. Der Plot in **??** zeigt das Verfahren für unterschiedliche Runs, d.h. unterschiedliches zufällig gewähltes Rauschen. Man erkennt, dass das Verfahren in den meisten Fällen gegen $\sqrt{2}$ konvergiert. Falls σ_k allerdings so groß wird, dass x_{k+1} negativ wird, konvergiert das Verfahren gegen $-\sqrt{2}$.

Konvergenz ist meist schwierig direkt zu beweisen. Wir können aber einen Umweg über Stabilität nehmen. Wir definieren Stabilität im wesentlichen als gleichgradige Stetigkeit der f_k, welche besagt, dass wir die Auswirkung von Störungen auf den Algorithmus kontrollieren

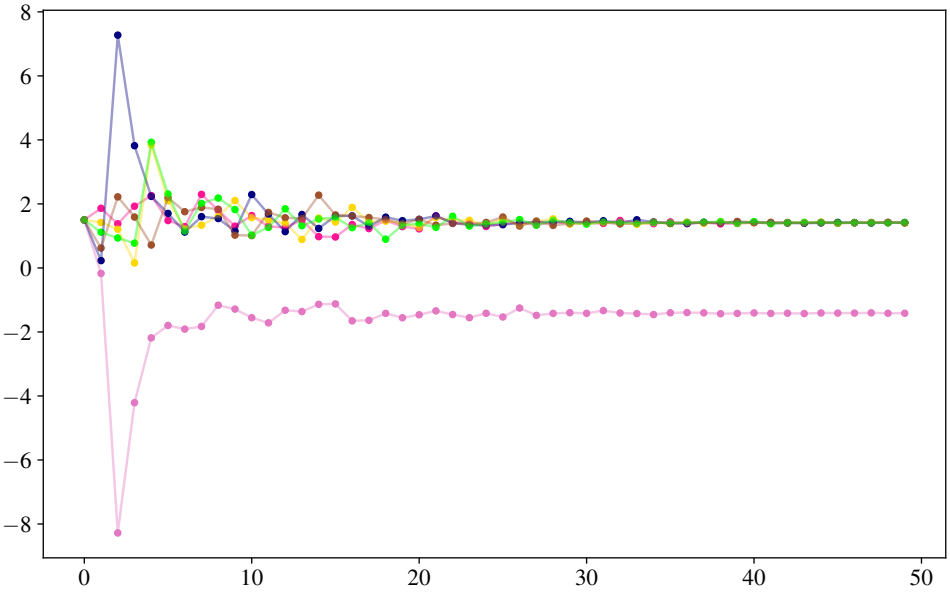


Abbildung 2.4: Visualisierung für Beispiel 2.42.

können. In diesem Sinne ist Stabilität verwandt mit dem Begriff der Kondition aus den letzten Abschnitten.

DEFINITION 2.43: Stabilität.

Ein Verfahren $f_k : D \to \mathbb{R}^m, k \in \mathbb{N}$ heißt **konsistent**, falls für jedes $\epsilon > 0$ ein $\delta > 0$, sodass für alle $k \in \mathbb{N}$ gilt

$$\|\Delta_d\| < \delta \Rightarrow \|f_k(d + \Delta_d) - f_k(d)\| < \epsilon.$$

Eine einfachere Definition ist im Falle Lipschitz-stetiger Abbildungen möglich, dann sprechen wir von Stabilität wenn eine Konstante L unabhängig von n existiert, sodass

$$\|f_k(d + \Delta_d) - f_k(d)\| \leq L\|\Delta_d\|$$

für alle $k \in \mathbb{N}$, alle zulässigen Daten $d \in D$ und zulässigen Störungen $d + \delta \in D$. Stabilität bedeutet also im wesentlichen, dass Fehler durch das Verfahren nicht zu sehr verstärkt werden.

Eine grundlegende Eigenschaft in der Numerik ist, dass sich Konvergenz aus Konsistenz und Stabilität ergibt.

THEOREM 2.44: Konvergenz eines Verfahrens.

Es sei $f_k : D \to \mathbb{R}^m, k \in \mathbb{N}$ ein Verfahren, dann gilt folgende Implikation,

$$\boxed{\textbf{Stabilität} + \textbf{Konsistenz} \Rightarrow \textbf{Konvergenz}.}$$

Beweis. Die Aussage folgt aus der Dreiecksungleichung. Sei dazu $d \in D$ und $d_k, k \in \mathbb{N}$ eine Folge von Daten, s.d. $\lim_{k\to\infty} d_k = d$. Dann gilt

$$|f_k(d_k) - f(d)| = |f_k(d_k) - f_k(d) + f_k(d) - f(d)| \leq \underbrace{|f_k(d_k) - f_k(d)|}_{\textbf{Stabilitäts-Term}} + \underbrace{|f_k(d) - f(d)|}_{\textbf{Konsistenz-Term}} .$$

Wegen der Stabilität wissen wir, dass

$$\lim_{k\to\infty} |f_k(d_k) - f_k(d)| = 0$$

und somit folgt zusammen mit der Konsistenz, dass

$$\lim_{k\to\infty} |f_k(d_k) - f(d)| = 0$$

das Verfahren ist also konvergent. □

Ist ein numerisches Verfahren nicht stabil, so können wir auch keine Konvergenz erwarten.

BEISPIEL 2.45: Stabilität des Heron-Verfahrens.

Um Stabilität zu zeigen, wiederholen wir kurz die Definition des Verfahrens. Für $d \in \mathbb{R}_0^+$ definieren wir

$$f_{k+1}(d) := \frac{1}{2}\left(f_k(d) + \frac{d}{f_k(d)}\right)$$

mit gegebenem $f_0(d)$. Die Auswertung von f_{k+1} an d, ist also die $k$te Heron-Iteration, welche stets mit d durchgeführt wurde. So erhalten wir ein wohldefiniertes Verfahren und können $f_k(d)$ mit $f_k(d + \Delta_d)$ vergleichen. Wir wollen also nun zeigen, dass ein $L > 0$ existiert, s.d.

$$|f_{k+1}(d + \Delta_d) - f_{k+1}(d)| \leq L\|\delta_d\|$$

gilt. Dazu berechnen wir

$$\begin{aligned} e_{k+1} := f_{k+1}(d + \Delta_d) - f_{k+1}(d) &= \frac{1}{2}\left(f_k(d + \Delta_d) + \frac{d + \Delta_d}{f_k(d + \Delta_d)} - f_k(d) - \frac{d}{f_k(d)}\right) \\ &= \frac{1}{2}\left((f_k(d + \Delta_d) - f_k(d))\left(1 - \frac{d + \Delta_d}{f_k(d + \Delta_d) \cdot f_k(d)}\right)\right) \\ &\quad - \frac{1}{2}\frac{d}{f_k(d)} + \frac{1}{2}\frac{d + \Delta_d}{f_k(d)} \end{aligned}$$

somit folgt

$$|e_{k+1}| \leq \frac{1}{2}|e_k|\left|1 - \frac{d + \Delta_d}{f_k(d + \Delta_d) \cdot f_k(d)}\right| + \frac{1}{2}\left|\frac{\Delta_d}{f_k(d)}\right|$$

Wir nehmen nun an, dass $f_0(d) > 0$ gilt, aus Beispiel 2.38 wissen wir dann, dass $f_k(d) \geq \sqrt{d}$ gilt und somit

$$\left|\frac{\Delta_d}{f_k(d)}\right| \leq \frac{1}{\sqrt{d}}|\Delta_d|$$

Zusätzlich wollen wir, dass für ein $q < 1$ gilt

$$\left|1 - \frac{d + \Delta_d}{f_k(d + \Delta_d) \cdot f_k(d)}\right| < 2q$$

gilt, was gegeben ist, falls

$$\frac{d + \Delta_d}{f_k(d + \Delta_d) \cdot f_k(d)} < 2q + 1.$$

Fordern wir, dass $|\Delta_d| < 2qd$ gilt so folgt

$$\frac{d + \Delta_d}{f_k(d + \Delta_d) \cdot f_k(d)} \leq \frac{\sqrt{d + \Delta_d}}{\sqrt{d}} \leq \sqrt{2q + 1} \leq 2q + 1$$

und somit insgesamt

$$|e_{k+1}| \leq q\,|e_k| + \frac{1}{\sqrt{d}}\,|\Delta_d| \leq \ldots \leq \frac{1}{\sqrt{d}}\,|\Delta_d| \sum_{j=0}^{k} q^k \leq \frac{1}{\sqrt{d}}\,|\Delta_d| \sum_{j=0}^{\infty} q^k = \frac{1}{(1-q)\sqrt{d}}\,|\Delta_d|\,.$$

Somit ist das Heron-Verfahren auf einer nach unten beschränkten Menge, $D := [a, \infty)$ mit $a > 0$ Lipschitz-stetig mit

$$|f(d + \Delta_d) - f(d)| \leq L\,|\Delta_d|$$

wobei $L = \frac{1}{(1-q)\sqrt{a}}$ mit $|\Delta_d| < 2qa$.

Kapitel 3

Über- und unterbestimmte lineare Gleichungssysteme

Nachdem wir uns in Kapitel 1 mit der numerischen Lösung eines linearen Gleichungssystems $Ax = b$ mit $A \in \mathbb{R}^{n\times n}$ und $x, b \in \mathbb{R}^n$ beschäftigt haben, wollen wir im Folgenden einen allgemeineren Fall untersuchen. Wir interessieren uns für lineare Gleichungssysteme der Form

$$Ax = b, \quad \text{für } A \in \mathbb{K}^{m\times n} \text{ und } x \in \mathbb{K}^n, b \in \mathbb{K}^m,$$

wobei $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$ wahlweise den Körper der reellen oder der komplexen Zahlen bezeichnet. Wir unterscheiden zwei Fälle für $m \neq n$.

DEFINITION 3.1.

Es sei $A \in \mathbb{K}^{m\times n}$ eine Matrix und $b \in \mathbb{K}^m$, dann heißt das Gleichungssystem

$$Ax = b \tag{UB-LGS}$$

- **überbestimmt**, falls $m > n$,
- **unterbestimmt**, falls $n > m$,

gilt.

Im Falle überbestimmter Systeme übersteigt die Anzahl der Gleichungen die der unbekannten Variablen. Wir haben also mehr Gleichungen als Variablen. Im Falle unterbestimmter Systeme übersteigt die Anzahl der Gleichungen die der unbekannten Variablen. Wir haben also weniger Gleichungen als Variablen.

BEMERKUNG 3.2. Im Allgemeinen sind sowohl unterbestimmte als auch überbestimmte lineare Gleichungssysteme in Problem UB-LGS nicht lösbar. Aus diesem Grund muss man sich andere Strategien und Kriterien überlegen, um sinnvolle Aussagen über den unbekannten Vektor $x \in \mathbb{K}^n$ treffen zu können. △

Der Fall eines überbestimmten Gleichungssystems ist äußerst relevant, da er in vielen Anwendungen auftritt, wie z.B. bei der Bildrekonstruktion in der Computertomographie wenn

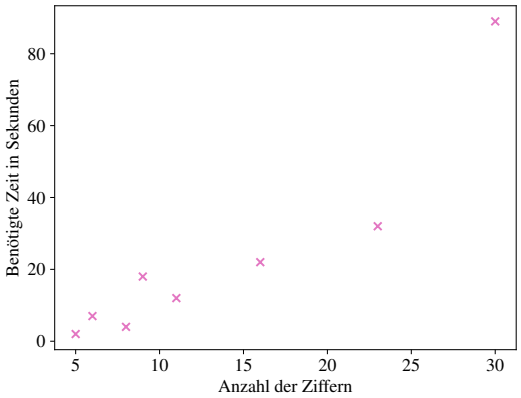


Abbildung 3.1: Visualisierung der Daten in Beispiel 3.3.

die Anzahl der gemessenen Röntgenstrahlen m die Anzahl der zu rekonstruierenden Pixel n übertrifft.

BEISPIEL 3.3: Polynom-Fit.

Eine Gruppe von $N \in \mathbb{N}$ Studierenden bekommt die Aufgabe jeweils $n_i \in \mathbb{N}$ verschieden natürliche Zahlen zu addieren. Wir messen jeweils Zeit t_i die dieser Vorgang benötigt und erhalten folgende Daten:

Anzahl der Ziffern n	5	6	8	9	11	16	30	23
Benötigte Zeit t in Sekunden	2	7	4	18	12	22	89	32

In Abb. 3.1 werden die Daten visualisiert. Wir erwarten einen Zusammenhang zwischen der Anzahl der Ziffern und der Berechnungszeit, welcher sich durch ein Polynom

$$t = f(t) = p_1 n + p_0$$

modellieren lässt mit unbekannten Koeffizienten $p_1, p_0 \in \mathbb{R}$. Für jeden Studierenden $i = 1, \ldots, N$ erhalten wir die Gleichung

$$p_1 n_i + p_0 = t_i$$

und ausgedrückt als Gleichungssystem ergibt dies

$$\underbrace{\begin{pmatrix} n_1 & 1 \\ \vdots & \\ n_N & 1 \end{pmatrix}}_{:=A} \begin{pmatrix} p_1 \\ p_0 \end{pmatrix} = \underbrace{\begin{pmatrix} t_1 \\ \vdots \\ t_N \end{pmatrix}}_{:=b}.$$

Wir erhalten also ein überbestimmtes Gleichungssystem.

3.1 Gaußsche Normalengleichungen und Ausgleichsprobleme

In diesem Abschnitt beschäftigen wir uns mit Lösungsbegriffen, welche es uns erlauben überbestimmte Gleichungssystem, d.h., Problem UB-LGS für $m > n$, zu lösen.

3.1.1 Lineare Ausgleichsprobleme

Da eine exakte Lösung eines überbestimmten linearen Gleichungssystems in der Regel nicht möglich ist, versucht man die Abweichung von der idealen Lösung bezüglich einer Vektornorm zu minimieren. Typischerweise wählt man die Euklidische Norm, so dass man anstatt ursprünglich Problem UB-LGS für $m > n$ nun ein Problem der folgenden Form hat, welches man auch lineares Ausgleichsproblem nennt.

DEFINITION 3.4: Lineares Ausgleichsproblem.

Es sei $A \in \mathbb{K}^{m\times n}$ eine Matrix und $b \in \mathbb{K}^m$ ein Vektor, dann heißt das Minimierungsproblem

$$\min_{x\in\mathbb{K}^n} ||b - Ax||_2 \tag{LSTSQ}$$

lineares Ausgleichsproblem. Eine Lösung $x \in \mathbb{K}^n$ nennt man **Kleinste-Quadrate-Lösung** oder auch **Least-squares**-Lösung.

BEISPIEL 3.5.

Wir betrachten das Least-Squares Ausgleichsproblem aus Beispiel 3.3 für $A \in \mathbb{R}^{N,2}, b \in \mathbb{R}^N$. In Abb. 3.2 plotten wir zunächst die Funktion $p \mapsto \|Ap-b\|$ über $\mathbb{R}^2$. Wir erkennen, dass die Funktion konvex ist mit elliptischen Niveaulinien. Zusätzlich markieren wir drei Punkte p^1, p^2, p^3, welche auf dem rechten Plot in ihrer Entsprechung als Polynom Funktion $n \mapsto (p^i)_1 x + (p^i)_0$ für $i = 1, 2, 3$. Der pink gefärbte Punkt bzw. die pink gefärbte Gerade stellt hierbei die Lösung von Problem LSTSQ dar

$$\hat{p} \approx \begin{pmatrix} 2.9 \\ -16.3 \end{pmatrix}.$$

Die vertikalen Linien im Plot visualisieren jeweils die Fehler zu den Daten, welche quadriert addiert werden.

Wir wollen uns im Folgenden mit numerischen Verfahren zur Berechnung von Kleinste-Quadrate-Lösungen und deren Eigenschaften beschäftigen.

3.1.2 Gaußsche Normalengleichung

Wir untersuchen zunächst die Eigenschaften einer Matrix, welche von links mit ihrer adjungierten Matrix multipliziert wurde, siehe Abschnitt 1.4. Hierbei verwenden für $A \in \mathbb{K}^{m\times n}$ wir die folgende Notation,

- der Kern der Matrix ist gegeben durch,

$$\mathcal{N}(A) := \{x \in \mathbb{K}^n : Ax = 0\}$$

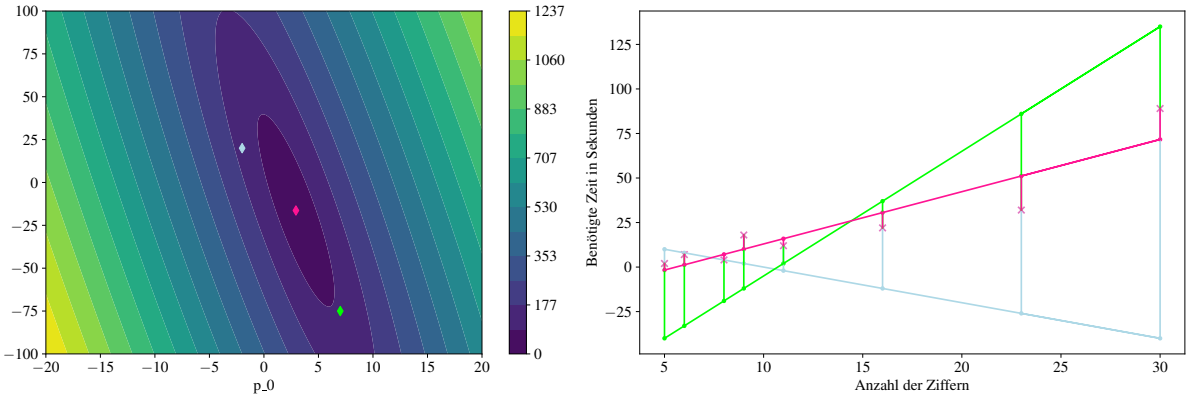


Abbildung 3.2: Visualisierung des Ausgleichsproblems für Beispiele 3.3 und 3.5.

- das Bild der Matrix ist gegeben durch,

$$\mathcal{R}(A) := A(\mathbb{K}^{m\times n}) = \{Ax : x \in \mathbb{K}^n\}.$$

Für einen Untervektorraum $U \subset \mathbb{K}^k$ wir definieren das orthogonale Komplement

$$U^\perp := \{z \in \mathbb{K}^k : \langle u, z\rangle = 0, \quad \forall z \in U\},$$

hier haben wir das Resultat (da $\mathbb{K}^k$ endlich-dimensional und das Skalarprodukt nicht ausgeartet ist), dass

$$\mathbb{K}^k = U \oplus U^\perp$$

wobei die Summe für zwei Untervektorräume $V, W \subset \mathbb{K}^k$ definiert ist durch $V + W := \{v + w : v \in V, w \in W\}$ und wir $V \oplus W$ schreiben, falls $V \cap W = \{0\}$ gilt. Aus der linearen Algebra wissen, wir, dass ein auf U orthogonal stehender Untervektorraum V genau dann der Komplementärraum ist, $V = U^\perp$ falls

$$k = \dim U + \dim V$$

gilt.

LEMMA 3.6.

Es sei $A \in \mathbb{K}^{m\times n}$ eine Matrix, dann gilt,

(i) $A^*A \in \mathbb{K}^{n\times n}$ ist hermitesch und positiv semi-definit,

(ii) die Matrix A^*A ist genau dann positiv definit, falls der Kern von A trivial ist, d.h., falls $\mathcal{N}(A) = \{0\}$ gilt,

(iii) für den Kern und das Bild gilt

$$\mathcal{N}(A^*A) = \mathcal{N}(A) \quad \text{und} \quad \mathcal{R}(A^*A) = \mathcal{R}(A^*) = \mathcal{N}(A)^\perp.$$

Beweis.
Ad (i).
Die Hermizität der Matrix A^*A folgt offensichtlich aus der Kommutativität der komponentenweisen Multiplikation in $\mathbb{K}$. Ferner gilt für alle $x \in \mathbb{K}^n$

$$\langle x, A^*Ax\rangle \;=\; x^*A^*Ax \;=\; (Ax)^*Ax \;=\; \langle Ax, Ax\rangle \;=\; \|Ax\|_2^2 \;\geq 0. \tag{3.33}$$

Also ist A^*A immer positiv semi-definit.
Ad (ii).
Da

$$\|Ax\|_2^2 > 0 \text{ für } x \neq 0 \Leftrightarrow \mathcal{N}(A) = \{0\}$$

sehen wir mit Gleichung (3.33), dass

$$A \text{ ist positiv definit} \Leftrightarrow \mathcal{N}(A) = \{0\}.$$

Ad (iii).
Aus Gleichung (3.33) folgt bereits, dass $\mathcal{N}(A^*A) \subset \mathcal{N}(A)$ gilt. Die umgekehrte Inklusion $\mathcal{N}(A) \subset \mathcal{N}(A^*A)$ gilt trivialerweise, da für ein beliebiges $x \in \mathcal{N}(A)$

$$(A^*A)x \;=\; A^*\underbrace{(Ax)}_{=0} \;=\; 0.$$

Somit haben wir also $\mathcal{N}(A^*A) = \mathcal{N}(A)$ bewiesen.
Wir zeigen nun, dass $\mathcal{R}(A^*)$ und $\mathcal{N}(A)$ orthogonale aufeinander stehen. Seien also $z \in \mathcal{R}(A^*)$ und $x \in \mathcal{N}(A)$ beliebig, dann gibt es ein $y \in \mathbb{K}^m$, s.d. $z = A^*y$ gilt und damit

$$\langle x, z\rangle \;=\; x^*z \;=\; x^*A^*y \;=\; (Ax)^*y \;=\; 0^*y \;=\; 0.$$

Da dies für beliebige Vektoren $z \in \mathcal{R}(A^*)$ und $x \in \mathcal{N}(A)$ gilt, müssen $\mathcal{N}(A)$ und $\mathcal{R}(A^*)$ orthogonal sein. Aus der **Dimensionsformel** wissen wir, dass

$$\dim \mathcal{N}(A) + \dim \mathcal{R}(A^*) \;=\; n$$

gilt, und daher $\mathcal{R}(A^*) = \mathcal{N}(A)^\perp$ gilt. Diese Identität gilt allgemein für lineare Operatoren, also auch $\mathcal{R}(A^*A) = \mathcal{N}(A^*A)^\perp$ und somit

$$\mathcal{R}(A^*) = \mathcal{N}(A)^\perp = \mathcal{N}(A^*A)^\perp = \mathcal{R}(A^*A). \tag{3.34}$$

□

Mit Hilfe der obigen Eigenschaften lässt sich nun ein Zusammenhang zwischen dem linearen Ausgleichsproblem und den **Gaußschen Normalengleichungen** herstellen.

THEOREM 3.7: Normalengleichungen.

Sei $A \in \mathbb{R}^{m\times n}$ und $b \in \mathbb{R}^m$, ein Vektor $\hat{x} \in \mathbb{R}^n$ löst genau dann die Gaußschen Normalengleichungen

$$A^TA\hat{x} \;=\; A^Tb \qquad \text{(Gauß-NGL)}$$

falls es das linearen Ausgleichsproblem Problem LSTSQ löst.

Beweis.
Für $A = 0$ ist die Aussage trivial. Wir nehmen also an, dass $A \neq 0$ gilt.
Schritt 1:
Es sei zunächst $\hat{x}$ eine Lösung von Problem LSTSQ. Unter Ausnutzung von Lemma 3.6 haben wir

$$\mathbb{K}^m \;=\; \mathcal{R}(A) \oplus \mathcal{R}(A)^\perp \;=\; \mathcal{R}(A) \oplus \mathcal{N}(A^*)$$

und daher können wir den Vektor $b \in \mathbb{K}^m$ schreiben als

$$b \;=\; b_\mathcal{R} + b_\mathcal{N} \quad \text{mit} \quad b_\mathcal{R} \in \mathcal{R}(A) \;\text{ und }\; b_\mathcal{N} \in \mathcal{N}(A^*).$$

Wir zeigen zunächst, dass $A\hat{x} = b_\mathcal{R}$ gilt. Da $b_\mathcal{R} \in \mathcal{R}(A)$ im Bild liegt, existiert mindestens ein $\tilde{x} \in \mathbb{K}^n$ mit $A\tilde{x} = b_\mathcal{R}$. Für beliebiges $x \in \mathbb{K}^n$ gilt dann

$$\begin{aligned}
\|b - Ax\|_2^2 \;&=\; \|b_\mathcal{R} + b_\mathcal{N} - Ax\|_2^2 \;=\; \|A\tilde{x} + b_\mathcal{N} - Ax\|_2^2 \;=\; \|A\underbrace{(\tilde{x} - x)}_{=:z} + b_\mathcal{N}\|_2^2 \\
&=\; \|Az - b_\mathcal{N}\|_2^2 \;=\; \langle Az, Az\rangle + \underbrace{\langle Az, b_\mathcal{N}\rangle + \langle b_\mathcal{N}, Az\rangle}_{=0} + \|b_\mathcal{N}\|_2^2 \\
&=\; \|Az\|_2^2 + \|b_\mathcal{N}\|_2^2
\end{aligned}$$

Hierbei haben wir ausgenutzt, dass $Az \in \mathcal{R}(A)$ orthogonal zu $b_\mathcal{N} \in \mathcal{N}(A^*)$ ist. Der obige Ausdruck ist minimal genau dann wenn $Az = 0$, d.h., für solche $x \in \mathbb{K}^n$ s.d.

$$\tilde{x} - x \in \mathcal{N}(A).$$

Da $\hat{x}$ das Problem löst, folgt also, dass

$$A\hat{x} = A\hat{x} + \underbrace{A(\tilde{x} - \hat{x})}_{=0} = A\tilde{x} = b_\mathcal{R}.$$

Wegen $A^* b_\mathcal{N} = 0$ gilt dann

$$A^*b \;=\; A^*(b_\mathcal{R} + b_\mathcal{N}) \;=\; A^* b_\mathcal{R} \;=\; A^*A\hat{x}.$$

Das heißt $\hat{x}$ ist Lösung der Normalengleichung Gauß-NGL.
Schritt 2:
Sei nun umgekehrt $\hat{x} \in \mathbb{R}^n$ eine Lösung der Normalengleichung, dann wissen wir, dass das Residuum $A\hat{x} - b$ im Kern der Matrix A^* liegen muss, denn es gilt

$$A^*A\hat{x} \;=\; A^*b \quad\Leftrightarrow\quad A^*(A\hat{x} - b) \;=\; 0 \quad\Leftrightarrow\quad A\hat{x} - b \in \mathcal{N}(A^*).$$

Für beliebiges $x \in \mathbb{R}^n$ definieren wir

$$z \;:=\; A(x - \hat{x}) \in \mathcal{R}(A), \quad r \;:=\; b - A\hat{x} \in \mathcal{N}(A^*),$$

so gilt wegen $\langle r, z\rangle = 0$

$$\|b - Ax\|_2^2 \;=\; \|r - z\|_2^2 \;=\; \|r\|_2^2 + \underbrace{\langle r, z\rangle + \langle z, r\rangle}_{=0} + \|z\|_2^2 \;\geq\; \|r\|_2^2 \;=\; \|b - A\hat{x}\|_2^2.$$

Damit haben wir gezeigt, dass

$$\|b - Ax\|_2^2 \geq \|b - A\hat{x}\|_2^2 \quad \forall x \in \mathbb{K}^n,$$

also, dass $\hat{x}$ eine Lösung des linearen Ausgleichsproblems Problem LSTSQ ist. □

In obigen Beweis haben wir für $A\tilde{x} = b_{\mathcal{R}}$ die Identität

$$\|b - Ax\|_2 = \|A(\tilde{x} - x)\|_2 + \|b_{\mathcal{N}}\|_2$$

gezeigt. Somit sehen wir, dass Problem LSTSQ stets mindestens eine Lösung, nämlich $\hat{x}$ hat. Für den Fall das $\mathcal{N}A = \{0\}$ existiert **genau ein** $\hat{x}$ s.d. $A\tilde{x} = b_{\mathcal{R}}$ gilt und dieses $\hat{x}$ ist dann auch die eindeutige Lösung des Problems. Dies ist der Inhalt des folgenden Korollars.

KOROLLAR 3.8.

Für eine Matrix $A \in \mathbb{K}^{m\times n}$ mit $\mathcal{N}(A) = \{0\}$ gibt es **genau eine** Lösung von Problem LSTSQ die auch die eindeutige Lösung der Normalengleichung Gleichung (Gauß-NGL).

BEMERKUNG 3.9. Falls die Matrix A^*A singulär ist, d.h. $\mathcal{N}(A) \neq \{0\}$ so haben die Gaußschen Normalengleichungen keine eindeutige Lösung und es existieren mehrere Vektoren $\hat{x} \in \mathbb{R}^n$, s.d.

$$A^*A\hat{x} = A^*b$$

△

BEMERKUNG 3.10. Für **Schritt 1** im Beweis von Theorem 3.7 gibt es folgenden alternativen Weg für $\mathbb{K} = \mathbb{R}$. Wir betrachten die Funktion $\phi : \mathbb{R}^n \to \mathbb{R}$,

$$\phi(x) := \frac{1}{2}\|b - Ax\|_2^2 = \frac{1}{2}\|b\|_2^2 - \langle A^*b, x\rangle + \frac{1}{2}\langle A^*Ax, x\rangle,$$

für welche wir in Problem LSTSQ ein Minimum $\hat{x}$ finden wollen. Eine notwendige Bedingung für ein Minimum ist hier durch

$$\nabla\phi(\hat{x}) = 0$$

gegeben. Um den Gradienten zu berechnen, benutzen wir, dass für zwei Funktionen $f, g : \mathbb{R}^n \to \mathbb{R}^m$ gilt

$$\nabla\ \langle f(x), g(x)\rangle = (\nabla f)(x)\ g(x) + (\nabla g)(x)\ f(x)$$

wobei $(\nabla f), (\nabla g)$ die Jacobi-Matrizen bezeichnen. Damit sehen wir, dass

$$\begin{aligned}
\nabla\phi(x) &= \underbrace{\nabla\frac{1}{2}\|b\|_2^2}_{=0} - \nabla\left(\langle A^*b, x\rangle\right) + \frac{1}{2}\nabla\left(\langle A^*Ax, x\rangle\right) \\
&= -\underbrace{\left(\nabla(A^*b)\right)}_{=0}x - \underbrace{\left(\nabla(x)\right)}_{=I}A^*b + \frac{1}{2}\underbrace{\left(\nabla(A^*Ax)\right)}_{=A^*A}x + \frac{1}{2}\underbrace{\left(\nabla(x)\right)}_{=I}A^*Ax \\
&= A^*b - A^*Ax
\end{aligned}$$

somit wissen wir, dass ein Minimum $\hat{x}$ von ϕ die Gleichung

$$A^*b - A^*Ax = 0 \Leftrightarrow A^*b = A^*Ax$$

also die Normalengleichung erfüllt. △

BEISPIEL 3.11.

Die Systemmatrix und die rechte Seite aus Beispiel 3.3 hatten die Form

$$A = \begin{pmatrix} 5 & 1 \\ 6 & 1 \\ 8 & 1 \\ 9 & 1 \\ 11 & 1 \\ 16 & 1 \\ 30 & 1 \\ 23 & 1 \end{pmatrix}, \qquad b = \begin{pmatrix} 2 \\ 7 \\ 4 \\ 18 \\ 12 \\ 22 \\ 89 \\ 32 \end{pmatrix}.$$

Die Normalengleichung führt hier dann auf

$$A^TAx = A^Tb \Leftrightarrow \begin{pmatrix} 2012 & 108 \\ 108 & 8 \end{pmatrix} x = \begin{pmatrix} 4136 \\ 186 \end{pmatrix}$$

was als Lösung gerade den Vektor $\hat{p} \approx (2.9, -16.3)^T$ aus Beispiel 3.5 hat. Die Norm des Fehlers beträgt hier

$$\|A\hat{p} - b\|^2 \approx 29.6$$

was mit dem Residuum der kleinsten Quadrate übereinstimmt,

$$\sqrt{\sum_{i=1}^{N} |t_i - (\hat{p}_1 n_i + \hat{p}_0)|^2} \approx 29.6,$$

welche in Abb. 3.2 visualisiert sind.

3.1.3 Normalengleichung und SVD

Zum Abschluss diskutieren wir noch eine Darstellung der Lösung der Normalengleichung mit Hilfe der Singulärwertzerlegung für $A \in \mathbb{R}^{m\times n}$, $A = U\Sigma V^T$.

LEMMA 3.12.

Für eine Matrix $A \in \mathbb{R}^{m\times n}$ mit SVD $A = U\Sigma V^T$ ist Lösung der Normalengleichung gegeben durch

$$x = V\Sigma^{-1}U^Tb.$$

BEMERKUNG 3.13. Für die Diagonalmatrix $\Sigma \in \mathbb{R}^{m\times n}$ mit r Singulärwerten $\sigma_1 \geq \ldots \geq \sigma_r > 0$, bezeichnen wir mit $\Sigma^{-1} \in \mathbb{R}^{n\times m}$ die Diagonalmatrix s.d.

$$\left(\Sigma^{-1}\right)_{ii} = \begin{cases} \frac{1}{\sigma_i} & \text{für } i = 1, \ldots, r, \\ 0 & \text{sonst.} \end{cases}$$

△

Beweis. Da die rechts-Singulärvektoren v_i auch die Eigenvektoren von A^TA sind, können wir damit schreiben,

$$A^TAx = \left(V\Sigma^TU^TU\Sigma V^T\right)x = \sum_{i=1}^{r} \sigma_i^2 (v_i^T x) v_i, \qquad A^Tb = \sum_{i=1}^{r} \sigma_i (u_i^T b) v_i.$$

Durch Koeffizientenvergleich sehen wir direkt

$$v_i^T x = \frac{1}{\sigma_i} u_i^T b$$

und damit

$$x = \sum_{i=1}^{r} \frac{1}{\sigma_i} (u_i^T b) v_i$$

bzw. in Matrixschreibweise

$$x = V\Sigma^{-1}U^Tb.$$

□

Die Singulärwertzerlegung erlaubt uns auch eine einfache Analyse der Kondition der Normalengleichung in der 2-Norm. Wir nehmen an, dass $\mathcal{N}(A) = \{0\}$ gilt und somit $r = n$ ist, da wir im überbestimmten Fall $m > n$ sind. Für die Kondition gilt dann

$$\kappa_2(A^TA) = \frac{\sigma_1^2}{\sigma_n^2}$$

Für beliebiges $z \in \mathbb{R}^m$ bekommen wir für eine Gleichung der Form $A^TAx = y$ also nur die Abschätzung

$$\delta_x^{\text{rel}} \leq \frac{\sigma_1^2}{\sigma_n^2} \frac{\|\Delta_z\|}{\|z\|}.$$

Im Fall einer regulären $n \times n$ Matrix ist dies das Quadrat der Kondition von A, somit könnten wir erwarten, dass die Normalengleichung deutlich schlechter konditioniert ist als das ursprüngliche Gleichungssystem. Dies stimmt aber nur bedingt, wie wir aus einem einfachen Argument sehen. Wenn die Störung nur in der rechten Seite b vorliegt, so ist die Differenz der Lösung

$$x - \tilde{x} = V\Sigma^{-1}U^T(b - \tilde{b}).$$

Damit gilt für den relativen Fehler

$$\delta_x^{\text{rel}} = \frac{\|V\Sigma^{-1}U^T(b-\tilde{b})\|}{\|V\Sigma^{-1}U^Tb\|} = \frac{\|\Sigma^{-1}U^T(b-\tilde{b})\|}{\|\Sigma^{-1}U^Tb\|},$$

wobei wir ausgenutzt haben, dass V eine orthogonale Matrix ist. Da aber

$$\frac{1}{\sigma_1} \leq (\Sigma^{-1})_{ii} \leq \frac{1}{\sigma_n}, \quad i = 1, \dots, n,$$

gilt, folgt für einen Vektor $z \in \mathbb{R}^m$

$$\frac{1}{\sigma_1}\|z\| \leq \|\Sigma^{-1}z\| \leq \frac{1}{\sigma_n}\|z\|.$$

Daraus folgern wir, dass

$$\frac{\|x-\tilde{x}\|}{\|x\|} \leq \frac{\sigma_1}{\sigma_n}\frac{\|U^T(b-\tilde{b})\|}{\|U^Tb\|} = \frac{\sigma_1}{\sigma_n}\frac{\|b-\tilde{b}\|}{\|b\|}$$

gilt. Also erhalten wir in der Fehlerabschätzung doch nur $\sqrt{\kappa_2(A^TA)}$ als Konstante und nicht $\kappa_2(A^TA)$. Der Grund dafür ist, dass wir bei der Normalengleichung nicht eine beliebige rechte Seite haben, sondern sehr speziell $y = A^Tb$.

BEMERKUNG 3.14 (Störung in der Matrix). Die Situation ändert sich, wenn wir auch Fehler in der Matrix A zulassen. Dann haben wir eine Gleichung der Form

$$\tilde{A}^T\tilde{A}(x-\tilde{x}) = (\tilde{A}^T\tilde{A} - A^TA)x + \tilde{A}^T\tilde{b} - A^Tb,$$

und es gibt keinen Grund warum die rechte Seite als Produkt von $\tilde{A}^T$ mit einem Fehler von Interesse dargestellt werden soll. In diesem Fall ist also $\kappa_2(A^TA)$ entscheidend und wir sehen, dass die Normalengleichungen deutlich sensitiver gegenüber einem Fehler in der Matrix sind als gegenüber einem Fehler in der rechten Seite. △

3.2 QR-Zerlegung für überbestimmte Systeme

Wir haben oben die Lösung überbestimmter Systeme auf die Lösung der Normalengleichung zurückgeführt, die wir z.B. mit der Cholesky-Zerlegung lösen können. Dies ist aber nicht in allen Fällen der optimale Weg um das Problem zu lösen. Erstens kann sich die Kondition der Matrix stark erhöhen. Bei einer invertierbaren Matrix gilt $\kappa_2(A^TA) = \kappa_2(A)^2$, siehe Bemerkung 3.14. Zweitens kann der numerische Aufwand, der zum Aufstellen der Normalengleichung benötigt wird, schon dominant sein. Die Berechnung von A^TA benötigt einen Aufwand in der Ordnung von $\mathcal{O}(mn^2)$, während die anschließende Lösung mit Hilfe der Cholesky-Zerlegung in Algorithmus 1.36 nur einen Rechenaufwand der Ordnung $\mathcal{O}(n^3)$ hat. Im stark überbestimmten Fall ($m >> n$) dominiert also der Aufwand zum Aufstellen der Normalengleichung.

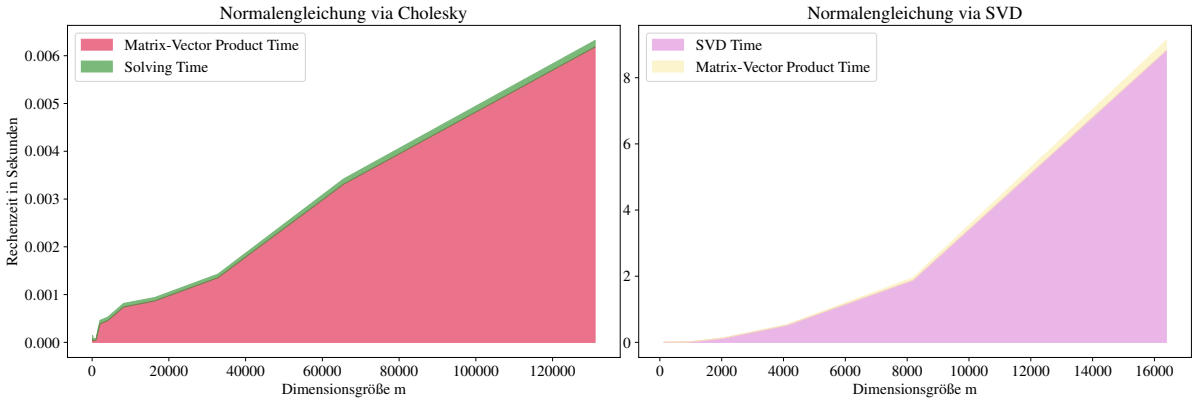


Abbildung 3.3: Visualsierung für Beispiel 3.15.

BEISPIEL 3.15: Rechenzeit Normalengleichung.

Wir wollen das Ausgleichsproblem mithilfe der Normalengleichung lösen. Hierfür fixieren wir $n = 5$ und ziehen zufällige Matrizen $A \in \mathbb{R}^{m\times n}$ für $m = 2^i, i = 1, \dots, N \in \mathbb{N}$. In Abb. 3.3 stellen wir links die Rechenzeit zur Lösung des Problems mit der Cholesky-Zerlegung dar. Der rot gefärbte Bereich stellt hierbei den Anteil der Produkte

$$A^T A, \quad A^T b$$

dar, der grüne den Anteil des Lösens mit Cholesky. Wir erkennen, dass die Zeit um das Produkt zu berechnen die Gesamtzeit dominiert, der Aufwand für Cholesky bleibt wie erwartet konstant, da wir $n = 5$ fixieren. Auf der rechten Seite, stellen wir die Variante dar, das Problem mithilfe der SVD zu lösen, was wesentlich mehr Zeit benötigt. Hier dominiert allerdings die Zeit die SVD zu berechnen.

3.2.1 Die QR-Zerlegung

Als Alternative betrachten wir wieder direkte Zerlegungsmethoden zur Lösung des Problems. Im Prinzip könnten wir versuchen genauso wie bei der LR-Zerlegung vorzugehen, allerdings ist unklar welche Art von Lösung wir damit berechnen würden. Insbesondere ist nicht zu erwarten, dass wir die Kleinste-Quadrate Lösung berechnen.

Problem: Die Anwendung einer Frobenius-Matrix kann die Norm des Residuums bzw. auch die Normalengleichungen relativ unkontrolliert verändern.

Es gibt also Fälle bei denen für eine Frobeniusmatrix L gilt

$$\|L(b - Ax)\| \neq \|b - Ax\|,$$
$$A^T L^T L A \neq A^T A$$

und somit verlieren wir so die Kontrolle über das Residuum und auch die Normalengleichung.

Wunsch: Transformationen welche das Residuum und die Normalengleichung nicht verändern!

Geeignet ist hier eine Transformation mit orthogonalen Matrizen Q, da diese das Residuum und die Normalengleichung nicht verändern,

$$\|Q(b - Ax)\| = \|b - Ax\|,$$
$$A^T Q^T Q A = A^T A.$$

DEFINITION 3.16: QR-Zerlegung.

ür eine Matrix $A \in \mathbb{R}^{m\times n}$ mit $m \geq n$ nennt man eine Zerlegung mit einer orthogonalen Matrix $Q \in \mathbb{R}^{m\times m}$ und einer oberen Dreiecksmatrix $R \in \mathbb{R}^{m\times n}$,

$$A = QR$$

eine **QR-Zerlegung** von A.

BEMERKUNG 3.17. Die obere Dreiecksmatrix $\mathbb{R} \in \mathbb{R}^{m\times n}$ ist hierbei von der Form

$$R = \begin{pmatrix} R^{(1)} \\ 0 \end{pmatrix}$$

wobei $R^{(1)} \in \mathbb{R}^{n\times n}$ eine quadratische obere Dreiecksmatrix ist und $0 \in \mathbb{R}^{(m-n)\times n}$ Nullmatrix ist, welche die Dimensionen entsprechend auffüllt. Somit sehen wir, wenn wir $Q = (Q^{(1)}|Q^{(2)})$ in die ersten n Spalten $Q^{(1)} \in \mathbb{R}^{m\times n}$ und die letzten $(m-n)$ Spalten $Q^{(2)} \in \mathbb{R}^{(m-n)\times n}$ aufteilen, dass

$$QR = (Q^{(1)}|Q^{(2)}) \begin{pmatrix} R^{(1)} \\ 0 \end{pmatrix} = Q^{(1)} R^{(1)}$$

gilt. Somit reicht es, $Q^{(1)}$ und $R^{(1)}$ zu berechnen, was die sogenannte **reduzierte** QR-Zerlegung ist. △

Haben wir nun eine QR-Zerlegung gegeben, so sehen wir, dass

$$\|b - Ax\|^2 = \|Q^T(b - Ax)\|^2 = \|Q^T b - Rx\|^2 = \|(Q^{(1)})^T b^{(1)} - R^{(1)}x\|^2 + \|Q^{(2)} b^{(2)}\|^2$$

wobei wir $b = (b^{(1)}|b^{(2)})$ setzen. Wir sehen also, dass das Residuum für

$$x = \left(R^{(1)}\right)^{-1} (Q^{(1)})^T b^{(1)}$$

minimiert, was wie bei der LR-Zerlegung über Rückwärtseinsetzen gelöst werden kann.

3.2.2 QR-Verfahren mit Reflexionen und Rotationen

Wir benötigen nun in Verfahren, welches eine QR-Zerlegung liefert.

Wunsch: Effizientes Verfahren um eine QR-Zerlegung zu berechnen!

Die Idee des **QR-Verfahrens** ist es eine Folge von orthogonalen Matrizen $Q_1, \ldots, Q_n$ zu konstruieren, sodass

$$R = Q_n \ldots Q_1 A$$

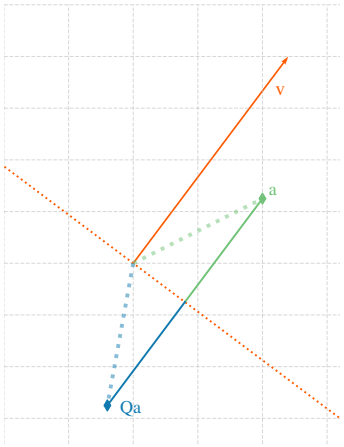


Abbildung 3.4: Visualisierung für Beispiel 3.20.

eine rechte obere Dreiecksmatrix ist. Um die Matrizen Q_j zu konstruieren haben wir im wesentlichen zwei Möglichkeiten, nämlich eine *Drehung* (genannt Givens-Rotation) oder eine *Spiegelung* (genannt Householder Reflexion).

DEFINITION 3.18: Householder-Reflexion.

Für einen Vektor $v \in \mathbb{R}^k$ definieren wir die **Householder**-Matrix

$$Q_v := I - 2vv^T \in \mathbb{R}^{k\times k}.$$

KOROLLAR 3.19.

Für einen Vektor $v \in \mathbb{R}^k, \|v\|_2 = 1$ ist die Householder-Matrix $Q_v \in \mathbb{R}^{k\times k}$ symmetrisch, orthogonal und selbstinvers.

Beweis. Siehe Hausaufgabe. □

BEISPIEL 3.20: Householder-Matrizen.

Wir betrachten für einen Vektor den Vektor $v = \frac{1}{5}(3,4)^T$ die Householder-Matrix

$$Q_v = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \frac{2}{25}\begin{pmatrix} 3 \\ 4 \end{pmatrix}(3,4) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \frac{2}{25}\begin{pmatrix} 9 & 12 \\ 12 & 16 \end{pmatrix} = \begin{pmatrix} 0.28 & -0.96 \\ -0.96 & -0.28 \end{pmatrix}.$$

In Abb. 3.4 visualisieren wir den Effekt der Matrix Q angewendet auf einen Vektor x. Wir erkennen, dass Qx gerade die Spiegelung von x an der Hyperebene ist, die durch v aufgespannt wird.

Ziel der Multiplikation mit Q_j ist es immer Nullen unter der Diagonale zu erzeugen, das bedeutet im Fall von Q_1, dass wir die erste Spalte der Matrix $A = (a_1|\dots|a_n)$ auf ein Vielfaches des ersten Einheitsvektors e_1 spiegeln wollen. Es soll also für ein $\lambda \in \mathbb{R}$ gelten

$$Q_1 a_1 = \lambda e_1.$$

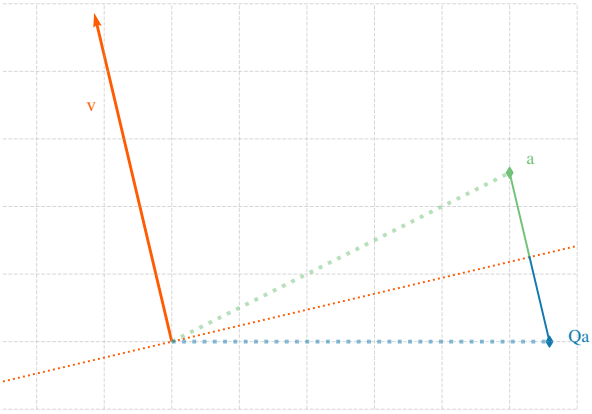


Abbildung 3.5: Visualisierung einer Spiegelung des Vektors a auf e_1.

In Beispiel 3.20 und Abb. 3.4 sehen wir, dass die Householder-Matrix Q_v eine Spiegelung an der Hyperebene mit Normalenvektor v realisiert. Wir wollen nun also ein $v \in \mathbb{R}^m$ bestimmen, s.d. der Vektor $a_1 \in \mathbb{R}^m$ gerade auf e_1 gespiegelt wird. Da sich 2-Norm bei einer Spiegelung nicht verändern kann, fordern wir also, dass

$$\begin{aligned} Q_1 a_1 = a_1 - 2\langle v_1, a_1\rangle v_1 &= \|a_1\|_2 e_1 \\ \Rightarrow v_1 = \frac{-1}{2\langle v_1, a_1\rangle}\,(a_1 - \|a_1\|_2 e_1) & \end{aligned}$$

Da wir aber zusätzlich fordern, dass $\|v_1\| = 1$ gilt, folgt

$$v_1 = \frac{a_1 - \|a_1\| e_1}{\|a_1 - \|a_1\| e_1\|}$$

für $a_1 \neq \lambda e_1, \lambda \in \mathbb{R}$. Falls a_1 schon ein Vielfaches von e_1 ist, brauchen wir in diesem Schritt keine Transformation und setzen $v_1 = 0$. Durch die Anwendung von Q_1 erhalten wir dann

$$Q_1 A = \underbrace{\begin{pmatrix} R_{1,1} & R_{1,2} & \dots & R_{1,n} \\ 0 & & & \\ \vdots & & \tilde{A}_1 & \\ 0 & & & \end{pmatrix}}_{:=A_1}$$

wir haben also die erste Spalte von A auf den Einheitsvektor gespiegelt. Im zweiten Schritt reicht es die Submatrix $\tilde{A}_1 \in \mathbb{R}^{(m-1)\times(m-1)}$ zu betrachten, wobei wir die erste Spalte $\tilde{a}_2 \in \mathbb{R}^{m-1}$, welche wir auf den Einheitsvektor $e_1^{(m-1)} \in \mathbb{R}^{m-1}$ spiegeln wollen, mit

$$\begin{aligned} \tilde{v}_2 &= \frac{\tilde{a}_2^1 - \|\tilde{a}_2^1\| e_1^{(m-1)}}{\|\tilde{a}_2^1 - \|\tilde{a}_2^1\| e_1^{(m-1)}\|} \\ \tilde{Q}_2 &= I - 2\tilde{v}_2 \tilde{v}_2^T. \end{aligned}$$

Die Householder Matrix, die die erste Zeile und Spalte unverändert lässt, ist von der Form

$$Q_2 = \begin{pmatrix} 1 & 0 \\ 0 & \tilde{Q}_2 \end{pmatrix} = I - 2v_2v_2^T, \qquad v_2 = \begin{pmatrix} 0 \\ \tilde{v}_2 \end{pmatrix}$$

und wir erhalten dann

$$Q_2\ (Q_1A) = Q_2 \begin{pmatrix} R_{1,1} & R_{1,2} & \dots & R_{1,n} \\ 0 & & & \\ \vdots & & \tilde{A}_1 & \\ 0 & & & \end{pmatrix} = \begin{pmatrix} R_{1,1} & R_{1,2} & \dots & R_{1,n} \\ 0 & & & \\ \vdots & & \tilde{Q}_2\tilde{A}_1 & \\ 0 & & & \end{pmatrix}$$

$$= \begin{pmatrix} R_{1,1} & R_{1,2} & \dots & R_{1,n} \\ 0 & R_{2,2} & \dots & R_{2,n} \\ \vdots & 0 & & \\ \vdots & \vdots & \tilde{A}_2 & \\ 0 & 0 & & \end{pmatrix}.$$

Für die weiteren Schritte gehen wir analog vor und erhalten somit folgenden Algorithmus.

ALGORITHMUS 3.21: Householder QR-Verfahren.

function QR(A) # QR-Verfahren für $A \in \mathbb{R}^{m\times n}$.

$Q \leftarrow I \in \mathbb{R}^{m\times m}$ # Initialisiere Matrix Q
$R \leftarrow A \in \mathbb{R}^{m\times n}$ # Initialisiere Matrix R

for $j = 0, \dots, n-1$ **do** # Schleife über alle Spalten.
$\tilde{v} \leftarrow R[j:m, j]$ # Spalte unter j-ten Diagonalelement
$\tilde{v}[0] \leftarrow \tilde{v}[0] - \|\tilde{v}\|$ # Konstruiere Spiegelvektor
$\tilde{v} \leftarrow \tilde{v}\ /\ \|\tilde{v}\|$ # Normalisierung

$\tilde{Q}_j \leftarrow I^{(m-j)} - 2 * v@v^T$ # Konstruiere Spiegelmatrix

$R[j:m, j:n] \leftarrow \tilde{Q}@R[j:m, j:n]$ # Spiegele auf $e_1^{(m-j)}$
$Q[j:m, :] \leftarrow \tilde{Q}@Q[j:m, :]$ # Update Q
end for
return Q^T, R
end function

Wir sehen direkt, dass uns dieses Verfahren tatsächlich eine QR-Zerlegung liefert.

KOROLLAR 3.22: QR-Verfahren.

Für eine Matrix $A \in \mathbb{R}^{m\times n}$ liefert das QR-Verfahren in Algorithmus 3.21 eine QR-Zerlegung,

$$A = QR$$

wobei insgesamt ca.

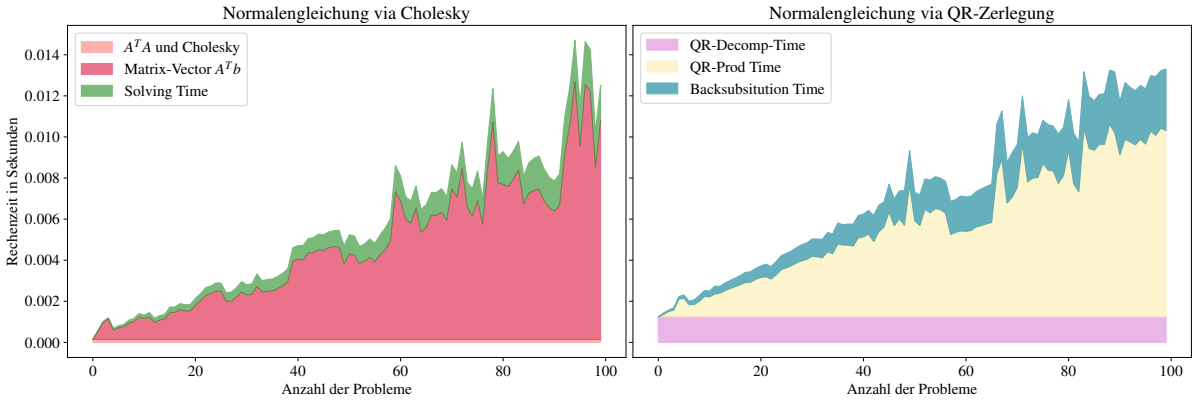


Abbildung 3.6: Visualisierung für Beispiel 3.23.

- $2m \cdot n^2 - 2/3n^3$ Operationen durchgeführt werden, falls nur R berechnet wird,
- $4(m^2 \cdot n + n^3/3 - m \cdot n^2)$ Operationen durchgeführt werden, falls auch Q berechnet wird.

Beweis. Siehe Hausaufgabe. □

BEISPIEL 3.23.

Um den Aufwand des QR-Verfahrens zu visualisieren, fixieren wir eine Matrix $A \in \mathbb{R}^{m \times n}$ für $m = 2^{16}, n = 5$ und ziehen jeweils $k \in \mathbb{N}$ zufällige Vektoren für welche wir das Ausgleichsproblem Problem LSTSQ lösen wollen,

$$A^T Ax = A^T b.$$

Da die Matrix A fixiert ist können wir für den Cholesky-Fall einmal das Produkt $A^T A$ berechnen, was $n^2 m$ Operationen benötigt, zusammen mit der Cholesky Zerlegung haben wir hier also $1/3n^3 + mn^2 + \mathcal{O}(n^2)$ Operationen. Hinzukommen, dann $m \cdot n$ Operationen für das Produkt $A^T b$ und $\mathcal{O}(n^2)$ Operationen für die Lösung des Systems. Für die Lösung mit der QR Zerlegung brauchen wir zunächst $m \cdot n^2 - 2/3n^3$ Operationen um die QR-Zerlegung zu berechnen, $m \cdot n$ Operationen für das Produkt $(Q^{(1)})^T b^{(1)}$ und nochmal $\mathcal{O}(n^2)$ Operationen für das Lösen via Rückwärtseinsetzen.
In der Praxis verwendet man allerdings Algorithmen zur Matrixmultiplikation $A^T A$, deren Aufwand besser ist, als $m \cdot n^2$. Bei der QR-Zerlegung ist liefert diese Optimierung kleinere Vorteile, weshalb das QR-Verfahren Nachteile in der Laufzeit hat, siehe Abb. 3.6.

3.3 Die Minimum-Norm-Lösung

Wir haben uns in den vorherigen Abschnitten hauptsächlich mit linearen Gleichungssystemen der Form $Ax = b$ mit $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$ und $b \in \mathbb{R}^n$ für den Fall $m \geq n$ und $\text{rank}(A) = n$ beschäftigt. Für diesen Fall konnten wir bisher immer eine eindeutige Lösung des linearen

Ausgleichsproblems

$$\min_{x\in\mathbb{R}^n} \|b - Ax\|_2$$

angeben. Wie wir jedoch in Bemerkung 3.1.2 bereits festgestellt haben liefern die Normalengleichungen keine eindeutige Lösung falls der Rang $\operatorname{rank}(A)$ der Matrix nicht voll ist und somit die Matrix A^*A singulär ist. Das folgende Beispiel verdeutlicht, dass man in diesem Fall keine eindeutige Lösung erwarten kann.

BEISPIEL 3.24.

Sei $A \in \mathbb{R}^{m\times n}$ und $b \in \mathbb{R}^m$ beliebig. Für eine Lösung der Normalengleichung $x \in \mathbb{R}^n$

$$A^*Ax \;=\; A^*b.$$

sehen wir, dass für jedes $n \in \mathcal{N}(A)$ gilt,

$$A^*A(x+n) = A^*Ax = A^*b.$$

Ist nun $\operatorname{rank}(A) < n$, so existiert $0 \neq n \in \mathcal{N}(A)$ und $x, x+n$ sind zwei verschiedene Lösungen.

Um selbst im nicht-eindeutigen Fall einen sinnvollen Begriff einer Lösung des linearen Gleichungssystems $Ax = b$ zu definieren führen wir eine zusätzliche Bedingung an den Lösungsvektor $x \in \mathbb{R}^n$ ein. Wir wählen aus allen Lösungen den mit der Kleinsten Norm aus.

DEFINITION 3.25: Minimum-Norm-Lösung.

Sei $A \in \mathbb{R}^{m\times n}$ und $b \in \mathbb{R}^n$. Ein Vektor $x^+ \in \mathbb{R}^n$ heißt **Minimum-Norm-Lösung** von $Ax = b$ genau dann, wenn folgende Bedingungen erfüllt sind,

(i) x^+ ist Kleinste-Quadrate-Lösung von $Ax = b$, (siehe Problem LSTSQ),

(ii) x^+ hat von allen existierenden Kleinste-Quadrate-Lösungen die kleinste Norm, d.h.,

$$\|x^+\|_2 \leq \|x\|_2$$

für alle Kleinste-Quadrate-Lösungen x.

BEMERKUNG 3.26. In der obigen Definition der Minimum-Norm-Lösung werden keinerlei Voraussetzungen an die Dimensionen m und n, sowie den Rang $\operatorname{rank}(A)$ von A gemacht. △

Um den Zusammenhang der Kleinste-Quadrat-Lösungen und der Minimum-Norm-Lösungen zu verstehen wollen wir das Problem wieder geometrisch deuten. Wir wissen, dass das lineare Gleichungssystem $Ax = b$ genau dann lösbar ist, wenn die rechte Seite im Bild der Matrix A liegt, d.h., wenn $b \in \mathcal{R}(A)$ gilt. Falls dem nicht so ist, so haben wir in Abschnitt 3.1 den Vektor b orthogonal auf das Bild von A projiziert. Die so erhaltenen Lösungen sind die Kleinste-Quadrate-Lösungen. Falls diese Lösungen nicht eindeutig sind so können wir wie folgt vorgehen. Da jede Kleinste-Quadrate-Lösung x sich darstellen lässt als

$$x \;=\; s + r, \quad s \in \mathcal{R}(A^*),\ r \in \mathcal{N}(A),$$

und der Anteil r im Kern von A keine Bedeutung für die Lösung von $Ax = b$ besitzt, so können wir orthogonal auf den Raum $\mathcal{R}(A^*)$ projizieren. Diese Projektion ist eindeutig bestimmt und liefert uns die Minimum-Norm-Lösung $x^+ \in \mathcal{R}(A^*)$.

Der folgende Satz präzisiert die obige geometrische Betrachtung weiter gehend.

THEOREM 3.27: Minimum-Norm-Lösung.

Sei $A \in \mathbb{R}^{m\times n}$ und $b \in \mathbb{R}^m$. Der Vektor $x^+ \in \mathbb{R}^n$ ist genau dann die eindeutige Minimum-Norm-Lösung von $Ax = b$, falls gilt

(i) $A^TAx^+ = A^Tb$, d.h., x^+ löst die Normalengleichung,

(ii) $x^+ \in \mathcal{R}(A^T)$.

Beweis. Es seien x_1, x_2 zwei Lösungen von Problem LSTSQ, dann existiert nach Lemma 3.6 jeweils $r_1, r_2 \in \mathcal{R}(A^T)$ und $n_1, n_2 \in \mathcal{N}(A)$, s.d.

$$x_1 = r_1 + n_1, \qquad x_2 = r_2 + n_2.$$

Da beide Problem LSTSQ lösen folgt,

$$\begin{aligned} A^TAr_1 = A^TA(r_1 + n_1) = A^Tb = A^TA(r_2 + n_2) = A^TAr_2 \\ \Rightarrow A^TA(r_1 - r_2) = 0 \\ \Rightarrow r_1 - r_2 \in \mathcal{N}(A^TA). \end{aligned}$$

Da aber $\mathcal{N}(A^TA) = \mathcal{R}(A^T)^\perp$ und $r_1 - r_2 \in \mathcal{R}(A^T)$ gilt folgt $r_1 - r_2 = 0$. Somit haben wir gezeigt, dass es ein eindeutig bestimmtes $r \in \mathcal{R}(A^T)$ gibt s.d. jede Lösung von Problem LSTSQ von der Form $\hat{x} = r + \hat{n}$ für $\hat{n} \in \mathcal{N}(A)$ ist. Wir setzten nun

$$x^+ = r.$$

Wir können außerdem für jede Kleinste-Quadrate-Lösung $\hat{x} \in \mathbb{R}^n$ folgende Abschätzung treffen

$$\|\hat{x}\|_2^2 \;=\; \|x^+ + \hat{n}\|_2^2 \;=\; \|x^+\|_2^2 + 2\underbrace{\langle x^+, n\rangle}_{=0} + \|n\|_2^2 \;=\; \|x^+\|_2^2 + \|n\|_2^2 \;\geq\; \|x^+\|_2^2.$$

Eine echte Gleichheit in obiger Ungleichung erhält man genau dann wenn $r = 0$ ist und somit $\hat{x} = x^+$ gilt. Die Minimum-Norm-Lösung ist also eindeutig durch x^+ bestimmt. □

3.4 Die Moore–Penrose Inverse

In diesem Abschnitt beschäftigen wir uns mit einer Verallgemeinerung der Matrix-Inversen, welche über die Minimum-Norm-Lösung gegeben ist.

DEFINITION 3.28: Moore–Penrose Inverse.

Für eine Matrix $A \in \mathbb{R}^{m\times n}$, nennen wir die Abbildung

$$\begin{aligned} A^+ \colon \mathbb{R}^m &\to \mathbb{R}^n \\ b &\mapsto A^+b := x^+ \end{aligned}$$

Moore–Penrose Inverse, wobei $A^+b = x^+$ jeweils die eindeutige Minimum-Norm-Lösung des LGS $Ax = B$ bezeichnet.

BEMERKUNG 3.29. Weitere Namen dieser Abbildung in der Literatur sind **Pseudo-Inverse** oder **verallgemeinerte Inverse**. Wir sehen, dass die Abbildung A^+ nach obiger Definition linear ist. Für eine Matrix $A \in \mathbb{R}^{m\times n}$ ist eine äquivalente Definition gegeben falls die folgenden Bedingungen gelten,

(i) $AA^+A = A$, d.h. A^+ ist eine sogenannte verallgemeinerte Inverse im allgemeinen Sinne,

(ii) $A^+AA^+ = A$, d.h. A^+ ist eine sogenannte schwache Inverse,

(iii) $(AA^+)^T = AA^+$,

(iv) $(A^+A)^T = A^+A$.

Eigenschaft (i) und (ii) sehen wir auch in Aufgabe 1, Theorie Blatt 07. △

Wir betrachten im Folgenden ein Szenario in dem die Matrix A vollen Rang hat, d.h., es gilt $\operatorname{rank}(A) = \min(m, n)$. In diesem Fall lässt sich nämlich die Minimum-Norm-Lösung aus Abschnitt 3.3 durch Matrix-Inversion berechnen.

THEOREM 3.30: Moore–Penrose Inverse.

Es sei $A \in \mathbb{R}^{m\times n}$ eine Matrix mit $\operatorname{rank}(A) = \min(m, n)$. Die Moore–Penrose Inverse $A^+ \in \mathbb{R}^{n\times m}$ ist linear und es lassen sich folgende Fälle unterscheiden:

(i) $\operatorname{rank}(\mathbf{A}) = \mathbf{m} = \mathbf{n}$: Die quadratische Matrix $A \in \mathbb{R}^{n\times n}$ ist invertierbar und es existiert eine eindeutige Lösung $x^+ \in \mathbb{R}^n$ des linearen Gleichungssystems $Ax = b$. In diesem Fall entspricht die Moore–Penrose Inverse der normalen Inversen von A, d.h. es gilt
$$A^+ = A^{-1}.$$

(ii) $\operatorname{rank}(\mathbf{A}) = \mathbf{n} < \mathbf{m}$: Im überbestimmten Fall ist die Matrix $A \in \mathbb{R}^{m\times n}$ injektiv und A^TA ist invertierbar. In diesem Fall ist die Moore–Penrose Inverse gegeben als
$$A^+ = (A^TA)^{-1}A^T.$$

(iii) $\operatorname{rank}(\mathbf{A}) = \mathbf{m} < \mathbf{n}$: Im unterbestimmten Fall ist die Matrix $A \in \mathbb{R}^{m\times n}$ surjektiv und AA^T ist invertierbar. In diesem Fall ist die Moore–Penrose Inverse gegeben als
$$A^+ = A^T(AA^T)^{-1}.$$

Beweis. Sei $b \in \mathbb{R}^m$ und $A \in \mathbb{R}^{m\times n}$ habe vollen Rang.

(i) Für den Fall $\operatorname{rank}(A) = m = n$ existiert eine eindeutige Lösung $x^+ \in \mathbb{R}^n$ des linearen Gleichungssystems nach Kapitel 1. Außerdem wissen wir aus der linearen Algebra, dass die Inverse A^{-1} einer Matrix A eindeutig bestimmt ist. Daher muss also schon gelten $A^+ = A^{-1}$.

(ii) Für den Fall $\text{rank}(A) = n < m$ ist die Dimension des Bildes $\mathcal{R}(A)$ größer als die Dimension des Raumes $\mathbb{R}^n$. Da der Rang voll ist wissen wir aus der linearen Algebra, dass die durch die Matrix A induzierte Abbildung injektiv sein muss. Nach Lemma 3.6 wissen wir, dass $\mathcal{N}(A^TA) = \mathcal{N}(A)$ gilt und somit auch die durch $A^TA \in \mathbb{R}^{n\times n}$ induzierte Abbildung injektiv sein muss. Dadurch ist A^TA aber bereits invertierbar und wir können schreiben

$$x^+ \;=\; (A^TA)^{-1}A^Tb.$$

Die Minimum-Norm-Lösung x^+ ist also wegen $\text{rank}(A) = n$ die einzige Kleinste-Quadrate-Lösung des überbestimmten linearen Gleichungssystems $Ax = b$ und für die Moore–Penrose Inverse gilt $A^+ = (A^TA)^{-1}A^T$.

(iii) Für den Fall $\text{rank}(A) = m < n$ ist die Dimension des Bildes $\mathcal{R}(A)$ kleiner als die Dimension des Raumes $\mathbb{R}^n$. Da der Rang voll ist wissen wir aus der linearen Algebra, dass die durch die Matrix A induzierte Abbildung surjektiv sein muss. Das bedeutet, dass es exakte Lösungen $\hat{x} \in \mathbb{R}^n$ mit $A\hat{x} = b$ geben muss, die alle bereits Kleinste-Quadrate-Lösungen mit Residuum $\|b - A\hat{x}\|_2 = 0$ sind. Außerdem wissen wir, dass $A^T \in \mathbb{R}^{n\times m}$ injektiv sein muss, denn es gilt

$$\text{rank}(A^T) = \text{rank}(A) \;=\; m.$$

Das bedeutet jedoch auch, dass $AA^T \in \mathbb{R}^{m\times m}$ injektiv und somit invertierbar sein muss. Da wir die Minimum-Norm-Lösung definiert haben über die Eigenschaft $x^+ \in \mathcal{R}(A^T)$ muss es mindestens ein $s \in \mathbb{R}^m$ geben mit $A^Ts = x^+$. Sei nun x^+ die Minimum-Norm-Lösung des unterbestimmten linearen Gleichungssystems $Ax = b$, dann gilt

$$b \;=\; Ax^+ \;=\; A(A^Ts) \;=\; AA^Ts.$$

Wegen der Invertierbarkeit von AA^T wissen wir also, dass $s = (AA^T)^{-1}b$ gilt. Damit können wir aber schon die Moore–Penrose Inverse von A wie folgt bestimmen:

$$x^+ \;=\; A^Ts \;=\; A^T(AA^T)^{-1}b$$

und Die Minimum-Norm-Lösung x^+ des unterbestimmten linearen Gleichungssystems $Ax = b$ lässt sich also für $\text{rank}(A) = m$ durch die Moore–Penrose Inverse $A^+ = A^T(AA^T)^{-1}$ bestimmen.

□

BEMERKUNG 3.31. Es ist naheliegend für die Berechnung einer Minimum-Norm-Lösung $x^+ \in \mathbb{R}^n$ die expliziten Formeln aus Theorem 3.30 zu benutzen. Wie wir jedoch in Abschnitt 3.2 festgestellt haben verhält sich die Kondition $\kappa_2(A^TA)$ der Matrix A^TA bei fehlerbehafteter Matrix A wie $(\kappa_2(A))^2$, was unter Umständen zu einer sehr großen Verstärkung von Anfangsfehlern führen kann. Außerdem liegt der Rechenaufwand zur Berechnung von A^TA in $\mathcal{O}(mn^2)$ was im Fall von wesentlich mehr Messdaten als unbekannte Variablen, d.h. für den Fall $m >> n$, besonders teuer werden kann. Zur Vermeidung dieser Probleme empfiehlt sich also die Inversion durch Anwendung der QR-Zerlegung von A (siehe Abschnitt 3.2). △

Kapitel 4

Iterative Lösungsverfahren für (nicht-)lineare Gleichungssysteme

Nachdem wir uns in den letzten Kapiteln mit der Lösung linearer Gleichungssysteme beschäftigt haben, wenden wir uns nun einer deutlich schwierigeren Aufgabe zu. Wir betrachten im Folgenden **nichtlineare Nullstellengleichungen** der allgemeinen Form

$$F(x) \;=\; 0, \tag{4.35}$$

für eine nichtlineare Abbildung $F\colon \mathbb{R}^n \to \mathbb{R}^n$. Mit Ausnahme von quadratischen und kubischen Gleichungen im Fall $n = 1$ existieren in der Regel keine Lösungsformeln. Es ist also von vornherein ausgeschlossen, dass wir eine direkte Lösungsmethode wie bei linearen Gleichungssystemen konstruieren können.

4.1 Banachscher Fixpunktsatz

Da sich im Allgemeinen keine geschlossenen Lösungen für nichtlineare Gleichungen der Form (4.35) angeben lassen, ist der Standardansatz eine *iterative Approximation der Lösung* zu berechnen, wie beim Beispiel des Heron-Verfahrens zur Berechnung der Quadratwurzel. Wir betrachten dazu **allgemeine Fixpunktiterationen** der Form

$$x_{k+1} = G(x_k), \tag{4.36}$$

für eine Funktion $G\colon \mathbb{R}^n \to \mathbb{R}^n$ mit denen wir sukzessive die Lösung der Fixpunktgleichung

$$x \;=\; G(x)$$

approximieren.

Man beachte, dass es in den meisten Fällen genügt Fixpunktiterationen zu betrachten, die nur auf der letzten Iterierten $x^k \in \mathbb{R}^n$ basieren, wie folgende Bemerkung erklärt.

BEMERKUNG 4.1. Liegt ein allgemeines Fixpunktverfahren der folgenden Form vor

$$x_{k+1} \;=\; G(x_k, x_{k-1}, \ldots, x_{k-m}),$$

mit $G\colon \underbrace{\mathbb{R}^n \times \ldots \times \mathbb{R}^n}_{m\text{-mal}} \to \mathbb{R}^n$ und $m \in \mathbb{N}, m > 1$, so müssen wir das Problem nur in den entsprechenden Vektorraum $\mathbb{R}^{m\times n}$ einbetten mit $\tilde{x}_k := (x_k, x_{k-1}, \ldots, x_{k-m})$ und erhalten x_{k+1} als Funktion von $\tilde{x}_k$ durch

$$x_{k+1} \;=\; \tilde{G}(\tilde{x}_k).$$

△

Natürlich sollte die Fixpunktgleichung (4.36) mit der allgemeinen Funktion G zum ursprünglichen Nullstellenproblem (4.35) mit der Funktion F äquivalent sein. Dazu müssen wir das Nullstellenproblem lediglich zu einer Fixpunktgleichung umformen, z.B. durch

$$x \;=\; \underbrace{x + F(x)}_{=:G(x)}, \qquad x \;=\; \underbrace{x - F(x)}_{=:G(x)}, \qquad x \;=\; \underbrace{x + H(F(x))}_{=:G(x)},$$

mit einer invertierbaren Funktion $H\colon \mathbb{R}^n \to \mathbb{R}^n$ für die $H(0) = 0$ gilt.

Wir haben also eine große Auswahl an Fixpunktiterationen und es stellt sich die Frage, welche sich am Besten für eine numerische Lösung eignen. Dies versuchen wir an dem folgenden einfachen Beispiel zu verstehen.

BEISPIEL 4.2: Verhalten von Fixpunktiterationsverfahren.

Betrachten wir zunächst das Nullstellenproblem

$$F(x) \;:=\; x - e^{x-1} \;=\; 0,$$

mit der eindeutigen Nullstelle $\overline{x} = 1$. Versuchen wir zunächst die kanonische Wahl einer Fixpunktfunktion

$$G_1(x) \;:=\; e^{x-1},$$

so kann man zeigen, dass für einen beliebigen Startwert $x_0 > 1$ der Fixpunktiteration $x_1 := e^{x_0-1} > x_0$ gilt auf Grund des Wachstums der Exponentialfunktion. Analog folgt $x_{k+1} > x_k, k \in \mathbb{N}$ für jede Iteration. Die Fixpunktiteration führt also von der Lösung $\overline{x} = 1$ weg und **divergiert**, egal wie nahe der Startwert $x_0 > 1$ bei der Nullstelle $\overline{x}$ von F liegt.

Als Alternative betrachten wir eine Fixpunktiteration der Form

$$G_2(x) \;:=\; 1 + \log x.$$

Diese kann ebenfalls zur Lösung des nichtlinearen Nullstellenproblems $F(x) = 0$ genutzt werden, da im Falle von $G_2(\overline{x}) = \overline{x}$ für die Nullstellengleichung gilt:

$$F(\overline{x}) \;=\; \overline{x} - e^{\overline{x}-1} \;=\; \overline{x} - e^{G_2(\overline{x})-1} \;=\; \overline{x} - \underbrace{e^{1+\log\overline{x}-1}}_{=\overline{x}} \;=\; 0.$$

In diesem Fall gilt nun für einen Startwert $x_0 > 1$, dass

$$1 \;\leq\; x_1 = 1 + \log x_0 \;<\; x_0$$

ist und analog erhalten wir, dass die Iterationen $x_{k+1} < x_k, k \in \mathbb{N}$ eine fallende Folge bilden, die nur durch 1 nach unten beschränkt ist. Man kann nun zeigen, dass die Fixpunktiteration tatsächlich gegen den Fixpunkt $\overline{x} = 1$ **konvergiert**, was gleichzeitig die Nullstelle der nichtlinearen Funktion F ist.

Der Grund für das unterschiedliche Verhalten der beiden diskutierten Fixpunktiterationen ist vor allem die lokale Steigung im Punkt $\overline{x} = 1$. Während $G_1'(1) > 1$ gilt, so ist $0 < G_2'(1) < 1$.

Wir sehen also, dass wir für Konvergenz einer Fixpunktiteration eine Schranke an die Variation der Funktion G brauchen. Das passende mathematische Konzept ist die Kontraktivität, die in folgender Definition eingeführt wird.

DEFINITION 4.3: Kontraktivität.

Eine Funktion $G : X \to X$ auf einem normierten Raum X heißt **kontraktiv** auf einer Teilmenge $D \subset X$, wenn sie Lipschitz-stetig mit Konstante kleiner eins ist, d.h. es existiert ein $q < 1$, so dass

$$\|G(x) - G(y)\| \ \leq \ q\|x - y\|$$

für alle $x, y \in D$ gilt. Die Konstante $q < 1$ wird manchmal auch als **Kontraktionskonstante** bezeichnet.

Unter einer Kontraktivitätsannahme zeigt der Banachsche Fixpunktsatz die Konvergenz der Fixpunktiteration.

THEOREM 4.4: Banachscher Fixpunktsatz.

Sei X ein Banachraum (ein vollständiger normierter Raum) und $D \subset X$ eine nichtleere, abgeschlossene Teilmenge. Sei außerdem $G : X \to X$ eine kontraktive Funktion auf D. Dann existiert ein eindeutiger Fixpunkt $\overline{x} \in D$ mit $\overline{x} = G(\overline{x})$ und die Fixpunktiteration $x_{k+1} = G(x_k)$ konvergiert gegen diesen Fixpunkt.

Beweis. Wir zeigen zunächst, dass die Folge der Iterationen $(x_k)_{k\in\mathbb{N}} \subset D$ eine Cauchy-Folge ist. Dazu wenden wir die Dreiecksungleichung der Norm auf eine konstruierte Teleskopsumme an und erhalten somit

$$\|x_{k+m} - x_k\| \ = \ \|\sum_{j=0}^{m-1} x_{k+j+1} - x_{k+j}\| \ \leq \ \sum_{j=0}^{m-1} \|x_{k+j+1} - x_{k+j}\| \ = \ \sum_{j=k}^{k+m-1} \|x_{j+1} - x_j\|.$$

Wegen der Definition der Fixpunktiteration und der angenommenen Kontraktivität von G folgt induktiv

$$\|x_{j+1} - x_j\| \ = \ \|G(x_j) - G(x_{j-1})\| \ \leq \ q\|x_j - x_{j-1}\| \ \leq \ \ldots \ \leq \ q^j\|x_1 - x_0\|.$$

Also erhalten wir insgesamt für den Abstand zweier Folgenglieder x_{k+m} und x_k:

$$\begin{aligned}\|x_{k+m} - x_k\| \ &\leq \ \sum_{j=k}^{k+m-1} q^j\|x_1 - x_0\| \ = \ q^k\|x_1 - x_0\| \sum_{j=0}^{m-1} q^j \\ &\leq \ q^k\|x_1 - x_0\| \sum_{j=0}^{\infty} q^j \ = \ \frac{q^k}{1-q}\|x_1 - x_0\|.\end{aligned}$$

Wegen $q < 1$ handelt es sich um eine geometrische Reihe und für $k \to \infty$ konvergiert außerdem die rechte Seite unabhängig von m gegen Null. Damit ist $(x_k)_{k\in\mathbb{N}} \subset D$ also eine Cauchy-Folge und wegen der Vollständigkeit des Banachraums X existiert ein Grenzwert $\overline{x} \in D$.

Der Grenzwert $\overline{x} \in D$ ist ein Fixpunkt von G, da wir wegen der Lipschitz-Stetigkeit von G folgern können:

$$\overline{x} \;=\; \lim_{k\to\infty} x_{k+1} \;=\; \lim_{k\to\infty} G(x_k) \;=\; G(\lim_{k\to\infty} x_k) \;=\; G(\overline{x}).$$

Die Eindeutigkeit des Fixpunktes $\overline{x} \in D$ können wir abschließend aus der Kontraktivität folgern. Seien $\overline{x}_1, \overline{x}_2 \in D$ zwei Fixpunkte, dann gilt

$$\|\overline{x}_1 - \overline{x}_2\| \;=\; \|G(\overline{x}_1) - G(\overline{x}_2)\| \;\leq\; q\|\overline{x}_1 - \overline{x}_2\|.$$

Dies ist wegen $q < 1$ nur möglich, falls $\overline{x}_1 = \overline{x}_2$ gilt. □

Neben der Konvergenz von Fixpunktverfahren können wir auch äußerst nützliche Fehlerabschätzungen für den Abstand der Iterierten vom Fixpunkt herleiten, wie das folgende Korollar zeigt.

KOROLLAR 4.5: A-priori und a-posteriori Abschätzungen.

Unter den Voraussetzungen von Theorem 4.4 gelten die **a-priori Fehlerabschätzung**

$$\|x_k - \overline{x}\| \;\leq\; q^k\|x_0 - \overline{x}\|$$

und die **a-posteriori** Fehlerabschätzung

$$\|x_k - \overline{x}\| \;\leq\; \frac{q^k}{1-q}\|x_0 - x_1\|.$$

Beweis. Die erste Abschätzung folgt induktiv aus

$$\|x_k - \overline{x}\| \;=\; \|G(x_{k-1}) - G(\overline{x})\| \;\leq\; q\|x_{k-1} - \overline{x}\| \;\leq\; \ldots \;\leq\; q^k\|x_0 - \overline{x}\|.$$

Die zweite Abschätzung folgt im Grenzwert $m \to \infty$ aus der Abschätzung

$$\|x_{k+m} - x_k\| \;\leq\; \frac{q^k}{1-q}\|x_0 - x_1\|,$$

die wir im Beweis des Banachschen Fixpunktsatzes hergeleitet haben. □

Die Aussage des Banachschen Fixpunktsatzes in Theorem 4.4 ist nur lokal für eine Umgebung $D \subset X$ des Fixpunktes $\overline{x} \in D$ gegeben, in der die Funktion G der Fixpunktiteration kontrahierend ist. Dies ist für viele mathematische Anwendungen in der Praxis sehr relevant, denn für die meisten nichtlinearen Gleichungen erhalten wir Konvergenz nur in einer lokalen Umgebung einer Lösung, d.h. wenn der Startwert $x_0 \in D$ nahe genug beim Fixpunkt $\overline{x} \in D$ liegt. Für andere Startwerte außerhalb der Teilmenge D kann die Iteration divergieren oder sogar gegen eine andere Lösung konvergieren.

Wir sehen dieses Verhalten zum Teil auch schon beim Heron-Verfahren in Beispiel 2.38. Für positive Startwerte konvergiert das Verfahren gegen die positive Wurzel, für negative Startwerte gegen die negative Wurzel. Dazwischen hat man für den speziellen Startwert $x_0 = 0$ Divergenz, die in diesem Fall eine extreme Form hat, da das Verfahren für diesen Punkt gar nicht definiert ist.

4.2 Das Newton-Verfahren

Bis jetzt haben wir allgemein diskutiert, wie sich Fixpunktiterationen für die Lösung von nichtlinearen Gleichungen nutzen lassen und wie man die Konvergenz dieser sicherstellt. Bisher haben wir jedoch die konkrete Konstruktion einer Fixpunktiteration nur für bestimmte Beispiele angegeben. Interessanter ist es natürlich allgemeine Verfahren kennen zu lernen, die sich für eine breite Klasse von mathematischen Problemen anwenden lassen. Als ein sehr prominentes Beispiel für ein solch allgemeines Verfahren betrachten wir das populäre Newton-Verfahren.

Die grundlegende Idee des Newton-Verfahrens ist es die nichtlineare Gleichung geeignet zu approximieren, so dass sich deren Lösung auf den bekannten Fall von linearen Gleichungssystemen zurückführen lässt. Nehmen wir an, dass die Funktion $F\colon \mathbb{R}^n \to \mathbb{R}^n$ differenzierbar sei im Folgenden. Konkret linearisiert man dann die Gleichung $F(x) = 0$ um die aktuelle Iterierte mittels einer **Taylor-Approximation 1. Ordnung**, d.h.

$$F(x) \;\approx\; F(x_k) + F'(x_k)(x - x_k),$$

wobei wir mit $F'(x_k)$ die Jacobi-Matrix der Funktion F bezeichnen.

Nun ist man wieder im Fall von linearen Gleichungssystemen und berechnet anstatt der gesuchten Nullstelle von F eine Lösung des linearen Gleichungssystems

$$Ax \;:=\; F'(x_k)x \;=\; F'(x_k)x_k - F(x_k) \;=:\; b.$$

Wir setzen nun die nächste Iterierte x_{k+1} als Lösung des linearen Gleichungssystems, so dass für die Approximation gilt

$$F(x_k) + F'(x_k)(x_{k+1} - x_k) \approx 0.$$

Nehmen wir nun an, dass die Jacobi-Matrix $F'(x_k)$ regulär ist, so lässt sich das **Newton-Verfahren** als Iterationsverfahren angeben:

$$x_{k+1} \;=\; x_k - (F'(x_k))^{-1} F(x_k). \tag{4.37}$$

Die Fixpunktfunktion $G\colon \mathbb{R}^n \to \mathbb{R}^n$ des Newton-Verfahrens lässt sich also angeben als

$$G(x) \;:=\; x - (F'(x))^{-1} F(x).$$

Das folgende Beispiel illustriert das Newton-Verfahren im Eindimensionalen.

BEISPIEL 4.6: Newton-Verfahren zur Wurzelberechnung.

Wir möchten analog zum Heron-Verfahren in Beispiel 2.38 die positive Quadratwurzel einer positiven reellen Zahl $a \in \mathbb{R}^+$ berechnen, d.h. wir suchen die Zahl $x \in \mathbb{R}^+$, so dass $x^2 = a$ gilt. Zunächst stellen wir dieses mathematische Problem zu einem Nullstellenproblem für eine nichtlineare Funktion $F\colon \mathbb{R} \to \mathbb{R}$ um:

$$F(x) \;:=\; 1 - \frac{a}{x^2} \;=\; 0.$$

Die Ableitung von F ist $F'(x) = \frac{2a}{x^3}$ und existiert offensichtlich falls $x \neq 0$.

Wir formulieren nun das Newton-Verfahren aus (4.37) in diesem eindimensionalen Beispiel als

$$x_{k+1} \;=\; x_k - \frac{F(x_k)}{F'(x_k)} \;=\; x_k - \frac{1 - a/x_k^2}{2a/x_k^3} \;=\; \frac{3x_k}{2} - \frac{x_k^3}{2a}$$

Wählen wir nun beispielsweise die Zahl $a = 2$ und den Startwert $x_0 = 1.5$ als Näherung der Quadratwurzel, so ergeben sich die folgenden Iterationen für das Newtonverfahren:

$$\begin{aligned} x_0 &= 1.5 \\ x_1 &= 1.40 \\ x_2 &= 1.414 \\ x_3 &= 1.414213514 \\ x_4 &= 1.41421356237309502 \\ x_5 &= 1.41421356237309504880168872420969 7 \end{aligned}$$

Das Verfahren konvergiert offensichtlich sehr schnell gegen $\sqrt{2}$. Wir werden später näher diskutieren warum dies so ist.

Das Newton-Verfahren lässt sich auch geometrisch anschaulich illustrieren, wie die folgende Bemerkung festhält.

BEMERKUNG 4.7. In einer Dimension lässt sich das Newton-Verfahren wie folgt veranschaulichen. Zunächst wird die nichtlineare Funktion F für die aktuelle Iteration x_k durch die Tangente an den Graphen $(x_k, F(x_k))$ approximiert. Anschließend wird die nächste Iterierte x_{k+1} aus dem Schnittpunkt dieser Tangente mit der x-Achse bestimmt. Abbildung **??** illustriert die geometrische Interpretation des Newton-Verfahren für eine eindimensionale Funktion. △

4.2.1 Lokale Regularität der Jacobi-Matrix

Bisher haben wir angenommen, dass die Funktion F differenzierbar ist und die Jacobi-Matrix F' stets regulär ist. Tatsächlich ist das Newton-Verfahren nur sinnvoll definiert in einer Umgebung des Fixpunkts $\overline{x}$ wenn $F'(x)$ für alle Punkte x in dieser Umgebung tatsächlich regulär ist. Um dies zu garantieren hilft uns das folgende nützliche Resultat.

LEMMA 4.8: Lokale Regularität.

Sei $F : \mathbb{R}^n \to \mathbb{R}^n$ eine Funktion, die in einer Umgebung des Fixpunkts $\overline{x}$ stetig differenzierbar ist und sei außerdem $F'(\overline{x})$ regulär. Dann existiert eine Umgebung $B_\delta(\overline{x})$ des Fixpunkts $\overline{x}$, so dass die Jacobi-Matrix $F'(x)$ regulär ist für alle Punkte $x \in B_\delta(\overline{x})$ in der Umgebung.

Beweis. Wegen der Stetigkeit der Ableitung gibt es zu jedem $\epsilon > 0$ ein $\delta > 0$, so dass

$$\|F'(x) - F'(\overline{x})\| \;<\; \epsilon, \qquad \forall\, x \in B_\delta(\overline{x}).$$

Wir betrachten zunächst die folgende nützliche Identität:

$$F'(x) \;=\; F'(\overline{x}) - (F'(\overline{x}) - F'(x))) \;=\; F'(\overline{x})(I_n - \underbrace{F'(\overline{x})^{-1}(F'(\overline{x}) - F'(x))}_{=:\,B}). \tag{4.38}$$

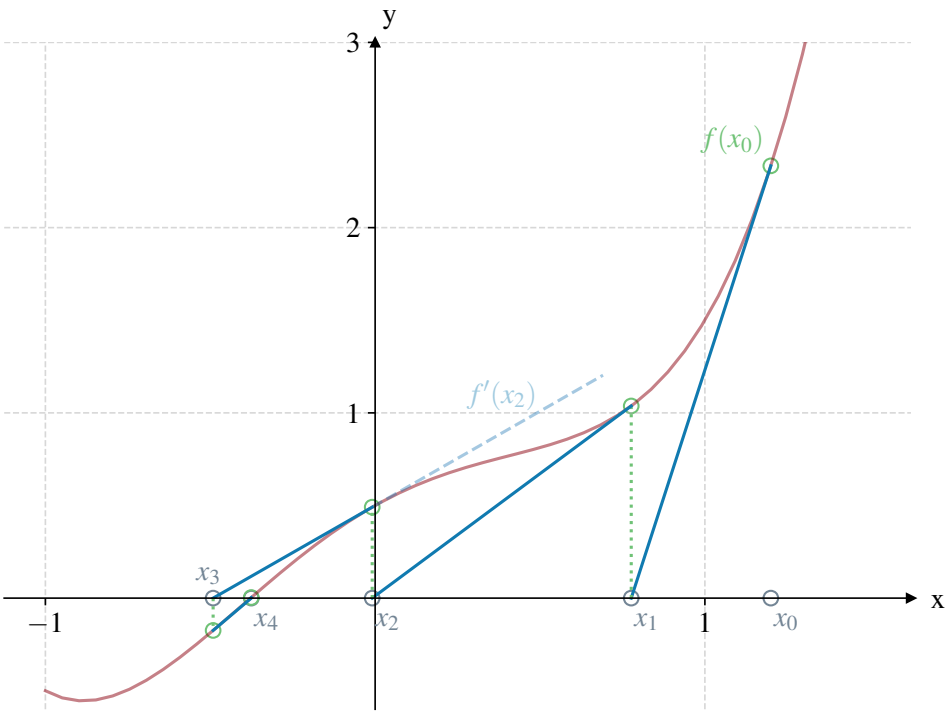


Abbildung 4.1: Geometrische Illustration des Newton-Verfahrens im Eindimensionalen durch Annäherung mittels Tangenten.

Sei nun konkret $\epsilon < \frac{1}{\|F'(\overline{x})^{-1}\|}$ gewählt, dann existiert ein $\delta > 0$, so dass für jedes $x \in B_\delta(\overline{x})$ gilt:

$$\|B\| \;=\; \|F'(\overline{x})^{-1}(F'(\overline{x}) - F'(x))\| \;\leq\; \underbrace{\|F'(\overline{x})^{-1}\|}_{<\epsilon^{-1}} \cdot \underbrace{\|F'(\overline{x}) - F'(x)\|}_{<\epsilon} \;<\; 1.$$

Nach Theorem 2.25 über die Neumannsche Reihe ist also $I_n - B$ regulär und da $F'(\overline{x})$ regulär nach Voraussetzung war ist das Produkt der beiden Matrizen auch wieder regulär. Somit folgt wegen der Identität (4.38), dass die Jacobimatrix $F'(x)$ für jedes $x \in B_\delta(\overline{x})$ in der lokalen Umgebung um den Fixpunkt $\overline{x}$ regulär sein muss. □

4.2.2 Konvergenz

Für den Fall, dass die Jacobi-Matrix in der Nullstelle der nichtlinearen Funktion $F\colon \mathbb{R}^n \to \mathbb{R}^n$ invertierbar ist, können wir nun die lokale Konvergenz des Newton-Verfahrens nachweisen.

THEOREM 4.9: Lokale Konvergenz des Newton-Verfahrens.

Sei $F : \mathbb{R}^n \to \mathbb{R}^n$ in einer Umgebung von $\overline{x}$ stetig differenzierbar und $\overline{x} \in \mathbb{R}^n$ eine Nullstelle von F mit $F(\overline{x}) = 0$. Sei außerdem die Jacobi-Matrix F' lokal Lipschitz-stetig und $F'(\overline{x})$ regulär in der Nullstelle.
Dann existiert eine lokale Umgebung $B_R(\overline{x})$, so dass das Newton-Verfahren für jeden Startwert $x_0 \in B_R(\overline{x})$ gegen die Nullstelle $\overline{x}$ konvergiert, d.h. $\lim_{k\to\infty} x_k = \overline{x}$.

Beweis. Wir betrachten zunächst nur Umgebungen mit $0 < R < \delta$, wobei δ der Radius aus Lemma 4.8 ist. Mit dessen Aussage wissen wir, dass $F'(x)$ in einer solchen Umgebung immer regulär für alle $x \in B_R(\overline{x})$ ist.

Es reicht nun die lokale Kontraktivität der Fixpunktiteration des Newton-Verfahrens mit $G(x) = x - F'(x)^{-1}F(x)$ nachzuweisen, damit wir den Banachschen Fixpunktsatz aus Theorem 4.4 anwenden können. Es gilt nun für alle Punkte $x, y \in B_R(\overline{x})$ die folgende Abschätzung

$$\begin{aligned}
\|G(x) - G(y)\| &= \|x - y - F'(x)^{-1}F(x) + F'(y)^{-1}F(y)\| \\
&= \|F'(x)^{-1}(F(y) - F(x) - F'(x)(y-x)) + (F'(y)^{-1} - F'(x)^{-1})F(y)\| \\
&\leq \|F'(x)^{-1}(F(y) - F(x) - F'(x)(y-x))\| + \|(F'(y)^{-1} - F'(x)^{-1})F(y)\| \\
&= \|F'(x)^{-1}(F(y) - F(x) - F'(x)(y-x))\| \\
&\qquad + \|(F'(y)^{-1} - F'(x)^{-1})(F(y) - F(\overline{x}))\|.
\end{aligned}$$

Wegen der Stetigkeit von F' können wir zunächst wegen Lemma 4.8 den Radius R so klein wählen, dass $\|F'(x)^{-1}\|$ beschränkt ist, z.B. durch $\|2F'(\overline{x})^{-1}\|$. Da F stetig differenzierbar ist, gilt insbesondere

$$\frac{\|F'(x)^{-1}(F(y) - F(x) - F'(x)(y-x))\|}{\|x-y\|} \;\to\; 0$$

für $\|x - y\| \to 0$. Da $\|x - y\| < 2R$ gilt können wir also für jedes q einen hinreichend kleinen Radius Rfinden, so dass gilt

$$\|F'(x)^{-1}(F(y) - F(x) - F'(x)(y-x))\| \;<\; \frac{q}{2}\|x-y\|.$$

Weiter können wir folgende Abschätzung treffen:

$$\begin{aligned}
\|(F'(y)^{-1} - F'(x)^{-1})(F(y) - F(\overline{x}))\| &= \|F'(y)^{-1}(F'(x) - F'(y))F'(x)^{-1}(F(y) - F(\overline{x}))\| \\
&\leq \|F'(y)^{-1}\|\,\|F'(x)^{-1}\|\,\|F'(x) - F'(y)\|\,\|F(y) - F(\overline{x})\|.
\end{aligned}$$

Da eine stetig differenzierbare Funktion insbesondere auch Lipschitz-stetig ist und wir die lokale Lipschitz-Stetigkeit von F' gefordert haben, gilt in einer Umgebung $B_R(\overline{x})$ des Fixpunktes $\overline{x}$:

$$\|F'(x) - F'(y)\| \;\leq\; C\|x-y\|, \qquad \|F(y) - F(\overline{x})\| \;\leq\; \|y - \overline{x}\| \;\leq\; C \cdot R.$$

Außerdem sind nach Theorem 2.25 über die Neumannsche Reihe $\|F'(x)^{-1}\|$ und $\|F'(y)^{-1}\|$ lokal beschränkt. Ist R hinreichend klein, so erhalten wir insgesamt also

$$\|(F'(y)^{-1} - F'(x)^{-1})(F(y) - F(\overline{x}))\| \;\leq\; \frac{q}{2}\|x-y\|.$$

Damit haben wir insgesamt die Kontraktivität der Fixpunktfunktion G nachgewiesen und somit erhalten wir nach dem Banachschen Fixpunktsatz in Theorem 4.4 die Konvergenz des Newton-Verfahrens. □

Wir sehen im Beweis, dass mit kleiner werdendem R ein beliebig kleines $q > 0$ in der Kontraktivität erreicht werden kann. Je näher wir also an die Lösung herankommen, desto schneller wird das Verfahren konvergieren. Dies motiviert die Frage des nächsten Abschnitts, nämlich wie wir unterschiedliche Konvergenzgeschwindigkeiten definieren und analysieren können.

4.3 Konvergenzgeschwindigkeit

Um zu beschreiben wie schnell die Folge von Punkten $(x_k)_{k\in\mathbb{N}}$ eines Iterationsverfahrens $x_{k+1} = G(x_k)$ gegen einen Fixpunkt $\overline{x}$ konvergiert verwenden wir den Begriff der Konvergenzgeschwindigkeit, den wir im Folgenden definieren wollen.

DEFINITION 4.10: Konvergenzgeschwindigkeit.

Sei X ein Banachraum, d.h. ein vollständiger normierter Vektorraum. Wir sagen, dass eine Folge von Punkten $(x_k)_{k\in\mathbb{N}} \subset X$ mit $\lim_{k\to\infty} x_k = \overline{x}$ gegen einen Punkt $\overline{x} \in X$ mindestens mit Ordnung $p \geq 1, p \in \mathbb{N}$ konvergiert, falls es ein $C > 0$ und ein $N \in \mathbb{N}$ gibt, so dass für alle $k \geq N$ gilt

$$||x_{k+1} - \overline{x}|| \;\leq\; C||x_k - \overline{x}||^p.$$

Im Fall $p = 1$ spricht man von **linearer Konvergenz** und für $p = 2$ von **quadratischer Konvergenz**.
Ein weiterer interessanter Spezialfall liegt vor, falls es eine gegen Null konvergente Zahlenfolge $(c_k)_{k\in\mathbb{N}}$ gibt, so dass man zeigen kann, dass gilt:

$$||x_{k+1} - \overline{x}|| \;\leq\; c_k||x_k - \overline{x}||.$$

In diesem Fall sprechen wir von **superlinearer Konvergenz**.

BEMERKUNG 4.11 (Konvergenzfaktor). Im Fall von linearer Konvergenzgeschwindigkeit, d.h. für $p = 1$ muss man in Definition 4.10 zusätzlich fordern, dass für die Konstante $C < 1$ gilt. Diese Konstante wird auch **Konvergenzfaktor** genannt und es ist klar, dass eine Folge umso schneller konvergiert, je kleiner der Konvergenzfaktor C ist. In diesem Fall entspricht der Konvergenzfaktor C der *Kontraktionskonstanten* $q < 1$ aus Definition 4.3, da

$$\|G(x_k) - G(\overline{x})\| \;=\; \|x_{k+1} - \overline{x}\| \;\leq\; C\|x_k - \overline{x}\| \;=\; q\|x_k - \overline{x}\|.$$

△

Um die Konvergenzgeschwindigkeit eines Iterationsverfahrens $x_{k+1} = G(x_k)$ zu analysieren können wir die Taylorentwicklung von G um den Fixpunkt $\overline{x}$ betrachten, falls G genügend oft in $\overline{x}$ differenzierbar ist. Hierbei spielen die Ableitungen des Iterationsverfahrens eine wichtige Rolle für die Konvergenzgeschwindigkeit, wie das folgende Theorem aussagt.

THEOREM 4.12: Konvergenzgeschwindigkeit.

Sei $U(\overline{x}) \subset \mathbb{R}$ eine lokale Umgebung von $\overline{x}$ und sei $G\colon \mathbb{R} \to \mathbb{R}$ ein in $U(\overline{x})$ konvergentes Iterationsverfahren mit $\lim_{k\to\infty} G(x_k) = \overline{x}$ für eine Folge $(x_k)_{k\in\mathbb{N}} \subset U(\overline{x})$.
Falls G in der lokalen Umgebung $U(\overline{x})$ mindestens p-Mal stetig differenzierbar ist für $p \geq 1, p \in \mathbb{N}$ und außerdem für die Ableitungen von G in $\overline{x}$ gilt:

$$G^{(j)}(\overline{x}) \;=\; 0, \quad \forall\, j = 1, \ldots, p-1,$$

so konvergiert das Iterationsverfahren mindestens mit Ordnung p.

Beweis. Sei $x_k \in U(\overline{x})$ ein Punkt der konvergenten Folge $(u_k)_{k\in\mathbb{N}}$. Dann betrachten wir die Taylorentwicklung von $G(x_k)$ im Fixpunkt $\overline{x}$ und erhalten

$$\begin{aligned}
|x_{k+1} - \overline{x}| &= |G(x_k) - \overline{x}| = \left| \sum_{j=0}^{\infty} G^{(j)}(\overline{x}) \frac{(x_k - \overline{x})^j}{j!} - G(\overline{x}) \right| \\
&= \Big| \underbrace{G(\overline{x}) \cdot 1 - G(\overline{x})}_{=0} + \underbrace{G'(\overline{x})(x_k - \overline{x})}_{=0} + \ldots + \underbrace{G^{(p-1)}(\overline{x}) \frac{(x_k - \overline{x})^{p-1}}{(p-1)!}}_{=0} \\
&\quad + G^{(p)}(\overline{x}) \frac{(x_k - \overline{x})^p}{p!} + \mathcal{O}((x_k - \overline{x})^{p+1}) \Big| \\
&= \left| \frac{G^{(p)}(\overline{x})}{p!} (x_k - \overline{x})^p + \mathcal{O}((x_k - \overline{x})^{p+1}) \right| \leq \underbrace{\left| \frac{G^{(p)}(\overline{x})}{p!} \right|}_{=:C} \cdot |x_k - \overline{x}|^p + \mathcal{O}(|x_k - \overline{x}|^{p+1})
\end{aligned}$$

Wenn wir die Terme höherer Ordnung vernachlässigen haben wir gezeigt, dass ein $C > 0$ existiert, so dass gilt

$$|x_{k+1} - \overline{x}| \leq C|x_k - \overline{x}|^p.$$

Wir sehen also, dass das Iterationsverfahren $x_{k+1} = G(x_k)$ in einer lokalen Umgebung $U(\overline{x})$ mit Ordnung p gegen den Fixpunkt $\overline{x}$ konvergiert. □

BEMERKUNG 4.13. Sei $x_{k+1} = G(x_k)$ ein *allgemeines Iterationsverfahren*, das in einer Umgebung $U(\overline{x})$ eines Fixpunktes $\overline{x}$ differenzierbar ist. Falls $0 < G'(\overline{x}) < 1$ gilt, so konvergiert das Iterationsverfahren linear und **monoton** gegen den Fixpunkt $\overline{x}$.
Falls jedoch $-1 < G'(\overline{x}) < 0$ gilt, so konvergiert das Iterationsverfahren linear und **alternierend** gegen den Fixpunkt $\overline{x}$. △

Wir können nun das hinreichende Kriterium aus Theorem 4.12 anwenden, um die Konvergenzgeschwindigkeit des Newton-Verfahrens genauer zu untersuchen.

KOROLLAR 4.14: Konvergenzgeschwindigkeit des Newton-Verfahrens.

Betrachten wir im Folgenden das *Newton-Verfahren* zur Approximation einer Nullstelle $F(\overline{x}) = 0$ der Funktion $F\colon \mathbb{R} \to \mathbb{R}$ mit der Iterationsvorschrift

$$x_{k+1} = G(x_k) = x_k - \frac{F(x_k)}{F'(x_k)}.$$

Hierbei nehmen wir an, dass die Funktion F in einer lokalen Umgebung $U(\overline{x})$ der Nullstelle genügend häufig stetig differenzierbar ist.
Zur Analyse der Konvergenzgeschwindigkeit des Newton-Verfahrens in der lokalen Umgebung $U(\overline{x})$ verwenden wir das Kriterium aus Theorem 4.12. Hierfür berechnen wir die ersten Ableitungen von G im Fixpunkt $\overline{x}$:

$$\begin{aligned}
G'(x) &= 1 - \frac{(F'(x))^2 - F(x)F''(x)}{(F'(x))^2} = \frac{F(x)F''(x)}{(F'(x))^2} \\
G''(x) &= \frac{(F'(x)F''(x) + F(x)F'''(x))(F'(x))^2 - 2F(x)F''(x)F'(x)F''(x)}{(F'(x))^4}
\end{aligned}$$

Betrachten wir nun die Ableitungen $G'(x)$ und $G''(x)$ des Iterationsverfahrens in der Nullstelle $F(\overline{x}) = 0$, so sehen wir, dass gilt:

$$\begin{aligned} G'(\overline{x}) &= \frac{F(\overline{x})F''(\overline{x})}{(F'(\overline{x}))^2} = 0, \\ G''(\overline{x}) &= \frac{F'(\overline{x})F''(\overline{x})(F'(\overline{x}))^2}{(F'(\overline{x}))^4} = \frac{F''(\overline{x})}{F'(\overline{x})}. \end{aligned}$$

Es gilt also, dass $G'(\overline{x}) = 0$ jedoch $G''(\overline{x}) \neq 0$ im Allgemeinen. Somit können wir mit Theorem 4.12 folgern, dass es eine lokale Umgebung $U(\overline{x})$ gibt in der das Newton-Verfahren eine **quadratische Konvergenzgeschwindigkeit** aufweist.

Das Newton-Verfahren konvergiert lokal quadratisch gegen die Nullstelle $F(\overline{x}) = 0$, wie wir im obigen Korollar festgestellt haben. Es gibt jedoch besondere Nullstellen, die wir im Folgenden definieren wollen.

DEFINITION 4.15: p-fache Nullstelle.

Sei $p \geq 0, p \in \mathbb{N}$. Wir nennen $\overline{x} \in \mathbb{R}$ eine *p-fache Nullstelle* einer Funktion $F\colon \mathbb{R}^n \to \mathbb{R}^m$, falls gilt

$$F^{(j)}(\overline{x}) = 0, \quad \forall j = 0, \ldots, p-1.$$

Im Fall einer p-fachen Nullstelle von F ändert sich das Konvergenzverhalten des Newton-Verfahrens überraschenderweise, wie das folgende Theorem besagt.

THEOREM 4.16: Lineare Konvergenz bei p-facher Nullstelle.

Sei $F\colon \mathbb{R} \to \mathbb{R}$ eine Funktion und $\overline{x} \in \mathbb{R}$ eine p-fache Nullstelle von F. Sei außerdem F in einer lokalen Umgebung $U(\overline{x})$ von $\overline{x}$ p-fach stetig differenzierbar.
Dann konvergiert das Newton-Verfahren für jeden Startwert $x_0 \in U(\overline{x})$ nur linear gegen den Fixpunkt $\overline{x}$.

Beweis. Sei $\overline{x}$ eine p-fache Nullstelle der Funktion F mit $p > 1$ und es gelte $F^{(p)}(\overline{x}) \neq 0$. Sei weiterhin $x \in U(\overline{x})$ ein Punkt in der lokalen Umgebung um die Nullstelle $\overline{x}$. Aus der Taylorentwicklung in Theorem 4.12 sehen wir, dass es eine differenzierbare Funktion g geben muss mit $g(\overline{x}) \neq 0$, so dass lokal gilt:

$$\begin{aligned} F(x) &= (x-\overline{x})^p g(x), \\ F'(x) &= p(x-\overline{x})^{p-1} g(x) + (x-\overline{x})^p g'(x). \end{aligned}$$

Wir können somit das Newton-Verfahren umschreiben zu

$$\begin{aligned} G(x) = x - \frac{F(x)}{F'(x)} &= x - \frac{(x-\overline{x})^p g(x)}{p(x-\overline{x})^{p-1} g(x) + (x-\overline{x})^p g'(x)} \\ &= x - \frac{(x-\overline{x}) g(x)}{p g(x) + (x-\overline{x}) g'(x)}. \end{aligned}$$

Zur Bestimmung der Konvergenzgeschwindigkeit betrachten wir wieder die Ableitung $G'(x)$ und erhalten:

$$G'(x) = 1 - \frac{(g(x)+(x-\overline{x})g'(x))(pg(x)+(x-\overline{x})g'(x)) - (x-\overline{x})g(x)(pg'(x)+g'(x)+(x-\overline{x})g''(x))}{(pg(x)+(x-\overline{x})g'(x))^2}$$

Glücklicherweise fallen viele Terme der Ableitung $G'(x)$ weg im Fixpunkt $\overline{x}$, so dass wir erhalten:

$$G'(\overline{x}) = 1 - \frac{p(g(\overline{x}))^2}{(pg(\overline{x}))^2} = 1 - \frac{1}{p} \neq 0.$$

Wir sehen also, dass im Fall einer p-fachen Nullstelle von F mit $p > 1$ die erste Ableitung des Newton-Verfahrens nicht verschwindet. Daher erhalten wir nur lineare Konvergenz. □

BEMERKUNG 4.17. Die quadratische Konvergenz des Newton-Verfahrens lässt sich im Fall einer p-fachen Nullstelle von F in $\overline{x}$ wiederherstellen indem man folgende Variante implementiert:

$$x_{k+1} = G(x_k) = x_k - p \cdot \frac{F(x_k)}{F'(x_k)}. \tag{4.39}$$

Leider ist dieses Verfahren *numerisch instabil*, da es bei der Division von F und F' in (4.39) nahe der p-fachen Nullstelle zu Auslöschung kommt (siehe Abschnitt 2.2). △

4.4 Fixpunktiterationen ohne Ableitungen

Im Folgenden diskutieren wir alternative Fixpunktiterationsverfahren, die keine Ableitungen verwenden. Gründe für den Einsatz solcher Verfahren können sein, dass entweder F nicht differenzierbar ist, oder häufiger noch, weil die Ableitung von F nicht einfach berechnet werden kann, z.B. weil die Funktion F selbst nur durch ein numerisches Verfahren ausgewertet werden kann. Das klassische Beispiel von Fixpunktiterationen ohne Ableitungen sind **Intervallschachtelungsverfahren** im Eindimensionalen, im einfachsten Fall mittels Bisektion.

BEISPIEL 4.18: Bisektionsverfahren.

Sei $f : \mathbb{R} \to \mathbb{R}$ stetig und $a < b$ so, dass $F(a)F(b) < 0$. Dann wissen wir aus dem Zwischenwertsatz, dass eine Nullstelle der Funktion im Intervall (a, b) liegt. Wir berechnen also $F(\frac{a+b}{2})$ und vergleichen das Vorzeichen mit dem von $F(a)$ und $F(b)$. Ist das Vorzeichen gleich dem von $F(a)$, so liegt die Nullstelle im Intervall $(\frac{a+b}{2}, b)$, andernfalls im Intervall $(a, \frac{a+b}{2})$. Nun können wir iterativ die Intervalle weiter verfeinern.
Mathematisch sauberer definieren wir eine Iterierte $x_k \in \mathbb{R}^2$ mit $x_0 = (a, b)$ und

$$x_{k+1} = \begin{cases} \left(x_k^1, \frac{x_k^1+x_k^2}{2}\right) & \text{falls } f(x_k^1) \cdot f(\frac{x_k^1+x_k^2}{2}) < 0, \\ \left(\frac{x_k^1+x_k^2}{2}, x_k^2\right) & \text{falls } f(x_k^2) \cdot f(\frac{x_k^1+x_k^2}{2}) < 0, \\ \left[\frac{x_k^1+x_k^2}{2}, \frac{x_k^1+x_k^2}{2}\right] & \text{falls } f(\frac{x_k^1+x_k^2}{2}) = 0. \end{cases}$$

In diesem Fall haben wir offensichtlich eine Fixpunktiteration und daher können wir das Verfahren bezüglich Kontraktivität untersuchen. Man kann zeigen, dass die Kontrakti-

onskonstante $q = \frac{1}{2}$ ist. Wir können die Konvergenz des Bisektionsverfahrens oben aber auch direkt analysieren. Es gilt offensichtlich:

$$\|x_k - \overline{x}\|_1 \;=\; |x_k^1 - x_k^2| \;\leq\; \frac{1}{2}|x_{k-1}^1 - x_{k-1}^2| \;\leq\; \ldots \;\leq\; \frac{1}{2^k}|x_0^1 - x_0^2| \;=\; \frac{|a-b|}{2^k}.$$

Als zweites Beispiel diskutieren wir ein klassisches Verfahren zur Approximation des Newton-Verfahrens ohne Ableitungen zu verwenden. Die Idee hierbei ist es die Tangente im Newton-Verfahren durch die Sekante zwischen den letzten beiden Iterierten zu ersetzen. Daraus lässt sich also das folgende **Sekantenverfahren** herleiten.

BEISPIEL 4.19: Sekantenverfahren.

Sei $F : \mathbb{R} \to \mathbb{R}$ eine Funktion deren Nullstelle wir bestimmen wollen. Es seien bereits mindestens zwei Iterierte x_k und x_{k-1} gegeben. Falls $F(x_k) \neq F(x_{k-1})$ gilt, berechnen wir

$$x_{k+1} \;=\; x_k - \frac{x_k - x_{k-1}}{F(x_k) - F(x_{k-1})} F(x_k).$$

Durch Taylor-Entwicklung erhalten wir, dass das Sekantenverfahren eine *superlineare Konvergenz* aufweist, wenn F hinreichend glatt ist und $F'(\overline{x}) \neq 0$ gilt (in diesem Fall ist in einer Umgebung auch $F(x) \neq F(y)$ für $x \neq y$).

BEMERKUNG 4.20. Das Sekantenverfahren lässt sich nicht direkt auf den mehrdimensionalen Fall erweitern. Eine einfachere Variante ist eine Differenzenapproximation der Ableitung mit fixer (kleiner) Schrittweite δ, d.h. also wir berechnen statt dessen im Eindimensionalen:

$$x_{k+1} \;=\; x_k - \frac{\delta}{F(x_k + \delta) - F(x_k)} F(x_k).$$

Für eine mehrdimensionale Funktion $F : \mathbb{R}^n \to \mathbb{R}^n$ können wir analog die Jacobi Matrix mittels

$$(F'(x))_{ij} \;\approx\; \frac{F_i(x + \delta e_j) - F_i(x)}{\delta}, \quad \forall\, 1 \leq i, j \leq n$$

approximieren, wobei e_j den j-ten Einheitsvektor im $\mathbb{R}^n$ bezeichnet. △

4.5 Gauss–Newton Verfahren

Im Fall einer Funktion $F : \mathbb{R}^n \to \mathbb{R}^m$ mit $m > n$ lässt sich die Jacobi-Matrix F' von F nicht invertieren, so dass wir das Newton-Verfahren aus Abschnitt 4.2 nicht zur Lösung der Gleichung $F(x) = 0$ verwenden können.

Stattdessen versuchen wir wieder ein Kleinste-Quadrate Problem zu lösen der Form

$$f(x) \;:=\; \frac{1}{2}\|F(x)\|^2 \to \min_{x \in \mathbb{R}^n}. \tag{4.40}$$

Ein Minimierer $\overline{x} \in \mathbb{R}^n$ von (4.40) muss offensichtlich die notwendige Bedingung erster Ordnung erfüllen, d.h.

$$f'(\overline{x}) \;=\; F'(\overline{x})^T F(\overline{x}) \;=\; 0.$$

Da f' eine Funktion ist mit $f'\colon \mathbb{R}^n \to \mathbb{R}^n$ können wir nun das Newton-Verfahren anwenden. Da wir im Allgemeinen nicht von Regularität der Ableitungen von f' ausgehen können, liefert uns das Newton-Verfahren zunächst das folgende lineare Gleichungssystem:

$$f''(x_k)(x_{k+1} - x_k) \;=\; -f'(x_k).$$

Hier bezeichnet f'' die Hesse-Matrix der Funktion f und es gilt

$$f''(x) \;=\; F''(x)^T F(x) + F'(x)^T F'(x). \tag{4.41}$$

Währen der zweite Teil $F'(x)^T F'(x)$ sicher positiv semi-definit ist, können wir wenig über den Term $F''(x)^T F(x)$ aussagen. Deshalb approximiert man häufig die Hesse-Matrix nur durch den zweiten Term und führt somit das sogenannte **Gauss–Newton Verfahren** durch mit der Iterationsvorschrift

$$F'(x_k)^T F'(x_k)(x_{k+1} - x_k) \;=\; -F'(x_k)^T F(x_k).$$

Man beobachtet, dass das Verfahren insbesondere eine gute Approximation ist, falls sich (4.40) exakt zu Null lösen lässt, d.h. im Fall $F(\overline{x}) = 0$. Hier gilt wegen (4.41) offensichtlich

$$f''(\overline{x}) = F'(\overline{x})^T F'(\overline{x}).$$

Mit Hilfe dieser Identität und der üblichen Taylor-Entwicklung können wir in diesem Fall die *superlineare Konvergenz* des Gauss–Newton Verfahrens nachweisen.

BEMERKUNG 4.21 (Varianten des Gauss–Newton Verfahrens). Da das Matrixprodukt $F'(x)^T F'(x)$ potentiell schlecht-konditioniert sein kann, ergibt sich daraus ein numerisches Problem für das Gauss–Newton Verfahren, denn hierdurch kann der Iterationsschritt $x_{k+1} - x_k$ zu groß werden. Als Alternative können wir Verfahren der folgenden Form betrachten

$$B_k(x_{k+1} - x_k) \;=\; -F'(x_k)^T F(x_k),$$

mit geeigneten positiv definiten Matrizen B_k.
Der einfachste Fall einer positiv definiten Matrix ist durch das **Gradientenverfahren** gegeben mit $B_k := \frac{1}{\tau_k} I$, d.h. wir erhalten die einfache Fixpunktiteration

$$x_{k+1} \;=\; x_k - \tau_k F'(x_k)^T F(x_k) \;=\; x_k - \tau_k f'(x_k).$$

Der Schrittweitenparameter $\tau_k > 0$ kann so gewählt werden, dass zu große Schritte vermieden werden. Das Gradientenverfahren ist ein sogenanntes Abstiegsverfahren, d.h. es gilt für hinreichend kleine Schrittweiten τ_k die folgende Abschätzung

$$f(x_{k+1}) \;=\; f(x_k - \tau_k f'(x_k)) \;=\; f(x_k) - \tau_k \|f'(x_k)\|^2 + o(\tau_k) < f(x_k).$$

Damit lässt sich gewährleisten, dass das Gradientenverfahren nicht gegen eine beliebige Lösung von $f'(x) = 0$ konvergiert, sondern zumindest die Funktion f dabei reduziert.
Ein Kompromiss zwischen Gradienten- und Gauss–Newton-Verfahren ist das sogenannte **Levenberg–Marquardt Verfahren**, das gegeben ist durch:

$$(F'(x_k)^T F'(x_k) + \tau_k I)(x_{k+1} - x_k) \;=\; -F'(x_k)^T F(x_k).$$

Um anfänglich zu große Schritte zu vermeiden, wählt man zunächst $\tau_k > 0$ groß. Um asymptotisch die superlineare Konvergenz zu erhalten, sollte τ_k dann im Verlauf der Iterationen gegen Null konvergieren. △

4.6 Gesamt- und Einzelschrittverfahren

Viele mathematische Probleme, die numerisch gelöst werden sollen, führen zu einem großen linearen Gleichungssystem $Au = f$, bei dem die Matrix nur an relativ wenigen Stellen Einträge ungleich Null besitzt. Dies tritt vor allem bei der Diskretisierung von Differentialgleichungen (DGL) auf, wie das folgende motivierende Beispiel demonstriert.

BEISPIEL 4.22: Diskretisierung einer Helmholtz-artigen DGL.

Wir betrachten ein eindimensionales Randwertproblem auf einem Intervall $\Omega = [0, 1]$ für $\sigma > 0$ der Form:

$$\begin{aligned} -u''(x) + \sigma u(x) &= f(x), \qquad x \in (0,1), \\ u(0) = u(1) &= 0. \qquad \text{(Dirichlet-Randbedingungen)} \end{aligned} \tag{4.42}$$

Diese Helmholtz-artige Differentialgleichung wird beispielsweise zur Entrauschung von Daten in der mathematischen Bildverarbeitung genutzt.
Die kontinuierliche Differentialgleichung in (4.42) lässt sich im Eindimensionalen auf einem diskreten Gitter

$$\Omega_h \coloneqq \{x_i \in [0,1] \mid x_i = i \cdot h\}$$

mit Schrittweite $h \coloneqq 1/N$ und $N + 1 \in \mathbb{N}$ Gitterpunkten approximieren (siehe Vorlesung „Numerik 2: Diskretisierung und Optimierung“). Dies führt zu einem linearen Gleichungssystem mit der (bis auf den Rand) unbekannten Lösung $u \in \mathbb{R}^{N+1}$ und der Diskretisierung $f_i \coloneqq f(x_i)$ der Form

$$\begin{aligned} \frac{-u_{i-1} + 2u_i - u_{i+1}}{h^2} + \sigma u_i &= f_i, \qquad i = 1, \dots, N-1, \\ u_0 = u_N &= 0. \end{aligned} \tag{4.43}$$

Dies können wir kompakt als lineares Gleichungssystem der Form $Au = f$, wobei nun $A \in \mathbb{R}^{(N-1)\times(N-1)}$ eine **Tridiagonalmatrix** und $u, f \in \mathbb{R}^{N-1}$ ist mit

$$A \coloneqq \frac{1}{h^2} \begin{pmatrix} 2+\sigma h^2 & -1 & 0 & \cdots & 0 \\ -1 & 2+\sigma h^2 & -1 & \ddots & \vdots \\ 0 & -1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & -1 & 2+\sigma h^2 \end{pmatrix}, \; u \coloneqq \begin{pmatrix} u_1 \\ \vdots \\ u_{N-1} \end{pmatrix}, \; f \coloneqq \begin{pmatrix} f_1 \\ \vdots \\ f_{N-1} \end{pmatrix}.$$

Eine Lösung $u \in \mathbb{R}^{N-1}$ des linearen Gleichungssystems (4.43) stellt dann eine diskrete Approximation einer Lösung der Differentialgleichung (4.42) dar.

Die Matrix A in Beispiel 4.22 ist von einer besonderen Gestalt, da sie symmetrisch ist und zum größten Teil nur Nullen als Einträge hat. Solche Matrizen nennt man auch **dünn-besetzt** (im Englischen: **sparse**). Man kann außerdem zeigen, dass die Matrix positiv-definit ist für $\sigma > 0$. Nun könnte man auf Grund dieser Eigenschaften eine direkte Lösung mittels der Cholesky-Zerlegung aus Abschnitt 1.4 umsetzen. Dies ist in der Regel jedoch weniger effizient

für dünn-besetzte Matrizen, für die sich häufig *iterative Lösungsverfahren* besser eignen. Solche Verfahren wollen wir uns im Folgenden genauer anschauen.

4.6.1 Gesamtschrittverfahren

Das Gesamtschrittverfahren, das oft auch **Jacobi-Verfahren** genannt wird, basiert auf einer relativ einfachen Idee. Möchte man nämlich ein allgemeines lineares Gleichungssystem

$$Au \;=\; f, \qquad A \in \mathbb{R}^{n\times n}, \quad u, f \in \mathbb{R}^n,$$

lösen, wobei A eine dünn-besetzte Matrix darstellt, so zerlegt man die Matrix (additiv) in folgende Matrizen:

$$A \;=\; L + D + R, \qquad L, D, R \in \mathbb{R}^{n\times n}.$$

Hierbei stellt D die Hauptdiagonale der Matrix A dar, und L und R sind jeweils linke untere und rechte obere Dreiecksmatrizen, die sich aus den entsprechenden Einträgen aus A ergeben.

Damit lässt sich das allgemeine lineare Gleichungssystem nun wie folgt umschreiben:

$$Au \;=\; (L + D + R)u \;=\; Du + (L + R)u \;=\; f.$$

Wir approximieren die obige Gleichung nun durch ein **iteratives Verfahren** der Form

$$Du^{k+1} + (L + R)u^k \;\approx\; f,$$

für einen geeigneten Startpunkt $x_0 \in \mathbb{R}^n$. Falls für die Einträge der Diagonalmatrix D gilt $d_{i,i} \neq 0, i = 1, \ldots, n$ ist eine Inversion der Matrix D wohldefiniert und darüber hinaus sehr einfach zu realisieren und wir erhalten

$$u^{k+1} \;=\; D^{-1}(f - (L + R)u^k). \tag{4.44}$$

Wenn man diese Iterationsvorschrift für die einzelnen Einträge des Lösungsvektors $u^{k+1} \in \mathbb{R}^n$ betrachtet ergibt sich folgende kompakte Form:

$$u_i^{k+1} \;=\; \frac{1}{a_{i,i}} \left(f_i - \sum_{j\neq i} a_{i,j} u_j^k \right), \qquad i = 1, \ldots, n. \tag{4.45}$$

Die Berechnung jedes Eintrags ist offensichtlich **unabhängig** von den anderen Einträgen und kann somit effizient parallelisiert werden. Im Gesamtschrittverfahren werden also pro Iteration erst alle Einträge des Lösungsvektors u aktualisiert und anschließend für die nächste Iteration verwendet.

BEMERKUNG 4.23 (Rechenaufwand des Gesamtschrittverfahrens). Betrachtet man die Rechenvorschrift des Gesamtschrittverfahrens in (4.45), so ist klar, dass man im schlechtesten Fall einer voll-besetzten Matrix A für die Berechnung eines Eintrags u_i^{k+1} genau n FLOPS benötigt und somit für die Berechnung der Iterierten $u^{k+1} \in \mathbb{R}^n$ insgesamt n^2 FLOPS benötigt.
Nehmen wir an, dass wir die echte Lösung $u \in \mathbb{R}^n$ des linearen Gleichungssystems in $m \in \mathbb{N}$ Iterationsschritten approximieren wollen, so ist der Rechenaufwand für das Gesamtschrittverfahren $m \cdot n^2$ und liegt somit in $\mathcal{O}(n^2)$. Für wenige Iterationen $m << n$ ist dies schon deutlich effizienter als direkte Lösungsverfahren, die wir in Kapitel 1 diskutiert haben.

Dazu kommt noch der Vorteil, dass für dünn-besetzte Matrizen mit nur $k \in \mathbb{N}$, $k << n$ Einträgen ungleich Null pro Zeile der Rechenaufwand nochmal signifikant sinkt und man insgesamt $m \cdot k \cdot n$ Rechenoperationen zur Approximation einer Lösung $\overline{u} \in \mathbb{R}^n$ benötigt. △

BEISPIEL 4.24: Gesamtschrittverfahren für Helmholtz-artige DGL.

Wenden wir das Gesamtschrittverfahren zur Lösung des linearen Gleichungssystems für die Helmholtz-artige Differentialgleichung aus Beispiel 4.22 an, so erhalten wir die folgende Iterationsvorschrift für die unbekannte Lösung $u \in \mathbb{R}^{N-1}$:

$$\begin{aligned} u_1^{k+1} &= \frac{1}{2/h^2+\sigma}\left(f_i + \frac{1}{h^2}u_2^k\right), \\ u_i^{k+1} &= \frac{1}{2/h^2+\sigma}\left(f_i + \frac{1}{h^2}(u_{i-1}^k + u_{i+1}^k)\right), \qquad i = 2,\dots,N-2, \\ u_{N-1}^{k+1} &= \frac{1}{2/h^2+\sigma}\left(f_i + \frac{1}{h^2}u_{N-2}^k\right). \end{aligned}$$

Da wir nur maximal 3 Einträge ungleich Null in der Matrix A haben, lässt sich eine Approximation der Lösung u in weniger als $3 \cdot m \cdot N$ FLOPS berechnen, wobei $m \in \mathbb{N}$ die Anzahl der Iterationen darstellt.

Nun stellt sich die Frage wann das Gesamtschrittverfahren gegen eine Lösung $\overline{u} \in \mathbb{R}^n$ des linearen Gleichungssystems $Au = f$ konvergiert. Wir werden zeigen, dass dies durch eine spezielle Klasse von Matrizen, die wir im Folgenden einführen werden, immer der Fall ist.

DEFINITION 4.25: Diagonaldominante Matrizen.

Eine Matrix $A \in \mathbb{R}^{n\times n}$ heißt **stark diagonaldominant**, falls die Beträge ihrer Diagonalelemente $a_{i,i}, i = 1,\dots,n$ jeweils größer sind als die Summer der Beträge der anderen jeweiligen Zeileneinträge $a_{i,j}$, d.h. wenn gilt

$$\sum_{j=1, j\neq i}^{n} |a_{i,j}| \;<\; |a_{i,i}|, \qquad \forall\, i = 1,\dots,n.$$

Man nennt diese Bedingung auch das starke Zeilensummenkriterium.
Ist die Ungleichung nicht strikt, d.h. es gilt lediglich

$$\sum_{j=1, j\neq i}^{n} |a_{i,j}| \;\leq\; |a_{i,i}|, \qquad \forall\, i = 1,\dots,n.$$

so ist die Matrix **schwach diagonaldominant**.

BEMERKUNG 4.26 (Spaltensummenkriterium). Analog zum Zeilensummenkriterium in Definition 4.25 gibt es auch ein starkes und schwaches Spaltensummenkriterium. Dieses ist offensichtlich äquivalent zum Zeilensummenkriterium der transponierten Matrix. △

Für stark diagonaldominante Matrizen können wir nun die Konvergenz des Gesamtschrittverfahrens nachweisen.

THEOREM 4.27: Konvergenz des Gesamtschrittverfahrens.

Das Gesamtschrittverfahren zur Lösung des linearen Gleichungssystems $Au = f$ für eine reguläre Matrix $A \in \mathbb{R}^{n\times n}$ und $u, f \in \mathbb{R}^n$ konvergiert für jeden Startvektor $u^0 \in \mathbb{R}^n$ gegen eine Lösung, falls A das starke Zeilensummenkriterium oder das starke Spaltensummenkriterium erfüllt.

Beweis. Wir verwenden die additive Zerlegung der Matrix A in $A = L + D + R$ und sehen ein, dass D nur Einträge ungleich Null auf der Hauptdiagonale besitzt, da A sonst nicht diagonaldominant wäre.

Um die Konvergenz des Gesamtschrittverfahrens zu zeigen, müssen wir zeigen, dass die Fixpunktiteration

$$u^{k+1} \;=\; G(x^k) \;:=\; -D^{-1}(L+R)u^k + D^{-1}f$$

kontraktiv ist. Hierzu definieren wir uns die Matrix $B \in \mathbb{R}^{n\times n}$ als $B := -D^{-1}(L+R)$. Aus der Gestalt von B wird klar, dass für die Einträge $B_{i,j} \in \mathbb{R}$ gilt:

$$b_{i,j} \;=\; \begin{cases} 0, & \text{falls } i = j, \\ \frac{a_{i,j}}{a_{i,i}}, & \text{sonst.} \end{cases}$$

Falls A das starke Zeilensummenkriterium erfüllt, folgt für die Matrix B schon direkt

$$\begin{aligned} ||B||_\infty \;&=\; \max_{i=1,\dots,n} |\sum_{j=1}^{n} b_{i,j}| \;=\; \max_{i=1,\dots,n} |\sum_{j\neq i} \frac{a_{i,j}}{a_{i,i}}| \\ &\leq\; \max_{i=1,\dots,n} \frac{\sum_{j\neq i} |a_{i,j}|}{|a_{i,i}|} \;<\; \max_{i=1,\dots,n} \frac{|a_{i,i}|}{|a_{i,i}|} \;=\; 1. \end{aligned}$$

Analog kann man zeigen, dass für die Spaltensummernnorm $||B||_1 < 1$ gilt, falls A das starke Spaltensummenkriterium erfüllt.

Nun können wir die Kontraktivität der Fixpunktiteration G leicht zeigen, denn es gilt:

$$\begin{aligned} ||G(u^k) - G(\overline{u})|| \;&=\; ||\underbrace{-D^{-1}(L+R)}_{=B} u^k + D^{-1}b - (\underbrace{-D^{-1}(L+R)}_{=B}\overline{u} + D^{-1}b)|| \\ &=\; ||Bu^k - B\overline{u}|| \;\leq\; \underbrace{||B||}_{=:\,q<1} \cdot ||u^k - \overline{u}||. \end{aligned}$$

Nun können wir den Banachschen Fixpunktsatz aus Theorem 4.4 anwenden, der uns die Konvergenz des Gesamtschrittverfahrens liefert. □

4.6.2 Einzelschrittverfahren

Wenn man auf eine parallele Berechnung der Einträge $u_i^{k+1}, i = 1, \dots, n$ in (4.45) verzichtet und statt dessen eine sequentielle Berechnung des Vektors $u^{k+1} \in \mathbb{R}^n$ vornimmt, so ergibt sich hieraus eine weitere Möglichkeit. Anstatt die Einträge vollkommen unabhängig voneinander zu berechnen könnte man auch auf die bereits aktualisierten, vorangehenden Einträge zurückgreifen, da diese schließlich der echten Lösung $u \in \mathbb{R}^n$ näher sein sollten.

Dies ist die grundlegende Idee beim Einzelschrittverfahren, das auch unter dem Namen **Gauss–Seidel-Verfahren** bekannt ist. Analog zum Gesamtschrittverfahren in Abschnitt 4.6.1

zerlegen wir die gegebene Matrix A wieder additiv zu $A = L + D + R$, jedoch schreiben wir das lineare Gleichungssystem nun wie folgt auf:

$$Au \;=\; (L + D + R)u \;=\; (L + D)u + Ru \;=\; f.$$

Um nun also auf die bereits aktualisierten Einträge der approximativen Lösung $u^{k+1} \in \mathbb{R}^n$ zurück zu greifen, approximieren wir das obige lineare Gleichungssystem durch

$$(L + D)u^{k+1} + Ru^k \;\approx\; f. \tag{4.46}$$

Die Matrix $(L + D) \in \mathbb{R}^{n\times n}$ ist nun eine linke untere Dreiecksmatrix, die genau dann invertierbar ist, falls für die Einträge der Diagonalmatrix D gilt $d_{i,i} \neq 0, i = 1, \ldots, n$. Unter dieser Annahme lässt sich dann eine Iterationsvorschrift für das Einzelschrittverfahren in Matrixschreibweise angeben als

$$x^{k+1} \;=\; (L + D)^{-1}(f - Ru^k).$$

Wenn man diese Iterationsvorschrift für die einzelnen Einträge des Lösungsvektors $u^{k+1} \in \mathbb{R}^n$ betrachtet ergibt sich aus (4.46) für alle $i = 1, \ldots, n$ folgende Form:

$$\begin{aligned}
&\sum_{j=1}^{i-1} a_{i,j}u_j^{k+1} + a_{i,i}u_i^{k+1} + \sum_{j=i+1}^{n} a_{i,j}u_j^k \;=\; f \\
\Rightarrow\quad & a_{i,i}u_i^{k+1} \;=\; f - \sum_{j=1}^{i-1} a_{i,j}u_j^{k+1} - \sum_{j=i+1}^{n} a_{i,j}u_j^k \\
\Rightarrow\quad & u_i^{k+1} \;=\; \frac{1}{a_{i,i}}\left(f_i - \sum_{j=0}^{i-1} a_{i,j}u_j^{k+1} - \sum_{j=i+1}^{n} a_{i,j}u_j^k\right).
\end{aligned} \tag{4.47}$$

BEMERKUNG 4.28 (Rechenaufwand des Einzelschrittverfahrens). Betrachtet man die Rechenvorschrift des Gesamtschrittverfahrens in (4.47), so ist klar, dass man analog zum Gesamtschrittverfahren im schlechtesten Fall einer voll-besetzten Matrix A für die Berechnung eines Eintrags u_i^{k+1} genau n FLOPS benötigt und somit für die Berechnung der Iterierten $u^{k+1} \in \mathbb{R}^n$ insgesamt n^2 FLOPS benötigt. Der Rechenaufwand beträgt also genauso $m \cdot n^2$ FLOPS für $m \in N$ Iterationsschritte wie beim Gesamtschrittverfahren in Bemerkung 4.23, lässt sich allerdings nicht mehr parallelisieren. △

BEISPIEL 4.29: Einzelschrittverfahren für Helmholtz-artige DGL.

Wenden wir das Einzelschrittverfahren zur Lösung des linearen Gleichungssystems für die Helmholtz-artige Differentialgleichung aus Beispiel 4.22 an, so erhalten wir die folgende Iterationsvorschrift für die unbekannte Lösung $u \in \mathbb{R}^{N-1}$:

$$\begin{aligned}
u_1^{k+1} &= \frac{1}{2/h^2 + \sigma}\left(f_i + \frac{1}{h^2}u_2^k\right), \\
u_i^{k+1} &= \frac{1}{2/h^2 + \sigma}\left(f_i + \frac{1}{h^2}(u_{i-1}^{k+1} + u_{i+1}^k)\right), \qquad i = 2, \ldots, N-2, \\
u_{N-1}^{k+1} &= \frac{1}{2/h^2 + \sigma}\left(f_i + \frac{1}{h^2}u_{N-2}^{k+1}\right).
\end{aligned}$$

Da wir nur maximal 3 Einträge ungleich Null in der Matrix A haben, lässt sich eine Approximation der Lösung u in weniger als $3 \cdot m \cdot N$ FLOPS berechnen, wobei $m \in \mathbb{N}$ die Anzahl der Iterationen darstellt.

BEMERKUNG 4.30 (Konvergenz von Gesamt- und Einzelschrittverfahren). Sowohl das Gesamt- als auch das Einzelschrittverfahren Verfahren *konvergieren linear* gemäß Definition 4.10. Dennoch bevorzugt man häufig das Einzelschrittverfahren, da es insgesamt schneller konvergiert als das Gesamtschrittverfahren und man somit Iterationsschritte einspart. Die Konvergenzgeschwindigkeit des Einzelschrittverfahrens lässt sich sogar noch weiter verbessern indem man ein sogenanntes **Successive-Over-Relaxation (SOR) Verfahren** verwendet.
Die Diagonaldominanz der Matrix A des linearen Gleichungssystems $Au = f$ ist ebenfalls eine hinreichendes Kriterium für die Konvergenz des Einzelschrittverfahrens, analog zur Aussage des Gesamtschrittverfahrens in Theorem 4.27. △

Kapitel 5

Interpolation

In diesem Kapitel wollen wir uns mit Interpolationsproblemen aus Sicht der numerischen Mathematik beschäftigen. Bei der Interpolation versucht man im Allgemeinen eine Approximation einer unbekannten Funktion zu finden, die durch eine endliche Anzahl gegebener Datenpunkte exakt verläuft und dazwischen sinnvoll interpoliert. Je nach Anwendung kann man verschiedene Bedingungen an die interpolierende Lösung stellen, so dass es eine Vielzahl von Methoden zur Interpolation gibt.

Bevor wir uns der mathematischen Formulierung des Problems widmen wollen wir ein einführendes Beispiel zur Motivation geben.

BEISPIEL 5.1: Interpolation für Bildskalierung.

Ein mögliches Einsatzgebiet von zweidimensionaler Interpolation ist das Skalieren von digitalen Bildern in der mathematischen Bildverarbeitung. Beim Vergrößern eines Bildes mit niedriger Auflösung in Abbildung 5.1a werden die Pixelwerte auf ein Pixelgitter mit höherer Auflösung projiziert. Die dabei entstehenden Lücken im Bild mit höherer Auflösung müssen durch Interpolation aufgefüllt werden. Je nach Interpolationsmethode erhält man ein Ergebnis mit klaren Kanten, jedoch blockartigen Strukturen (siehe Abbildung 5.1b) oder verwaschenen Kanten, jedoch glatten Strukturen (siehe Abbildung 5.1c).

Man beachte, dass durch Interpolation keine neuen Informationen über eine den Daten zu Grunde liegende, unbekannte analytische Funktion hinzugewonnen werden können. Vielmehr versucht man eine glatte Fortsetzung der gegebenen Daten zu finden, die plausibel für die Anwendung ist.

DEFINITION 5.2: Interpolationsproblem.

Wir betrachten im Folgenden eine Menge mit $n+1 \in \mathbb{N}$ Datenpunkten

$$(x_i, f_i) \in \mathbb{R}^2, \quad i = 0, \dots, n. \tag{5.48}$$

Hierbei geht man davon aus, dass die gegebenen **Stützwerte** f_i die beobachteten Funktionswerte einer unbekannten, kontinuierlichen Funktion $f : \mathbb{R} \to \mathbb{R}$ an den **Stützstellen** x_i darstellen.

(a) Computergrafik mit ursprünglicher Skalierung

(b) Nearest-Neighbor Interpolation

(c) Bikubische Interpolation

(d) Digitales Foto mit ursprünglicher Skalierung

(e) Nearest-Neighbor Interpolation

(f) Bikubische Interpolation

Abbildung 5.1: Skalierung einer Computergrafik und eines digitalen Fotos durch Interpolation.

Das allgemeine Interpolationsproblem besteht nun darin eine **Ansatzfunktion**

$$g\colon \mathbb{R} \times \mathbb{R}^{n+1} \to \mathbb{R},$$

mit freien Parametern $a_0, \ldots, a_n \in \mathbb{R}$ zu finden, so dass diese die gegebenen Daten interpoliert, d.h.,

$$g(x_i; a_0, \ldots, a_n) \;=\; f_i, \qquad i = 0, \ldots, n. \tag{5.49}$$

Das heißt wir sind interessiert an einer Funktion g, die die gegeben Daten exakt repräsentiert und zwischen den Daten sinnvoll (möglichst stetig) verläuft. Mögliche Beispiele für die Wahl von g sind Polynomfunktionen, trigonometrische Funktionen oder Splines.

Wenn man diese Aufgabe mit dem Ziel der linearen Ausgleichsrechnung in Kapitel 3 vergleicht, so stellt man fest, dass die Forderungen an eine interpolierende Funktion *restriktiver* sind. Der Grund hierfür ist, dass im Fall der linearen Ausgleichsrechnung eine Approximation der gegebenen Daten gesucht wird, die die Werte nicht exakt trifft, jedoch die Verteilung der Daten gut annähert und dabei den unvermeidbaren Fehler minimiert.

Ein Spezialfall des allgemeinen Interpolationsproblems in Definition 5.2 ist die lineare Interpolation mit der wir uns in diesem Kapitel hauptsächlich beschäftigen möchten.

DEFINITION 5.3: Lineares Interpolationsproblem.

Wir nennen ein Interpolationsproblem **linear** falls die Ansatzfunktion g in (5.49) nur linear von den Parametern $a_0, \ldots, a_n$ abhängt, d.h. es gilt

$$g(x; a_0, \ldots, a_n) \;:=\; a_0 g_0(x) + a_1 g_1(x) + \ldots + a_n g_n(x). \tag{5.50}$$

Die Funktionen $g_k\colon \mathbb{R} \to \mathbb{R}, k = 0, \ldots, n$ werden in der Regel als *Basisfunktionen* des Funktionenraumes gewählt in dem die Ansatzfunktion g zu bestimmen ist.
Das lineare Interpolationsproblem kann also als ein lineares Gleichungssystem der Form $Ax = b$ dargestellt werden mit:

$$A \;:=\; \begin{pmatrix} g_0(x_0) & \cdots & g_n(x_0) \\ \vdots & \ddots & \vdots \\ g_0(x_n) & \cdots & g_n(x_n) \end{pmatrix}, \qquad x \;:=\; \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix}, \qquad b \;:=\; \begin{pmatrix} f_0 \\ \vdots \\ f_n \end{pmatrix}. \tag{5.51}$$

Jetzt wird der Zusammenhang zwischen linearer Ausgleichsrechnung und linearer Interpolation deutlich. Da wir $n+1$ gemessene Datenpunkte vorliegen haben und wir $n+1$ unbekannte Variablen suchen ist klar, dass wir es mit einem linearen Ausgleichsproblem mit quadratischer Matrix $A \in \mathbb{R}^{(n+1)\times(n+1)}$ zu tun haben. Das lineare Interpolationsproblem (5.50) ist also ein Spezialfall des linearen Ausgleichsproblems (LSTSQ).

BEMERKUNG 5.4 (Wahl der Basisfunktionen). Wir werden im folgenden Abschnitt erkennen, dass auch bei unterschiedlicher Wahl der Basisfunktionen $g_k\colon \mathbb{R} \to \mathbb{R}, k = 0, \ldots, n$ in Definition 5.3 eine eindeutige Ansatzfunktion g als Lösung des linearen Interpolationsproblems bestimmt ist.

Dennoch spielt die Wahl dieser Basisfunktionen g_i eine entscheidende Rolle für die Numerik, da sie Auswirkungen auf den Rechenaufwand und die Entwicklung von numerischen Fehlern hat. △

5.1 Interpolationsformel nach Lagrange

Wir haben in Beispiel 3.5 bereits eine wichtige Klasse von linearen Ausgleichsproblemen betrachtet, nämlich Probleme in denen ein polynomieller Zusammenhang zwischen den gemessenen Daten angenommen wird. Wir wollen uns im Fall der Interpolation im Folgenden auch mit **Polynomfunktionen** als Ansatzfunktion

$$g(x; a_0, \ldots, a_n) \;:=\; P(x) \;=\; \sum_{k=0}^{n} a_k x^k$$

in (5.50) beschäftigen.

DEFINITION 5.5: Funktionenraum der Polynome.

Sei $n \in \mathbb{N}$, dann bezeichnen wir mit Π_n den Funktionenraum aller reellen (oder komplexen) Polynome P vom Grad $p \leq n$ der Form

$$P(x) \;=\; \sum_{k=0}^{n} a_k x^k \;=\; a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n. \tag{5.52}$$

In Kapitel 1 haben wir festgestellt, dass das lineare Gleichungssystem $Ax = b$ genau dann eindeutig lösbar ist, wenn die Matrix $A \in \mathbb{R}^{n\times n}$ vollen Rang $\operatorname{rank}(A) = n$ hat. Wir wollen im Folgenden eine hinreichende Bedingung für den vollen Rang von A mit Hilfe der *Lagrangschen Interpolationsformel* angeben.

THEOREM 5.6: Eindeutigkeit des Interpolationspolynoms.

Seien $(x_i, f_i) \in \mathbb{R}^2, i = 0, \ldots, n$ eine Menge von $n+1$ gemessenen Datenpunkten deren Stützpunkte paarweise verschieden sind, d.h., es gilt $x_i \neq x_j, \forall i \neq j$.
Dann existiert ein *eindeutiges Polynom* $P(x) \in \Pi_n$, das die gemessenen Daten interpoliert mit

$$P(x_i) \;=\; f_i, \qquad \forall\, i = 0, \ldots, n.$$

Beweis. Wir beginnen damit die *Eindeutigkeit* eines Interpolationspolynoms $P \in \Pi_n$ zu beweisen, falls es existiert. Wir nehmen an es gäbe zwei Polynome $P_1, P_2 \in \Pi_n$ die an den Datenpunkten übereinsimmen, d.h. es gelte

$$P_1(x_i) \;=\; P_2(x_i) \;=\; f_i, \quad \forall i = 0, \ldots, n.$$

Betrachten wir nun das Polynom $P(x) = P_1(x) - P_2(x)$ als Differenz der vorigen Polynome, so ist klar, dass P maximal Ordnung n haben kann und somit $P \in \Pi_n$ gilt. Andererseits muss P mindestens $n+1$ Nullstellen, nämlich genau in den Stützstellen $x_i, i = 0, \ldots, n$ besitzen. Aus der Algebra wissen wir jedoch, dass dies nur sein kann, wenn P die Nullfunktion ist, d.h. wenn $P(x) \equiv 0$ gilt. Daraus folgt aber auch schon die Eindeutigkeit, denn dann war bereits $P_1 \equiv P_2$ überall.

Nun widmen wir uns der *Existenz* eines solchen Interpolationspolynoms. Hierbei kann man nach Lagrange spezielle Interpolationspolynome $L_k \in \Pi_n, k = 0, \dots, n$ konstruieren von der folgenden Form

$$L_k(x) \coloneqq \prod_{i=0, i\neq k}^{n} \frac{(x - x_i)}{(x_k - x_i)} = \frac{(x - x_0)\cdots(x - x_{k-1})(x - x_{k+1})\cdots(x - x_n)}{(x_k - x_0)\cdots(x_k - x_{k-1})(x_k - x_{k+1})\cdots(x_k - x_n)}. \tag{5.53}$$

Man bezeichnet diese Polynome auch als **Lagrange-Polynome**. Da die Stützstellen $x_i, i = 0, \dots, n$ als paarweise verschieden vorausgesetzt waren, sind die Polynome L_k in (5.53) wohldefiniert. Außerdem gilt offensichtlich, dass die $L_k, k = 0, \dots, n$ Polynome vom Grad n sind. Des Weiteren können wir die Lagrange-Polynome L_k noch umschreiben in

$$L_k(x) = \frac{\omega_{n+1}(x)}{\omega'_{n+1}(x_k)(x - x_k)}, \quad \text{mit} \quad w_{n+1}(x) \coloneqq \prod_{i=0}^{n}(x - x_i) \in \Pi_{n+1}$$

und es lässt sich folgende nützliche Eigenschaften zeigen:

$$L_k(x_i) = \delta_{k,i} = \begin{cases} 1, & \text{für } k = i, \\ 0, & \text{für } k \neq i. \end{cases} \tag{5.54}$$

Wegen der bereits oben gezeigten Eindeutigkeit des Interpolationspolynoms sind die Polynome L_k also eindeutig bestimmt.

Nun lässt sich mit Hilfe der **Lagrangschen Interpolationsformel** die Lösung des linearen Interpolationsproblems mittels Polynomen angeben:

$$P(x) = \sum_{k=0}^{n} f_k L_k(x) = \sum_{k=0}^{n} f_k \prod_{i=0, i\neq k}^{n} \frac{(x - x_i)}{(x_k - x_i)}. \tag{5.55}$$

Offensichtlich gilt $P(x_i) = f_i, \forall i = 0, \dots, n$ und damit haben wir die Existenz eines solchen Polynoms bewiesen. □

BEMERKUNG 5.7. Setzt man diese Polynome als Basisfunktionen $g_k(x_i) \coloneqq L_k(x_i)$ mit $k, i = 0, \dots, n$ in die Matrix A des linearen Interpolationsproblems in (5.51) ein, so ergibt sich wegen des Kronecker-Deltas in (5.54) die Einheitsmatrix. Das bedeutet, dass die unbekannten Koeffizienten einfach $a_k = f_k, k = 0, \dots, n$ sind, da offensichtlich $x = b$ in (5.51) gilt.
Wir bemerken, dass die Lagrange-Polynome $L_k, k = 0, \dots, n$ in (5.53) völlig unabhängig von den *Stützwerten* definiert sind. Sind diese also erst einmal für fixe, paarweise verschiedene Stützstellen $x_i \in \mathbb{R}, i = 0, \dots, n$ berechnet, so lässt sich das Polynom für beliebige Mengen von Stützwerten $f_i \in \mathbb{R}, i = 0, \dots, n$ effizient auswerten.
Der Nachteil der Lagrange-Interpolation liegt jedoch darin, dass man alle Lagrange-Polynome neu berechnen muss, wenn man eine neue Stützstelle hinzunimmt. △

5.2 Vandermonde-Matrix

Anstatt für eine gegebene Menge von $n+1$ Datenpunkten ein Interpolationspolynom mit Hilfe der Lagrangschen Interpolationsformel (5.55) zu berechnen, lassen sich auch Monome als Basisfunktionen $g_k(x) \coloneqq x^k, k = 0, \dots, n$ wählen, so dass man ein anderes lineares Gleichungssystem in (5.51) erhält, bei der die Matrix A die sogenannte **Vandermonde-Matrix** darstellt.

DEFINITION 5.8: Vandermonde-Matrix.

Seien $(x_i, f_i) \in \mathbb{R}^2, i = 0, \ldots, n$ eine Menge von $n+1$ gemessenen Datenpunkten. Wir bezeichnen die Matrix

$$A = \begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \tag{5.56}$$

als **Vandermonde-Matrix**.
Mit ihr lässt sich das eindeutige Interpolationspolynome als Lösung des Gleichungssystems $Ax = b$ bestimmen, wobei $x = (a_0, \ldots, a_n)^T$ die unbekannten Koeffizienten des Polynoms darstellen und $b = (f_0, \ldots, f_n)^T$ die beobachteten Stützwerte sind.

BEMERKUNG 5.9. Wir können folgende Aussagen zur Bestimmung des Interpolationspolynoms $P(x)$ mit Hilfe der Vandermonde-Matrix machen:

(i) Es lässt sich leicht zeigen, dass die Vandermonde-Matrix in (5.56) genau dann **invertierbar** ist, wenn die Datenpunkte $(x_i, f_i) \in \mathbb{R}^2, i = 0, \ldots, n$ paarweise verschieden sind.

(ii) Die numerische Lösung des resultierenden linearen Gleichungssystems $Ax = b$ ist in der Regel **numerisch ineffizient**, da die Vandermonde-Matrix im Allgemeinen voll besetzt ist und keine weiteren günstigen Eigenschaften aufweist, z.B., Hermizität. Aus diesem Grund sollte man diese Herangehensweise für eine große Anzahl $n \in \mathbb{N}, n \gg 1$ von Datenpunkten nur dann wählen, wenn man eine Reihe von $m \in \mathbb{N}$ Interpolationsproblemen mit gleichen Stützstellen $x_0, \ldots, x_n$ betrachtet bei der sich nur die Stützwerte $f_0, \ldots, f_n$ ändern. Denn dann muss man die Matrix A nur ein einziges Mal invertieren.

(iii) Die numerische Lösung des resultierenden linearen Gleichungssystems $Ax = b$ ist in der Regel **numerisch instabil**, da die Vandermonde-Matrix sehr schlecht konditioniert ist.

△

Um den Wert eines Polynoms $P \in \Pi_n$, mit $P(x) = \sum_{k=0}^{n} a_k x^k$ an einer Stelle $\xi \in \mathbb{R}$ auszurechnen benötigen wir $n+1$ Additionen und $2n$ Multiplikationen. Jedoch gibt es eine alternative Schreibweise des Polynoms $P(x)$, die uns eine günstigere Auswertung erlaubt.

DEFINITION 5.10: Horner-Schema.

Sei $P(x) \in \Pi_n$ ein Polynom von Grad $n \in \mathbb{N}$ mit der Darstellung

$$P(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n.$$

Dann lautet eine äquivalente Darstellung von P im **Horner-Schema** wie folgt:

$$P(x) = (\ldots(a_n x + a_{n-1})x + a_{n-2})x \ldots)x + a_1)x + a_0. \tag{5.57}$$

Der numerische Rechenaufwand zur Berechnung des Polynoms P an einer Stelle $\xi \in \mathbb{R}$ reduziert sich durch die Verwendung des Horner-Schemas in Definition 5.10 signifikant, da wir nun nur noch n Multiplikationen und n Additionen durchführen müssen.

5.3 Interpolationsformel nach Newton

Die in Abschnitt 5.1 beschriebenen Verfahren, d.h. die Lagrangesche Interpolationsformel und die Vandermonde-Matrix, haben den Nachteil, dass man alle durchgeführten Berechnungen zur Bestimmung des Interpolationspolynoms durch $n+1$ Stützpunkte nicht erneut verwenden kann, sobald ein weiterer Stützpunkt hinzugenommen wird. Das heißt bei Hinzunahme weiterer Daten ist der bisher betriebene numerische Rechenaufwand umsonst, da man zur Bestimmung der neuen Polynomkoeffizienten wieder von vorne beginnen muss.

Glücklicherweise gibt es eine alternative Herangehensweise, die uns einen signifikanten Teil der Neuberechnung von Koeffizienten erspart. Anstatt die herkömmliche Polynomdarstellung wie in (5.52) zu verwenden verfolgt man den folgenden Ansatz nach Newton.

DEFINITION 5.11: Newtonsche Interpolationsformel.

Seien $(x_i, f_i) \in \mathbb{R}^2, i = 0, \dots, n$ eine Menge von $n+1$ gegebenen Datenpunkten. Das unbekannte Interpolationspolynom $P \in \Pi_n$ vom Grad n lässt sich mit der **Newtonschen Interpolationsformel** wie folgt darstellen:

$$\begin{aligned} P(x) &= a_0 + a_1(x - x_0) + a_2(x - x_1)(x - x_0) + \dots + a_n(x - x_{n-1}) \dots (x - x_0) \\ &= \sum_{k=0}^{n} a_k N_k(x), \end{aligned} \tag{5.58}$$

wobei wir die **Newtonschen Basispolynome** $N_k, k = 0, \dots, n$ definieren als

$$N_k(x) := \prod_{i=0}^{k-1} (x - x_i) \quad \text{und} \quad N_0(x) \equiv 1. \tag{5.59}$$

BEMERKUNG 5.12. Wir können folgende Beobachtungen zur Newtonschen Interpolationsformel machen:

(i) Durch die Wahl der Basispolynome $g_k(x) := N_k(x), k = 0, \dots, n$ lässt sich die Bestimmung der unbekannten Koeffizienten $a_0, \dots, a_n \in \mathbb{R}$ eines Polynoms $P \in \Pi_n$ mit $P(x_i) = f_i, i = 0, \dots, n$ als ein lineares Gleichungssystem $Lx = b$ schreiben mit einer linken, unteren Dreiecksmatrix $L \in \mathbb{R}^{(n+1)\times(n+1)}$ von der Form:

$$\begin{pmatrix} N_0(x_0) & 0 & & 0 \\ N_0(x_1) & N_1(x_1) & 0 & \\ \vdots & \vdots & \ddots & 0 \\ N_0(x_n) & N_1(x_n) & \cdots & N_n(x_n) \end{pmatrix} \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f_0 \\ \vdots \\ f_n \end{pmatrix}.$$

Man sieht ein, dass man durch Hinzufügen von weiteren Daten nur eine zusätzliche Zeile und Spalte in L erhält, die an der Dreiecksgestalt nichts ändern.

(ii) Für bekannte Koeffizienten $a_0, \dots, a_n$ benötigt die Auswertung des Lagrange-Polynoms an einer Stelle ξ insgesamt $2n$ Additionen und $2n$ Multiplikationen. Das effizientere Horner-Schema der Newtonschen Interpolationsformel lässt sich folgendermaßen ange-

ben:

$$P(x) \;=\; (\ldots(a_n(x-x_{n-1})+a_{n-1})(x-x_{n-2})+\ldots+a_1)(x-x_0)+a_0. \tag{5.60}$$

In dieser Form benötigt die Berechnung des Polynoms an einer Stelle ξ insgesamt nur noch $2n$ Additionen und n Multiplikationen. Der numerische Rechenaufwand ist also höher als in der herkömmlichen Polynomdarstellung (5.52). Dennoch bietet die Newtonsche Interpolationsformel den Vorteil, dass man weitere Datenpunkte hinzunehmen kann ohne alle Berechnungen erneut durchzuführen, wie wir im weiteren Verlauf noch genauer diskutieren werden.

△

Offensichtlich handelt es sich in (5.58) um ein Polynom von Grad n mit $n+1$ unbekannten Koeffizienten $a_0,\ldots,a_n$, die nach Satz 5.6 eindeutig bestimmt sind, da wir die Stützstellen $x_i, i=0,\ldots,n$ als paarweise verschieden angenommen haben.

Zur numerischen Berechnung dieser unbekannten Koeffizienten kann man folgenden sukzessiven Algorithmus verwenden.

ALGORITHMUS 5.13: Interpolationsschema nach Newton.

Es sei eine Menge von $n+1$ Datenpunkten $(x_i,f_i)\in\mathbb{R}^2, i=0,\ldots,n$ gegeben, deren Stützstellen paarweise verschieden sind, d.h. es gilt $x_i\neq x_j$ für $i\neq j$. Bei der Betrachtung der Newtonschen Interpolationsformel (5.58) fällt auf, dass alle Summanden bis auf den ersten für die Stützstelle $x=x_0$ wegfallen und wir die folgende Gleichung erhalten:

$$P(x_0) \;=\; a_0.$$

Da $P\in\Pi_n$ jedoch gleichzeitig Interpolationspolynom bezüglich der gegebenen Daten sein muss, wissen wir, dass auch $P(x)=f_0$ gelten muss. Also setzen wir im **ersten Schritt**

$$a_0 \;=\; f_0.$$

Wir können nun wieder die Newtonsche Interpolationsformel betrachten für den Fall $x=x_1$ und erkennen, dass alle Summanden bis auf die ersten beiden wegfallen. Wir erhalten somit die Gleichung

$$P(x_1) \;=\; a_0+(x_1-x_0)a_1 \;=\; f_0+(x_1-x_0)a_1.$$

Da $P(x_1)=f_1$ gelten muss können wir den unbekannten Koeffizienten a_1 also im **zweiten Schritt** bestimmen als

$$a_1 \;=\; \frac{f_1-f_0}{x_1-x_0}.$$

Dieses Vorgehen lässt sich nun sukzessiv anwenden und wir können die Berechnung des Koeffizienten $a_j, 1\leq j\leq n$ im **j-ten Schritt** allgemein angeben als:

$$\begin{aligned} a_j \;&=\; \frac{f_j-a_0}{\prod_{i=0}^{j-1}(x_j-x_i)} - \frac{a_1}{\prod_{i=1}^{j-1}(x_j-x_i)} - \ldots - \frac{a_{j-1}}{(x_j-x_{j-1})} \\ &=\; \frac{f_j}{\prod_{i=0}^{j-1}(x_j-x_i)} - \sum_{k=0}^{j-1}\frac{a_k}{\prod_{i=k}^{j-1}(x_j-x_i)}. \end{aligned}$$

BEMERKUNG 5.14 (Rechenaufwand des Newtonschen Interpolationsschemas). Die Berechnung der unbekannten Koeffizienten $a_i, i = 0, \dots, n$ benötigt bei der Nutzung des Algorithmus 5.13 einen numerischen Rechenaufwand von n Divisionen, $n(n-1)$ Multiplikationen und $n(n+1)$ Additionen. △

Wir stellen bei der Berechnung der unbekannten Koeffizienten $a_i, i = 0, \dots, n$ in der Newtonschen Interpolationsformel (5.58) fest, dass der j-te Koeffizient a_j nur von Koeffizienten und Datenpunkten mit niedrigerem Index abhängt. Dies ist eine sehr nützliche Eigenschaft, denn dies erlaubt es uns zu einem bereits berechneten Interpolationspolynom in $n+1$ Datenpunkten weitere Daten hinzuzufügen ohne die bisher berechneten Koeffizienten neu bestimmen zu müssen. Man kann sogar zeigen, dass die Anordnung der Stützstellen $x_i, i = 0, \dots, n$ bei der Berechnung der Koeffizienten keine Rolle spielen. Das bedeutet, dass wir auch neue Daten zwischen zwei bekannten Stützstellen hinzunehmen können, wie im folgenden Beispiel.

BEISPIEL 5.15: Newton-Interpolation.

Es seien zunächst die folgenden Datenpunkte $(x_i, f_i) \in \mathbb{R}^2, i = 0, 1$ gegeben:

x_i	0	2
f_i	0	4

Wir wollen das eindeutige Interpolationspolynom $P \in \Pi_1$ mit dem Newton Verfahren für Polynominterpolation bestimmen. Hierfür bestimmen wir die unbekannten Koeffizienten a_0 und a_1 wie in Algorithmus 5.13 hergeleitet als:

$$a_0 = f_0 = 0, \qquad a_1 = \frac{f_1 - f_0}{x_1 - x_0} = \frac{4-0}{2-0} = 2.$$

Durch Einsetzen in die Newtonsche Interpolationsformel (5.58) erhalten wir das eindeutige, lineare Interpolationspolynom $P \in \Pi_1$ als:

$$P(x) = a_0 + (x - x_0)a_1 = 0 + (x-0)2 = 2x.$$

Nehmen wir nun an, dass wir einen weiteren Datenpunkt $(x_2, f_2) = (1, 1)$ zusätzlich erhalten, dessen Stützstelle $x_2 = 1$ zwischen den vorigen Stützstellen $x_0 = 0$ und $x_1 = 2$ liegt. Wir wollen nun das eindeutige, quadratische Interpolationspolynom $P \in \Pi_2$ finden, das alle Daten interpoliert. Hierfür müssen wir nur den zusätzlichen Koeffizienten a_2 mit Algorithmus 5.13 berechnen:

$$a_2 = \frac{f_2 - a_0}{(x_2 - x_0)(x_2 - x_1)} - \frac{a_1}{(x_2 - x_1)} = \frac{1}{-1} - \frac{2}{-1} = 1.$$

Durch Erweitern des bereits bekannten linearen Interpolationspolynoms aus den ursprünglichen Daten erhalten wir:

$$P(x) = 2x + a_2(x - x_1)(x - x_0) = 2x + 1(x-2)(x-0) = 2x + x^2 - 2x = x^2.$$

Wie aus dem obigen Beispiel 5.15 klar werden sollte, handelt es sich bei den Koeffizienten a_i nicht um die Vorfaktoren der üblichen Polynombasis aus Monomen x^i. Daher muss man die geläufige Darstellung eines Polynoms wie in (5.52) erst aus der Newtonschen Interpolationsformel (5.58) durch Einsetzen berechnen.

	$k=0$	$k=1$	$k=2$	$\dots$
x_0	$\mathbf{f_0}$			
		$\mathbf{f_{0,1}}$		
x_1	f_1		$\mathbf{f_{0,1,2}}$	
		$f_{1,2}$		
x_2	f_2			
$\vdots$				

Tabelle 5.1: Ein Ausschnitt des dividierte Differenzen Schemas.

5.3.1 Dividierte Differenzen

Anstatt die unbekannten Koeffizienten $a_i, i = 0, \dots, n$ des Interpolationspolynoms $P \in \Pi_n$ mit dem oben beschriebenen Algorithmus 5.13 zu berechnen verwendet man eine effizientere Methode, die auf die Idee der dividierten Differenzen basiert. Wir betrachten hierzu noch einmal die Newtonsche Interpolationsformel (5.58) für das eindeutige Interpolationspolynom Polynom $P_{0,\dots,n} \in \Pi_n$ vom Grad n durch $n+1$ Datenpunkte $(x_i, f_i) \in \mathbb{R}^2, i = 0, \dots, n$. Wir tauschen hierbei die Bezeichnung der unbekannten Koeffizienten a_i wie folgt:

$$P_{0,\dots,n}(x) \;=\; f_0 \,+\, f_{0,1}N_1x \,+\, \dots \,+\, f_{0,\dots,n}N_n(x). \tag{5.61}$$

Die Zahlen $f_{0,\dots,n}$ nennt man **dividierte Differenzen** und es lässt sich zeigen, dass folgende Rekursionsformel für sie gilt:

$$f_{i,\dots,i+k} \;=\; \frac{f_{i+1,\dots,i+k} \,-\, f_{i,\dots,i+k-1}}{x_{i+k} \,-\, x_i}. \tag{5.62}$$

Die rekursive Abhängigkeit der dividierten Differenzen lässt sich mit Hilfe des folgenden Differenzenschemas veranschaulichen, welches sich spaltenweise beginnend mit der Spalte $k = 0$ berechnen lässt.

BEMERKUNG 5.16 (Rekursionsformel nach Neville-Aitken). ie rekursive Abhängigkeit der dividierten Differenzen in (5.62) sind ein Spezialfall der allgemeineren Rekursionsformel von Neville-Aitken. △

Mit steigendem Grad n des zu bestimmenden Polynoms $P_{0,\dots,n} \in \Pi_n$ wächst die rekursive Abhängigkeit der neu zu bestimmenden Koeffizienten. Hierbei lässt sich eine zu berechnende dividierte Differenzen einfach aus den beiden linken Nachbarn aus der vorigen Spalte bestimmen. Zum Beispiel gilt folgender Zusammenhang im obigen Schema in Tabelle 5.1 für die dividierten Differenzen:

$$f_{0,1} \;=\; \frac{f_1 - f_0}{x_1 - x_0}, \quad f_{1,2} \;=\; \frac{f_2 - f_1}{x_2 - x_1}, \quad f_{0,1,2} \;=\; \frac{f_{1,2} - f_{0,1}}{x_2 - x_0}.$$

Die fett markierten dividierten Differenzen in der obersten Schrägzeile des Schemas ergeben die eindeutig bestimmten Koeffizienten des Interpolationspolynoms $P \in \Pi_n$ in (5.61).

BEMERKUNG 5.17 (Rechenaufwand für dividierte Differenzen). Der numerische Rechenaufwand zur Bestimmung der dividierten Differenzen für ein Interpolationspolynom

$P \in \Pi_n$ vom Grad n beträgt $n(n+1)/2$ Divisionen und $n(n+1)$ Additionen und ist damit deutlich effizienter als das Newtonsche Interpolationsschema (vgl. Bemerkung 5.14). △

Folgendes Beispiel erklärt die Berechnung der dividierten Differenzen anschaulich.

BEISPIEL 5.18: Dividierte Differenzen.

Es seien die folgenden Datenpunkte $(x_i, f_i) \in \mathbb{R}^2, i = 0, 1, 2$ gegeben:

x_i	0	1	3
f_i	1	3	2

Wir wollen das eindeutige Interpolationspolynom $P_{0,1,2} \in \Pi_2$ mittels dividierter Differenzen bestimmen. Hierfür verwenden wir das Schema 5.1.

	$k=0$	$k=1$	$k=2$	$\dots$
$x_0 = 0$	$\mathbf{f_0 = 1}$			
		$\mathbf{f_{0,1} = 2}$		
$x_1 = 1$	$f_1 = 3$		$\mathbf{f_{0,1,2} = -\frac{5}{6}}$	
		$f_{1,2} = -\frac{1}{2}$		
$x_2 = 3$	$f_2 = 2$			
$\vdots$				

Durch Ablesen der Koeffizienten $f_0, f_{0,1}, f_{0,1,2}$ des Newtonschen Interpolationspolynoms $P_{0,1,2}$ entlang der obersten Schrägzeile erhalten wir das Polynom

$$P_{0,1,2}(x) \;=\; f_0 + f_{0,1}(x - x_0) + f_{0,1,2}(x - x_0)(x - x_1) \;=\; 1 + 2(x-0) - \frac{5}{6}(x-0)(x-1).$$

Wollen wir das Polynom an einer Stelle $\xi = 2$ auswerten so berechnen wir in dieser Darstellung mittels Horner-Schema in (5.60)

$$P_{0,1,2}(2) \;=\; (-\frac{5}{6}(2-1) + 2)(2-0) + 1 \;=\; \frac{10}{3}.$$

5.3.2 Dämpfung von Rundungsfehlern

Es lässt sich zeigen, dass man die Indizierung der Datenpunkte $(x_i, f_i), i = 0, \dots, n$ beliebig durch Permutation ändern kann in der Newtonschen Interpolationsformel (5.58) und man trotzdem das gleiche Interpolationspolynom erhält, wie wir bereits angemerkt haben. Das heißt, wir würden bei der Methode der dividierten Differenzen für eine beliebige Permutation $i_0, \dots, i_n = \Pi(0, \dots, n)$ das gleiche Interpolationspolynom $P_{i_0,\dots,i_n} \equiv P_{0,\dots,n}$ erhalten. Leider spielen bei der Implementierung des Verfahrens Rundungsfehler eine Rolle, so dass dieses theoretische Resultat numerisch nicht immer zutrifft. Jedoch hilft uns diese Aussage, um bei der Auswertung eines gegebenen Polynoms weniger Rundungsfehler zu machen, wie man im Folgenden sehen wird.

Wir nehmen an, dass die gegebenen Stützstellen $x_i, i = 0, \dots, n$ monoton steigend angeordnet sind, d.h. es gilt $x_0 < x_1 < \dots < x_n$. Das Ziel wird es sein das gegebene Interpolationspolynom $P_{0,\dots,n} \in \Pi_n$ vom Grad n an einer Stelle ξ so auszuwerten, dass möglichst wenig Rundungsfehler

entstehen. Sei nun x_{i_0} die zu ξ nächstgelegene Stützstelle, d.h. es gilt

$$|\xi - x_{i_0}| \;=\; \min\{|x - x_i| \mid i = 0, \ldots, n\}.$$

Weiter können wir die Stützstellen $x_i, i = 0, \ldots, n$ nach ihrer Entfernung zu ξ sortieren und dabei die folgende Indizierung wählen:

$$|\xi - x_{i_k}| \;=\; \min\{|x - x_i| \mid i \in I_k\},$$

mit der Indexmenge $I_k = \{0, \ldots, n\} \backslash \{i_0, \ldots, i_{k-1}\}$. Durch diese Umsortierung wird eine Permutation $\Pi(0, \ldots, n) = i_0, \ldots, i_n$ induziert. Dadurch erhalten wir eine alternative Darstellung des Newtonschen Interpolationspolynoms mit

$$P_{i_0,\ldots,i_n}(x) \;=\; f_{i_0} + f_{i_0,i_1}(x - x_{i_0}) + \ldots + f_{i_0,\ldots,i_n}(x - x_{i_0}) \ldots (x - x_{i_{n-1}}).$$

Schreiben wir das Polynom $P_{i_0,\ldots,i_n}(x)$ bei der Auswertung an der Stelle ξ in das zugehörige Horner-Schema (5.60) um, so erkennt man den Vorteil dieser Permutation:

$$P_{i_0,\ldots,i_n}(\xi) \;=\; (\ldots(f_{i_0,\ldots,i_n}(\xi - x_{i_{n-1}}) + f_{i_0,\ldots,i_{n-1}})(\xi - x_{i_{n-2}}) + \ldots + f_{i_0,i_1})(\xi - x_{i_0}) + f_{i_0}.$$

Es ist davon auszugehen, dass eine **Dämpfung der Rundungsfehler** bei der Auswertung von $P_{i_0,\ldots,i_n}(\xi)$ auftritt, da jeder Term mit einem immer kleiner werdenden Faktor multipliziert wird.

Da wir angenommen haben, dass die Stützstellen $x_0 < x_1 < \ldots < x_n$ monoton angeordnet sind folgt sofort, dass die von uns gewählte Permutation eine zusammenhängende Indexmenge liefert, d.h. es gilt

$$\{i_l, \ldots, i_{l+k}\} \;=\; \{0 \le i \le n \mid \min_{I_{l,k}} i_j \le i \le \max_{I_{l,k}} i_j\},$$

mit der Indexmenge $I_{l,k} = \{i_l, \ldots, i_{l+k}\}$. Dadurch lassen sich die dividierten Differenzen aus dem Schema 5.1 verwenden. Jedoch wird bei der Auswertung nicht die oberste Schrägzeile genutzt, sondern ein Zickzack-Pfad im Schema verwendet, der durch die Permutation induziert wird.

BEISPIEL 5.19: Dämpfung von Rundungsfehlern.

Es seien die gleichen Datenpunkte wie in Beispiel 5.18 gegeben. Wir wollen die bereits berechneten dividierten Differenzen in der Art umsortieren, so dass bei Auswertungen mittels Horner-Schema ein möglichst geringer Rundungsfehler entsteht. Sei die Auswertestelle nun $\xi = 2$, so erhalten wir nach dem oben beschriebenen Vorgehen eine Permutation der Indizes mit:

$$i_0 \;=\; 1, \quad i_1 = 2, \quad i_2 \;=\; 0.$$

Der zugehörige Pfad im dividierte Differenzenschema ist fett markiert:

	$k=0$	$k=1$	$k=2$	$\dots$
$x_0=0$	$f_0=1$			
		$f_{0,1}=2$		
$x_1=1$	$\mathbf{f_1=3}$		$\mathbf{f_{0,1,2}=-\frac{5}{6}}$	
		$\mathbf{f_{1,2}=-\frac{1}{2}}$		
$x_2=3$	$f_2=2$			
$\vdots$				

Durch Ablesen der fett markierten Koeffizienten $f_1, f_{1,2}, f_{0,1,2}$ des Newtonschen Interpolationspolynoms $P_{0,1,2}$ erhalten wir das Polynom

$$P_{0,1,2}(x) \;=\; f_1 + f_{1,2}(x-x_1) + f_{0,1,2}(x-x_1)(x-x_2) \;=\; 3 - \frac{1}{2}(x-1) - \frac{5}{6}(x-1)(x-3).$$

Wollen wir das Polynom nun an der Stelle $\xi = 2$ auswerten, so berechnen wir in dieser Darstellung mittels Horner-Schema (5.60)

$$P_{0,1,2}(2) \;=\; (-\frac{5}{6}(2-3) - \frac{1}{2})(2-1) + 3 \;=\; \frac{10}{3}.$$

5.4 Stabilität und Fehlerabschätzung der Polynominterpolation

Wir wollen uns im Folgenden mit der Frage beschäftigen, welche Aussagen man zur numerischen Stabilität der Polynominterpolation machen kann und ob es nützliche Fehlerabschätzungen gibt. Nehmen wir hierfür zuerst an, dass die bisher beobachteten Datenpunkte $(x_i, f_i) \in \mathbb{R}^2, i = 0, \dots, n$ Abtastungen einer zu Grunde liegenden Funktion f sind, für die gilt $f(x_i) = f_i$. Dann ergibt sich natürlich die Frage, wie gut das eindeutige Interpolationspolynom $P \in \Pi_n$, das die Datenpunkte (x_i, f_i) interpoliert, die Funktion f approximiert.

5.4.1 Fehlerabschätzung

Es wird klar, dass die Abweichung $|P(x) - f(x)|$ für beliebiges $x \neq x_i, i = 0, \dots, n$ bei geeigneter Wahl von f beliebig groß werden kann, falls man keine weiteren Forderungen an die Funktion f stellt. Folgender Satz liefert eine Fehlerabschätzung für genügend glatte Funktionen

THEOREM 5.20: Fehler der Polynominterpolation.

Sei f eine $(n+1)$-mal stetig differenzierbare Funktion, d.h. es gilt $f \in C^{n+1}(\mathbb{R})$. Seien weiter $(x_i, f_i) \in \mathbb{R}^2, i = 0, \dots, n$ eine Menge von $n+1$ Datenpunkten deren Stützstellen paarweise verschieden sind mit $f(x_i) = f_i, i = 0, \dots, n$. Es sei $P \in \Pi_n$ das eindeutig bestimmte Interpolationspolynom von Grad n, das die Datenpunkte interpoliert mit $P(x_i) = f_i, i = 0, \dots, n$.
Dann gibt es für jedes $\overline{x} \in I$ ein $\xi \in I$ aus dem kleinsten Intervall I, das alle Stützstellen x_i und $\overline{x}$ enthält, so dass gilt:

$$f(\overline{x}) - P(\overline{x}) \;=\; \frac{w_{n+1}(\overline{x})}{(n+1)!} f^{(n+1)}(\xi),$$

wobei $w_{n+1}(x) = (x-x_0)(x-x_1)\dots(x-x_n)$ die Basispolynome aus der Lagrangeschen Interpolationsformel (5.55) sind.

Beweis. Für $\overline{x} = x_i, i \in \{0,\dots,n\}$ ist nichts zu zeigen, denn sowohl der Fehler $f(x_i) - P(x_i)$ als auch der Wert $w_{n+1}(\overline{x})$ sind Null in den Stützstellen. Sei nun also $\overline{x} \in I$ mit $\overline{x} \neq x_i, i = 0\dots,n$ beliebig. Wir betrachten nun die Hilfsfunktion

$$F(x) \;:=\; f(x) - P(x) - K \cdot w_{n+1}(x).$$

Wählt man die Konstante $K \in \mathbb{R}$ nun so, dass die Funktion F an der Stelle $x = \overline{x}$ verschwindet, so stellt man fest, dass F mindestens die $n+2$ Nullstellen $x_0,\dots,x_n,\overline{x}$ auf dem Intervall I besitzt.

Nach dem **Satz von Rolle** besitzt somit F' mindestens $n+1$ Nullstellen in I, F'' mindestens n Nullstellen in I, usw. bis schließlich $F^{(n+1)}$ mindestens eine Nullstelle ξ in I besitzt. Da das Interpolationspolynom $P \in \Pi_n$ vom Grad n ist, gilt aber auch schon $P^{(n+1)}(x) \equiv 0$ und somit erhalten wir für ein $\xi \in I$

$$F^{(n+1)}(\xi) \;=\; f^{(n+1)}(\xi) \;-\; \underbrace{P^{(n+1)}(\xi)}_{=0} + K \cdot (n+1)! \;=\; 0.$$

Damit wissen wir, dass die Konstante K wie folgt gewählt werden muss

$$K \;:=\; \frac{f^{(n+1)}(\xi)}{(n+1)!}.$$

Mit dieser Wahl folgt dann die Behauptung

$$f(\overline{x}) - P(\overline{x}) \;=\; K \cdot w_{n+1}(\overline{x}) \;=\; \frac{w_{n+1}(\overline{x})}{(n+1)!} f^{(n+1)}(\xi).$$

□

Um den Fehler in Theorem 5.20 noch besser interpretieren zu können, müssen wir zunächst den Term w_{n+1} vernünftig abschätzen.

KOROLLAR 5.21: Fehlerabschätzung in der ∞-Norm.

Für ein gegebenes Intervall I, das die Stützstellen $x_i, i = 0,\dots,n$ enthält, lässt sich die folgende Abschätzung treffen

$$||w_{n+1}||_\infty \;=\; \max_{x\in I} |(x-x_0)\cdot\ldots\cdot(x-x_n)| \;\leq\; \frac{n!}{4} h_{n+1}, \tag{5.63}$$

wobei h_{n+1} der größte Abstand zweier benachbarter Stützstellen ist. Mit Hilfe dieser Abschätzung können wir eine obere Schranke für den Fehler der Polynominterpolation auf dem Intervall I angeben:

$$||f - P||_\infty \;\leq\; \frac{||f^{(n+1)}||_\infty}{4(n+1)} h_{n+1}.$$

Diese Fehlerabschätzung erlaubt es uns in Spezialfällen zu zeigen, dass der Approximationsfehler der Polynominterpolation gleichmäßig gegen 0 konvergiert, wie das folgende Lemma zeigt.

KOROLLAR 5.22: Gleichmäßige Konvergenz.

Sei $f \in C^\infty(I)$ und es seien alle Ableitung von f beschränkt, d.h. $||f^{(n)}||_\infty \leq C, C > 0$. Dann lässt sich zeigen, dass der Approximationsfehler der Polynominterpolation für wachsende Anzahl von Stützstellen verschwindet, d.h.,

$$\lim_{n\to\infty} ||f - P_n||_\infty \;=\; 0.$$

Beweis. Die Aussage folgt direkt aus dem Abschnitt 5.4.1, denn es gilt:

$$\begin{aligned}\lim_{n\to\infty} ||f - P_n||_\infty \;&\leq\; \lim_{n\to\infty} \frac{||f^{(n+1)}||\infty}{4(n+1)} h_{n+1} \\ &\leq\; \lim_{n\to\infty} \frac{C}{4(n+1)} h_{n+1} \;=\; 0.\end{aligned}$$

□

Wir können im Allgemeinen nicht davon ausgehen, dass wir bei der Interpolation einer Funktion f auf einem festen Intervall I immer mehr Stützstellen hinzunehmen können und damit die Funktion f in I immer besser approximieren. Es lässt sich nämlich zeigen, dass zu gegebenen Datenpunkten $(x_i, f_i) \in \mathbb{R}^2, i = 0, \ldots, n$ auf einem Intervall I immer eine stetige Funktion $f \in C^0(I)$ existiert, so dass das Interpolationspolynom $P \in \Pi_n$ nicht gleichmäßig, d.h. bezüglich der Maximumsnorm, gegen f konvergiert. Dies können wir mit Hilfe des folgenden berühmten Beispiels zeigen.

BEISPIEL 5.23: Runges Gegenbeispiel.

Wir nehmen an, dass wir die Funktion

$$f(x) \;=\; \frac{1}{1+x^2}, \qquad -5 \leq x \leq 5$$

mit Hilfe der Langrangeschen Interpolationsformel auf einem äquidistanten Gitter approximieren wollen. Es kann gezeigt werden, dass es Punkte x innerhalb des Interpolationsintervalls gibt, so dass für die Folge der Interpolationspolynome $\{P_n\}_{n\in\mathbb{N}}$ gilt

$$\lim_{n\to\infty} |f(x) - P_n(x)| \;\neq\; 0.$$

Speziell kann man zeigen, dass die Folge der Interpolationspolynome P_n divergiert für alle Punkte x mit $|x| > 3.63\ldots$. Dieses Phänomen ist auf die Wahl von äquidistanten Stützstellen x_i zurückzuführen. Betrachten wir die n-te Ableitung der Funktion f, welche gegeben ist durch

$$f^{(n)}(x) \;=\; (-1)^n \cdot n! \cdot \frac{\sin[(n+1)\,\mathrm{arccot(x)}]}{(1+x^2)^{\frac{n+1}{2}}},$$

so können wir approximativ sagen, dass sich $f^{(n)}$ asymptotisch verhält wie $n!$. Mit Hilfe der Fehlerabschätzung in Korollar 5.4.1 können wir damit das asympotische Verhalten

des Approximationsfehlers in Runges Gegenbeispiel angeben als:

$$||f - P_n||_\infty \;\leq\; \frac{||f^{(n+1)}||_\infty}{4(n+1)} h_{n+1} \;\simeq\; \frac{2\pi \cdot 10}{4} \cdot \left(\frac{10}{e}\right)^n \cdot \frac{1}{\sqrt{n}}.$$

Mit der Regel von L'Hospital lässt sich zeigen, dass die obere Schranke an den Fehler für $n \to \infty$ divergiert und man somit mit wachsender Anzahl von Stützstellen potentiell einen immer größeren Approximationsfehler macht. Dies lässt sich auch numerisch belegen.

5.4.2 Stabilität unter Störungen

Wir wollen uns abschließend mit der Frage beschäftigen wie stabil die Bestimmung des eindeutigen Interpolationspolynoms P mit Bezug auf kleine Änderungen der Stützwerte $f_i = f(x_i)$ ist. Solche kleinen Abweichungen können durch verschiedene Störungen wie Messfehler oder Rundungsfehler auftreten.

Betrachten wir hierzu eine Menge von Funktionswerten $\tilde{f}(x_i), i = 0, \ldots, n$, die eine Störung der Daten $f(x_i)$ in den Stützstellen $x_i, i = 0, \ldots, n$ im Intervall I sei. Bezeichnen wir nun mit $P \in \Pi_n$ das Interpolationspolynom durch die *ungestörten Stützwerte* $f(x_i), i = 0, \ldots, n$ und mit $\tilde{P} \in \Pi_n$ das Interpolationspolynom durch die *gestörten Stützwerte* $\tilde{f}(x_i), i = 0, \ldots, n$. Dann betrachten wir die maximale Abweichung zwischen den beiden Polynomen in der Lagrangeschen Interpolationsdarstellung aus Abschnitt 5.1 auf dem Intervall I durch

$$\begin{aligned} ||P - \tilde{P}||_\infty &= \max_{x \in I} \left| \sum_{i=0}^{n} (f(x_i) - \tilde{f}(x_i)) L_i(x) \right| \\ &\leq \Lambda_n \max_{i=0,\ldots,n} |f(x_i) - \tilde{f}(x_i)|, \end{aligned}$$

wobei $L_i(x)$ das Lagrangesche Basispolynom in (5.53) ist und $\Lambda_n := || \sum_{i=0}^{n} |L_i| \, ||_\infty$ die **Lebesguekonstante des Interpolationsproblems**, für die man zeigen kann, dass gilt

$$\Lambda_n \;>\; \frac{2}{\pi} \log(n+1) - C.$$

Hierbei ist $C > 0$ eine Konstante, die nur von den Datenpunkten $(x_i, f_i) \in \mathbb{R}^2, i = 0, \ldots, n$ abhängt. Das heißt es gilt, dass $\Lambda_n \to \infty$ für $n \to \infty$. Wir können also folgern, dass kleine Abweichungen in den Stützwerten $f(x_i)$ nur dann zu kleinen Änderungen im Interpolationspolynom führen, wenn die Lebesguekonstante klein ist. Sie spielt hier die Rolle der *Konditionszahl des Interpolationsproblems*.

Als abschließendes Fazit lässt sich sagen, dass das Interpolationsproblem mit wachsender Anzahl an Datenpunkten $(x_i, f_i) \in \mathbb{R}^2$ zunehmend instabiler wird und zugleich in der Regel keine gute Approximation an eine gegebene Funktion f darstellt. Mögliche Optionen zu Vermeidung dieser Probleme sind die Verwendung von **Tschebyscheff-Polynomen**, welche die ∞-Norm des Approximationsfehlers minimieren oder ein lokaler Ansatz basierend auf **Splines**. Mit Letzeren wollen wir uns im Folgenden näher beschäftigen.

5.5 Spline-Interpolation

Wie wir im vorigen Abschnitt gesehen haben, führt die Verwendung von global definierten Interpolationspolynomen $P \in \Pi_n$ bezüglich gegebener Datenpunkte $(x_i, f(x_i)) \in \mathbb{R}^2, i = 0, \ldots, n$ einer Funktion f auf einem Intervall I für $n \to \infty$ zu Lösungen, die immer stärker oszillieren und damit die zu Grunde liegende Funktion f immer schlechter approximieren. Aus diesem Grund geht man dazu über das Intervall I in Teilintervalle aufzuteilen bezüglich dieser Unterteilung die Datenpunkte im Intervall I mit stückweise stetigen Polynomen von relativ niedrigem Grad zu interpolieren. Es handelt sich also um einen **lokalen Ansatz** für das Interpolationsproblem.

Eine typische Wahl für diese Art von Interpolation sind die sogenannten Splines oder Splinefunktionen. Das englische Wort „Spline“leitet sich von dem deutschen Wort „Straklatte“ab, welche in der Vergangenheit im Schiffsbau verwendet wurden. Bei diesem handelt es sich um ein sehr biegsames Lineal aus Holz, welches vom Schiffsbauer an vorgegebenen Punkten fixiert wurde und so eine glatte Verbindung zwischen diesen Punkten schuf. Die Straklatte nimmt rein pyhsikalisch bedingt hierbei eine Form mit minimaler Biegeenergie und Krümmung an.

In der Mathematik verstehen wir unter einer Splinefunktion glatte Funktion, die auf Teilintervallen mit Polynomen übereinstimmt, wie folgende Definition beschreibt.

DEFINITION 5.24: Splinefunktion.

Sei $\Delta := \{a = x_0 < x_1 < \ldots < x_n = b\}$ eine Unterteilung des Intervalls $I = [a, b] \subset \mathbb{R}$. Unter einer zu Δ zugehörigen **Splinefunktion** $S_\Delta \colon [a, b] \to \mathbb{R}$ vom Grad $k \in \mathbb{N}$ versteht man eine reellwertige Funktion mit den folgenden Eigenschaften:

(i) $S_\Delta \in C^{k-1}([a, b])$ ist auf dem Intervall $I = [a, b]$ $(k-1)$-mal stetig differenzierbar.

(ii) auf jedem Teilintervall $[x_i, x_{i+1}], i = 0, \ldots, n-1$ stimmt S_Δ mit einem Polynom vom Grad k überein.

5.5.1 Lineare Splinefunktionen

Die einfachste Art von Splines sind diejenigen, die nur stetig auf dem Intervall $I = [a, b]$ sind, d.h. man versucht die Funktion f lokal durch stückweise lineare Funktionen zu approximieren. Diese Art von Streckenzug lässt sich bereits mit einfachen Mitteln bestimmen, da man lokal nur die eindeutige Verbindungsgerade zwischen zwei Punkten bestimmen muss.

Seien hierfür zwei Datenpunkte (x_i, f_i) und (x_{i+1}, f_{i+1}) auf einem Teilintervall $[x_i, x_{i+1}] \subset [a, b]$ gegeben. Zur Berechnung des **linearen Splines** S auf diesem Teilintervall benutzen wir die Geradengleichung in $\mathbb{R}$:

$$S_{\Delta,i}(x) := S_\Delta|_{[x_i, x_{i+1}]}(x) = m_i \cdot x + b_i = \frac{f_{i+1} - f_i}{x_{i+1} - x_i} \cdot x + f_i - \frac{f_{i+1} - f_i}{x_{i+1} - x_i} \cdot x_i$$

Zur Bestimmung der beiden unbekannten m_i, b_i auf dem Teilintervall $[x_i, x_{i+1}]$ haben wir lediglich die beiden zu erfüllenden Gleichungen für die Polynominterpolation ausgenutzt, d.h.

$$S_\Delta(x_i) = f_i, \quad S_\Delta(x_{i+1}) = f_{i+1}.$$

5.5.2 Kubische Splinefunktionen

Die Interpolation mit linearen Splines liefert eine stetige Approximation der Funktion f auf dem Intervall $[a, b]$, die in der Regel zu ungenau für reale Anwendungen ist. Um bessere Ergeb-

nisse zu erhalten verwendet man typischerweise **kubische Splines** $S_\Delta \in C^2([a,b])$ vom Grad 3. Hierbei lassen sich zusätzliche Bedingungen an die Glattheit des Splines stellen, so dass die Funktion stetig differenzierbar an den Übergängen, d.h. den Stützstellen wird. In diesem Fall verwendet man kubische Polynome auf jedem Teilintervall $[x_i, x_{i+1}]$ zur Interpolation der Datenpunkte $(x_i, f_i), i = 0, \dots, n$. Man verwendet häufig eine Darstellung der Polynome mit Hilfe der Newtonschen Interpolationsformel aus Kapitel 5.3 von der Form:

$$S_{\Delta,i}(x) \;=\; a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \quad i = 0, \dots, n-1.$$

Wir erhalten also ein Gleichungssystem mit $4n$ unbekannten Variablen $(a_i, b_i, c_i, d_i), i = 0, \dots, n$.

Für jedes der n Teilintervalle lassen sich nun folgende zwei **Bedingungen der Polynominterpolation** angeben:

$$S_{\Delta,i}(x_i) \;=\; f_i, \quad S_{\Delta,i}(x_{i+1}) \;=\; f_{i+1}, \quad i = 0, \dots, n-1.$$

Das ergibt also bereits $2n$ Bedingungen für das lineare Gleichungssystem.

Nun lassen sich weitere $2(n-1)$ Glattheitsbedingungen an den Spline in den inneren Stützstellen $x_i, i = 1, \dots, n-1$ stellen indem man fordert, dass der Spline an den Übergängen **zweimal stetig differenzierbar** sein soll, d.h. es soll gelten:

$$S'_{\Delta,i}(x_{i+1}) = S'_{\Delta,i+1}(x_{i+1}), \quad S''_{\Delta,i}(x_{i+1}) = S''_{\Delta,i+1}(x_{i+1}), \quad i = 0, \dots, n-2.$$

Die übrig bleibenden 2 Freiheitsgrade werden in der Regel mit Hilfe von **Randbedingungen** festgelegt. Hierbei gibt es in der Praxis verschiedene Ansätze:

(i) *natürliche Randbedingungen:* $S''_{\Delta,0}(x_0) = S''_{\Delta,n-1}(x_n) = 0$,

(ii) *Neumann-Randbedingungen:* $S'_{\Delta,0}(x_0) = f'(x_0),\ S'_{\Delta,n-1}(x_n) = f'(x_n)$,

(iii) *periodische Randbedingungen:* $S'_{\Delta,0}(x_0) = S'_{\Delta,n-1}(x_n),\ S''_{\Delta,0}(x_0) = S''_{\Delta,n-1}(x_n)$.

Wie man sieht benötigt man die erste und zweite Ableitung des Splines auf den jeweiligen Teilintervallen $[x_i, x_{i+1}]$, welche sich wie folgt berechnen lassen:

$$\begin{aligned} S'_{\Delta,i}(x) &= b_i + 2c_i(x - x_i) + 3d_i(x - x_i)^2, \\ S''_{\Delta,i}(x) &= 2c_i + 6d_i(x - x_i). \end{aligned}$$

BEMERKUNG 5.25 (Fehlerabschätzungen und Stabilität von Splines). Für die Spline-Interpolation gelten lokal auf jedem Teilintervall analoge Fehlerabschätzungen wie bei der Polynominterpolation in Abschnitt 5.4, aber auf jedem Teilintervall. Durch die geringe Länge der Teilintervalle kann der Fehler mit linearen und kubischen Splines klein gehalten werden, und gleichzeitig eine hohe Stabilität erzielt werden.
Wir werden im folgenden Abschnitt auch eine globale Fehlerabschätzung für lineare Splines in einer speziellen Norm herleiten. △

5.5.3 Variationsprinzip für Splinefunktionen

Wir können eine alternative Theorie der Splines basierend auf Variationsprinzipien entwickeln. Die Idee ist es hierbei Minimierungsprobleme für Funktionen zu lösen, die physikalisch motiviert sind. Im folgenden Theorem suchen wir eine interpolierende Funktion, die global die geringste Steigung, d.h., die minimale Variations besitzt.

THEOREM 5.26: Variationsmethode bzgl. der ersten Ableitung.

Seien $(x_i, f_i) \in \mathbb{R}^2, i = 0, \ldots, n$ eine Menge von Datenpunkten deren Stützstellen monoton steigend sortiert sind und im Intervall $I := [a, b] \subset \mathbb{R}$ liegen. Sei außerdem $\Delta := \{a = x_0 < x_1 < \ldots < x_n = b\}$ die durch die Stützstellen induzierte Unterteilung des Intervalls I.
Dann ist der lineare Spline $S := S_\Delta$ der eindeutige Minimierer des Funktionals

$$E(f) := \int_a^b |f'(x)|^2 \, \mathrm{d}x, \tag{5.64}$$

über der Menge der stetigen und bzgl. Δ stückweise stetig differenzierbaren Funktionen, die gleichzeitig die Interpolationsbedingung $f(x_i) = f_i$ erfüllen.

Beweis. Sei f eine stetige und bzgl. Δ stückweise stetig differenzierbaren Funktion, die die Interpolationsbedingungen erfüllt. Dann gilt

$$\begin{aligned} E(f) - E(S) &= \int_a^b |f'(x)|^2 \, \mathrm{d}x - \int_a^b |S'(x)|^2 \, \mathrm{d}x \\ &= \int_a^b (f'(x) + S'(x))(f'(x) - S'(x)) \, \mathrm{d}x \\ &= \sum_{i=1}^n \int_{x_{i-1}}^{x_i} (f'(x) + S'(x))(f'(x) - S'(x)) \, \mathrm{d}x. \end{aligned}$$

Nun könnnen wir partiell integrieren und dabei ausnutzen, dass an den Stützstellen die Funktionen übereinstimmen, d.h., dass $f(x_i) = S(x_i) = f_i$ gilt und dass die zweite Ableitung des linearen Splines auf jedem Teilintervall (x_{i-1}, x_i) verschwindet mit $S''(x) = 0$. Damit erhalten wir

$$\begin{aligned} E(f) - E(S) &= \sum_{i=1}^n \int_{x_{i-1}}^{x_i} (f'(x) + S'(x))(f'(x) - S'(x)) \, \mathrm{d}x \\ &= \sum_{i=1}^n \underbrace{[(f'(x) + S'(x))(f(x) - S(x))]_{x_{i-1}}^{x_i}}_{=0} - \int_{x_{i-1}}^{x_i} (f''(x) + \underbrace{S''(x)}_{=0})(f(x) - S(x)) \, \mathrm{d}x \\ &= -\sum_{i=1}^n \int_{x_{i-1}}^{x_i} (f''(x) - S''(x))(f(x) - S(x)) \, \mathrm{d}x. \end{aligned}$$

Integrieren wir nun wieder in die andere Richtung partiell, so folgt

$$\begin{aligned} E(f) - E(S) &= -\sum_{i=1}^n \int_{x_{i-1}}^{x_i} (f''(x) - S''(x))(f(x) - S(x)) \, \mathrm{d}x \\ &= -\sum_{i=1}^n \underbrace{[(f'(x) - S'(x))(f(x) - S(x))]_{x_{i-1}}^{x_i}}_{=0} - \int_{x_{i-1}}^{x_i} (f'(x) - S'(x))(f'(x) - S'(x)) \, \mathrm{d}x \\ &= \sum_{i=1}^n \int_{x_{i-1}}^{x_i} (f'(x) - S'(x))(f'(x) - S'(x)) \, \mathrm{d}x = \int_a^b |f'(x) - S'(x)|^2 \, \mathrm{d}x \geq 0. \end{aligned}$$

Da die Energie des Variationsproblems $E(S)$ also immer kleiner gleich der Energie $E(f)$ für eine beliebige Funktion f ist, ist der Spline S in Minimierer von E.

Diskutieren wir nun noch die Eindeutigkeit dieses Minimierers. Für die Energie einer beliebigen, zulässigen Funktion f kann $E(f) = E(S)$ nur dann gelten, wenn $f' = S'$ fast überall gilt, wie man an obigen Argument erkennt. Dies ist nur möglich, wenn sich f und S nur durch eine Konstante in jedem Teilintervall (x_{i-1}, x_i) unterscheiden. Da die Randwerte dieser Teilintervalle wegen der Interpolationsbedingung jedoch gleich sind, muss die Konstante schon Null sein. Also ist der lineare Spline S der eindeutige Minimierer der Energie E und es gilt $E(f) > E(S)$ für $f \neq S$. □

BEMERKUNG 5.27 (Funktionalanalytische Diskussion). Theorem 5.26 zeigt, dass wir die Splineinterpolation als Verallgemeinerung der *Minimum-Norm-Lösung* aus Abschnitt 3.3 interpretieren können. Ausgangspunkt ist das unterbestimmte Problem eine stetige und stückweise stetig differenzierbare Funktion zu finden, die die Interpolationsbedingung erfüllt. Auf dieser unendlichdimensionalen Menge minimieren wir die Energie $E(f)$, welche über eine Halbnorm definiert ist. Es handelt sich um eine Halbnorm, da der Nullraum dieser Abbildung auch alle konstanten Funktionen umfasst und nicht nur die Nullfunktion. Andererseits sind die Konstanten durch die Interpolation bestimmt, also ist E auf einer geeigneten Menge auch eine Norm.
Tatsächlich können wir einen leicht größeren Raum betrachten, den Sobolev-Raum $H^1([a,b]) := W^{1,2}([a,b])$ der Funktionen mit quadratisch integrierbarer (schwacher) Ableitung, d.h.

$$H^1([a,b]) \;:=\; \{f \in L^2([a,b] \mid f' \in L^2([a,b]\}.$$

Wie wir in Theorem 5.26 gesehen haben sind lineare Splines die eindeutigen Minimierer auf diesem Funktionenraum.
Man kann zeigen, dass in einer Raumdimension $H^1([a,b])$ nur stetige Funktionen enthält. Die Interpolationsbedingung ist also wohldefiniert. Andererseits sind nicht alle Funktionen in $H^1([a,b])$ stückweise stetig. Die Minimierung eines Funktionals wie E über einem solchen Raum ist ein klassisches Problem aus dem mathematischen Feld der **Variationsrechnung und partiellen Differentialgleichungen**.
Resultate wie Satz 5.26 nennt man auch „*Representer Theorem*“, denn hier wird eine Darstellung des Minimums eines Variationsproblems in unendlicher Dimension als Kombination endlich vieler Funktionen (hier: Splines) erhalten. Diese Ergebnisse sind in allgemeinerer Form auch im Bereich Machine Learning wichtig. Dort wird normalerweise statt der Interpolation eine Minimierung des empirischen Risikos mit Regularisierung betrachtet, d.h.

$$J_\alpha(f) \;=\; \sum_{i=0}^{n} (f(x_i) - f_i)^2 + \alpha E(f),$$

mit einem geeigneten Parameter $\alpha > 0$. Die Minimierer von J_α besitzen auch hier die gleiche Gestalt wie Splinefunktionen. △

Aus dem Beweis von Satz 5.26 erhalten wir auch eine Fehlerabschätzung für die linearen Splines in Bezug auf die unbekannte den Daten zu Grunde liegende Funktion f, wie das folgende Korollar zeigt.

KOROLLAR 5.28: Fehlerabschätzung für lineare Splines.

Es gelten die Voraussetzungen aus Theorem 5.26. Außerdem sei $f\colon I \to \mathbb{R}$ eine zweimal stetig differenzierbare Funktion, die den Datenpunkten $(x_i, f_i) \in \mathbb{R}^2, i = 0, \dots, n$ zu Grunde liegt. Sei

$$M := \sup_{x\in[a,b]} |f''(x)| \qquad \text{und} \qquad h_{n+1} := \max_{i=1,\dots,n} |x_i - x_{i-1}|.$$

Dann gilt für den eindeutig bestimmten linearen Spline S die folgende Fehlerabschätzung

$$\sqrt{\int_a^b |f'(x) - S'(x)|^2 \,\mathrm{d}x} \;\le\; M(b-a)h.$$

Beweis. Da wir angenommen haben, dass die Funktion f zweimal stetig differenzierar ist erhalten wir wieder durch partielle Integration

$$\begin{aligned}\int_a^b |f'(x) - S'(x)|^2 \,\mathrm{d}x &= -\sum_{i=1}^n \int_{x_{i-1}}^{x_i} (f''(x) - \underbrace{S''(x)}_{=0})(f(x) - S(x)) \,\mathrm{d}x \\ &= -\sum_{i=1}^n \int_{x_{i-1}}^{x_i} f''(x)(f(x) - S(x)) \,\mathrm{d}x.\end{aligned}$$

Nun können wir den Hauptsatz der Differential- und Integralrechnung anwendung und es gilt für jeden Punkt $x \in (x_{i-1}, x_i)$ wegen der Gleichheit der Randwerte und der Cauchy-Schwarz-Ungleichung

$$\begin{aligned}|f(x) - S(x)| &= \Big|\int_{x_{i-1}}^x (f'(y) - S'(y)) \,\mathrm{d}y\Big| \;\le\; \sqrt{x - x_i} \cdot \sqrt{\int_{x_{i-1}}^x (f'(y) - S'(y))^2 \,\mathrm{d}y} \\ &\le \sqrt{h_{n+1}} \cdot \sqrt{\int_{x_{i-1}}^{x_i} (f'(y) - S'(y))^2 \,\mathrm{d}y},\end{aligned}$$

Mit der Definition der Konstante M können wir nun abschätzen, dass gilt

$$\begin{aligned}\int_a^b |f'(x) - S'(x)|^2 \,\mathrm{d}x &\le \sum_{i=1}^n \int_{x_{i-1}}^{x_i} |f''(x)| \, |f(x) - S(x)| \,\mathrm{d}x \\ &\le M\sqrt{h_{n+1}} \cdot \sum_{i=1}^n \int_{x_{i-1}}^{x_i} 1 \,\mathrm{d}x \cdot \sqrt{\int_{x_{i-1}}^{x_i} (f'(y) - S'(y))^2 \,\mathrm{d}y}.\end{aligned}$$

Mit der Cauchy-Schwarz Ungleichung für die Summe gilt weiter

$$\int_a^b |f'(x) - S'(x)|^2 \,\mathrm{d}x \;\le\; M\sqrt{h_{n+1}} \cdot \sqrt{\sum_{i=1}^n (x_i - x_{i-1})^2} \cdot \sqrt{\int_a^b (f'(y) - S'(y))^2 \,\mathrm{d}y}.$$

Die letzte Wurzel können wir kürzen und die vorletzte Wurzel weiter abschätzen, um letztendlich zu erhalten

$$\sqrt{\int_a^b |f'(x) - S'(x)|^2 \,\mathrm{d}x} \;\le\; M(b-a)h.$$

□

BEMERKUNG 5.29. Die Fehlerabschätzung aus Korollar 5.28 besagt, dass der Approximationsfehler durch die lineare Splinefunktion S an eine Funktion f in der Halbnorm des Sobolevraums $H^1([a,b])$ im Wesentlichen von der zweiten Ableitung der Funktion f und Auflösung h_{n+1} der Unterteilung Δ des Intervalls $I = [a,b]$ in Teilintervalle abhängt. Das bedeutet, dass der Approximationsfehler mit zusätzlichen Stützstellen $(x_n)_{n\in\mathbb{N}} \subset I$ in der Halbnorm gegen Null konvergiert. △

Zum Abschluss diskutieren wir noch ein analoges Resulat für kubische Splines. Hierbei versucht man in der Variationsmethode eine Funktion zu finden, die minimale Krümmung aufweist und trotzdem noch interpoliert.

THEOREM 5.30: Variationsmethode bzgl. der zweiten Ableitung.

Seien $(x_i, f_i) \in \mathbb{R}^2, i = 0, \dots, n$ eine Menge von Datenpunkten deren Stützstellen monoton steigend sortiert sind und im Intervall $I := [a,b] \subset \mathbb{R}$ liegen. Sei außerdem $\Delta := \{a = x_0 < x_1 < \dots < x_n = b\}$ die durch die Stützstellen induzierte Unterteilung des Intervalls I.
Dann ist der kubische Spline $S := S_\Delta$ mit Neumann-Randbedingungen $S'(a) = S'(b) = f'(a) = f'(b) = 0$ der eindeutige Minimierer des Funktionals

$$E(f) := \int_a^b |f''(x)|^2 \,\mathrm{d}x, \tag{5.65}$$

über der Menge der stetig differenzierbaren und bezüglich Δ stückweise zweimal stetig differenzierbaren Funktionen, die gleichzeitig die Interpolationsbedingung $f(x_i) = f_i$ erfüllen.

Beweis. Es sei f eine stetig differenzierbare und bezüglich Δ stückweise zweimal stetig differenzierbaren Funktion. Mit Hilfe von partieller Integration gilt

$$\begin{aligned}\int_a^b S''(x)f''(x)\,\mathrm{d}x &= \sum_{i=1}^n \int_{x_{i-1}}^{x_i} S''(x)f''(x)\,\mathrm{d}x = -\sum_{i=1}^n \int_{x_{i-1}}^{x_i} S^{(3)}(x)f'(x)\,\mathrm{d}x + \underbrace{\left[S''(x)f'(x)\right]_a^b}_{=0} \\ &= \sum_{i=1}^n \int_{x_{i-1}}^{x_i} \underbrace{S^{(4)}(x)}_{=0} f(x)\,\mathrm{d}x - \left[S^{(3)}(x)f(x)\right]_a^b.\end{aligned}$$

Den letzten Term können wir mit Hilfe des Hauptsatzes der Differential- und Integralrechnung, sowie der Produktregel für Differentiation umformen zu

$$\begin{aligned}\left[S^{(3)}(x)f(x)\right]_a^b = \left[S^{(3)}(x)S(x)\right]_a^b &= \sum_{i=1}^n \int_{x_{i-1}}^{x_i} \frac{\mathrm{d}}{\mathrm{d}x}(S^{(3)}(x)S(x))\,\mathrm{d}x \\ &= \sum_{i=1}^n \int_{x_{i-1}}^{x_i} \underbrace{S^{(4)}(x)}_{=0} S(x) + S^{(3)}(x)S'(x)\,\mathrm{d}x \\ &= \sum_{i=1}^n \int_{x_{i-1}}^{x_i} S^{(3)}(x)S'(x)\,\mathrm{d}x\end{aligned}$$

Nun können wir wieder partielle Integration anwenden und erhalten

$$\begin{aligned}\sum_{i=1}^{n}\int_{x_{i-1}}^{x_i} S^{(3)}(x)S'(x)\,\mathrm{d}x &= -\sum_{i=1}^{n}\int_{x_{i-1}}^{x_i} S''(x)S''(x)\,\mathrm{d}x + \underbrace{\left[S''(x)S'(x)\right]_a^b}_{=0}\\ &= \int_a^b (S''(x))^2\mathrm{d}x.\end{aligned}$$

Insgesamt bekommen wir hierdurch die Identität

$$\int_a^b S''(x)f''(x)\,\mathrm{d}x = \int_a^b (S''(x))^2\mathrm{d}x.$$

Somit haben wir für den Unterschied der Energie für die Funktion f und den kubischen Spline S

$$\begin{aligned}E(f)-E(S) &= \int_a^b (f''(x))^2\,\mathrm{d}x - \int_a^b (S''(x))^2\,\mathrm{d}x\\ &= \int_a^b (f''(x))^2\,\mathrm{d}x - 2\int_a^b (S''(x))^2\,\mathrm{d}x + \int_a^b (S''(x))^2\,\mathrm{d}x\\ &= \int_a^b (f''(x))^2\,\mathrm{d}x - 2\int_a^b S''(x)f''(x)\,\mathrm{d}x + \int_a^b (S''(x))^2\,\mathrm{d}x\\ &= \int_a^b (f''(x)-S''(x))^2\,\mathrm{d}x \geq 0,\end{aligned}$$

und somit gilt offensichtlich

$$E(S) \leq E(f).$$

Bezüglich der Eindeutigkeit des Minimierers stellen wir fest, dass nur $E(f) = E(S)$ gelten kann, wenn $f'' \equiv S''$ fast überall gilt. Letzeres ist jedoch wegen der Randwerte und der Interpolationsbedingung schon gleichbedeutend mit $f \equiv S$. □

5.6 Trigonometrische Interpolation

Wir betrachten in diesem Abschnitt noch einen Spezialfall der Polynominterpolation, nämlich die Interpolation auf äquidistanten Stützstellen auf dem komplexen Einheitskreis. Dieses sehr spezielle Interpolationsproblem führt uns zu einem der wichtigsten Hilfsmittel der angewandten Mathematik - der *diskreten Fouriertransformation*.

Da wir im Gegensatz zu den vorigen Abschnitten nun mit Polynomen über dem Körper $\mathbb{C}$ arbeiten wollen, bezeichnen wir ab nun mit Π_n die Menge der Polynome mit komplexen Koeffizienten vom Grad $n \in \mathbb{N}$. Ein Polynom $P \in \Pi_n$ mit $P\colon \mathbb{C} \to \mathbb{C}$ ist nun von der Gestalt

$$P(x) = \sum_{k=0}^{n} a_k x^k, \qquad a_k \in \mathbb{C}.$$

Es sei angemerkt, dass sich die Aussage zur Eindeutigkeit des Interpolationspolynoms aus Theorem 5.6 direkt vom Körper $\mathbb{R}$ auf $\mathbb{C}$ verallgemeinern lässt.

Wir wählen im Folgenden $n \in \mathbb{N}$ Stützstellen $x_j \in \mathbb{S}^1, j = 0, \dots, n-1$ auf dem komplexen Einheitskreis, die gegeben sind durch

$$x_j \;=\; e^{it_j} \;=\; \cos t_j + i \sin t_j, \qquad t_j \;=\; \frac{2\pi j}{n}, \qquad j = 0, \dots, n-1.$$

Darüberhinaus nehmen wir an, dass wir zu den n Stützstellen oben ebenfalls n Stützwerte $f_j, j = 0, \dots, n-1$ gegeben haben.

Da die Stützstellen $x_j, j = 0, \dots, n-1$ paarweise unterschiedlich sind existiert also ein eindeutiges Interpolationspolynom $P \in \Pi_{n-1}$ dessen unbekannte Koeffizienten a_k wir im Folgenden mit $\hat{f}_k$ bezeichnen wollen und das die Form hat

$$P(x) \;=\; \sum_{k=0}^{n-1} \hat{f}_k x^k.$$

Da es sich um interpolierendes Polynom handelt muss also folgende Bedingung gelten

$$P(x_j) \;=\; \sum_{k=0}^{n-1} \hat{f}_k x_j^k \;=\; \sum_{k=0}^{n-1} \hat{f}_k e^{ikt_j} \;=\; \sum_{k=0}^{n-1} \hat{f}_k e^{2\pi ikj/n} \;\stackrel{!}{=}\; f_j, \qquad j = 0, \dots, n-1.$$

In diesem Fall können wir eine sehr einfache Lösungsformel für die unbekannten Koeffizienten $\hat{f}_k$ angeben, wie das folgende Theorem zeigt.

THEOREM 5.31: Diskrete Fouriertransformation.

Es seien n Datenpunkte $(x_j, f_j) \in \mathbb{C}^2$ gegeben, wobei die Stützstellen $x_j = e^{2\pi ij/n}$ äquidistant auf dem komplexen Einheitskreis liegen. Sei $P(x) \coloneqq \sum_{k=0}^{n-1} \hat{f}_k x^k$ das eindeutig bestimmte Interpolationspolynom.
Dann lassen sich die unbekannten Koeffizienten $\hat{f}_k, k = 0, \dots, n-1$ von P wie folgt ausrechnen

$$\hat{f}_k \;=\; \frac{1}{n} \sum_{j=0}^{n-1} f_j e^{-2\pi ikj/n}. \tag{5.66}$$

Diese Darstellung der Koeffizienten nennt man die **diskrete Fourier-Transformation** der Länge n zu den Daten $f_j, j = 0, \dots, n-1$.

Beweis. Wir zeigen die Identität des Lemmas indem wir die diskrete Fourier-Transformation der Daten in das Interpolationspolynom einsetzen. Sei also $x_j \in \mathbb{S}^1$ eine beliebige Stützstelle auf dem komplexen Einheitskreis. Dann gilt für das eindeutig bestimmte Interpolationspolynom P

$$\begin{aligned} P(x_j) \;&=\; \sum_{k=0}^{n-1} \hat{f}_k e^{2\pi ikj/n} \;=\; \sum_{k=0}^{n-1} \frac{1}{n} \sum_{\ell=0}^{n-1} f_\ell e^{-2\pi ik\ell/n} e^{2\pi ikj/n} \\ &=\; \frac{1}{n} \sum_{k=0}^{n-1} \sum_{\ell=0}^{n-1} f_\ell e^{2\pi i(j-\ell)k/n} \;=\; \frac{1}{n} \sum_{\ell=0}^{n-1} f_\ell \sum_{k=0}^{n-1} \left(e^{2\pi i(j-\ell)/n} \right)^k \end{aligned}$$

Die innere Summe kann als Partialsumme einer geometrischen Reihe $s_{n-1}(j,\ell) \coloneqq \sum_{k=0}^{n-1} q^k(j,\ell)$ für $q(j,\ell) \coloneqq e^{2\pi i(j-\ell)/n}$ interpretiert und berechnet werden, so dass wir erhalten

$$s_{n-1}(j,\ell) \;=\; \frac{q^n(j,\ell) - 1}{q(j,\ell) - 1} \;=\; \frac{e^{2\pi i(j-\ell)} - 1}{e^{2\pi i(j-\ell)/n} - 1} \;=\; \begin{cases} 0, & \text{falls } j \neq \ell, \\ n, & \text{falls } j = \ell. \end{cases}$$

Damit berechnen wir also insgesamt

$$P(x_j) \;=\; \frac{1}{n}\sum_{\ell=0}^{n-1} f_\ell s_{n-1}(j,\ell) \;=\; f_j.$$

Da dies für jede beliebige Stützstelle $x_j, j = 0,\dots,n-1$ gilt, haben wir gezeigt, dass die Koeffizienten $\hat{f}_k$ das eindeutige Interpolationspolynom ergeben. □

KOROLLAR 5.32: Orthogonalitätseigenschaft.

Im Beweis von Theorem 5.31 können wir sehr schön eine Orthogonalitätseigenschaft der trigonometrischen Funktionen ablesen, denn es gilt

$$\frac{1}{n}\sum_{j=0}^{n-1} e^{2\pi jk/n} \;=\; \begin{cases} 1, & \text{falls } k \in n\mathbb{Z}, \\ 0, & \text{sonst .} \end{cases}$$

Die unbekannten Koeffizienten $\hat{f}_k$ in der diskreten Fouriertransformation werden auch **Fourierkoeffizienten** zu den Daten $f_j, j = 0,\dots,n-1$ genannt. Aus dem Beweis von Theorem 5.31 wird klar, dass man die Berechnungen auch umkehren kann, d.h., dass man aus einer Menge von Fourierkoeffizienten $\hat{f}_k, k = 0,\dots,n-1$ die ursprünglichen Stützwerde $f_j, j = 0,\dots,n-1$ ausrechnen kann. Dies lässt sich nämlich wie folgt realisieren

$$f_j \;=\; \sum_{k=0}^{n-1} \hat{f}_k e^{2\pi ijk/n}. \tag{5.67}$$

Diese Identität nennt man daher auch die **inverse diskrete Fouriertransformation** der Länge n zu den Fourierkoeffizienten $\hat{f}_k, k = 0,\dots,n-1$. Man beachte, dass je nach wissenschaftlicher Disziplin die Definition der diskreten Fouriertransformation und ihrer Inversen leicht abweichen kann, wie z.B. die Vertauschung der Vorzeichens im Exponenten der Stützstellen oder die Normalisierung mit $1/n$.

BEMERKUNG 5.33 (Rechenaufwand der diskreten Fourier-Transformation). Der Rechenaufwand zur Berechnung der diskreten Fourier-Transformation in (5.66) und ihrer Umkehrabbildung in (5.67) benötigt bei Vorberechnung der n Koeffizienten $t_j = \frac{2\pi j}{n}$ genau n^2 FLOPS und somit liegt der Rechenaufwand in $\mathcal{O}(n^2)$. △

5.6.1 Schnelle Fouriertransformation

Die Berechnung der Fourierkoeffizienten für gegebene Daten erlaubt es diese in ein Spektrum aus **Frequenzen** zu zerlegen und darzustellen. Anders ausgedrückt kann die Fouriertransformation als Zerlegung eines Signals in seine verschiedenen Schwingungsanteile interpretiert werden und hat deshalb vielfache Anwendungen. Daher ist es naheliegend, dass die Fouriertransformation besonders in der Signalverarbeitung eine zentrale Rolle spielt. Ein Großteil der modernen Kompressionsalgorithmen, wie zum Beispiel *MP3* oder *JPEG* verwenden die diskrete Fouriertransformation oder eine ihrer Varianten (z.B. die diskrete Kosinustransformation), um Daten effizient in ihrer Größe zu reduzieren ohne große Qualitätseinbußen hinnehmen zu müssen.

Durch die vielseitige Anwendbarkeit der diskreten Fouriertransformation ist es besonders wichtig, diese so effizient wie möglich zu berechnen. Eine formelle Beschreibung eines effizienten Algorithmus, der sogenannten *schnellen Fouriertransformation* (im Englischen: Fast Fourier Transform (FFT)) basiert auf der Pionierarbeit von Cooley-Tukey aus dem Jahr 1965 und basiert auf Ideen von Gauss. Die zentrale Idee des Algorithmus ist es die diskrete Fouriertransformation der Größe $n \in \mathbb{N}$ in zwei diskrete Fouriertransformationen der Größe $n/2$ aufzuteilen und diese Zerlegung rekursiv fortzusetzen.

Sei im Folgenden $n = 2m$ eine gerade natürliche Zahl. Dann können wir die diskrete Fouriertransformation der Länge n

$$\hat{f}_k \;=\; \frac{1}{n}\sum_{j=0}^{n-1} f_j q^{jk}, \qquad \text{mit} \qquad q \;\coloneqq\; e^{-2\pi i/n},$$

zur Berechnung eines Fourierkoeffizienten $f_k \in \mathbb{C}$ in **gerade** und **ungerade Indizes** wie folgt aufteilen:

$$\begin{aligned} n\hat{f}_k \;&=\; \sum_{j=0}^{m-1} f_{2j} q^{(2j)k} + \sum_{j=0}^{m-1} f_{2j+1} q^{(2j+1)k} \\ &=\; \sum_{j=0}^{m-1} f_{2j} (q^2)^{jk} + q^k \sum_{j=0}^{m-1} f_{2j+1} (q^2)^{jk} \\ &=:\; \hat{g}_k + q^k \hat{u}_k. \end{aligned}$$

Man sieht also, das wir die ursprüngliche diskrete Fouriertransformation $\hat{f}_k, k = 0\,ldots, n-1$ der Länge n durch zwei diskrete Fouriertransformationen $\hat{g}_k, k = 0, \ldots, m-1$ und $\hat{u}_k, k = 0, \ldots, m-1$ der Länge $m = n/2$ berechnen können. Dies sieht man ein, da gilt

$$q^2 \;=\; e^{(-2\pi i/n)^2} \;=\; e^{-4\pi i/n} \;=\; e^{-2\pi i/(n/2)} \;=\; e^{-2\pi i/m}.$$

Wie man nachrechnen kann gilt außerdem

$$\begin{aligned} \hat{g}_{k+m} \;&=\; \sum_{j=0}^{m-1} f_{2j} (q^2)^{j(k+m)} \;=\; \sum_{j=0}^{m-1} f_{2j} e^{-2\pi i j(k+m)/m} \\ &=\; \sum_{j=0}^{m-1} f_{2j} e^{-2\pi i jk/m} \cdot \underbrace{e^{-2\pi i j}}_{=1} \;=\; \hat{g}_k \end{aligned}$$

und analog $\hat{u}_{k+m} = \hat{u}_k$. Wegen $m = n/2$ gilt außerdem

$$q^{k+m} \;=\; e^{(-2\pi i/n)^{(k+m)}} \;=\; e^{-2\pi i k/n} \cdot \underbrace{e^{-2\pi i (n/2)/n}}_{=-1} \;=\; -q^k.$$

Vernachlässigen wir den Normalisierungsfaktor n vor den Fourierkoeffizienten, so lassen sich alle ursprünglichen Fourierkoeffizienten $\hat{f}_l = 0, \ldots, n-1$ wie folgt über die Fouriertransformationen g_k und u_k für $k = 0, \ldots, m-1$ berechnen:

$$\begin{aligned} \hat{f}_k \;&=\; \hat{g}_k + q^k \hat{u}_k, \\ \hat{f}_{k+m} \;&=\; \hat{g}_{k+m} + q^{k+m} \hat{u}_{k+m} \;=\; \hat{g}_k - q^k \hat{u}_k. \end{aligned} \tag{5.68}$$

Mit Hilfe dieser typischen *Divide & Conquer Strategie* können wir den Rechenaufwand für die schnelle Fouriertransformation bestimmen.

LEMMA 5.34: Rechenaufwand der schnellen Fouriertransformation.

Der Rechenaufwand der schnellen Fouriertransformation der Länge $n = 2^p, p \in \mathbb{N}$ liegt in $\mathcal{O}(n \log n)$.

Beweis. Sei $M_n = M_{2^p}$ der gesamte Rechenaufwand zur Berechnung der diskreten Fouriertransformation der Länge $n = 2^p$. Wie bemerken zunächst, dass wir alle n Terme $q^k, k = 0, \ldots, n-1$ einmalig als fixe Einheitswurzeln vorberechnen können mit einem Rechenaufwand von $\mathcal{O}(n)$.

Dann gilt mit der Zerlegung aus der schnellen Fouriertransformation in (5.68)

$$M_n \;=\; 2M_{n/2} + n + n/2 \;=\; 2M_m + 3m,$$

da wir n zusätzliche Additionen und m Multiplikationen durchführen müssen um alle Fourierkoeffizienten zu berechnen. Da $n = 2^p$ als eine Zweierpotenz gewählt ist, können wir mit $m = 2^{p-1}$ rekursiv folgern

$$\begin{aligned} M_{2^p} \;&=\; 2M_{2^{p-1}} + 3 \cdot 2^{p-1} \;=\; 2(2M_{2^{p-2}} + 3 \cdot 2^{p-2}) + 3 \cdot 2^{p-1} \\ &=\; 4M_{2^{p-2}} + 2 \cdot 3 \cdot 2^{p-1} \;\overset{\cdots}{=}\; 2^p M_1 + 3 \cdot p \cdot 2^{p-1}. \end{aligned} \tag{5.69}$$

Da für die Fouriertransformation der Länge 1 gilt, dass man die Daten selbst erhält, d.h.

$$\hat{f}_j \;=\; \frac{1}{1} \sum_{j=0}^{0} f_j q^{j \cdot 0} \;=\; f_j,$$

ist der Rechenaufwand also $M_1 = 1$. Und somit erhalten wir aus (5.69), dass $M_{2^p} = \mathcal{O}(p \cdot 2^p)$ bzw. $M_n = \mathcal{O}(n \log n)$ gilt. □

Kapitel 6

Numerische Integration

In diesem Abschnitt widmen wir uns der numerischen Integration, d.h. der numerischen Approximation bestimmter Integrale der Form $\int_a^b f(x)\,\mathrm{d}x$.

Die numerische Integration ist insbesondere in solchen Fällen von Interesse in denen eine Stammfunktion sich nicht durch Elementarfunktionen ausdrücken lässt, eine numerische Auswertung der Stammfunktion zu komplex ist oder ein Integrand nur durch Messungen an endlich vielen Punkten bekannt ist. Sie hat daher vielfältige Anwendungen, insbesondere in höheren Dimensionen. Der Einfachheit halber beschränken wir uns in diesem Kapitel zumeist aber auf den eindimensionalen Fall.

Die grundlegende Idee der numerischen Integration ist es das Intervall $[a,b] \subset \mathbb{R}$ zu *diskretisieren*, d.h. in kleinere Teilintervalle aufzuteilen und das Integral durch die Lösung eines Ersatzproblems auf diesen Teilintervallen zu approximieren. Hierbei wird die Feinheit der Diskretisierung durch eine Schrittweite $h > 0$ bestimmt, die entscheidend für die Güte der Approximation des Integrals ist, wie wir später feststellen werden. Wir gehen zunächst davon aus, dass die Stützstellen für die numerische Integration *äquidistant* gewählt werden und die Anfangspunkte $a, b \in \mathbb{R}$ mit einschließen, d.h. wir diskretisieren das Intervall $[a,b] \subset \mathbb{R}$ wie folgt:

$$x_i \coloneqq a + i \cdot h, \quad i = 0, \ldots, n \qquad \text{für } h \coloneqq \frac{b-a}{n}, n \in \mathbb{N}_+. \tag{6.70}$$

Die wichtigste Definition zur numerischen Approximation des bestimmten Integrals einer Funktion f ist im Folgenden gegeben.

DEFINITION 6.1: Quadraturformel.

Sei $f\colon [a,b] \to \mathbb{R}$ eine integrierbare Funktion. Wir nennen eine gewichtete Summe der Form

$$I(f) \;=\; \sum_{i=0}^{n} w_i f(x_i) \;\approx\; \int_a^b f(x)\,\mathrm{d}x$$

eine **Quadraturformel** auf dem Intervall $[a,b] \subset \mathbb{R}$.
Die Punkte $x_0, \ldots, x_n \in [a,b]$ werden **Stützstellen** genannt und die Koeffizienten $w_0, \ldots, w_n \in \mathbb{R}$ sind **Gewichte** der Quadraturformel, die unabhängig von der zu integrierenden Funktion f sind.

Die wohl einfachste Quadraturformel ist im folgenden Beispiel illustriert.

ToDo

Abbildung 6.1: Illustration zur numerischen Approximation des bestimmten Integrals einer Funktion f mit Hilfe der Mittelpunktsregel.

BEISPIEL 6.2: Mittelpunktsregel.

Für $n = 0$ erhalten wir das triviale Beispiel einer Quadraturformel, die sogenannte **Mittelpunktsregel** , als

$$I_0(f) \;=\; w_0 f(x_0) \;\coloneqq\; (b-a) f\left(\frac{a+b}{2}\right).$$

Hierbei ist das einzige Gewicht $w_0 \coloneqq (b-a)$ die Intervalllänge und der einzige Funktionswert $f(x_0)$ wird am Mittelpunkt $x_0 \coloneqq \frac{a+b}{2}$ des Intervalls ausgewertet. Man beachte, dass es für $n = 0$ nicht möglich ist das Intervall $[a, b] \subset \mathbb{R}$ entsprechend (6.70) zu diskretisieren. Dennoch besitzt die Mittelpunktsregel die Form einer Quadraturformel in Definition 6.1.
Abb. 6.1 visualisiert die Approximation des Integrals einer Funktion mit Hilfe der Mittelpunktsregel.

BEMERKUNG 6.3 (Quadraturformeln und Treppenfunktionen). Wie man an Definition 6.1 und Beispiel 6.2 erkennt, werden in der Quadraturformel nur Funktionsauswertungen einer Funktion an $n+1$ Stützstellen verwendet. Anschaulich bedeutet dies, dass man das bestimmte Integral einer Funktion durch eine Treppenfunktion approximiert, deren Plateaus durch die Funktionswerte $f(x_i), i = 0, \ldots, n$ festgelegt sind und deren Breite der konstante Teilstücke durch die Gewichte $w_i, i = 0, \ldots, n$ definiert werden.
Dies entspricht der grundlegenden Idee bei der Herleitung des *Riemann-Integrals*. △

Bei der Konstruktion von Quadraturformeln sind einige strukturelle Eigenschaften von Quadraturformeln besonders interessant, zum Beispiel die *Erhaltung der Nichtnegativität*, d.h. $I(f) \geq 0$ für nichtnegative Funktionen f. Man sieht leicht ein, dass dies äquivalent zur Nichtnegativität der Gewichte $w_i \in \mathbb{R}$ ist. Eine andere oft geforderte Eigenschaft ist die *Erhaltung der Intervalllänge*, d.h. $I(\mathbf{1}) = b - a$ für die konstante Einsfunktion $\mathbf{1}(x) \equiv 1$ für $x \in [a, b]$.

Wir interessieren uns zunächst für eine mögliche Fehlerverstärkung bei der Berechnung von $I(\tilde{f})$, wenn $\tilde{f}$ eine gestörte Version von f ist. Tatsächlich sind die beiden oben genannten Eigenschaften schon hinreichend für die Stabilität einer Quadraturformel, wie das folgende Theorem zeigt.

THEOREM 6.4: Stabilität von Quadraturformeln.

Sei $I : C([a, b]) \to \mathbb{R}$ eine Quadraturformel mit nichtnegativen Gewichten $w_i \in \mathbb{R}, i = 0, \ldots, n$ und $I(\mathbf{1}) = b - a$.
Dann erhalten wir die Abschätzung

$$|I(f) - I(\tilde{f})| \;\leq\; (b-a) \cdot \epsilon,$$

für alle Funktionen $f, \tilde{f} \in C([a,b])$, für die gilt

$$|f(x) - \tilde{f}(x)| \;\leq\; \epsilon, \qquad \text{für alle } \; x \in [a,b].$$

Beweis. Wegen der Nichtnegativität der Gewichte $w_i, i = 0, \ldots, n$ folgt direkt

$$b - a \;=\; I(\mathbf{1}) \;=\; \sum_{i=0}^{n} w_i \;=\; \sum_{i=0}^{n} |w_i|.$$

Damit erhalten wir schon

$$\begin{aligned} |I(f) - I(\tilde{f})| \;&=\; |\sum_{i=0}^{n} w_i (f(x_i) - \tilde{f}(x_i))| \\ &\leq\; \sum_{i=0}^{n} |w_i| \cdot |f(x_i) - \tilde{f}(x_i)| \;\leq\; \sum_{i=0}^{n} |w_i| \cdot \epsilon \;=\; (b-a) \cdot \epsilon. \end{aligned}$$

□

6.1 Interpolatorische Quadraturformeln

Man erkennt aus der Definition einer Quadraturformel einen Zusammenhang zur Interpolation aus Kapitel 5, da wir auch hier mit Stützwerten arbeiten. Diese Beobachtung liefert auch den ersten Ansatz zur expliziten Konstruktion von Quadraturformeln, nämlich die sogenannte *interpolatorische Quadratur*.

Die Idee der interpolatorischen Quadratur ist es die Funktion f an den Stützstellen $x_i \in [a,b], i = 0, \ldots, n$ durch eine einfach zu integrierende Funktion $P \in C([a,b])$ zu interpolieren und dann das Integrals von f durch das Integral von P zu approximieren, d.h.

$$I(f) \;=\; \int_a^b P(x)\,\mathrm{d}x. \tag{6.71}$$

6.1.1 Integrationsformeln nach Newton-Cotes

Der einfachste Fall ist eine Interpolation durch Polynome $P \in \Pi_n$. In diesem Fall verwenden wir die Lagrange-Polynome aus Abschnitt 5.1 mit

$$P(x) \;:=\; \sum_{i=0}^{n} f(x_k) L_k(x), \qquad \text{und} \qquad L_k(x) \;:=\; \prod_{i=0, i\neq k}^{n} \frac{(x - x_i)}{(x_k - x_i)},$$

da diese die Stützwerte von f direkt verwenden und somit die Form einer Quadraturformel haben.

Setzen wir diese in die interpolatorische Quadratur (6.71) ein, erhalten wir die sogenannten **Integrationsformeln nach Newton-Cotes** durch

$$I_n(f) \;=\; \int_a^b P(x)\,\mathrm{d}x \;=\; \int_a^b \sum_{k=0}^{n} f(x_k) L_k(x)\,\mathrm{d}x \;=\; \sum_{k=0}^{n} f(x_k) \int_a^b L_k(x)\,\mathrm{d}x.$$

Wir erkennen, dass es sich bei dieser Integration über das Interpolationspolynom P um eine Quadraturformel nach Definition 6.1 handelt, deren Gewichte unabhängig von der Funktion f als Integral über die Langrange-Polynome gegeben sind mit

$$w_k \coloneqq \int_a^b L_k(x)\,\mathrm{d}x = \int_a^b \prod_{i=0,i\neq k}^{n} \frac{(x-x_i)}{(x_k-x_i)}\,\mathrm{d}x, \qquad k=0,\ldots,n. \tag{6.72}$$

Substituieren wir nun den Punkt $x \in [a,b]$ mit Hilfe einer Funktion $\varphi\colon [0,n] \to [a,b]$ mit $\varphi(t) \coloneqq h\cdot t + a = x$, so sehen wir, dass gilt

$$\varphi(0) = h\cdot 0 + a = a, \qquad \varphi(n) = h\cdot n + a = \frac{b-a}{n}\cdot n + a = b,$$
$$\varphi(i) = h\cdot i + a = x_i.$$

Wenden wir diese Subsitution auf das Integral für die Gewichte der Quadraturformel in (6.72) an, so erhalten wir:

$$\begin{aligned} w_k &= \int_a^b \prod_{i=0,i\neq k}^{n} \frac{(x-x_i)}{(x_k-x_i)}\,\mathrm{d}x = \int_{\varphi(0)}^{\varphi(n)} \prod_{i=0,i\neq k}^{n} \frac{(x-x_i)}{(x_k-x_i)}\,\mathrm{d}x \\ &= \int_0^n \prod_{i=0,i\neq k}^{n} \frac{h(t-i)}{h(k-i)}\cdot \underbrace{\varphi'(t)}_{=h}\,\mathrm{d}t = h\cdot \int_0^n \prod_{i=0,i\neq k}^{n} \frac{t-i}{k-i}\,\mathrm{d}t =: hW_k, \qquad k=0,\ldots,n. \end{aligned}$$

Damit lassen sich die **Integrationsformeln nach Newton-Cotes** schreiben als

$$I_n(f) = h\sum_{k=0}^{n} f(x_k)W_k, \qquad n\in\mathbb{N}_+,$$

wobei die Gewichte W_k nicht von der Funktion f abhängen und sich durch einfach zu berechnende Integrale bestimmen lassen.

Wir betrachten im folgenden einige Beispiele für interpolatorische Quadraturformeln.

BEISPIEL 6.5: Quadraturformeln.

Wir wollen in diesem Beispiel zwei bekannte Quadraturformeln für die Approximation des Integrals für unterschiedliche Anzahl der Stützpunkte $n\in\mathbb{N}_+$ aus den Integrationsformeln nach Newton-Cotes herleiten.

1. Für $n=1$ erhalten wir eine Quadraturformel, die auf zwei Funktionswerten beruht, die sogenannte **Trapezregel** mit

   $$I_1(f) = h(f(x_0)W_0 + f(x_1)W_1) = h\,(f(a)W_0 + f(b)W_1)$$

   Hierbei ist $h = \frac{b-a}{n} = b-a$ die Länge des Intervalls.

   Berechnen wir die Gewichte W_0 und W_1 durch Integration so erhalten wir

   $$W_0 \coloneqq \int_0^1 \frac{t-1}{0-1}\,\mathrm{d}t = \int_0^1 1-t\,\mathrm{d}t = \left[t-\frac{1}{2}t^2\right]_0^1 = -\frac{1}{2}+1 = \frac{1}{2}$$

   und

   $$W_1 \coloneqq \int_0^1 \frac{t-0}{1-0}\,\mathrm{d}t = \int_0^1 t\,\mathrm{d}t = \left[\frac{1}{2}t^2\right]_0^1 = \frac{1}{2}$$

Somit erhalten wir insgesamt für die Trapezregel die Quadraturformel

$$I_1(f) \;=\; \frac{h}{2}(f(a)+f(b)) \;=\; \frac{b-a}{2}(f(a)+f(b)).$$

?? visualisiert die Approximation des Integrals einer Funktion mit Hilfe der Trapezregel, deren Namen sich an der Form der Fläche der Quadraturformel ablesen lässt.

2. Für $n=2$ erhalten wir eine relativ einfache Quadraturformel, die sich jedoch als erstaunlich genau herausstellt. Wir erhalten die sogenannte **Simpsonregel** (auch bekannt als **Keplersche Fassregel** durch

$$\begin{aligned} I_2(f) \;&=\; h((f(x_0)W_0 + f(x_1)W_1 + f(x_2)W_2) \\ &=\; h\left(f(a)W_0 + f\left(\frac{a+b}{2}\right)W_1 + f(b)W_2\right) \end{aligned}$$

Hierbei ist die Schrittweite gegeben durch $h = \frac{b-a}{n} = \frac{b-a}{2}$.

Berechnen wir die Gewichte W_0, W_1 und W_2 durch Integration so erhalten wir

$$\begin{aligned} W_0 \;&\coloneqq\; \int_0^2 \frac{t-1}{0-1}\cdot\frac{t-2}{0-2}\,\mathrm{d}t \;=\; \int_0^2 (1-t)(1-\frac{t}{2})\,\mathrm{d}t \\ &=\; \int_0^2 \frac{1}{2}t^2 - \frac{3}{2}t + 1\,\mathrm{d}t \;=\; \left[\frac{1}{6}t^3 - \frac{3}{4}t^2 + t\right]_0^2 \;=\; \frac{8}{6} - 3 + 2 \;=\; \frac{1}{3}, \\ W_1 \;&\coloneqq\; \int_0^2 \frac{t-0}{1-0}\cdot\frac{t-2}{1-2}\,\mathrm{d}t \;=\; \int_0^2 t\cdot\frac{t-2}{-1}\,\mathrm{d}t \\ &=\; \int_0^2 -t^2 + 2t\,\mathrm{d}t \;=\; \left[-\frac{1}{3}t^3 + t^2\right]_0^2 \;=\; -\frac{8}{3} + 4 \;=\; \frac{4}{3}, \end{aligned}$$

und

$$\begin{aligned} W_2 \;&\coloneqq\; \int_0^2 \frac{t-0}{2-0}\cdot\frac{t-1}{2-1}\,\mathrm{d}t \;=\; \int_0^2 \frac{t}{2}\cdot(t-1)\,\mathrm{d}t \\ &=\; \int_0^2 \frac{1}{2}(t^2-t)\,\mathrm{d}t \;=\; \left[\frac{1}{6}t^3 - \frac{1}{4}t^2\right]_0^2 \;=\; \frac{8}{6} - 1 \;=\; \frac{1}{3}. \end{aligned}$$

Somit erhalten wir insgesamt für die Simpsonregel die Quadraturformel

$$I_2(f) \;=\; \frac{h}{3}\left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right) \;=\; \frac{b-a}{6}\left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right).$$

?? visualisiert die Approximation des Integrals einer Funktion mit Hilfe der Simpsonregel.

BEMERKUNG 6.6 (Exaktheit interpolatorischer Quadraturen).

(i) Es ist in der Tat möglich noch genauere interpolatorische Quadraturen mit Hilfe der Newton-Cotes Integrationsformeln herzuleiten. Tabelle 6.1 zeigt eine kompakte Darstel-

n	W_k / Faktor							Faktor	Name
1	1	1						$\frac{1}{2}$	Trapez-Regel
2	1	4	1					$\frac{1}{3}$	Simpson-Regel
3	1	3	3	1				$\frac{3}{8}$	Newton $\frac{3}{8}$-Regel
4	7	32	12	32	7			$\frac{2}{45}$	Milne-Regel
5	19	75	50	50	75	19		$\frac{5}{288}$	6-Punkt-Regel
6	41	216	27	272	27	216	41	$\frac{1}{140}$	Weddle-Regel

Tabelle 6.1: Kompakte Darstellung der ersten sechs interpolatorischen Quadraturen für die numerische Integration.

lung der ersten sechs Quadraturen basierend auf den Gewichten W_k. Für $n \geq 8$ treten jedoch negative Gewichte W_k in der Quadraturformel auf (siehe [Quad]), was sie numerisch ungünstig macht, da es ein hinreichendes Kriterium für die Stabilität aufhebt, wie man in Theorem 6.4 sieht.

Typischerweise verwendet man für die Integration nur interpolatorische Quadraturen bis $n \geq 3$, d.h. die Simpsonregel, da die erreichte Genauigkeit häufig ausreichend ist für die meisten Anwendungen, wie wir in den folgenden Abschnitten sehen werden.

(ii) Man sieht ein, dass interpolatorische Quadraturformeln mit Polynomen $P \in \Pi_n$ alle Polynome vom Grad kleiner gleich n exakt integriert, da diese offensichtlich exakt interpoliert werden. Allgemein nennt man eine Quadraturformel vom **Exaktheitsgrad** $r \in \mathbb{N}$, wenn alle Polynome vom Grad kleiner gleich r exakt integriert werden.

△

Das folgende Beispiel vergleicht die Genauigkeit der numerischen Integration für verschiedene Werte von n.

BEISPIEL 6.7: Genauigkeit numerischer Integration.

In diesem Beispiel wollen wir die numerische Integration mit Hilfe der Mittelspunktregel und den ersten interpolatorischen Quadraturen für die einfache Funktion $f(x) \coloneqq e^x$ für verschiedene Werte von n miteinander vergleichen. Zunächst bestimmen wir analytisch den Wert des bestimmten Integrals

$$\int_0^1 f(x)\,\mathrm{d}x \;=\; \int_0^1 e^x\,\mathrm{d}x \;=\; [e^x]_0^1 \;=\; e-1 \;\approx\; \mathbf{1.7183}.$$

Wenden wir zunächst die Mittelpunktsregel an, so erhalten wir

$$I_0 \;=\; (b-a)f\left(\frac{a+b}{2}\right) \;=\; (1-0)\cdot e^{\frac{1}{2}} \;=\; e^{\frac{1}{2}} \;\approx\; \mathbf{1.6487}.$$

Nun betrachten wir die interpolatorischen Quadraturen für $n = 1, 2, 3$:

$$\begin{aligned} I_1 &= \frac{b-a}{2}(f(a)+f(b)) = \frac{1}{2}(e^0+e^1) = \frac{1}{2}(1+e) && \approx \mathbf{1.8591}, \\ I_2 &= \frac{b-a}{6}\left(f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right) = \frac{1}{6}\left(1+4e^{\frac{1}{2}}+e\right) && \approx \mathbf{1.7189}, \\ I_3 &= \frac{b-a}{8}\left(f(a)+3f\left(\frac{a+b}{3}\right)+3f\left(\frac{2(a+b)}{3}\right)+f(b)\right) \\ &= \frac{1}{8}(1+3e^{\frac{1}{3}}+3e^{\frac{2}{3}}+e) && \approx \mathbf{1.7185}. \end{aligned}$$

6.1.2 Numerischer Integrationsfehler

Es ist klar, dass man bei der Annäherung des bestimmten Integrals durch eine Quadraturformel einen *Approximationsfehler* macht. Dieser wird maßgeblich durch die Wahl der Stützstellen und Gewichte der Quadraturformel in Definition 6.1 beeinflusst. Aus Sicht der Numerik ist es interessant den Fehler dieser Approximation zu berechnen, d.h., $|\int_a^b f(x)\,\mathrm{d}x - I(f)|$ abzuschätzen. Darüberhinaus wollen wir die Entwicklung des Fehlers der interpolatorischen Quadraturformeln

$$I_n(f) = h\sum_{k=0}^{n} f(x_k)W_k,$$

für $n \to \infty$ untersuchen.

Aus den Fehlerabschätzungen zur Polynominterpolation erhalten wir direkt eine Abschätzung für die Integrationsformeln nach Newton-Cotes Formel, wie das folgende Theorem zeigt.

THEOREM 6.8: Fehlerabschätzung für die interpolatorische Quadratur.

Wir können im Folgenden zwei unterschiedliche Fehlerabschätzungen basierend auf der Wahl von $n \in \mathbb{N}_+$ formulieren. Entscheidend ist, dass man bei geradem n durch den Übergang zu $n+1$ keine Potenz der Schrittweite h in der Fehlerabschätzung hinzugewinnt.
Sei im Allgemeinen $I_n(f)$ eine Quadraturformel auf dem Intervall $[a,b] \subset \mathbb{R}$ für eine integrierbare Funktion $f\colon [a,b] \to \mathbb{R}$ und sei $h := \frac{b-a}{n}$ die Schrittweite der äquidistanten Diskretisierung des Intervalls.

(i) Sei $f \in C^{n+1}([a,b])$ für $n \in \mathbb{N}_+$ beliebig. Dann gilt:

$$\left|\int_a^b f(x)\,\mathrm{d}x - I_n(f)\right| \leq c_n h^{n+2} \max_{x\in[a,b]} |f^{n+1}(x)|$$

mit

$$c_n := \frac{1}{(n+1)!}\int_0^n \prod_{k=0}^{n} |t-k|\,\mathrm{d}t.$$

(ii) Sei $f \in C^{n+2}([a,b])$ für $n \in \mathbb{N}_+$ gerade. Dann gilt:

$$\left|\int_a^b f(x)\,\mathrm{d}x - I_n(f)\right| \leq c_n^* h^{n+3} \max_{x\in[a,b]} |f^{n+2}(x)|, \qquad \text{für } c_n^* := \frac{n}{2}c_n.$$

Beweis.

(i) Da es sich bei $I_n(f)$ um eine interpolatorische Quadraturformel handelt gilt

$$\int_a^b f(x)\,\mathrm{d}x - I_n(f) \;=\; \int_a^b f(x) - P(x)\,\mathrm{d}x$$

für das eindeutig bestimmte Interpolationspolynom $P \in \Pi_n$.

Nach Theorem 5.20 können wir den Interpolationsfehler wie folgt angeben:

$$f(x) - P(x) \;=\; \frac{w_{n+1}(x)}{(n+1)!} f^{(n+1)}(\xi), \qquad \text{mit} \quad w_{n+1}(x) \;=\; \prod_{k=0}^{n}(x - x_k),$$

für ein $\xi \in [a,b]$. Somit erhalten wir für den Fehler der interpolatorischen Quadraturformel schon direkt die Abschätzung

$$\left|\int_a^b f(x)\,\mathrm{d}x - I_n(f)\right| \;\leq\; \frac{1}{(n+1)!} \max_{x\in[a,b]} |f^{(n+1)}(x)| \cdot \int_a^b |w_{n+1}(x)|\,\mathrm{d}x.$$

Wir können nun die Substitution $\varphi(t) = t{\cdot}h{+}a = x$ wieder auf das Integral $\int_a^b |w_{n+1}(x)|\mathrm{d}x$ anwenden und erhalten

$$\int_a^b |w_{n+1}(x)|\,\mathrm{d}x \;=\; \int_{\varphi(0)}^{\varphi(n)} \prod_{k=0}^{n} |x{-}x_k|\,\mathrm{d}x \;=\; \int_0^n \prod_{k=0}^{n} |h(t{-}k)|{\cdot}h\,\mathrm{d}t \;=\; h^{n+2} \int_0^n \prod_{k=0}^{n} |t{-}k|\,\mathrm{d}t.$$

Somit erhalten wir insgesamt

$$\int_a^b |w_{n+1}(x)|\,\mathrm{d}x \;\leq\; c_n h^{n+2} \max_{x\in[a,b]} |f^{n+1}(x)|.$$

(ii) Wir bemerken zunächst, dass für ein gerades $n \in \mathbb{N}_+$ das Polynom $w_{n+1} \in \Pi_{n+1}$ eine *ungerade Funktion* ist bezüglich des Mittelpunktes des Intervalls $c := \frac{a+b}{2}$ und somit gilt

$$\int_a^b w_{n+1}(x)\,\mathrm{d}x \;=\; 0.$$

Nach Theorem 5.20 können wir den Interpolationsfehler wieder wie folgt angeben:

$$f(x) - P(x) \;=\; \frac{w_{n+1}(x)}{(n+1)!} f^{(n+1)}(\xi),$$

für ein $\xi \in [a,b]$. Da wir die Funktion $f \in C^{n+2}([a,b])$ angenommen haben, können wir den Wert $f^{n+1}(\xi)$ nochmal Taylorentwickeln in $c \in [a,b]$ und erhalten somit für den

Integrationsfehler

$$\begin{aligned}\int_a^b f(x) - P(x)\,\mathrm{d}x &= \frac{1}{(n+1)!}\int_a^b w_{n+1}(x) f^{(n+1)}(\xi(x))\,\mathrm{d}x \\ &= \frac{1}{(n+1)!}\int_a^b w_{n+1}(x)[f^{(n+1)}(c) + (\xi(x)-c)f^{n+2}(\eta(x))]\,\mathrm{d}x \\ &= \frac{1}{(n+1)!} f^{(n+1)}(c) \underbrace{\int_a^b w_{n+1}(x)\,\mathrm{d}x}_{=0} \\ &\qquad + \frac{1}{(n+1)!}\int_a^b w_{n+1}(x)(\xi(x)-c)f^{n+2}(\eta(x))\,\mathrm{d}x \\ &= \frac{1}{(n+1)!}\int_a^b w_{n+1}(x)(\xi(x)-c)f^{n+2}(\eta(x))\,\mathrm{d}x.\end{aligned}$$

Wegen

$$|\xi - c| \;\leq\; c - a \;=\; \frac{b-a}{2} \;=\; \frac{n}{2}\cdot h$$

können wir mit dem Resultat aus (i) insgesamt abschätzen

$$\begin{aligned}\left|\int_a^b f(x) - P(x)\,\mathrm{d}x\right| &\leq \frac{1}{(n+1)!}\int_a^b |w_{n+1}(x)|\,\mathrm{d}x \cdot \frac{n}{2}\cdot h \cdot \max_{x\in[a,b]} |f^{n+2}(x)| \\ &= h^{n+2}\cdot c_n \cdot \frac{n}{2}\cdot h \cdot \max_{x\in[a,b]} |f^{n+2}(x)| \;=\; h^{n+3}\cdot c_n^* \cdot \max_{x\in[a,b]} |f^{n+2}(x)|.\end{aligned}$$

□

Die Fehlerabschätzung in Theorem 6.8 ist eher von theoretischer Natur und selten von praktischer Relevanz, da es in Anwendungsfällen in der Regel sehr schwierig ist das Maximum hoher Ableitungen einer Funktion f konkret zu bestimmen. Dennoch liefert uns das Theorem die Aussage, dass der Fehler hauptsächlich vom Maximum der $(n+1)$-en Ableitung von f abhängt und mit welcher Potenz der Schrittweite h dieser Fehler gegen 0 geht.

BEMERKUNG 6.9 (Divergenz des numerischen Integrationsfehlers). Da wir im Allgemeinen keine obere Schranke für hohe Ableitungen einer Funktion $f \in C^\infty([a,b])$ haben, kann es passieren, dass der Approximationsfehler durch die numerische Integration bei Verwendung von interpolatorischen Quadraturformeln für $n \to \infty$ **divergiert**. Insbesondere erhalten wir nicht die gewünschte Konvergenz des Fehlers gegen Null, d.h., es gilt im Allgemeinen:

$$\left|\int_a^b f(x)\,\mathrm{d}x - I_n(f)\right| \;\not\longrightarrow\; 0 \qquad \text{für} \quad n \to \infty.$$

△

Das folgende Beispiel berechnet allgemeine Fehlerabschätzungen für die ersten beiden interpolatorischen Quadraturformeln.

BEISPIEL 6.10: Fehlerabschätzungen für interpolatorische Quadraturen.

In diesem Beispiel untersuchen wir den allgemeinen Fehler der interpolatorischen Quadraturformeln aus Beispiel 6.5 für $n = 1$ und $n = 2$.

(i) Für $n = 1$ betrachten wir die **Trapezregel** und es gilt die Identität

$$\int_0^1 |t-0| \cdot |t-1| \, \mathrm{d}t \;=\; \int_0^1 t \cdot (1-t) \, \mathrm{d}x \;=\; \int_0^1 t - t^2 \, \mathrm{d}x \;=\; \left[\frac{1}{2}t^2 - \frac{1}{3}t^3\right]_0^1 \;=\; \frac{1}{6}.$$

Nehmen wir nun an, dass eine Funktion $f \in C^2([a,b])$ vorliegt, so können wir das Theorem 6.8 anwenden und erhalten die Fehlerabschätzung

$$\begin{aligned}\left|\int_a^b f(x)\,\mathrm{d}x - I_1(f)\right| \;&\leq\; c_1 \cdot h^3 \cdot \max_{x\in[a,b]} |f''(x)| \\ &=\; (b-a)^3 \cdot \frac{1}{2}\int_0^1 |t-0|\cdot|t-1|\,\mathrm{d}t \cdot \|f''(x)\|_\infty \\ &=\; \frac{(b-a)^3}{12} \cdot \|f''\|_\infty.\end{aligned}$$

(ii) Für $n = 2$ berachten wir die **Simpsonregel** und es gilt die Identität

$$\begin{aligned}\int_0^2 |t-0|\cdot|t-1|\cdot|t-2|\,\mathrm{d}t &= \int_0^1 t\cdot(1-t)\cdot(2-t)\,\mathrm{d}t + \int_1^2 t\cdot(t-1)\cdot(2-t)\,\mathrm{d}t \\ &=\; \int_0^1 t^3 - 3t^2 + 2t\,\mathrm{d}t + \int_1^2 -t^3 + 3t^2 - 2t\,\mathrm{d}t \\ &=\; \left[\frac{1}{4}t^4 - t^3 + t^2\right]_0^1 + \left[-\frac{1}{4}t^4 + t^3 - t^2\right]_1^2 \\ &=\; \frac{1}{4} - 1 + 1 + (-4 + 8 - 4) + \frac{1}{4} - 1 + 1 \;=\; \frac{1}{2}.\end{aligned}$$

Da $n = 2$ gerade ist nehmen wir eine Funktion $f \in C^4([a,b])$ an und wenden Theorem 6.8 an um die folgende Fehlerabschätzung zu erhalten

$$\begin{aligned}\left|\int_a^b f(x)\,\mathrm{d}x - I_2(f)\right| \;&\leq\; h^5 \cdot c_2^* \cdot \max_{x\in[a,b]} |f^{(4)}(x)| \\ &=\; \left(\frac{b-a}{2}\right)^5 \cdot \frac{1}{6}\int_0^2 |t-0|\cdot|t-1|\cdot|t-2|\,\mathrm{d}t \cdot \|f^{(4)}(x)\|_\infty \\ &=\; \frac{(b-a)^5}{384} \cdot \|f^{(4)}\|_\infty.\end{aligned}$$

BEMERKUNG 6.11 (Peanosche Fehlerdarstellung). Die in Theorem 6.8 gegebene Fehlerabschätzung ist nur eine obere Schranke. In der Tat ist es mittels der *Peanoschen Fehlerdarstellung* [FH07, Kapitel 3.2] möglich den numerischen Integrationsfehler exakter zu berechnen. Die zugehörige Theorie liegt jedoch außerhalb des Umfangs dieser Vorlesung. △

6.1.3 Stückweise interpolatorische Quadratur

Wie wir in Runges Gegenbeispiel in Beispiel 5.23 gesehen haben geht der Interpolationsfehler $||f - P_n||_\infty$ für ein Interpolationspolynom $P_n \in \Pi_n$ für $n \to \infty$ im Allgemeinen nicht gegen Null, wenn man das Intervall mit äquidistanten Stützstellen diskretisiert, was an der Unbeschränktheit des Terms $||f^{n+1}(x)||_\infty$ liegt. Dementsprechend können wir nicht erwarten, dass die interpolatorischen Quadraturen eine immer bessere Integral-Approximation für immer größer werdende $n \in \mathbb{N}$ liefern.

Im Falle der numerischen Interpolation haben wir in Abschnitt 5.5 die Idee der **Interpolation mit Splines** von niedrigem Grad $n \in \mathbb{N}$ eingeführt und deren Vorteile diskutiert. Als interpolierende Funktion für die Integrationsformeln nach Newton-Cotes können wir neben regulären Polynomen natürlich auch hier stückweise zusammengesetzte Polynome verwenden.

Wir verwenden in diesem Abschnitt wieder eine Diskretisierung des Intervalls $[a, b] \subset \mathbb{R}$ mit fester Schrittweite $h = \frac{b-a}{n}$ und $n+1$ äquidistanten Stützstellen $x_0, \ldots, x_n \in [a, b]$ wie in (6.70). Anstatt die interpolatorischen Quadraturformeln aus dem letzten Abschnitt für das gesamte Intervall $[a, b]$ anzuwenden, beschränken wir uns im Folgenden auf eine Reihe geeigneter Teilintervalle von $[a, b]$ und summieren schließlich die Teilapproximationen für eine Annäherung des bestimmten Integrals $\int_a^b f(x)\mathrm{d}x$. Der Vorteil dieses Vorgehens ist es, dass wir den Maximalwert der höchsten auftretenden Ableitung beschränken können und somit Fehlerabschätzungen abhängig von der Schrittweite h der Diskretisierung erhalten.

Im Fall eines linearen Polynoms, d.h für $n = 1$, wenden wir also die interpolatorische Quadraturformel I_1 (was der Trapezregel entspricht) für jedes Teilintervall $[x_i, x_{i+1}], i = 0, \ldots, n-1$ an. Dementsprechend definieren wir die Annäherungswerte der numerischen Integration mittels Trapezregel auf den n Teilintervallen von $[a, b]$ wie folgt:

$$I_1^i(h) \;:=\; \frac{h}{2}(f(x_i) + f(x_{i+1})) \;\approx\; \int_{x_i}^{x_{i+1}} f(x)\,\mathrm{d}x, \qquad i = 0, \ldots, n-1.$$

Wir erhalten somit für das gesamte Intervall $[a, b]$ durch Summation über die n Teilintervalle die sogenannte **zusammengesetzte Trapezregel** oder **Trapezsumme** als

$$T(h) \;:=\; \sum_{i=0}^{n-1} I_1^i(h) \;=\; \sum_{i=0}^{n-1} \frac{h}{2}(f(x_i) + f(x_{i+1})) \;=\; \frac{h}{2}\left(f(a) + 2\sum_{i=1}^{n-1} f(x_i) + f(b)\right). \tag{6.73}$$

Man beachte, dass es sich hierbei in der Tat um einen Spline vom Grad 1 gemäß Definition 5.24 handelt.

Dieses Prinzip können wir auch auf andere Quadraturformeln anwenden, indem wir diese einfach auf geeignete Teilintervallen anwenden. So können wir zum Beispiel für gerades $n \in \mathbb{N}, n \geq 4$ auf jedes Teilintervall $[x_{2i}, x_{2i+2}] \subset [a, b], i = 0, \ldots, \frac{n}{2} - 1$ die Simpsonregel mit Hilfe eines quadratischen Polynoms anwenden. In diesem Fall definieren wir die Annäherungswerte der numerischen Integration auf den $\frac{n}{2}$ Teilintervallen von $[a, b]$ wie folgt:

$$I_2^i(h) \;:=\; \frac{h}{3}(f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})) \;\approx\; \int_{x_{2i}}^{x_{2i+2}} f(x)\,\mathrm{d}x, \qquad i = 0, \ldots, \frac{n}{2} - 1.$$

Wir erhalten somit für das gesamte Intervall $[a,b]$ durch Summation über die $\frac{n}{2}$ Teilintervalle die sogenannte **zusammengesetzte Simpsonregel** als

$$\begin{aligned} S(h) &:= \sum_{i=0}^{\frac{n}{2}-1} I_2^i(h) = \sum_{i=0}^{\frac{n}{2}-1} \frac{h}{3}(f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})) \\ &= \frac{h}{3}\left(f(a) + 4\sum_{i=0}^{\frac{n}{2}-1} f(x_{2i+1}) + 2\sum_{i=0}^{\frac{n}{2}-2} f(x_{2i+2}) + f(b)\right). \end{aligned}$$

Mit Hilfe der Fehlerabschätzungen für die Trapez- und die Simpsonregel in Beispiel 6.10 können wir den Fehler für die stückweisen interpolatorischen Quadraturen mit linearen und quadratischen Polynomen angeben.

KOROLLAR 6.12: Fehler für stückweise interpolatorische Quadratur.

Sei $f\colon [a,b] \to \mathbb{R}$ eine integrierbare, hinreichend glatte Funktion und sei $h := \frac{b-a}{n}$ eine feste Schrittweite für eine Diskretisierung des Intervalls $[a,b] \subset \mathbb{R}$ in $n \in \mathbb{N}_+$ gleich große Teilintervalle. Dann können wir den Fehler für die stückweise interpolatorischen Quadraturen mit linearen und quadratischen Polynomen nach oben wie folgt abschätzen.

(i) Für $f \in C^2([a,b])$ gilt für die **zusammengesetzte Trapezregel** die Fehlerabschätzung:

$$\begin{aligned} \left|\int_a^b f(x)\,\mathrm{d}x - T(h)\right| &= \left|\int_a^b f(x)\,\mathrm{d}x - \sum_{i=0}^{n-1} I_1^i(h)\right| \\ &= \left|\sum_{i=0}^{n-1}\int_{x_i}^{x_{i+1}} f(x)\,\mathrm{d}x - I_1^i(h)\right| \\ &\le \sum_{i=0}^{n-1}\left|\int_{x_i}^{x_{i+1}} f(x)\,\mathrm{d}x - \frac{h}{2}(f(x_i) + f(x_{i+1}))\right| \end{aligned}$$

Wir können nun die Fehlerabschätzung für die Trapezregel aus Beispiel 6.10 einsetzen und erhalten somit

$$\left|\int_a^b f(x)\,\mathrm{d}x - T(h)\right| \le \sum_{i=0}^{n-1}\frac{h^3}{12}\|f''\|_\infty = n\cdot\frac{b-a}{n}\cdot\frac{h^2}{12}\|f''\|_\infty = \frac{b-a}{12}h^2\|f''\|_\infty.$$

(ii) Für $f \in C^4([a,b])$ und eine gerade Anzahl $n \in \mathbb{N}_+$ von Teilintervallen gilt für die **zusammengesetzte Simpsonregel** die Fehlerabschätzung:

$$\begin{aligned} \left|\int_a^b f(x)\,\mathrm{d}x - S(h)\right| &= \left|\int_a^b f(x)\,\mathrm{d}x - \sum_{i=0}^{\frac{n}{2}-1} I_2^i(h)\right| = \left|\sum_{i=0}^{\frac{n}{2}-1}\int_{x_{2i}}^{x_{2i+2}} f(x)\,\mathrm{d}x - I_2^i(h)\right| \\ &\le \sum_{i=0}^{\frac{n}{2}-1}\left|\int_{x_{2i}}^{x_{2i+2}} f(x)\,\mathrm{d}x - \frac{h}{3}\left(f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})\right)\right| \end{aligned}$$

Wir können wiederum die Fehlerabschätzung für die Simpsonregel aus Beispiel 6.10 einsetzen und erhalten somit

$$\left|\int_a^b f(x)\,\mathrm{d}x - S(h)\right| \leq \sum_{i=0}^{\frac{n}{2}-1} \frac{h^5}{12}\cdot\|f^{(4)}\|_\infty = \frac{n}{2}\cdot\frac{b-a}{n}\cdot\frac{h^4}{12}\|f^{(4)}\|_\infty = \frac{b-a}{24}h^4\|f^{(4)}\|_\infty.$$

BEMERKUNG 6.13 (Konvergenz des numerischen Integrationsfehlers). Durch die Fehlerabschätzungen in Korollar 6.12 sehen wir, dass der Approximationsfehler der numerischen Integration für die oben hergeleiteten stückweise interpolatorischen Quadraturformeln im Gegensatz zu den einfachen interpolatorischen Quadraturformeln (wie in Bemerkung 6.9 diskutiert) gegen Null konvergiert, da in diesem Fall die Maximumsnorm der Ableitung der Funktion f unabhängig von $n \in \mathbb{N}$ ist und somit beschränkt ist. Das heißt es gilt:

$$\left|\int_a^b f(x)\,\mathrm{d}x - T(h)\right| \longrightarrow 0 \quad \text{und} \quad \left|\int_a^b f(x)\,\mathrm{d}x - S(h)\right| \longrightarrow 0 \qquad \text{für} \quad h \to 0.$$

△

6.2 Numerische Integration mit unterschiedlichen Schrittweiten

Bisher haben wir interpolatorische Quadraturformeln für die numerische Integration hergeleitet, die auf einer festen Schrittweite $h > 0$ und $n \in \mathbb{N}_+$ Stützpunkten basierten. Wir haben bereits im letzten Abschnitt festgestellt, dass dies für $n \geq 8$ zu Instabilitäten führt. Der Grund hierfür ist, dass das Interpolationspolynom durch die Wahl von äquidistanten Stützstellen eine beliebig große Abweichung von der zu Grunde liegenden Funktion haben kann im Sinne der Maximumsnorm (vergleiche Beispiel 5.23).

Aus diesem Grund wollen wir in diesem Abschnitt Ingegrationsformeln **mit unterschiedlichen Schrittweiten** $h > 0$ konstruieren, deren Approximationsfehler mit einer hohen Potenz von h abfallen. Im Speziellen widmen wir uns dem *Romberg-Verfahren*, das im Wesentlich auf der Trapezregel basiert.

6.2.1 Euler-Maclaurinsche Summenformel

Wir beginnen zunächst mit einer zentralen Idee, die wir als Grundlage für die numerische Integration mit verschiedenen Schrittweiten nutzen wollen, der *Euler-Maclaurinschen Summenformel*. Ursprünglich leitete Euler sich diese Formel her zur Berechnung einer Summe von Funktionswerten durch die Werte der Ableitungen dieser Funktion an den Summationsgrenzen. Maclaurin wandelte sie dann später ab für die numerische Approximation eines bestimmten Integrals über einzelne Werte des Integranden und seiner Ableitungen.

Zur Formulierung der Euler-Maclaurinschen Summenformel benötigen wir zunächst die Definition sogenannter Bernoulli-Polynome und Bernoulli-Zahlen. Diese tauchen in unterschiedli-

chen Gebieten der Mathematik auf und spielen beispielsweise bei der Riemannschen Zetafunktion ebenfalls eine wichtige Rolle.

Wir wollen zunächst die grundlegenden Bernoulli-Polynome und die damit eng verknüpften Bernoulli-Zahlen einführen.

DEFINITION 6.14: Bernoulli-Polynome und Bernoulli-Zahlen.

Sei $n \in \mathbb{N}$, dann ist das **Bernoulli-Polynom** $\mathrm{B_n} \in \Pi_n$ mit $\mathrm{B_n}\colon [0,1] \to \mathbb{R}$ durch folgende Rekursionsgleichungen

$$\begin{aligned} \mathrm{B}_0(x) &:= 1 \\ \mathrm{B}'_{n+1}(x) &:= (n+1)\cdot \mathrm{B_n}(x), \end{aligned} \tag{6.74}$$

und die zusätzliche Integralbedingung

$$\int_0^1 \mathrm{B_n}(x)\,\mathrm{d}x = 0, \qquad n \geq 1 \tag{6.75}$$

vollständig charakterisiert.
Wir definieren die **Bernoulli-Zahlen** (erster Art) $B_n \in \mathbb{R}$ als konstante Terme der Bernoulli-Polynome $\mathrm{B_n}(x) \in \Pi_n$, d.h. wir setzen

$$B_n := \mathrm{B_n}(0).$$

Das folgende Beispiel gibt die ersten Bernoulli-Polynome explizit an und zeigt, dass die Bedingungen in Definition 6.14 diese schon eindeutig festlegen.

BEISPIEL 6.15: Erste Bernoulli-Polynome.

Wir wollen im Folgenden die ersten Bernoulli-Polynome $\mathrm{B_n}(x) \in \Pi_n$ für $n = 0,\dots,6$ aus den Rekursionsgleichungen (6.74) und der Integralbedingung (6.75) berechnen.
Das erste Bernoulli-Polynom B_0 für $n = 0$ ist per Definition gegeben und es gilt $\mathrm{B}_0(x) \equiv 1$ für $x \in [0,1]$.
Für $n = 1$ müssen wir nun die folgende Rekursionsbedingung erfüllen

$$\mathrm{B}'_1(x) = 1 \cdot \mathrm{B}_0(x) = 1.$$

Daraus folgt, dass das Bernoulli-Polynom $\mathrm{B}_1(x)$ die Form $\mathrm{B}_1(x) = x + c$ mit einer unbestimmten Integrationskonstante $c \in \mathbb{R}$ haben muss. Zur Bestimmung von c wenden wir nun die zusätzliche Bedingung (6.75) an, die besagt, dass für Bernoulli-Polynome gelten muss:

$$0 = \int_0^1 \mathrm{B}_1(x)\,\mathrm{d}x = \int_0^1 x + c\,\mathrm{d}x = \left[\frac{1}{2}x^2 + cx\right]_0^1 = \frac{1}{2} + c.$$

Daraus folgt direkt, dass $c = -\frac{1}{2}$ sein muss und somit hat das Bernoulli-Polynom B_1 die Form

$$\mathrm{B}_1(x) = x - \frac{1}{2}.$$

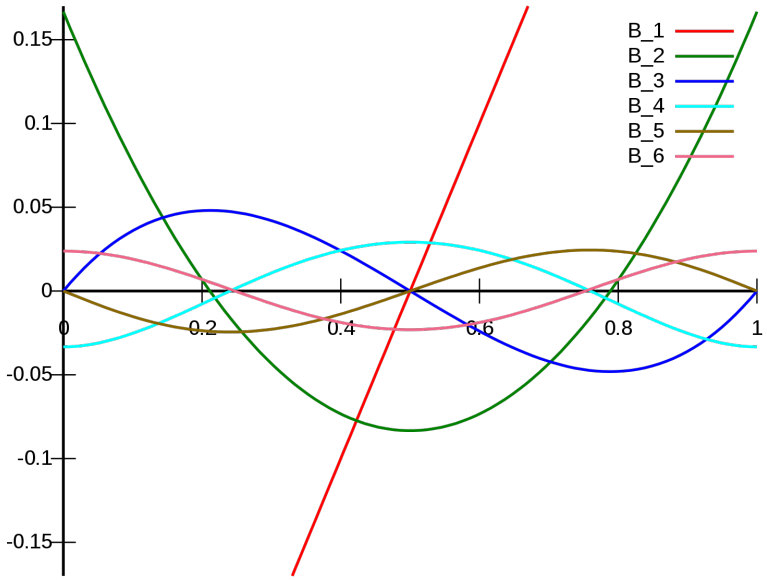


Abbildung 6.2: Visualisierung der Bernoulli-Polynome B_n vom Grad 1 bis 6 auf dem Intervall $[0,1]$ [Poly]

Dieses Vorgehen kann man nun sukzessive weiter anwenden und erhält so die ersten Bernoulli-Polynome für $n = 0, \ldots, 6$ durch:

$$\begin{aligned}
B_0(x) &= 1, \\
B_1(x) &= x - \frac{1}{2}, \\
B_2(x) &= x^2 - x + \frac{1}{6}, \\
B_3(x) &= x^3 - \frac{3}{2}x^2 + \frac{1}{2}x, \\
B_4(x) &= x^4 - 2x^3 + x^2 - \frac{1}{30}, \\
B_5(x) &= x^5 - \frac{5}{2}x^4 + \frac{5}{3}x^3 - \frac{1}{6}x, \\
B_6(x) &= x^6 - 3x^5 + \frac{5}{2}x^4 - \frac{1}{2}x^2 + \frac{1}{42}.
\end{aligned}$$

Abb. 6.2 visualisiert die Bernoulli-Polynome B_n für $n = 1, \ldots, 6$ auf dem Intervall $[0,1]$.

BEMERKUNG 6.16 (Alternative Definition von Bernoulli-Polynomen). In der Literatur werden Bernoulli-Polynome anstatt durch die Integralbedingung (6.75) manchmal auch durch eine alternative Randbedingung beschrieben. Hierbei fordert man, dass die konstanten

Terme der Bernoulli-Polynome ungeraden Grades verschwinden, d.h. es soll gelten

$$\mathrm{B}_{2n+1}(0) \;=\; \mathrm{B}_{2n+1}(1) \;=\; 0, \qquad n \in \mathbb{N}_+. \tag{6.76}$$

Diese Bedingung lässt sich als Eigenschaft ebenfalls aus der Integralbedingung (6.75) folgern (siehe Lemma 6.19).
Für ein gegebenes Polynom B_{2n-1} ist B_{2n} durch die Rekursionsbedingung (6.74) bis auf eine additive Integrationskonstante eindeutig bestimmt. Diese lässt sich durch die zusätzliche Randbedingung (6.76)für Bernoulli-Polynome B_{2n+1} von ungeradem Grad eindeutig bestimmen. Um dies zu verstehen betrachten wir zunächst für $n \in \mathbb{N}_+$ das allgemeine Bernoulli-Polynom

$$\mathrm{B}_{2n-1}(x) \;=\; x^{2n-1} + c_{2n-2}x^{2n-2} + \ldots + c_1 x + c_0$$

mit bekannten Koeffizienten $c_0, \ldots, c_{2n-2} \in \mathbb{R}$. Durch Integration und unter Einhaltung der Bedingung (6.74) folgt dann, dass

$$\mathrm{B}_{2n}(x) \;=\; x^{2n} + \frac{2n}{2n-1} \cdot c_{2n-2}x^{2n-1} + \ldots + \frac{2n}{2}c_1 x^2 + 2nc_0 x + c$$

mit einer noch unbestimmten Integrationskonstante $c \in \mathbb{R}$ gilt. Unter Verwendung des gleichen Arguments erhalten wir durch nochmalige Integration

$$\mathrm{B}_{2n+1}(x) \;=\; x^{2n+1} + \frac{(2n+1)2n}{2n(2n-1)}c_{2n-2}x^{2n} + \ldots + \frac{(2n+1)2n}{3\cdot 2}c_1 x^3 + \frac{(2n+1)2n}{2}c_0 x^2 + (2n+1)cx + d$$

mit unbestimmten Integrationskonstanten $c, d \in \mathbb{R}$. Wendet man nun die zusätzliche Randbedingungen (6.79) an, so erhält man zunächst, dass wegen $\mathrm{B}_{2n+1}(0) = 0$ schon $d = 0$ gelten muss. Die Integrationskonstante c ist ebenfalls eindeutig bestimmt, denn wir können aus der Bedingung $\mathrm{B}_{2n+1}(1) = 0$ und $d = 0$ einfach berechnen:

$$\begin{aligned} 0 \;&\overset{!}{=}\; 1 + \frac{(2n+1)2n}{2n(2n-1)}c_{2n-2} + \ldots + + \frac{(2n+1)2n}{3\cdot 2}c_1 + \frac{(2n+1)2n}{2}c_0 + (2n+1)c \\ \Rightarrow \quad c \;&=\; -\frac{1 + \frac{(2n+1)2n}{2n(2n-1)}c_{2n-2} + \ldots + \frac{(2n+1)2n}{3\cdot 2}c_1 + \frac{(2n+1)2n}{2}c_0}{2n+1}. \end{aligned}$$

Aus diesem Grund sind Bernoulli-Polynome durch die Bedingungen in (6.76) ebenfalls vollständig charakterisiert. △

Die folgende Bemerkung nennt weitere Darstellungsformen der Bernoulli-Polynome aus verschiedenen mathematischen Gebieten.

BEMERKUNG 6.17 (Darstellung von Bernoulli-Polynomen und -zahlen). Bernoulli-Polynome tauchen in sehr unterschiedlichen mathematischen Teilgebieten auf, z.B. in der algebraischen Zahlentheorie, in der Kombinatorik oder der algebraischen Topologie. Daher gibt es auch alternative Darstellungsformen als die in Definition 6.14 gegebene, wie wir im Folgenden festhalten wollen.

(i) Man kann zeigen, dass sich das n-te Bernoulli-Polynom $\mathrm{B_n} \in \Pi_n$ durch folgenden geschlossenen Ausduck *rekursiv* berechnen lässt:

$$\mathrm{B_n}(x) \;=\; \sum_{k=0}^{n} \binom{n}{k} B_k x^{n-k},$$

wobei $B_k \in \mathbb{R}$ hier die Bernoulli-Zahlen (erster Art) aus Definition 6.14 bezeichnen.

Auf Grund der Rekursionsbeziehung (6.74) lassen sich die Bernoulli-Zahlen $B_n \in \mathbb{R}$ für den Startwert $B_0 \coloneqq 1$ ebenfalls *rekursiv* wie folgt berechnen:

$$B_n \;\coloneqq\; -\frac{1}{n+1} \sum_{k=0}^{n-1} \binom{n+1}{k} B_k. \tag{6.77}$$

(ii) Die Bernoulli-Zahlen $B_n \in \mathbb{R}$ lassen sich mit Hilfe einer **generierenden Funktion** implizit definieren als Koeffizienten der folgenden Reihendarstellung

$$\frac{x}{e^x - 1} \;=\; \sum_{k=0}^{\infty} B_k \frac{x^k}{k!}.$$

Daraus lässt sich ein ähnlicher Zusammenhang für die Bernoulli-Polynome $\mathrm{B_n} \in \Pi_n$ ableiten, denn es gilt für $x, t \in \mathbb{R}$:

$$\frac{te^{tx}}{e^t - 1} \;=\; \sum_{k=0}^{\infty} \mathrm{B_k}(x) \frac{t^k}{k!}.$$

△

Wir wollen im Folgenden die ersten drei Bernoulli-Zahlen mit Hilfe der Rekursionformel in (6.77) explizit berechnen.

BEISPIEL 6.18: Bernoulli-Zahlen.

Wir wollen in diesem Beispiel die ersten drei Bernoulli-Zahlen $B_1, B_2, B_3 \in \mathbb{R}$ explizit mittels Rekursion berechnen. Für $n = 1$ berechnen wir die einzige Bernoulli-Zahl mit ungeradem Index $B_1 \neq 0$ als:

$$B_1 \;=\; -\frac{1}{2} \cdot \binom{2}{0} B_0 \;=\; -\frac{1}{2} \cdot \frac{2!}{0! \cdot 2!} \cdot 1 \;=\; -\frac{\mathbf{1}}{\mathbf{2}}$$

Für $n = 2$ berechnen wir mit Hilfe der vorigen Bernoulli-Zahlen B_0, B_1 die Bernoulli-Zahl $B_2 \in \mathbb{R}$ als:

$$\begin{aligned} B_2 \;&=\; -\frac{1}{3} \cdot \left[\binom{3}{0} B_0 + \binom{3}{1} B_1 \right] \;=\; -\frac{1}{3} \cdot \left[\frac{3!}{0! \cdot 3!} \cdot 1 - \frac{3!}{1! \cdot 2!} \cdot \frac{1}{2} \right] \\ &=\; -\frac{1}{3} \cdot \left[1 - \frac{3}{2} \right] \;=\; \frac{\mathbf{1}}{\mathbf{6}}. \end{aligned}$$

Für $n = 3$ berechnen wir mit Hilfe der vorigen Bernoulli-Zahlen B_0, B_1, B_2 die Bernoulli-Zahl $B_3 \in \mathbb{R}$ als:

$$\begin{aligned} B_3 &= -\frac{1}{4} \cdot \left[\binom{4}{0} B_0 + \binom{4}{1} B_1 + \binom{4}{2} B_2 \right] \\ &= -\frac{1}{4} \cdot \left[\frac{4!}{0! \cdot 4!} \cdot 1 - \frac{4!}{1! \cdot 3!} \cdot \frac{1}{2} + \frac{4!}{2! \cdot 2!} \cdot \frac{1}{6} \right] \\ &= -\frac{1}{4} \cdot [1 - 2 + 1] \;=\; \mathbf{0}. \end{aligned}$$

Das folgende Hilfslemma formuliert wichtige Eigenschaften der Bernoulli-Polynome, die für den Beweis von Theorem 6.20 entscheidend sein werden.

LEMMA 6.19: Eigenschaften der Bernoulli-Polynome.

Wir können die folgenden Eigenschaften für die Bernoulli-Polynome $\mathrm{B_n} \in \Pi_n$ festhalten.

(i) Für $n \in \mathbb{N}, n \geq 2$ gelten für das Bernoulli-Polynom $\mathrm{B_n}$ die Randbedingungen

$$\mathrm{B_n}(1) \;=\; \mathrm{B_n}(0) \;=\; B_n, \tag{6.78}$$

wobei B_n die n-te Bernoulli-Zahl bezeichnet.

(ii) Für $n \in \mathbb{N}_+$ gilt das folgende *Umbral-Kalkül* für das Bernoulli-Polynom $\mathrm{B_n}$:

$$F_n(x) \;:=\; \mathrm{B_n}(x+1) - \mathrm{B_n}(x) \;=\; n \cdot x^{n-1}.$$

(iii) Falls $P \in \Pi_n$ für $n \in \mathbb{N}$ ein Polynom mit der Eigenschaft aus (ii) ist, d.h. es gilt

$$P(x+1) - P(x) \;=\; n \cdot x^{n-1},$$

so ist P schon von der Form $P(x) = \mathrm{B_n}(x) + c$ mit einer Konstanten $c \in \mathbb{R}$.

(iv) Für $n \in \mathbb{N}$ gilt die folgende Symmetriebeziehung für ein Bernoulli-Polynom $\mathrm{B_n}$:

$$(-1)^n \mathrm{B_n}(1-x) \;=\; \mathrm{B_n}(x), \qquad \forall\, x \in [0,1].$$

(v) Für $n \in \mathbb{N}_+$ gelten für alle Bernoulli-Polynome von ungeradem Grad $\mathrm{B_{2n+1}}$ die Randbedingungen

$$\mathrm{B_{2n+1}}(0) \;=\; \mathrm{B_{2n+1}}(1) \;=\; 0. \tag{6.79}$$

Beweis. In der Hausaufgabe zu zeigen. □

Mit den obigen Vorüberlegungen zu Bernoulli-Polnomen und -zahlen sind wir nun in der Lage die zentrale Aussage dieses Abschnitts zu formulieren und beweisen.

THEOREM 6.20: Euler-Maclaurinsche Summenformel.

Sei $f \in C^{2m+2}([0,1])$ eine auf dem Intervall $[0,1]$ mindestens $(2m+2)$-mal stetig differenzierbare Funktion für ein beliebiges $m \in \mathbb{N}$. Dann existiert ein $\xi \in (0,1)$, so dass die sogenannte **Euler-Maclaurinsche Summenformel** gilt:

$$\int_0^1 f(t)\,\mathrm{d}t \;=\; \frac{f(0)}{2} + \frac{f(1)}{2} + \sum_{k=1}^{m} \frac{B_{2k}}{(2k)!}\left(f^{(2k-1)}(0) - f^{(2k-1)}(1)\right) - \frac{B_{2m+2}}{(2m+2)!} f^{(2m+2)}(\xi).$$

Hierbei stellen die Koeffizienten $B_{2k} \in \mathbb{R}, k = 1, \ldots, m+1$ die Bernoulli-Zahlen aus Definition 6.14 dar.

Beweis. Zunächst formen wir das bestimmte Integral $\int_0^1 f(t)\mathrm{d}t$ mit Hilfe der *partiellen Integration* für das Bernoulli-Polynom $\mathrm{B}_1 \in \Pi_1$ mit $\mathrm{B}_1(x) \coloneqq x - \frac{1}{2}$ um zu:

$$\int_0^1 f(t)\,\mathrm{d}t \;=\; \int_0^1 \underbrace{\mathrm{B}_1'(t)}_{=1} \cdot f(t)\,\mathrm{d}t \;=\; \Big[\mathrm{B}_1(t)\cdot f(t)\Big]_0^1 - \int_0^1 \mathrm{B}_1(t)\cdot f'(t)\,\mathrm{d}t$$

Den ersten Term berechnen wir explizit als

$$\Big[\mathrm{B}_1(t)\cdot f(t)\Big]_0^1 \;=\; \left(1-\frac{1}{2}\right)\cdot f(1) - \left(0-\frac{1}{2}\right)\cdot f(0) \;=\; \frac{1}{2}(f(0)+f(1)). \tag{6.80}$$

Wir können das auftretende bestimmte Integral $\int_0^1 \mathrm{B}_1(t) f'(t)\,\mathrm{d}t$ nun sukzessive weiter durch partielle Integration umformen, indem wir die Rekursioneigenschaft der Bernoulli-Polynome B_n in (6.74) ausnutzen. Dadurch erhalten wir die folgenden Identitäten:

$$\begin{aligned}
\int_0^1 \mathrm{B}_1(t)\cdot f'(t)\,\mathrm{d}t \;&=\; \int_0^1 \frac{1}{2}\mathrm{B}_2'(t)\cdot f'(t)\,\mathrm{d}t \\
&=\; \frac{1}{2}\Big[\mathrm{B}_2(t)\cdot f'(t)\Big]_0^1 - \frac{1}{2}\int_0^1 \mathrm{B}_2(t)\cdot f''(t)\,\mathrm{d}t \\
&\;\;\vdots \\
\int_0^1 \mathrm{B}_{\mathrm{k}-1}(t)\cdot f^{(k-1)}(t)\,\mathrm{d}t \;&=\; \int_0^1 \frac{1}{k}\mathrm{B}_\mathrm{k}'(t)\cdot f^{(k-1)}(t)\,\mathrm{d}t \\
&=\; \frac{1}{k}\Big[\mathrm{B}_\mathrm{k}(t)\cdot f^{(k-1)}(t)\Big]_0^1 - \frac{1}{k}\int_0^1 \mathrm{B}_\mathrm{k}(t)\cdot f^{(k)}(t)\,\mathrm{d}t
\end{aligned} \tag{6.81}$$

Nun können wegen der Randbedingungen (6.78) der Bernoulli-Polynome B_k mit $k > 1$ für die Auswertungen der partiellen Integrationen in (6.81) folgern, dass gilt

$$\Big[\frac{1}{k}\mathrm{B}_\mathrm{k}(t) f^{(k-1)}(t)\Big]_0^1 \;=\; \frac{1}{k}B_k f^{(k-1)}(1) - \frac{1}{k}B_k f^{(k-1)}(0) \;=\; -\frac{B_k}{k}\left(f^{(k-1)}(0) - f^{(k-1)}(1)\right).$$

Wegen der Randbedingung (6.79) wissen wir außerdem, dass für die Bernoulli-Polynome B_k von ungeradem Grad $k \in N$ gilt:

$$\mathrm{B}_\mathrm{k}(0) \;=\; \mathrm{B}_\mathrm{k}(1) \;=\; 0.$$

Dadurch fallen in (6.81) also alle Terme der Form $\frac{1}{k}\Big[\mathrm{B_k}(t)\cdot f^{(k-1)}(t)\Big]_0^1$ für ungerades $k \in N$ weg und somit lässt sich mit (6.80) die Euler-Maclaurinsche Summenformel nun schreiben als

$$\int_0^1 f(t)\,\mathrm{d}t \;=\; \frac{1}{2}(f(0)+f(1)) + \sum_{k=1}^{m} \frac{B_{2k}}{(2k)!}\left(f^{(2k-1)}(0) - f^{(2k-1)}(1)\right) + r_{m+1},$$

wobei für das Restglied r_{m+1} gilt:

$$r_{m+1} \;:=\; -\frac{1}{(2m+1)!}\int_0^1 \mathrm{B_{2m+1}}(t) f^{(2m+1)}(t)\,\mathrm{d}t.$$

Integrieren wir das Integral des Restglieds r_{m+1} nochmals partiell erhalten wir

$$\int_0^1 \mathrm{B_{2m+1}}(t) f^{(2m+1)}(t)\,\mathrm{d}t \;=$$
$$\Big[\frac{1}{2m+2}\left(\mathrm{B_{2m+1}}(t) - B_{2m+2}\right) f^{(2m+2)}(t)\Big]_0^1 - \frac{1}{2m+2}\int_0^1 (\mathrm{B_{2m+2}}(t) - B_{2m+2}) f^{(2m+2)}(t)\,\mathrm{d}t.$$

Der erste Term verschwindet wegen (6.78), so dass für das Restglied insgesamt gilt:

$$r_{m+1} \;=\; \frac{1}{(2m+2)!}\int_0^1 (\mathrm{B_{2m+2}}(t) - B_{2m+2}) f^{(2m+2)}(t)\,\mathrm{d}t$$

Wir können das Restglied r_{m+1} weiter vereinfachen, wenn wir annehmen, dass die Hilfsfunktion

$$g(t) \;:=\; \mathrm{B_{2m+2}}(t) - B_{2m+2}$$

auf dem Intervall $[0,1]$ nicht das Vorzeichen wechselt, d.h., es gilt entweder die Bedingung $g(t) \geq 0$ oder die Bedingung $g(t) \leq 0$ für alle $t \in [0,1]$. Das dem wirklich so ist, werden wir im Anschluss in Lemma 6.21 beweisen.

Nun können wir den **Mittelwertsatz der Integralrechnung** anwenden und damit das Restglied wie folgt für ein $\xi \in [0,1]$ vereinfachen:

$$\begin{aligned} r_{m+1} \;&=\; \frac{1}{(2m+2)!}\int_0^1 (\mathrm{B_{2m+2}}(t) - B_{2m+2}) f^{(2m+2)}(t)\,\mathrm{d}t \;=\; \frac{1}{(2m+2)!}\int_0^1 g(t) f^{(2m+2)}(t)\,\mathrm{d}t \\ &=\; \frac{f^{(2m+2)}(\xi)}{(2m+2)!}\int_0^1 g(t)\,\mathrm{d}t \;=\; \frac{f^{(2m+2)}(\xi)}{(2m+2)!}\int_0^1 \mathrm{B_{2m+2}}(t) - B_{2m+2}\,\mathrm{d}t, \end{aligned}$$

für ein $\xi \in [0,1]$. Wegen der Integrationseigenschaft der Bernoulli-Polynome in Lemma 6.19 verschwindet das Integral und somit gilt für das Restglied insgesamt

$$r_{m+1} \;=\; -\frac{B_{2m+2}}{(2m+2)!} f^{(2m+2)}(\xi), \qquad \xi \in [0,1].$$

Damit haben wir die Euler-Maclaurinsche Summenformel schließlich bewiesen und es gilt insgesamt für ein $\xi \in [0,1]$:

$$\int_0^1 f(t)\,\mathrm{d}t \;=\; \frac{1}{2}(f(0)+f(1)) + \sum_{k=1}^{m} \frac{B_{2k}}{(2k)!}\left(f^{(2k-1)}(0) - f^{(2k-1)}(1)\right) - \frac{B_{2m+2}}{(2m+2)!} f^{(2m+2)}(\xi).$$

□

Zur Vervollständigung des Beweises der Euler-Maclaurinschen Summenformel müssen wir noch folgendes Lemma über das Vorzeichen von Bernoulli-Polynome und Bernoulli-Zahlen formulieren.

LEMMA 6.21: Vorzeichen von Bernoulli-Polynomen und -zahlen.

Sei $n \in \mathbb{N}_+$ und $\mathrm{B}_n \in \Pi_n$ und $B_n \in \mathbb{R}$ seien das entsprechende Bernoulli-Polynom bzw. die entsprechende Bernoulli-Zahl. Dann gelten die folgenden Vorzeicheneigenschaften:

(i) $(-1)^n \mathrm{B}_{2n-1}(x) > 0$ für $0 < x < \frac{1}{2}$,

(ii) $(-1)^n (\mathrm{B}_{2n}(x) - B_{2n}) > 0$ für $0 < x < 1$,

(iii) $(-1)^{n+1} B_{2n} > 0$.

Beweis. In der Hausaufgabe zu zeigen. □

Aus der Euler-Maclaurinschen Summenformel in Theorem 6.20 lassen sich direkt Verfahren zur numerischen Approximation des bestimmten Integrals einer Funktion f ableiten, wie das folgende Korollar zeigt.

KOROLLAR 6.22: Einfache Interpolationsformel mit Ableitungen.

Für $m = 1$ erhalten wir für eine Funktion $f \in C^1([a,b])$ folgende Approximation des bestimmten Integrals auf dem Intervall $[0,1] \subset \mathbb{R}$:

$$\begin{aligned} \int_0^1 f(t)\,\mathrm{d}t &\approx \frac{1}{2}(f(0)+f(1)) + \frac{B_2}{2!}(f'(0)-f'(1)) \\ &= \frac{1}{2}(f(0)+f(1)) + \frac{1}{12}(f'(0)-f'(1)) \end{aligned} \tag{6.82}$$

Sei $[a,b] \subset \mathbb{R}$ nun ein beliebiges Intervall. Wir führen nun eine Variablentransformation mit einer Funktion $\varphi\colon [0,1] \to [a,b]$ und $\varphi(t) \coloneqq t \cdot h + a$ durch, wobei $h \coloneqq b - a$ die Breite des Intervalls $[a,b] \subset \mathbb{R}$ bezeichnet. Dann gilt

$$\varphi(0) = 0 \cdot h + a = a, \quad \varphi(1) = 1 \cdot h + a = b, \quad \varphi'(t) = h.$$

Basierend der Approximationsregel (6.82) können wir mittels Integration durch Substitution herleiten, dass für eine beliebige Funktion $f \in C^1([a,b])$ gilt

$$\begin{aligned} \int_a^b f(x)\,\mathrm{d}x &= \int_{\varphi(0)}^{\varphi(1)} f(x)\,\mathrm{d}x = \int_0^1 f(t\cdot h + a)\cdot \underbrace{\varphi'(t)}_{=h}\,\mathrm{d}t \\ &= h \cdot \left(\frac{1}{2}(f(a)+f(b)) + \frac{1}{12}(h\cdot f'(a) - h\cdot f'(b))\right) \\ &= \frac{h}{2}(f(a)+f(b)) + \frac{h^2}{12}(f'(a)-f'(b)) \;=:\; M(h). \end{aligned}$$

Man beachte, dass der zusätzliche Faktor h als innere Ableitung von $f'(th+a)$ auftaucht. Mit Hilfe der Peanoschen Fehlerdarstellung in [FH07, Kapitel 3.2] lässt sich zeigen, dass für den numerischen Approximationsfehler der Integrationsregel $M(h)$ für eine Funktion $C^4([a,b])$ gilt:

$$\int_a^b f(x)\,\mathrm{d}x - M(h) = \frac{h^5}{720} f^{(4)}(\xi), \qquad \xi \in [a,b].$$

Somit liegt die Fehlerordnung der numerischen Integrationsformel $M(h)$ in der Größenordnung der Simpsonregel, obwohl nur Funktionsauswertungen und Ableitungen am Rand des Intervalls $[a,b] \subset \mathbb{R}$ verwendet werden.

6.2.2 Romberg-Verfahren

Wir wollen zunächst das bestimmte Integral einer Funktion $f \in C^{2m+2}([0,n])$ auf einem Intervall $[0,n] \subset \mathbb{R}$ für $n \in \mathbb{N}_+$ mit Hilfe der Euler-Maclaurinschen Summenformel approximieren. Dieses zerlegen wir in n Teilintervalle der Länge 1 und definieren eine passende Folge von Transformationen $\varphi_i \colon [0,1] \to [i,i+1], i = 0,\dots,n-1$ durch $\varphi_i(t) \coloneqq i+t = x$ betrachten, für die gilt:

$$\varphi_i(0) \;=\; i, \quad \varphi_i(1) \;=\; i+1, \quad \varphi_i'(t) = 1, \qquad i = 0,\dots,n-1.$$

Nun können wir das ursprüngliche bestimmte Integral umschreiben zu:

$$\int_0^n f(x)\,\mathrm{d}x \;=\; \sum_{i=0}^{n-1} \int_i^{i+1} f(t)\,\mathrm{d}t \;=\; \sum_{i=0}^{n-1} \int_0^1 f(i+t)\,\mathrm{d}t.$$

Nun lässt sich für jeden Summanden die Euler-Maclaurinschen Summenformel in Theorem 6.20 anwenden, so dass wir insgesamt erhalten

$$\int_0^n f(x)\,\mathrm{d}x \;=\; \sum_{i=0}^{n-1} \int_0^1 f(i+t)\,\mathrm{d}t \;=$$
$$\sum_{i=0}^{n-1} \left(\frac{f(i)}{2} + \frac{f(i+1)}{2} + \sum_{k=1}^{m} \frac{B_{2k}}{(2k)!} \left(f^{(2k-1)}(i) - f^{(2k-1)}(i+1) \right) - \frac{B_{2m+2}}{(2m+2)!} f^{(2m+2)}(i+\xi_i) \right)$$

Wir erkennen, dass sich die Terme in der Differenz durch die Summation praktischerweise wegheben, bis auf die beiden Randterme. Die $\xi_i \in [0,1]$ hängen vom i-ten Teilintervall ab und da f^{2m+2} nach Voraussetzung eine stetige Funktion ist lässt sich auf Grund des **Zwischenwertsatzes** ein $\xi \in [0,n]$ finden, so dass für den Mittelwert der Funktionswerte gilt:

$$\frac{1}{n} \sum_{i=0}^{n-1} f^{(2m+2)}(\xi_i) \;=\; f^{(2m+2)}(\xi) \quad \Rightarrow \quad \sum_{i=0}^{n-1} f^{(2m+2)}(\xi_i) \;=\; n f^{(2m+2)}(\xi).$$

Damit erhalten wir insgesamt für ein $\xi \in [0,n]$ die folgende Darstellung der Euler-Maclaurinschen Summenformel:

$$\int_0^n f(x)\,\mathrm{d}x \;=\; \frac{f(0)}{2} + f(1) + f(2) + \ldots + f(n-1) + \frac{f(n)}{2}$$
$$+ \sum_{k=1}^{m} \frac{B_{2k}}{(2k)!} \left(f^{(2k-1)}(0) - f^{(2k-1)}(n) \right) - n \cdot \frac{B_{2m+2}}{(2m+2)!} f^{(2m+2)}(\xi).$$

Für ein beliebiges Intervall $[a,b] \subset \mathbb{R}$ mit äquidistanten Stützpunkten $x_i = a + ih, i = 0,\dots,n$ mit Schrittweite $h = \frac{b-a}{n}$ und einer Funktion $f \in C^{2m+2}[a,b])$ erhalten wir nun durch Variablentransformation mit $\varphi \colon [0,n] \to [a,b]$ und $\varphi(x) \;=\; th + a$ die Darstellung

$$\int_a^b f(x)\,\mathrm{d}x \;=\; \overbrace{h \cdot \left(\frac{f(a)}{2} + f(a+h) + f(a+2h) + \ldots + f(a+(n-1)h) + \frac{f(b)}{2} \right)}^{=T_1(h)}$$
$$+ \sum_{k=1}^{m} h^{2k} \frac{B_{2k}}{(2k)!} \left(f^{(2k-1)}(a) - f^{(2k-1)}(b) \right) - h^{2m+2} \cdot \frac{B_{2m+2}}{(2m+2)!} (b-a) f^{(2m+2)}(\xi).$$

Hierbei bezeichnet $T_1(h) := T(h)$ die **zusammengesetzte Trapezregel** aus Abschnitt 6.1.3 mit

$$T(h) \;:=\; \frac{h}{2}\left(f(a) + 2\sum_{i=1}^{n-1} f(x_i) + f(b)\right).$$

Stellen wir diese Darstellung des Integrals nach $T_1(h)$ um, so erhalten wir

$$T_1(h) \;=\; \int_a^b f(x)\,\mathrm{d}x + \sum_{k=1}^{m} c_{2k}h^{2k} + c_{2m+2}(b-a)f^{(2m+2)}(\xi)h^{2m+2}, \tag{6.83}$$

wobei die Konstanten $c_k \in \mathbb{R}$ definiert sind als $c_{2k} := \frac{B_{2k}}{(2k)!}\left(f^{(2k-1)}(b) - f^{(2k-1)}(a)\right)$ für $k = 1, \ldots, m+1$.

Folgendes Korollar bemerkt eine spezielle Eigenschaft der Trapezsumme für periodische Funktionen.

KOROLLAR 6.23: Trapezsumme für periodische Funktionen.

Sei $f \in C^{2m+2}(\mathbb{R})$ eine 2π-periodische Funktion, d.h., es gilt $f^{(2k)}(a) = f^{(2k)}(a + 2\pi)$ für $k = 0, \ldots, m$.
Dann ist die Trapezsumme $T(h)$ für eine feste Schrittweite $h = \frac{2\pi}{n}$ und einen Startpunkt $a \in \mathbb{R}$ eine numerische Approximation des bestimmten Integrals von f von Ordnung $\mathcal{O}(h^{2m+2})$, d.h.

$$T(h) \;=\; \int_a^{a+2\pi} f(x)\,\mathrm{d}x + \mathcal{O}(h^{2m+2}).$$

Es lässt sich sogar zeigen, dass für jede Funktion $f \in C^\infty(\mathbb{R})$ gilt:

$$\left|\int_a^{a+2\pi} f(x)\,\mathrm{d}x - T(h)\right| \;\xrightarrow{h\to 0}\; 0,$$

schneller als jede Potenz von h. Wir erkennen also, dass sich periodische, glatte Funktionen außerordentlich gut mit der zusammengesetzten Trapezregel integrieren lassen. Diese vereinfacht sich durch die Periodizität in diesem Fall zu:

$$T(h) \;=\; h\cdot\left(\frac{f(a)}{2} + \sum_{i=1}^{n-1} f(a+ih) + \underbrace{\frac{f(a+2\pi)}{2}}_{=f(a)}\right) \;=\; h\sum_{i=0}^{n-1} f(a+ih).$$

Bisher haben wir eine Diskretisierung des Intervalls $[a, b] \subset \mathbb{R}$ mit fester Schrittweite $h > 0$ und $n \in \mathbb{N}_+$ Stützpunkten betrachtet, für die gilt

$$h \;:=\; \frac{b-a}{n}, \quad x_i \;:=\; a + ih, \qquad i = 0, \ldots, n.$$

In diesem Abschnitt wollen wir Verfahren für die numerische Integration herleiten, die unterschiedliche Folgen von Schrittweiten $h_1 > h_2 > \ldots > h_k > 0$ kombinieren, um somit eine hohe Approximationsgenauigkeit zu erhalten.

Wir bezeichnen im Folgenden das bestimmte Integral der Funktion $f \in C^{2m+2}([a, b])$ mit

$$I \;:=\; \int_a^b f(x)\,\mathrm{d}x.$$

Sei die Schrittweite $h := b - a$ zunächst als die Breite des Intervalls gewählt. Dann gilt mit der Darstellung der Trapezsumme aus (6.83), dass gilt:

$$T_1(h) \;=\; I + c_1 h^2 + c_2 h^4 + \ldots + c_m h^{2m} + \mathcal{O}(h^{2m+2}).$$

Identitäten dieser Form nennt man auch eine Entwicklung von T_1 nach den Potenzen von h.

Wenn wir nun die Anzahl der Teilintervalle verdoppeln, das heißt anstatt $n \in \mathbb{N}_+$ verwenden wir $2n \in \mathbb{N}_+$ viele Teilintervalle für die Diskretisierung von $[a, b] \subset \mathbb{R}$, so erhalten wir für die Trapezsumme die Entwicklung

$$\begin{aligned} T_1\left(\frac{h}{2}\right) &= I + c_1\left(\frac{h}{2}\right)^2 + c_2\left(\frac{h}{2}\right)^4 + \ldots + c_m\left(\frac{h}{2}\right)^{2m} + \mathcal{O}(h^{2m+2}) \\ &= I + c_1 2^{-2} h^2 + c_2 2^{-4} h^4 + \ldots + c_m 2^{-2m} h^{2m} + \mathcal{O}(h^{2m+2}). \end{aligned}$$

Die Idee des sogenannten **Romberg-Verfahrens** ist es nun, durch eine geschickte Linearkombination von $T_1(h)$ und $T_1(\frac{h}{2})$ den ersten Term $c_1 h^2$ verschwinden zu lassen und somit ein Verfahren der Ordnung $\mathcal{O}(h^4)$ zu erhalten. Hierzu bestrachtet man nun die folgende Differenz:

$$2^2 \cdot T_1\left(\frac{h}{2}\right) - T_1(h) \;=\; (2^2 - 1)I + c_2(2^{-2} - 1)h^4 + \ldots + c_m(2^{2-2m} - 1)h^{2m} + \mathcal{O}(h^{2m+2})$$

Dies können wir wiederum nach dem bestimmten Integral I umstellen und erhalten so

$$I \;=\; \frac{1}{2^2 - 1}(2^2 T_1\left(\frac{h}{2}\right) - T_1(h)) - c_2\frac{2^{-2} - 1}{2^2 - 1}h^4 - \ldots - c_m\frac{2^{2-2m} - 1}{2^2 - 1}h^{2m} + \mathcal{O}(h^{2m+2}).$$

Wir erhalten also durch Linearkombination zweier zusammengesetzter Trapezregeln mit unterschiedlicher Schrittweite ein Verfahren höherer Ordnung, nämlich

$$T_2(h) \;:=\; \frac{1}{2^2 - 1}\left[2^2 \cdot T_1\left(\frac{h}{2}\right) - T_1(h)\right].$$

Durch sukzessives Anwenden dieser Idee lassen sich Formeln für die numerische Integration von noch höherer Ordnung herleiten. Sie hierzu im Allgemeinen $T_k(h)$ eine Formel der Ordnung $\mathcal{O}(h^{2k})$. Dann betrachten wir

$$\begin{aligned} T_k(h) &= I + c_k h^{2k} + c_{k+1} h^{2k+2} + \ldots + c_m h^{2m} + \mathcal{O}(h^{2m+2}), \\ T_k\left(\frac{h}{2}\right) &= I + 2^{-2k} \cdot c_k h^{2k} + \ldots + 2^{-2m} \cdot c_m h^{2m} + \mathcal{O}(h^{2m+2}). \end{aligned}$$

Durch passende Linearkombination erhalten wir wiederum

$$2^{2k} \cdot T_k\left(\frac{h}{2}\right) - T_k(h) \;=\; (2^{2k} - 1) \cdot I + (2^{-2} - 1) \cdot c_{k+1} h^{2k+2} + \ldots + (2^{2k-2m} - 1) \cdot c_m h^{2m} + \mathcal{O}(h^{2m+2}).$$

Mit

$$T_{k+1}(h) \;:=\; \frac{1}{2^{2k} - 1}\left[2^{2k} \cdot T_k\left(\frac{h}{2}\right) - T_k(h)\right]$$

erhalten wir also eine numerische Approximation des bestimmten Integrals durch

$$I \;=\; T_{k+1}(h) + \mathcal{O}(h^{2k+2}).$$

Die rekursive Abhängigkeit dieser Formeln lassen sich sehr übersichtlich in einem sogenannten **Romberg-Schema** darstellen wie in Tabelle 6.2 illustriert wird.

Im folgenden Beispiel wenden wir das Romberg-Verfahren zur numerischen Approximation eines bestimmten Integrals an und vergleichen die Genauigkeit der Integrationsformeln $T_k(h)$ für unterschiedliche $k \in \mathbb{N}$.

$$
\begin{array}{c|ccccccc}
h & T_1(h) & & & & & & \\
 & & \searrow & & & & & \\
\frac{h}{2} & T_1\left(\frac{h}{2}\right) & \rightarrow & T_2(h) & & & & \\
 & & \searrow & & \searrow & & & \\
\frac{h}{4} & T_1\left(\frac{h}{4}\right) & \rightarrow & T_2\left(\frac{h}{2}\right) & \rightarrow & T_3(h) & & \\
 & & \searrow & & \searrow & & \searrow & \\
\frac{h}{8} & T_1\left(\frac{h}{8}\right) & \rightarrow & T_2\left(\frac{h}{4}\right) & \rightarrow & T_3\left(\frac{h}{2}\right) & \rightarrow & T_4(h)
\end{array}
$$

Tabelle 6.2: Illustration der rekursiven Abhängigkeit von numerischen Integrationsverfahren basierend auf der Trapezsumme mit unterschiedlichen Genauigkeiten in einem Romberg-Schema.

BEISPIEL 6.24: Romberg-Integrationsverfahren.

Es sei $f(x) \coloneqq e^x$ und wir wollen das bestimmte Integral

$$I \coloneqq \int_0^1 f(x)\,\mathrm{d}x \approx 1.718281828$$

mit Hilfe des Romberg-Integrationsverfahrens für unterschiedliche Integrationsformeln $T_k(h)$ mit $k = 1, \ldots, 4$ berechnen.
Tragen wir die Ergebnisse in eine Tabelle ein, so erhalten wir:

h	T_1	T_2	T_3	T_4
1	1.859140914			
$\frac{1}{2}$	1.753931092	1.718861151		
$\frac{1}{4}$	1.727221904	1.718318841	1.718282687	
$\frac{1}{8}$	1.720518592	1.718284155	1.718281842	1.718281829

BEMERKUNG 6.25 (Auslöschung beim Romberg-Verfahren). Da die Trapezsummen $T_k(h)$ und $T_k(\frac{h}{2})$ beide für große $k >> 1$ und kleine Schrittweiten h jeweils gute Approximationen für das bestimmte Integral $I = \int_a^b f(x)dx$ liefern, kommt bei der Linearkombination dieser Formeln zu numerischen Rundungsfehlern durch Auslöschung (siehe Abschnitt 2.2). Daher lohnt es sich im Allgemeinen nicht über $k = 6$ hinaus die Schrittweiten zu verfeinern.
Alternativ lässt sich anstatt der ursprünglichen Folge von Schrittweiten

$$h_0 = b - a, \quad h_1 = \frac{h_0}{2}, \quad \ldots, \quad h_k = \frac{h_{k-1}}{2}, \qquad k = 2, 3, \ldots$$

aus dem Romberg-Verfahren zu verwenden, kann man die langsam fallendere *Bulirsch-Folge* wie folgt verwenden:

$$h_0 = b - a, \quad h_1 = \frac{h_0}{2}, \quad h_2 = \frac{h_0}{3}, \quad \ldots, \quad h_k = \frac{h_{k-2}}{2}, \quad h_{k+1} = \frac{h_{k-2}}{3} \qquad k = 3, 5, \ldots$$

△

Betrachten wir die zusammengesetzte Trapezregel aus Abschnitt 6.1.3 nochmal genauer, so stellt sich heraus, dass diese eine sehr praktische Eigenschaft für die numerische Berechnung des Integrals besitzt. Hat man nämlich bereits für eine Funktion $f \in C([a,b])$ eine Approximation $T(h)$ des bestimmten Integrals $\int_a^b f(x)\mathrm{d}x$ mittels der zusammengesetzten Trapezregel ermittelt und ist mit der Genauigkeit noch nicht zufrieden, so kann man bei einer Verdopplung der Teilintervalle von $[a,b]$ auf die **vorangegangene Berechnung zurückgreifen** für die Berechnung der neuen Approximation $T(\frac{h}{2})$, denn es gilt mit $h := \frac{b-a}{n}$ für $n \in \mathbb{N}$:

$$\begin{aligned} T\left(\frac{h}{2}\right) &= \sum_{i=0}^{2n-1} \frac{h}{4}(f(x_i) + f(x_{i+1})) \\ &= \underbrace{\frac{1}{2}\sum_{i=0}^{n-1} \frac{h}{2}(f(x_{2i}) + f(x_{2i+2}))}_{=T(h)} + \sum_{i=1}^{n-1} \frac{h}{2}(f(x_{2i-1}) + f(x_{2i+1})) \\ &= \frac{T(h)}{2} + \sum_{i=1}^{n-1} \frac{h}{2}(f(x_{2i-1}) + f(x_{2i+1})). \end{aligned}$$

Diese Beobachtung ist einer der Gründe, warum das oben eingeführte *Romberg-Verfahren* numerisch effizient berechnet werden kann.

6.3 Gauss Quadratur

Zuletzt wollen wir uns mit einem weiteren Verfahren für die numerische Integration beschäftigen, bei der die Stützstellen nicht äquidistant gewählt werden, sondern so, dass man einen möglichst hohen Exaktheitsgrad erzielt (siehe Bemerkung 6.6). Hierbei verzichtet man bei der Diskretisierung des Intervalls $[a,b] \subset \mathbb{R}$ also inbesondere auf eine fixe Schrittweite $h = \frac{b-a}{n}$.

In diesem Abschnitt beschäftigen wir uns, anders als zuvor, mit einem **gewichteten Integral** der Form

$$I(f) := \int_a^b f(x)\sigma(x)\,\mathrm{d}x,$$

wobei $\sigma\colon [a,b] \to \mathbb{R}_{\geq 0}$ eine gegebene nichtnegative Gewichtsfunktion ist. Wir lassen hierbei auch unendliche Integrale wie $[0,\infty]$ oder $[-\infty,\infty]$ zu. Die Gewichtsfunktion σ soll folgende Voraussetzungen erfüllen:

(i) $\sigma(x)$ ist für alle $x \in [a,b]$ nichtnegativ und messbar,

(ii) Alle Momente der Form

$$\mu_k := \int_a^b x^k \sigma(x)\,\mathrm{d}x, \qquad k = 0,1,\ldots$$

existieren und sind endlich,

(iii) Für jedes Polynom $p(x)$ mit $\int_a^b \sigma(x)p(x)\mathrm{d}x = 0$ und $p(x) \geq 0$ für alle $x \in [a,b]$ gilt schon $p(x) \equiv 0$.

Man kann zeigen, dass jede Gewichtsfunktion σ, die auf dem Intervall $[a,b]$ positiv und stetig ist, die obigen Voraussetzungen erfüllt. Für den Spezialfall $\sigma(x) \equiv 1$ für alle $x \in [a,b]$ erhalten wir das herkömmliche bestimmte Integral.

Wir betrachten zunächst wieder Quadraturformeln aus Definition 6.1 der Form

$$G_n(f) \;=\; \sum_{i=1}^{n} w_i f(x_i) \;\approx\; \int_a^b f(x)\sigma(x)\,\mathrm{d}x \;=\; I(f), \tag{6.84}$$

mit Gewichten $w_i \in \mathbb{R}, i = 1, \ldots, n$. Im Gegensatz zu den bisher betrachteten Quadraturformeln nehmen wir hier jedoch keine äquidistanten Stützstellen $x_i \in [a, b], i = 1, \ldots, n$ an, so dass diese frei gewählt werden können. Somit hat die Quadraturformel (6.84) insgesamt $2n$ Freiheitsgrade.

Nach Bemerkung 6.6 wissen wir bereits, dass die interpolatorischen Quadraturformeln nach Newton-Cotes mit $n \in \mathbb{N}_+$ Stützstellen ein Polynom $n$ten Grades exakt integrieren. Hierdurch motiviert wollen wir in diesem Abschnitt spezielle Quadraturformeln G_n, genannt **Gauss-Quadraturen**, herleiten, die einen Exaktheitsgrad von $2n-1$ besitzen, d.h., für alle Polynome $p \in \Pi_{2n-1}$ soll folgende Bedingung gelten:

$$G_n(p) \;=\; I(p).$$

Dies führt zu genau $2n$ Bedingungen für unsere $2n$ Freiheitsgrade in (6.84).

Der folgende Satz besagt, dass diese Forderung schon maximal ist.

THEOREM 6.26: Maximaler Exaktheitsgrad für Quadraturformeln.

Es gibt keine Quadraturformel G_n wie in (6.84) mit $2n$ Freiheitsgraden, die vom Exaktheitsgrad $2n$ ist.

Beweis. Nehmen wir an G_n sei eine Quadraturformel mit $2n$ Freiheitsgraden und Exaktheitsgrad $2n$. Dann gilt offensichtlich $G_n(p) = I(p)$ für alle Polynome $p \in \Pi_{2n}$. Dies würde also auch gelten für das spezielle Polynom

$$p(x) \;:=\; \prod_{i=1}^{n} (x - x_i)^2,$$

für die Stützstellen $x_i \in [a, b], i = 1, \ldots, n$.

Im Allgemeinen gilt offensichtlich $I(p) \neq 0$, jedoch ist

$$G_n(p) \;=\; \sum_{i=1}^{n} w_i p(x_i) \;=\; \sum_{i=1}^{n} w_i \prod_{j=1}^{n} (x_i - x_j)^2 \;=\; 0.$$

Damit ist also $G_n(p) \neq I(p)$ und somit kann es keine Quadraturformel der Form (6.84) mit $2n$ Freiheitsgraden geben, die Exaktheitsgrad $2n$ besitzt. □

Es stellt sich nun die Frage, wie man eine Quadraturformel G_n finden kann, die den Exaktheitsgrad $2n-1$ besitzt. Eine theoretische Grundlage für die Beantwortung dieser Frage liefert uns das folgende Theorem.

THEOREM 6.27: Exaktheitsgrad einer Quadraturformel.

Seien $n, m \in \mathbb{N}_+$. Eine Quadraturformel G_n der Form (6.84) für das gewichtete Integral hat den Exaktheitsgrad $(n + m - 1) \in \mathbb{N}_+$, genau dann wenn sie die folgenden Bedingungen erfüllt:

(i) G_n ist eine interpolatorische Quadraturformel,

(ii) das zu den n Stützstellen $x_i \in [a,b], i = 1,\dots,n$ gehörende Polynom $\omega_n \in \Pi_n$ der Form $\omega_n(x) = \prod_{i=1}^n (x - x_i)$ erfüllt die folgende *Orthogonalitätsbedingung*

$$\int_a^b \omega_n(x) p(x) \sigma(x)\,\mathrm{d}x \;=\; 0 \tag{6.85}$$

für alle Polynome $p \in \Pi_{m-1}$ vom Grad kleiner oder gleich $m-1$.

Beweis. Wir zeigen beide Richtungen der Aussage des Theorems getrennt.

„⇒“: Sei zunächst $G_n(f) = \sum_{i=1}^n w_i f(x_i)$ eine Quadraturformel vom Exaktheitsgrad $n+m-1$ mit den Stützstellen $x_i \in [a,b], i = 1,\dots,n$. Wir betrachten nun ein Polynom $P(x) := \omega_n(x)p(x)$ für $\omega_n \in \Pi_n$ mit $w_n(x) := \prod_{i=1}^n (x - x_i)$ und $p \in \Pi_{m-1}$ sei ein beliebiges Polynom vom Grad kleiner oder gleich $m-1$. Dann ist klar, dass $P(x)$ ein Polynom vom Grad kleiner gleich $n+m-1$ ist. Da wir angenommen hatten, dass die Quadraturformel den Exaktheitsgrad $n+m-1$ besitzt gilt per Definition $I(P) = G_n(P)$ und somit

$$G_n(P) \;=\; I(P) \;=\; \int_a^b P(x)\sigma(x)\,\mathrm{d}x.$$

Wir sehen also, dass G_n eine interpolatorische Quadratur ist. Außerdem gilt:

$$\int_a^b \omega_n(x)p(x)\sigma(x)\,\mathrm{d}x \;=\; I(P) \;=\; G_n(P) \;=\; \sum_{i=1}^n w_i P(x_i) \;=\; \sum_{i=1}^n w_i \cdot \omega_n(x_i)p(x_i) \;=\; 0,$$

da die $x_i \in [a,b], i = 1,\dots,n$ gerade die Nullstellen von ω_n sind. Also erfüllt das Polynom $\omega_n \in \Pi_n$ die Orthogonalitätsbedingungen für alle Polynome $p \in \Pi_{m-1}$.

„⇐“: Wir nehmen nun an, dass G_n eine interpolatorische Quadraturformel ist und das Polynom $\omega_n \in \Pi_n$ die Orthogonalitätsbedingung (6.85) für alle Polynome $p \in \Pi_{m-1}$ erfüllt.

Sei $p \in \Pi_n$ ein beliebiges Polynom vom Grad $n+m-1$, so können wir p mittels **Polynomdivision** in der Form

$$p \;=\; \omega_n q_1 + q_2$$

schreiben, mit einem Polynom $q_1 \in \Pi_{m-1}$ vom Grad kleiner oder gleich $m-1$ und einem Restpolynom $q_2 \in \Pi_{n-1}$ vom Grad kleiner oder gleich $n-1$. Da die interpolatorische Quadraturformel G_n Exaktheitsgrad n besitzt und somit Polynome vom Grad kleiner oder gleich n exakt integriert, folgt

$$G_n(q_2) \;=\; \sum_{i=1}^n w_i q_2(x_i) \;=\; \int_a^b q_2(x)\sigma(x)\,\mathrm{d}x \;=\; \int_a^b (p(x) - \omega_n q_1(x))\sigma(x)\,\mathrm{d}x.$$

Da wir die Orthogonalitätsbedingung (6.85) angenommen haben folgt

$$\begin{aligned} G_n(p) &= \sum_{i=1}^{n} w_i p(x_i) = \sum_{i=1}^{n} w_i(\omega_n(x_i)q_1(x_i) + q_2(x_i)) = \underbrace{\sum_{i=1}^{n} w_i\omega_n(x_i)q_1(x_i)}_{=0} + \sum_{i=1}^{n} w_i q_2(x_i) \\ &= \sum_{i=1}^{n} w_i q_2(x_i) = \int_a^b (p(x) - \omega_n(x)q_1(x))\sigma(x)\,\mathrm{d}x \\ &= \int_a^b p(x)\sigma(x)\,\mathrm{d}x - \underbrace{\int_a^b \omega_n(x)q_1(x)\sigma(x)\,\mathrm{d}x}_{=0} = \int_a^b p(x)\sigma(x)\,\mathrm{d}x. \end{aligned}$$

Dies bedeutet, dass die Quadraturformel G_n den Exaktheitsgrad $n+m-1$ hat. □

Das Theorem 6.27 sagt uns also, dass wir für einen Exaktheitsgrad $n+m-1$ die Nullstellen $x_i \in [a,b], i = 1,\ldots,n$ des Polynoms ω_n (die die Stützstellen der Quadraturformel G_n sind) so geschickt wählen müssen, dass die Orthogonalitätsbedingung für alle Polynome $p \in \Pi_{m-1}$ vom Grad kleiner oder gleich $m-1$ erfüllt wird. Wir wollen natürlich den nach Theorem 6.26 maximal möglichen Exaktheitsgrad von $2n-1$ erreichen, d.h. wir setzen $m = n$ in Theorem 6.27. Dann muss die Orthogonalitätsbedingung (6.85) entsprechend für alle Polynome $p \in \Pi_n$ vom Grad $n-1$ gelten.

Um die Gausschen Quadraturformeln herzuleiten wollen wir zunächst noch einige grundlegende Eigenschaften von Orthogonalpolynomen wiederholen.

DEFINITION 6.28: Menge der normierten Polynome.

Sei Π_n der Vektorraum aller reellen Polynome vom Grad kleiner oder gleich $n \in \mathbb{N}$. Dann definieren wir durch

$$\overline{\Pi}_n := \{p \in \Pi_n \,|\, p(x) = x^n + a_1 x^{n-1} + \ldots + a^{n-1}x + a_n\}$$

die **Menge aller normierten, reellen Polynome** vom Grad $n \in \mathbb{N}$.

Weiterhin betrachten wir den Raum $L^2([a,b])$ aller quadratisch integrierbaren Funktionen versehen mit dem Skalarprodukt

$$\langle f, g\rangle_\sigma := \int_a^b f(x)g(x)\sigma(x)\,\mathrm{d}x,$$

wobei $\sigma\colon [a,b] \to \mathbb{R}_+$ eine stetige, positive Gewichtsfunktion ist.

Mit Hilfe dieses Skalarprodukts können wir nun orthogonale Polynome auf $L^2[a,b]$ definieren.

DEFINITION 6.29: Orthogonale Funktionen.

Seien $f, g \in L^2([a,b])$ quadratisch integrierbare Funktionen für ein Intervall $[a,b] \subset \mathbb{R}$. Wir nennen f und g **orthogonal**, falls $\langle f, g\rangle_\sigma = 0$ gilt.

Das folgende Theorem beweist, dass es eindeutig bestimmte Orthogonalpolynome bezüglich einer gewählten Gewichtsfunktion σ gibt.

THEOREM 6.30: Existenz und Eindeutigkeit von Orthogonalpolynomen.

Es gibt für $n = 0, 1, \ldots$ eindeutig bestimmte Polynome $p_n \in \overline{\Pi}_n$ für die die folgende Orthogonalitätseigenschaft gilt:

$$\langle p_n, p_m \rangle_\sigma \;=\; 0, \qquad \text{für} \quad n \neq m.$$

Diese Orthogonalpolynome genügen der Rekursionformel

$$\begin{aligned} p_0(x) &\equiv 1, \\ p_{n+1}(x) &= (x - \delta_{n+1}) p_n(x) - \gamma_{n+1}^2 p_{n-1}(x) \qquad \text{für} \quad n \in \mathbb{N}, \end{aligned} \tag{6.86}$$

wobei wir $p_{-1}(x) \equiv 0$ definieren und

$$\begin{aligned} \delta_{n+1} &:= \langle x p_n, p_n \rangle_\sigma \,/\, \langle p_n, p_n \rangle_\sigma \qquad \text{für } n \in \mathbb{N}, \\ \gamma_{n+1}^2 &:= \begin{cases} 1, & \text{für } n = 0, \\ \langle p_n, p_n \rangle_\sigma \,/\, \langle p_{n-1}, p_{n-1} \rangle_\sigma & \text{für } n \in \mathbb{N}_+. \end{cases} \end{aligned}$$

Beweis. Siehe [FH07, Theorem 3.6.3]. □

Die Orthogonalpolynome $p_n \in \Pi_n$ erfüllen bereits die gewünschte Orthogonalitätseigenschaft aus (6.85), wie das folgende Korollar aussagt.

KOROLLAR 6.31.

Sei $p_n \in \overline{\Pi}_n$ ein Orthogonalpolynom. Dann gilt $\langle p_n, p \rangle_\sigma$ für alle Polynome $p \in \Pi_{n-1}$.

Beweis. Da die Orthogonalpolynome $p_n \in \overline{\Pi}_n$ eine Basis des Vektorraums der Polynome vom Grad kleiner oder gleich $n \in \mathbb{N}$ bilden, lässt sich jedes Polynom $p \in \Pi_{n-1}$ als Linearkombination von Orthogonalpolynomen $p_0, \ldots, p_{n-1} \in \overline{\Pi}_{n-1}$ darstellen. Daher gilt schon:

$$\langle p_n, p \rangle_\sigma \;=\; \langle p_n, \sum_{k=1}^{n-1} \alpha_k p_k \rangle_\sigma \;=\; \sum_{k=1}^{n-1} \alpha_k \underbrace{\langle p_n, p_k \rangle_\sigma}_{=0} \;=\; 0.$$

□

Folgendes Resultat beschreibt die Eigenschaften der Nullstellen der eindeutig bestimmten Orthogonalpolynome $p_n \in \overline{\Pi}_n$.

THEOREM 6.32: Nullstellen der Orthogonalpolynome.

Die Nullstellen $x_i, i = 1, \ldots, n$ des Orthogonalpolynoms $p_n \in \overline{\Pi}_n$ sind reell, einfach und liegen alle im offenen Intervall (a, b).

Beweis. Siehe [FH07, Theorem 3.6.10]. □

Nach Theorem 6.30 kann ein System von Orthogonalpolynomen bezüglich der Gewichtungsfunktion σ schrittweise durch die Rekursionsformel (6.86) mittels des Gram-Schmidt-Verfahrens berechnet werden. Mit Hilfe der Nullstellen des n-ten Orthogonalpolynoms $\omega_n \in \overline{\Pi}_n$

lassen sich so die optimalen Stützstellen für die Gauss-Quadratur G_n finden. Das Vorgehen ist im folgenden Algorithmus nochmal zusammengefasst.

ALGORITHMUS 6.33: Gausssches Integrationsverfahren.

Sei zunächst ein Intervall $[a,b] \subset \mathbb{R}$ und eine stetige, positive Gewichtungsfunktion $\sigma\colon [a,b] \to \mathbb{R}_+$ gegeben. Für die numerische Approximation des bestimmten, gewichteten Integrals $I(f) = \int_a^b f(x)\sigma(x)\mathrm{d}x$ führen wir folgende Schritte durch:

1. Berechne entsprechend zur Gewichtsfunktion σ das Orthogonalpolynom $\omega_n \in \overline{\Pi}_n$ vom Grad n.
2. Berechne die Nullstellen $x_i \in (a,b), i = 1,\dots,n$ von ω_n.
3. Verwende die bestimmten Nullstellen von ω_n als Stützstellen für die Integrationsformeln nach Newton-Cotes (siehe Abschnitt 6.1.1)

$$G_n(f) \;=\; \sum_{i=1}^{n} w_i f(x_i), \qquad w_k = \int_a^b L_k(x)\sigma(x)\,\mathrm{d}x.$$

Je nach Wahl der Gewichtsfunktion $\sigma\colon [a,b] \to \mathbb{R}_{\geq 0}$ und des Intervalls $[a,b] \subset \mathbb{R}$ erhält man ein anderes Orthogonalsystem von normierten Polynomen, wie die folgende Bemerkung zusammenfasst.

BEMERKUNG 6.34 (Unterschiedliche Systeme von Orthogonalpolynomen). Die folgende Tabelle fasst die wichtigsten Gewichtsfunktionen σ, die zugehörigen Intervalle $[a,b] \subset \mathbb{R}$ und die daraus resultierenden Systeme von Orthogonalpolynomen zusammen.

$[a,b]$	σ	Bezeichnung der Orthogonalpolynome
$[-1,1]$	1	Legendre-Polynome
$[-1,1]$	$(1-x^2)^{-\frac{1}{2}}$	Tschebyscheff-Polynome 1. Art
$[-1,1]$	$(1-x^2)^{\frac{1}{2}}$	Tschebyscheff-Polynome 2. Art
$[-1,1]$	$(1-x)^\alpha(1+x)^\beta$	Jacobi-Polynome
$(-\infty,\infty)$	$e^{-\frac{x^2}{2}}$	Hermite'sche Polynome
$(0,\infty)$	e^{-x}	Laguerre'sche Polynome

△

Das folgende Beispiel bestimmt die optimalen Stützstellen für eine Gauss-Quadratur mit $n = 2$.

BEISPIEL 6.35: Stützstellen einer Gauss-Quadratur.

Wir betrachten im folgenden Beispiel das Intervall $[0,1]$ und die Gewichtsfunktion $\sigma(x) \equiv 1$ für $x \in [0,1]$. Unser Ziel ist es die Nullstellen des Orthogonalpolynoms $\omega_n \in \overline{\Pi}_n$ für $n = 2$ zu berechnen.
Wegen der Rekursionsformel (6.86) gilt für das 0-te Orthogonalpolynom

$$\omega_0(x) \;=\; 1.$$

Da wir wissen, dass die Orthogonalpolynome normiert sind, können wir für ω_1 den Ansatz $\omega_1(x) = x + a$ machen Aus der Orthogonalitätsbedingung

$$\int_0^1 \underbrace{\omega_0(x)}_{\equiv 1}\,\omega_1(x)\,\underbrace{\sigma(x)}_{\equiv 1}\,\mathrm{d}x \;=\; 0$$

erhalten wir mit dem Hauptsatz der Integral- und Differentialrechnung

$$0 \;=\; \int_0^1 x + a\,\mathrm{d}x \;=\; \left[\frac{1}{2}x^2 + ax + c\right]_0^1 \;=\; \frac{1}{2} + a.$$

Daher muss $a = -\frac{1}{2}$ gelten und wir erhalten die Stützstelle $x_1 = \frac{1}{2}$ als Nullstelle des Orthogonalpolynoms $\omega_1(x) = (x - \frac{1}{2})$. Die entsprechende Gauss-Quadratur entspricht in diesem Fall der **Mittelpunktsregel** aus Beispiel 6.2.
Um das nächste Orthogonalpolynom $\omega_2 \in \overline{\Pi}_2$ zu bestimmen machen wir den Ansatz $\omega_2(x) = x^2 + bx + c$. Die unbekannten Parameter bestimmen wir aus den beiden Orthogonalitätsbedingungen

$$\int_0^1 \omega_0(x)\omega_2(x)\sigma(x)\,\mathrm{d}x \;=\; 0, \qquad \int_0^1 \omega_1(x)\omega_2(x)\sigma(x)\,\mathrm{d}x \;=\; 0.$$

Für die *erste Orthogonalitätsbedingung* erhalten wir

$$0 \;=\; \int_0^1 x^2 + bx + c\,\mathrm{d}x \;=\; \left[\frac{1}{3}x^3 + \frac{b}{2}x^2 + cx + d\right]_0^1 \;=\; \frac{1}{3} + \frac{b}{2} + c.$$

Für die *zweite Orthogonalitätsbedingung* erhalten wir

$$\begin{aligned} 0 \;&=\; \int_0^1 (x - \frac{1}{2})(x^2 + bx + c)\,\mathrm{d}x \;=\; \int_0^1 x^3 + (b - \frac{1}{2})x^2 + (c - \frac{b}{2})x - \frac{c}{2}\,\mathrm{d}x \\ &=\; \left[\frac{1}{4}x^4 + (\frac{b}{3} - \frac{1}{6})x^3 + (\frac{c}{2} - \frac{b}{4})x^2 - \frac{c}{2}x + d\right]_0^1 \\ &= \frac{1}{4} + \frac{b}{3} - \frac{1}{6} + \frac{c}{2} - \frac{b}{4} - \frac{c}{2} \;=\; \frac{1}{12} + \frac{b}{12} \;=\; \frac{1}{12}(1 + b). \end{aligned}$$

Hieraus können wir direkt folgern, dass $b = -1$ sein muss. Setzen wir dies in die erste Orthogonalitätsbedingung wieder ein, so erhalten wir $c = \frac{1}{2} - \frac{1}{3} = \frac{1}{6}$. Damit ist das zweite Orthogonalpolynom $\omega_2 \in \overline{\Pi}_2$ eindeutig bestimmt mit

$$\omega_2(x) \;=\; x^2 - x + \frac{1}{6}.$$

Mit Hilfe der p-q-Formel lassen sich nun die Nullstellen von ω_2 bestimmen als:

$$x_1 \;=\; \frac{1}{2} + \frac{1}{2\sqrt{3}}, \qquad x_2 \;=\; \frac{1}{2} - \frac{1}{2\sqrt{3}}.$$

BEMERKUNG 6.36 (Fehlerabschätzung für die Gauss-Quadratur). Da die Gauss-Quadratur einen Exaktheitsgrad von $2n-1$ hat lässt sich mit der Fehlerabschätzung für den Approximationsfehler der numerischen Interpolation in Theorem 5.20 für eine Funktion $f \in C^{2n}([a,b])$ zeigen, dass gilt

$$\left| G_n(f) - \int_a^b f(x)\sigma(x)\,\mathrm{d}x \right| \;\leq\; \frac{\|f^{(2n)}\|_\infty}{(2n)!}\,\frac{(b-a)^{2n+1}}{2^n}.$$

Natürlich kann die Gauss-Quadratur auch als summierte Quadraturformel angewandt werden (siehe Abschnitt 6.1.3) indem man das Intervall $[a,b]$ in mehrere Teilintervalle zerlegt und dort die Nullstellen der Orthogonalpolynome verwendet. △

Kapitel 7

Eigenwertprobleme

In diesem Kapitel beschäftigen wir uns mit Eigenwertproblemen. Für eine Matrix $A \in \mathbb{C}^{n\times n}$ betrachten wir das Problem

$$Ax = \lambda x$$

mit Eigenwert $\lambda \in \mathbb{C}$ und Eigenwert $x \in \mathbb{C}^n$. In manchen ist man nur an einzelnen Eigenwerten interessiert (z.B. die Bestimmung des größten Eigenwerts als Spektralradius um Konvergenz iterativer Verfahren abzusichern), in anderen an vielen (oder allen) Eigenwerten und Eigenvektoren (z.B. bei der Singulärwertzerlegung) und in manchen nur an einzelnen Eigenvektoren. Ein modernes Beispiel für den letzten Fall sind Ranking-Probleme wie sie z.B. Google für Suchergebnisse anwendet.

BEISPIEL 7.1: PageRank Algorithmus.

Wir betrachten $n \in \mathbb{N}$ Dokumente (oder Seiten) $p_1, \dots, p_n$ welche sich jeweils gegenseitig verlinken. Der Wert $\tilde{l}(p_i, p_j) \in \{0, 1\}$ beschreibt hier, ob die Seite p_i auf die Seite p_j verweist oder nicht. Für eine Seite p_i berechnen wir die Anzahl der Seiten auf die p_i verweist durch

$$d(p_i) = \sum_{j=1}^{n} \tilde{l}(p_i, p_j)$$

womit wir die normalisierte Größe $L_{i,j} := \tilde{l}(p_i, p_j)/d(p_i)$ definieren und so eine stochastische Matrix $L \in \mathbb{R}^{n,n}$ erhalten. Wir gehen hier davon aus, dass jede Seite auch Ausganglinks hat, d.h. dass heißt, dass $d(p_i) > 0, i = 1, \dots, n$. Wir wollen nun die Relevanz der Dokumente bestimmen, d.h. wir möchten ein Ranking erstellen, welches wir als Vektor $R \in \mathbb{R}_+^N$ modellieren. Der Eintrag R_i modelliert hier die Relevanz der Seite p_i. Dazu betrachten wir einen random Surfer, d.h. eine Person welche zufällig auf Links klickt und wollen die Wahrscheinlichkeit bestimmen, dass der random Surfer auf Seite p_i landet. Für ein gegebene Verteilung nach k Klicks $R^k \in \mathbb{R}^n$ mit $\sum_{i=1}^n R_i^k = 0$ erhalten wir die Verteilung nach $k + 1$ Klicks durch

$$R^{k+1} = LR^k$$

was einen sogenannte Markov Chain beschreibt. Der stationäre Zustand $R \in \mathbb{R}^n$ dieses Prozesses ist durch die Gleichung

$$R = L^T R$$

gegeben. Für den klassichen Google PageRank Algorithmus [BP98] geht man zusätzlich davon aus, dass der random Surfer mit einer Wahrscheinlichkeit von $1 - d$ zu einer zufälligen Seite springt unabhängig von den Links auf der momentanen Seite, wobei $d \in (0, 1)$ der sogenannte Dämpfungsfaktor ist. In diesem Fall erhält man die Gleichung

$$R = \underbrace{\left(\frac{1-d}{n} I + d\, L^T\right)}_{:=P^T} R$$

wobei $P \in \mathbb{R}^{n,n}$ die sogenannte Google-Matrix ist. Da auch P eine stochastische Matrix ist und somit $\lambda =!$ als größten Eigenwert hat, suchen wir also einen Eigenvektor zum größten Eigenwert.

BEISPIEL 7.2: Ranking im Sport.

Wir betrachten eine Liga mit n Teams und modellieren mit $s_i \in \mathbb{R}^n_+$ die Spielstärke des i-ten Teams. Ist A nun eine Matrix von Spielergebnissen,

$$A_{ij} = \begin{cases} 0 & \text{falls } i \text{ gegen } j \text{ verloren hat,} \\ 1 & \text{falls } i \text{ gegen } j \text{ unentschieden oder nicht gespielt hat,} \\ 2 & \text{falls } i \text{ gegen } j \text{ gewonnen hat.} \end{cases}$$

dann ist der Rang durch $r = As$ gegeben. Für eine faire Bewertung sollte $r = \lambda s$ gelten, also können wir die Spielstärke aus dem Eigenwertproblem $\lambda s = As$ bzw. das Ranking aus

$$\lambda r = Ar$$

bestimmen. Hier ist natürlich nur ein nicht-negativer Eigenwert und Eigenvektor interessant, welcher nach dem Satz von Perron–Frobenius existiert (und der betragsmäßig größte ist). Wir interessieren uns also für den Eigenvektor zum größten Eigenwert.

7.1 Potenzmethode und inverse Iteration

Wir beginnen mit einer Methode zur Berechnung des größten Eigenwerts einer Matrix A, die sogenannte Potenzmethode oder Vektoriteration. Betrachten wir dazu zunächst eine symmetrische Matrix $A \in \mathbb{R}^{n \times n}$, dann können wir die Diagonalisierung $A = V \Lambda V^T$ verwenden. Hier ist V eine orthogonale Matrix, deren Spalten die Eigenvektoren von A sind, und Λ ist eine Diagonalmatrix bestehend aus den Eigenwerten. Berechnen wir für ein x^0 die Anwendung

$Ax^0 = VDV^T x^0$, so können wir dies als

$$Ax^0 = \sum_{j=1}^{n} \lambda_j v_j (v_j^T x^0)$$

schreiben und sehen, dass die Komponenten von x_0 bezüglich der Orthonormalbasis (v_j) mit λ_j verstärkt werden, am meisten natürlich der zum größten Eigenwert gehörige Anteil. Dies induziert die Idee einfach immer wieder mit der Matrix A zu multiplizieren, um am meisten den Anteil zum größten Eigenwert zu verstärken. Um keine Divergenz zu erhalten, muss dann natürlich noch normiert werden. Diese Idee lässt sich auf allgemeine Matrizen übertragen und führt auf die sogenannte Potenzmethode.

ALGORITHMUS 7.3: Potenzmethode.

Für eine Matrix $A \in \mathbb{C}^{n\times n}$ und gegebener Anfangsvektor $x_0 \in \mathbb{C}^n$ ist die Potenzmethode oder Vektoriteration durch die Vorschrift

$$x^{k+1} = \frac{1}{\|Ax^k\|} Ax^k.$$

gegeben.

Wir zeigen im Folgenden, dass die Potenzmethode tatsächlich einen Eigenwert zum größten Eigenvektor liefert. Wir nutzten im folgenden eine Zerlegung des Raums in die Eigenvektoren $v_1, \ldots, v_n$, s.d. sich der Startvektor durch

$$x^0 = \sum_{i=1}^{n} \alpha_i v_i$$

ausdrücken lässt. Hierbei, gehen wir davon aus, dass

$$\alpha_1 \neq 0 \tag{7.87}$$

gilt, also dass der Anteil zum betragsmäßig größten Eigenwert nicht null ist.

LEMMA 7.4: Konvergenz der Potenzmethode.

Es sei $A \in \mathbb{C}^{n\times n}$ eine Matrix mit Eigenwerten $\lambda_1, \ldots, \lambda_n \in \mathbb{C}$ mit $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n|$. Zu einem gegebenen Startvektor $x_0 \in \mathbb{C}^n$ der Gleichung (7.87) erfüllt konvergiert die Vektoriteration in Algorithmus 7.3 gegen einen Eigenvektor zum Eigenwert λ_1.

Beweis. Wir gehen vereinfachend von einer reell diagonalisierbaren Matrix $A \in \mathbb{R}^{n\times n}$ aus, der allgemeine Fall lässt sich analog mit der Jordanschen Normalform zeigen. In diesem Fall existiert für $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ eine Matrix $V = (v_1|\ldots|v_n)$, wobei $v_i \in \mathbb{R}^n$ ein Eigenvektor zum Eigenwert λ_i ist, s.d.

$$A = V\Lambda V^{-1}.$$

Insbesondere bilden die v_i eine Basis des $\mathbb{R}^n$ und somit existieren Koeffizienten α_i, s.d. $x^0 = \sum_{i=1}^n \alpha_i v_i$. Für die k-te Iterierte der Potenzmethode haben wir

$$x^{k+1} = \frac{\|A^k x^0\|^k}{A} x^0$$

wobei wir sehen, dass

$$\begin{aligned} A^k x_0 &= (V\Lambda V^{-1})^k x_0 = V\Lambda^k V^{-1} x_0 \\ &= (V\Lambda^k) \sum_{i=1}^n \alpha_i V^{-1} v_i \\ &= (V\Lambda^k) \sum_{i=1}^n \alpha_i e_i \\ &= V\lambda_1^k \left(\alpha_1 e_1 + \sum_{i=2}^n \alpha_i \left(\frac{\lambda_i}{\lambda_1} \right)^k e_i \right) \\ &= \alpha_1 \lambda_1^k v_1 + \sum_{i=2}^n \alpha_i \left(\frac{\lambda_i}{\lambda_1} \right)^k v_i \end{aligned}$$

Wegen $\left(\frac{\lambda_i}{\lambda_1}\right)^k \to 0$ für $k \to \infty, i > 1$ folgt dann

$$\frac{1}{\|A^k x^0\|} A^k x_0 \to \frac{1}{\|v_1\|} v_1$$

für $k \to \infty$. □

BEMERKUNG 7.5. Um z.B. den Eigenvektor zum zweitgrößten Eigenwert zu berechnen müsste beim Startwert in Gleichung (7.87) $\alpha_1 = 0, \alpha_2 \neq 0$ gelten. Falls A eine symmetrische Matrix ist könne wir dazu x^0 orthogonal zu v_1 wählen. Mit dieser Strategie können wir alle Eigenvektoren berechnen. △

Um den Eigenwert zu berechnen betrachtet man den sogenannten **Rayleigh-Quotienten**

$$\lambda_1^k := \frac{(x^k)^T A x^k}{(x^k)^T x^k}$$

wofür aus Lemma 7.4 folgendes Korollar erhalten.

KOROLLAR 7.6.

Unter den Voraussetzungen von Lemma 7.4 konvergiert der Rayleigh-Quotient gegen den betragsmäßig größten Eigenwert λ_1.

BEISPIEL 7.7.

Wir betrachten die Matrix

$$A = \begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix}$$

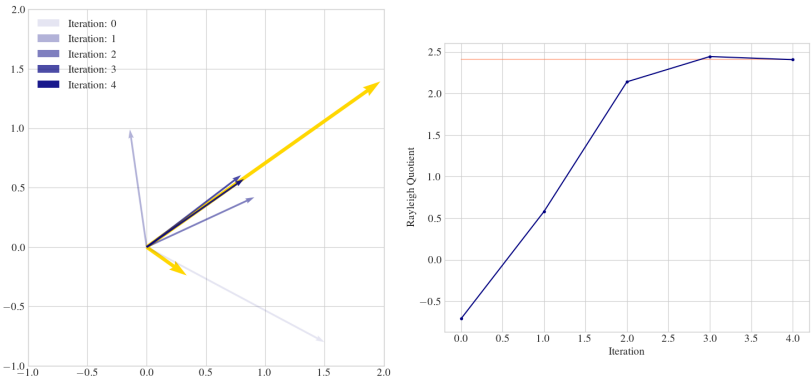


Abbildung 7.1: Die ersten vier Iterationen der Potenzmethode aus Beispiel 7.7 in blau. In gelb sind die Eigenvektoren skalirt mit den Eigenwerten dargestellt. Im rechten Plot ist das verhalten des Rayleigh-Quotienten abgebildet, wobei der eigentliche Wert des größten Eigenwerts in hell-rot dargestellt ist.

und wenden darauf die Potenzmethode an. Die ersten vier Iterationen sind in Abb. 7.1 visualisiert.

Wollen wir direkt einen anderen Eigenwert von A berechnen, so kann dazu die inverse Iteration genutzt werden. Grundlage dafür ist folgendes Resultat.

LEMMA 7.8.

Seien $A \in \mathbb{C}^{n\times n}$ und $\lambda \in \mathbb{C}$ gegeben. Sei λ_i ein Eigenwert von A mit minimalem Abstand zu λ, d.h.

$$|\lambda_i - \lambda| \leq |\lambda_j - \lambda|$$

für alle Eigenwerte λ_j von A. Dann ist $(\lambda - \lambda_i)^{-1}$ ein betragsmäßig größter Eigenwert von

$$B = (\lambda I - A)^{-1}.$$

Beweis. Wir sehen sofort, dass μ_j ein Eigenwert von B mit Eigenvektor x_j, wenn $\lambda_j = \lambda - \frac{1}{\mu_j}$ ein Eigenwert von A mit dem selben Eigenvektor ist und umgekehrt. Die Eigenwerte von B sind also $\frac{1}{\lambda-\lambda_j}$ und dies ist betragsmässig maximal, wenn $|\lambda - \lambda_j|$ minimal ist. □

Damit sehen wir, dass die Iteration

$$x^{k+1} = \frac{1}{\|y^k\|} y^k, \qquad y^k = (\lambda I - A)^{-1} x^k \tag{7.88}$$

unter analogen Bedingungen gegen einen Eigenvektor zum Eigenwert $\frac{1}{\lambda-\lambda_j}$ konvergiert, wobei λ_i der zu λ nächstgelegene Eigenwert ist. Die Matrix dabei ist auch invertierbar, falls λ kein Eigenwert ist, in diesem Fall wären wir ohnehin schon im ersten Schritt fertig. Um ein effizientes Verfahren zu erhalten, sollte man für großes n natürlich die Berechnung der inversen Matrix vermeiden und statt dessen in jedem Schritt das lineare Gleichungssystem

$$\lambda y^k - A y^k = x^k$$

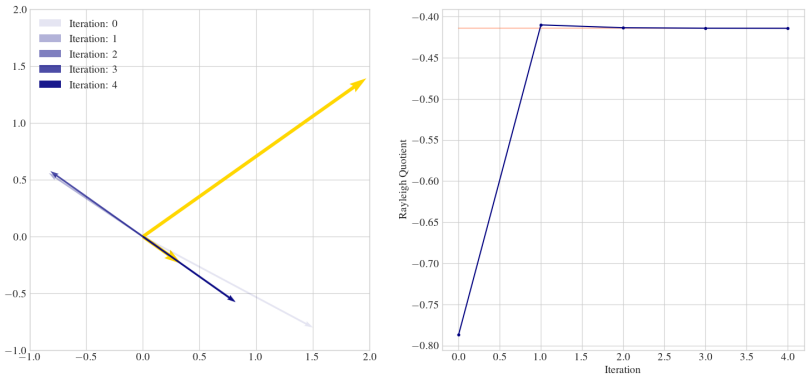


Abbildung 7.2: Die inverse Iteration für die Matrix aus Beispiel 7.7 mit $\lambda = -1$.

lösen. Hat man eine hinreichend gute Näherung für den Eigenvektor x^k, kann eine Approximation des Eigenwertes mithilfe eines modifizierten Rayleigh-Quotienten berechnet werden. Denn aus

$$\frac{1}{\lambda - \lambda_j} \approx \mu_j^k =: \frac{(x^k)^T (\lambda I - A)^{-1} x^k}{(x^k)^T x^k}$$

folgt sofort $\lambda_j \approx \lambda - \frac{1}{\mu_j^k}$.

7.2 Der Satz von Gerschgorin

In vielen Fällen benötigt man zumindest grobe Abschätzungen über die Lage von Eigenwerten, etwa um den Spektralradius abzuschätzen oder ein vernünftiges λ für die inverse Iteration zu wählen. Ein zentrales Hilfsmittel sind die Gerschgorin-Kreise.

DEFINITION 7.9: Gerschgorin-Kreise.

Für eine Matrix $A \in \mathbb{C}^{n \times n}$ sind die zeilenweisen Gerschgorin-Kreise für $i = 1, \ldots, n$ definiert durch

$$R_i := \{\lambda \in \mathbb{C} \mid |\lambda - A_{ii}| \leq \sum_{j \neq i} |a_{ij}|\}.$$

Die spaltenweisen Gerschgorin-Kreise definiert man analog durch

$$S_i := \{\lambda \in \mathbb{C} \mid |\lambda - A_{ii}| \leq \sum_{j \neq i} |A_{ji}|\}.$$

Der Satz von Gerschgorin zeigt, dass alle Eigenwerte in diesen Kreisen liegen

THEOREM 7.10: Satz von Gerschgorin.

Sei $A \in \mathbb{C}^{n\times n}$ und die Gerschgorin Kreise definiert wie oben. Dann gilt für alle Eigenwerte λ von A

$$\lambda \in \left(\bigcup_{j=1}^{n} R_j\right) \cap \left(\bigcup_{j=1}^{n} S_j\right).$$

Beweis. Es genügt zu zeigen, dass $\lambda \in \bigcup_{j=1}^{n} R_j$ gilt, denn da die Eigenwerte von A jenen von A^T entsprechen, folgt dann sofort auch $\lambda \in \bigcup_{j=1}^{n} S_j$. Wir schreiben $A = D + E$ mit $D = \mathrm{diag}(A_{11}, \ldots, A_{nn})$ einer Diagonalmatrix $E = A - D$ eine Matrix mit Nulldiagonale. Sei λ ein Eigenwert und v der zugehörige Eigenvektor, so folgt aus $Av = \lambda v$ auch

$$(D - \lambda I)v = Dv - \lambda v = -Ev.$$

Fall I.: Falls $\lambda = A_{ii}$ für ein $i \in \{1, \ldots, n\}$, dann gilt $\lambda \in R_i$ und die Aussage ist gezeigt.
Fall II.: Andernfalls ist $D - \lambda I$ invertierbar und somit

$$v = (\lambda I - D)^{-1} E v,$$

also

$$v_i = \sum_{j\neq i} \frac{A_{ij}}{\lambda - A_{ii}} v_j.$$

Daraus folgt

$$\max_i |v_i| \leq \max_i \sum_{j\neq i} \frac{|A_{ij}|}{|\lambda - A_{ii}|} |v_j| \leq \max_i \sum_{j\neq i} \frac{|A_{ij}|}{|\lambda - A_{ii}|} \max_k |v_k|$$
$$\Rightarrow 1 \leq \max_i \sum_{j\neq i} \frac{|A_{ij}|}{|\lambda - A_{ii}|}$$

Es existiert also $i \in \{1, \ldots, n\}$, s.d.

$$|\lambda - A_{ii}| \leq \sum_{j\neq i} |A_{ij}|$$

. □

Der Satz von Gerschgorin liefert auch sofort ein interessantes Resultat über eine Klasse von Matrizen, die in der Numerik eine wichtige Rolle spielen.

KOROLLAR 7.11.

Sei $A \in \mathbb{C}^{n\times n}$ strikt diagonaldominant, d.h.

$$|A_{ii}| > \sum_{j\neq i} |A_{ij}|$$

für alle i. Dann liegt $\lambda = 0$ in keinem Gerschgorin-Kreis, d.h. A ist invertierbar.

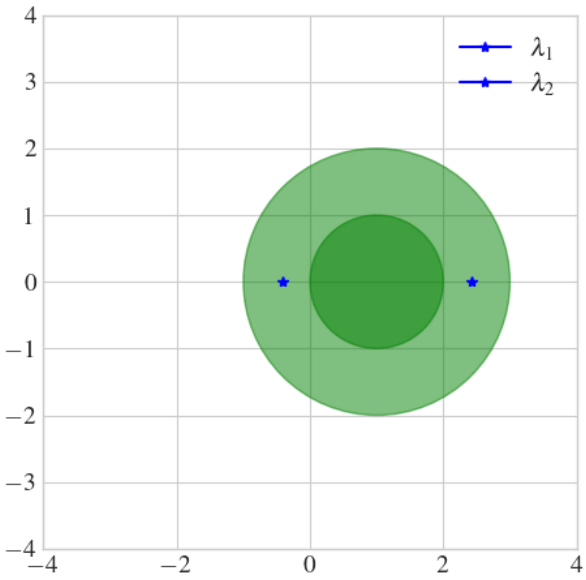


Abbildung 7.3: Die Gerschgorin-Kreise für die Matrix aus Beispiel 7.7.

7.3 Die QR-Iteration

Zum Abschluss wollen wir noch kurz ein Verfahren diskutieren, das alle Eigenwerte einer Matrix auf einmal berechnen kann. Hierbei wollen wir A durch Transformationen auf obere Dreicksstruktur oder zumindest Blockdreickstruktur bringen, um die Eigenwerte von der Diagonale ablesen zu können. Grundidee dabei ist, dass orthogonale Ähnlichkeitstransformationen die Eigenwerte einer Matrix unverändert lassen. Gilt $Ax = \lambda x$, so folgt für $y = Qx$ auch

$$QAQ^T y = \lambda y$$

wegen $Q^T Q = I$. Es ist also naheliegend einen Zusammenhang zur QR-Zerlegung einer Matrix zu nutzen, die wir stabil mit dem Householder-Verfahren berechnen können. Haben wir eine QR-Zerlegung $A = QR$ gegeben, so folgt

$$Q^T AQ = Q^T(QR)Q = RQ$$

was auf den QR-Algorithmus führt.

ALGORITHMUS 7.12.

Wir starten mit $A^0 = A$ mit einer gegebenen Matrix $A \in \mathbb{C}^{n\times n}$. Im k-ten Schritt berechnen wir die QR-Zerlegung $A^{(k)} = Q^{(k)}R^{(k)}$ und setzten dann

$$A^{(k+1)} := R^{(k)}Q^{(k)} = Q^{(k)}A^{(k)}Q^{(k)}.$$

Unter gewissen Voraussetzungen lässt sich zeigen, dass die QR-Iteration das gewünschte Ergebniss liefert. Der Beweis ist relativ technisch, weshalb wir hier auf die Literatur verweisen.

THEOREM 7.13.

Es sei $A \in \mathbb{C}^{n\times n}$ eine diagonalisierbare Matrix mit $A = V\Lambda V^{-1}$ und Eigenwerten $|\lambda_1| > |\lambda_2| > \ldots |\lambda_n| > 0$. Weiterhin existiere für V^{-1} eine LR-Zerlegung. Dann konvergiert die Folge $A^{(k)}$ aus der QR-Iteration Algorithmus 7.12 gegen eine obere Dreicksmatrix, deren Diagonaleinträge die Eigenwerte von A sind.

Beweis. Siehe z.B. [MS19, S. 198]. □

BEISPIEL 7.14.

Wir betrachten als Beispiel die Matrix

$$A = \begin{pmatrix} 8 & 1 & 9 \\ 1 & 3 & 5 \\ 9 & 5 & 4 \end{pmatrix}$$

und wenden die QR-Iteration darauf an. Nach 10 Iterationen erhalten wir die Matrix

$$A^{(10)} \approx \begin{pmatrix} 16.4074456 & 0.00006 & 0.000009 \\ 0.00006 & -4.58988353 & 1.14034769 \\ 0.000009 & 1.14034769 & 3.18243789 \end{pmatrix}$$

und nach 50 Iterationen die Matrix

$$A^{(50)} \approx \begin{pmatrix} 16.4074456 & 0. & 0. \\ 0. & -4.75373985 & 0. \\ 0. & 0. & 3.34629421 \end{pmatrix}$$

wobei die Werte auf den Diagonalen approximativ den Eigenwerten von A entsprechen. Wir konvergieren hier gegen eine Diagonalmatrix, da A symmetrisch ist.

Falls $A \in \mathbb{R}^{n\times n}$ eine reelle Matrix ist und wir damit auch reelle QR-Zerlegungen durchführen, A allerdings auch komplexe Eigenwerte $\lambda \in \mathbb{C}$ hat, so erhalten wir eine Blockdreickstruktur, wie folgendes Beispiel illustriert.

BEISPIEL 7.15.

Wir wenden nun die QR-Iteration auf die Matrix

$$A = \begin{pmatrix} 8 & 10 & 9 \\ 1 & 3 & 5 \\ 9 & 5 & 4 \end{pmatrix}$$

an welche auch komplexe Eigenwerte hat. Nach 50 Iterationen erhalten wir die Matrix

$$A^{(50)} \approx \begin{pmatrix} 18.0 & -4.5 & -5.7 \\ 0. & -1.78 & 4.3 \\ 0. & -0.87 & -1.2 \end{pmatrix}$$

welche keine echte Dreicksstruktur hat. Der obere rechte Eintrag enthält den reellen Eigenwert $\lambda_1 = 18$ von A. Der linke untere Block enthält nun die komplexen Eigenwerte

von A, in dem Sinne, dass die Submatrix

$$\tilde{A}^{(50)} \approx \begin{pmatrix} -1.78 & 4.3 \\ -0.87 & -1.2 \end{pmatrix}$$

die Eigenwerte $\lambda_2 \approx -1.5+1.9i, \lambda_3 \approx -1.5-1.9i$ enthält was den komplexen Eigenwerten von A entspricht. Hat eine Matrix also komplexe Eigenwerte liefert die QR-Iteration jeweils 2×2-Blöcke in welche diese Eigenwerte beinhalten.

Literatur

Books

[FH07] R. W. Freund und R. H. Hoppe. *Stoer/Bulirsch: Numerische Mathematik 1*. Springer, 2007.

[MS19] A. Meister und T. Sonar. *Numerik*. Springer Berlin Heidelberg, 2019. DOI: 10.1007/978-3-662-58358-6.

Articles

[BP98] S. Brin und L. Page. „The anatomy of a large-scale hypertextual web search engine“. In: *Computer networks and ISDN systems* 30.1-7 (1998), S. 107–117.

Online

[FLOPS] *Floating-Point Operations Per Second*. URL: https://en.wikichip.org/wiki/flops (besucht am 18. 10. 2022).

[2-Kompl] *Zweierkomplement Darstellung von Maschinenzahlen*. URL: https://de.wikipedia.org/wiki/Zweierkomplement (besucht am 31. 10. 2022).

[IEEE754] *IEEE 754 Standard für Gleitkommazahlen*. URL: https://de.wikipedia.org/wiki/IEEE_754 (besucht am 30. 10. 2022).

[Quad] *Abgeschlossene Newton-Cotes-Formeln*. URL: https://de.wikipedia.org/wiki/Newton-Cotes-Formeln#Abgeschlossene_Newton-Cotes-Formeln (besucht am 10. 01. 2023).

[Poly] *Visualisierung der ersten Bernoulli-Polynome*. URL: https://de.m.wikipedia.org/wiki/Bernoulli-Zahl#/media/Datei%3ABernoulli_Polynome.svg (besucht am 23. 01. 2023).