

# NUMERISCHE MATHEMATIK

Prof. Dr. Hans Babovsky

Institut für Mathematik

Technische Universität Ilmenau

WS 2011/2012

# Inhaltsverzeichnis

<b>1</b>	<b>Die Kondition numerischer Probleme am Beispiel linearer Gleichungssysteme</b>	<b>3</b>
<b>2</b>	<b>Lineare Gleichungssysteme I – Direkte Löser</b>	<b>8</b>
2.1	Gestaffelte Systeme . . . . .	8
2.2	Gauß-Elimination (ohne Pivotisierung) . . . . .	10
2.3	LR-Zerlegung einer Matrix . . . . .	12
2.4	Spaltenpivotisierung . . . . .	14
2.5	Das Cholesky-Verfahren . . . . .	16
<b>3</b>	<b>Lineare Gleichungssysteme II – Iterationsverfahren</b>	<b>19</b>
3.1	Das Jacobi- und das Gauß-Seidel-Verfahren . . . . .	19
3.2	Lineare Gleichungssysteme als Fixpunktprobleme . . . . .	21
<b>4</b>	<b>Nichtlineare Gleichungen</b>	<b>25</b>
4.1	Ableitungsfreie Verfahren bei einer Unbekannten . . . . .	25
4.2	Fixpunktverfahren . . . . .	27
4.3	Das Newton-Verfahren . . . . .	29
<b>5</b>	<b>Interpolation und Approximation</b>	<b>35</b>
5.1	Polynominterpolation . . . . .	35
5.2	Tschebyscheff-Interpolation . . . . .	39
5.3	Spline-Interpolation . . . . .	42
5.3.1	Polynom-Splines . . . . .	42
5.3.2	Kubische Splines . . . . .	44
5.4	Hermite-Interpolation . . . . .	47
<b>6</b>	<b>Bestimmte Integrale</b>	<b>52</b>
6.1	Quadraturformeln . . . . .	52
6.2	Newton-Cotes-Formeln . . . . .	54
6.3	Extrapolationsverfahren . . . . .	59

<b>7</b>	<b>Numerik gewöhnlicher Differentialgleichungen</b>	<b>65</b>
7.1	Anfangswertprobleme . . . . .	65
7.2	Konsistenz und Konvergenz von ESV . . . . .	66
7.3	Runge-Kutta-Verfahren . . . . .	68
7.4	Die Stabilität von ESV . . . . .	71

# 1 Die Kondition numerischer Probleme am Beispiel linearer Gleichungssysteme

## Maschinenzahlen

In Computern können reelle Zahlen nur approximativ dargestellt werden, da zur exakten Darstellung unendlich viele Speicherstellen nötig wären. Ist  $x \in \mathbb{R} \setminus \{0\}$  eine beliebige reelle Zahl, so können wir sie zunächst als Dezimalzahl schreiben in der Form

$$x = \pm 0.x_1x_2x_3 \dots \cdot 10^p$$

wobei  $p \in \mathbf{Z}$  so gewählt ist, dass  $x_1 \neq 0$ . Zur Beschreibung von  $x$  ist i.A. eine unendliche Folge  $(x_i)_{i \in \mathbf{N}}$  nötig. In einem Rechner, der Zahlen in Dezimalform in  $N$ -stelliger Genauigkeit darstellt, wird  $x$  approximiert durch eine Zahl der Form

$$\hat{x} = 0.\hat{x}_1\hat{x}_2\hat{x}_3 \dots \hat{x}_N \cdot 10^p$$

Diese Zahl wird in der Regel gemittelt durch Abschneiden oder durch Runden von  $x$  nach der  $N$ -ten Stelle.

Alle Rechnungen auf dem Computer mit reellen Zahlen unterliegen damit einem **Abschneidefehler**. Die **Kondition** einer Rechenoperation gibt Auskunft darüber, wie stark das Ergebnis durch diesen Fehler beeinflusst wird.

## Ein Beispiel

**Beispiel 1:** Die exakte Lösung des Gleichungssystems  $Ax = b$  mit

$$A = \begin{pmatrix} 2.09 & 11.32 \\ 8.84 & 47.83 \end{pmatrix}, \quad b = \begin{pmatrix} 2.96 \\ 12.47 \end{pmatrix}$$

ist  $x = (-4, 1)^T$ . Wir definieren die gerundeten Matrizen

$$\hat{A} = \begin{pmatrix} 2.1 & 11.3 \\ 8.8 & 47.8 \end{pmatrix}, \quad \hat{b} = \begin{pmatrix} 3.0 \\ 12.5 \end{pmatrix}$$

Die Lösung von  $Ax = \hat{b}$  ist  $x = (-19.116, 3.794)^T$ , die von  $\hat{A}x = b$  ist  $x = (0.614, 0.148)^T$  und die von  $\hat{A}x = \hat{b}$  gleich  $x = (2.287, -0.16)^T$ . Beide Näherungslösungen sind offenbar

unbrauchbar. Ähnliche Abhängigkeiten von Rundungsfehlern treten z.B. bei der Matrix

$$\tilde{A} = \begin{pmatrix} 2.1 & 11.3 \\ -8.8 & 47.8 \end{pmatrix}$$

nicht auf. Versuchen Sie eine geometrische Begründung!

Zur Untersuchung dieses Phänomens benötigen wir einige Vorbereitungen.

## Normen

Einem Vektor  $x \in \mathbb{R}^n$  lässt sich auf anschauliche Weise eine *Länge* zuordnen (*„Euklidische Länge“*). Durch Verallgemeinerung kommen wir auf den Begriff einer (Vektor-) Norm.

Eine **Vektornorm** auf  $\mathbb{R}^n$  ist eine Abbildung  $\|\cdot\|_V : \mathbb{R}^n \rightarrow \mathbb{R}_+$ , welche die *Vektor-Normeigenschaften* erfüllt:

- (i)  $\|x\|_V > 0$ , falls  $x \neq 0$  (Nichtnegativität),
- (ii)  $\|\lambda x\| = |\lambda| \|x\|$  für alle  $\lambda \in \mathbb{R}$ ,  $x \in \mathbb{R}^n$  (Homogenität) und
- (iii)  $\|x + y\| \leq \|x\| + \|y\|$  für alle  $x, y \in \mathbb{R}^n$  (Dreiecksungleichung).

Entsprechend lässt sich auch einer quadratischen Matrix eine nichtnegative Zahl zuordnen, welche mit der Größe der Matrixkoeffizienten in Verbindung steht und gewissen Rechengesetzen unterliegt.

Eine **Matrixnorm** auf der Menge der  $n \times n$ -Matrizen ist eine Abbildung  $\|\cdot\|_M : \mathbb{R}^{n \times n} \rightarrow \mathbb{R}_+$ , welche die *Matrix-Normeigenschaften* erfüllt:

- (i)  $\|A\|_M > 0$ , falls  $A \neq 0$ ,
- (ii)  $\|\lambda A\|_M = |\lambda| \|A\|_M$  für alle  $\lambda \in \mathbb{R}$ ,  $A \in \mathbb{R}^{n \times n}$ ,
- (iii)  $\|A + B\| \leq \|A\| + \|B\|$  für alle  $A, B \in \mathbb{R}^{n \times n}$  und
- (iv)  $\|A \cdot B\|_M \leq \|A\|_M \cdot \|B\|_M$  für alle  $A, B \in \mathbb{R}^{n \times n}$ .

Zu vorgegebener Vektornorm  $\|\cdot\|_V$  und Matrixnorm  $\|\cdot\|_M$  wünschen wir uns als weitere

Rechenregel die Abschätzung

$$\|Ax\|_V \leq \|A\|_M \|x\|_V$$

Diese ist leider nicht für beliebige Kombinationen von Vektor- und Matrixnorm erfüllt. Ein Normpaar  $(\|\cdot\|_M, \|\cdot\|_V)$ , für welches diese Zusatzregel gilt, heißt **kompatibel**.

Das für uns wichtigste Beispiel eines kompatiblen Normpaars ist gegeben durch die Vektornorm für  $v = (v_1, \dots, v_n)^T$

$$\|x\|_2 = \sqrt{v_1^2 + \dots + v_n^2} \quad (\text{Euklidische Norm})$$

und die Matrixnorm für  $A = (a_{ij})_{1 \leq i, j \leq n}$

$$\|A\|_F = \sqrt{\sum_{i,j=1}^n a_{ij}^2} \quad (\text{Frobeniusnorm})$$

Ein weiteres wichtiges kompatibles Normpaar besteht aus der Vektornorm

$$\|v\|_\infty = \max_{i=1, \dots, n} |v_i| \quad (\text{Maximumnorm})$$

und der Matrixnorm

$$\|A\|_Z = \max_{i=1, \dots, n} \sum_{j=1}^n |a_{ij}| \quad (\text{Zeilensummennorm})$$

## Die Kondition linearer Gleichungssysteme

Im Folgenden seien  $A$  regulär,  $x$  Lösung von  $Ax = b$  und  $\tilde{x}$  Lösung des gestörten Systems  $A\tilde{x} = \tilde{b}$  mit  $\tilde{b} = b + \Delta b$ .  $(\|\cdot\|_M, \|\cdot\|_V)$  sei ein kompatibles Normpaar. Gesucht ist eine Abschätzung des Fehlers

$$\Delta x := \tilde{x} - x.$$

Offenbar erfüllt  $\Delta x$  die Gleichung

$$A \cdot \Delta x = \Delta b.$$

Aus den beiden Abschätzungen

$$\|b\|_V = \|Ax\|_V \leq \|A\|_M \cdot \|x\|_V \quad \text{und} \quad \|\Delta x\|_V = \|A^{-1}\Delta b\|_V \leq \|A^{-1}\|_M \cdot \|\Delta b\|_V$$

folgt als Beziehung zwischen dem relativen Fehler der Lösung und dem der rechten Seite die Abschätzung

$$\frac{\|\Delta x\|_V}{\|x\|_V} \leq \|A\|_M \cdot \|A^{-1}\|_M \cdot \frac{\|\Delta b\|_V}{\|b\|_V}$$

Zu ähnlichen Ergebnissen kommt man, wenn man zusätzlich zur rechten Seite  $b$  auch die Matrix  $A$  stört.

**Definition und Satz:** (a) Sei  $\|\cdot\|_M$  eine Matrixnorm. Zu gegebener regulärer  $n \times n$ -Matrix  $A$  heißt

$$\kappa_M(A) = \|A\|_M \cdot \|A^{-1}\|_M$$

die **Konditionszahl** von  $A$  bzgl. der Matrixnorm  $\|\cdot\|_M$ .

(b) Es seien  $x$  die Lösung des Gleichungssystems  $Ax = b$  und  $\tilde{x} = x + \Delta x$  die Lösung des gestörten Systems  $\tilde{A}\tilde{x} = \tilde{b}$  mit  $\tilde{A} = A + \Delta A$  und  $\tilde{b} = b + \Delta b$ . Ist

$$\frac{\|\Delta A\|_M}{\|A\|_M} < \frac{1}{\kappa_M(A)}$$

so gilt für den relativen Fehler der gestörten Lösung

$$\frac{\|\Delta x\|_V}{\|x\|_V} \leq \frac{\kappa_M(A)}{1 - \kappa_M(A) \frac{\|\Delta A\|_M}{\|A\|_M}} \cdot \left( \frac{\|\Delta A\|_M}{\|A\|_M} + \frac{\|\Delta b\|_V}{\|b\|_V} \right)$$

**Beispiel 1 (Fortsetzung):** In Beispiel 1 sind

$$A = \begin{pmatrix} 2.09 & 11.32 \\ 8.84 & 47.83 \end{pmatrix}, \quad A^{-1} = \begin{pmatrix} -459.462 & 108.742 \\ 84.918 & -20.077 \end{pmatrix}$$

Es folgt

$$\|A\|_F = 49.98, \quad \|A^{-1}\|_F = 480.10, \quad \kappa_F(A) = 23995.61 \approx 24000$$

Die Störungen von  $A$  und  $b$  sind gegeben durch

$$\Delta A = \begin{pmatrix} 0.01 & -0.02 \\ -0.04 & -0.03 \end{pmatrix}, \quad \Delta b = \begin{pmatrix} 0.04 \\ 0.03 \end{pmatrix}$$

Da  $\kappa_F(A)$  zu groß ist, ist die Formel des Satzes nicht anwendbar. Für die Approximation des Systems  $Ax = \tilde{b}$  erhalten wir aus unserer ersten Abschätzung

$$\frac{\|\Delta x\|_2}{\|x\|_2} \leq \kappa_F(A) \frac{\|\Delta b\|_2}{\|b\|_2} \approx 93.6$$

## 2 Lineare Gleichungssysteme I – Direkte Löser

Im Folgenden sei  $A$  eine reguläre  $n \times n$ -Matrix und  $b \in \mathbb{R}^n$  ein vorgegebener Vektor. Wir stellen eine Methode zur Lösung des linearen Gleichungssystems

$$A \cdot x = b$$

vor, welche als **Gaußsches Eliminationsverfahren** bezeichnet wird.

### 2.1 Gestaffelte Systeme

Für gewisse Matrizen  $A$  lässt sich das Gleichungssystem  $Ax = b$  besonders einfach lösen.

**Beispiel 1:** Es sei

$$(A) = \begin{pmatrix} 1 & 1 & -2 \\ 0 & -1 & 1 \\ 0 & 0 & 3 \end{pmatrix}$$

und  $b = (-1, 0, 2)^T$ . Ausführlich lässt sich das Gleichungssystem  $Ax = b$  offenbar schreiben als

$$\begin{aligned} x_1 + x_2 - 2x_3 &= -1 \\ -x_2 + x_3 &= 0 \\ 3x_3 &= 2 \end{aligned}$$

Dieses System lässt sich einfach “von unten nach oben” lösen. Aus Gleichung 3 folgt nämlich  $x_3 = 2/3$ . Eingesetzt in Gleichung 2 erhalten wir  $x_2 = x_3 = 2/3$ , und aus Gleichung 1 leiten wir her, dass  $x_1 = -1 - x_2 + 2x_3 = -1/3$ . Dieses Verfahren heißt **Rückwärtseinsetzen**.

**Definition:** (a) Eine Matrix  $A$  ist eine **rechte (obere) Dreiecksmatrix**, falls  $A$  die

Form hat

$$A = \begin{pmatrix} a_{11} & \cdots & & a_{1n} \\ 0 & a_{22} & & \vdots \\ \vdots & \ddots & \ddots & \\ 0 & \cdots & 0 & a_{nn} \end{pmatrix},$$

wenn also gilt  $a_{ij} = 0$  für  $i > j$ .

(b) Ist dagegen  $A$  von der Form

$$A = \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ \vdots & a_{22} & \ddots & \vdots \\ & & \ddots & 0 \\ a_{n1} & \cdots & & a_{nn} \end{pmatrix}$$

(d.h.  $a_{ij} = 0$  falls  $i < j$ ), so heißt  $A$  **linke (untere) Dreiecksmatrix**.

(c) Ein lineares Gleichungssystem  $Ax = b$  mit einer oberen oder unteren Dreiecksmatrix  $A$  mit heißt **gestaffeltes System**.

**Bemerkung:** Ist  $A$  obere oder untere Dreiecksmatrix, so ist die Determinante

$$\det(A) = \prod_{i=1}^n a_{ii}.$$

Da  $A$  als regulär vorausgesetzt ist, folgt

$$a_{ii} \neq 0, \quad i = 1, \dots, n.$$

Ist  $A$  obere Dreiecksmatrix, so lässt sich das Gleichungssystem  $A \cdot x = b$  leicht durch Rückwärtseinsetzen lösen. Aus der letzten der  $n$  Gleichungen folgt nämlich  $x_n = b_n/a_{nn}$ , aus der vorletzten  $x_{n-1} = (b_{n-1} - a_{n-1,n}x_n)/a_{n-1,n-1}$ , etc. Der folgende Algorithmus löst das Gleichungssystem.

**Algorithmus (Rückwärtseinsetzen)** zur Lösung eines linearen Gleichungssystems in oberer Dreiecksgestalt.

S1: Berechne  $x_n := b_n/a_{nn}$ ;

S2: Für  $k = n - 1(-1)1$  berechne

$$x_k := (b_k - a_{k,k+1}x_{k+1} - \cdots - a_{k,n}x_n)/a_{kk}.$$

Ein ähnlicher Algorithmus zum **Vorwärtseinsetzen** für Gleichungssysteme mit einer **unteren Dreiecksmatrix** lässt sich leicht als Modifikation des obigen Algorithmus herleiten.

## 2.2 Gauß-Elimination (ohne Pivotisierung)

Die Idee der Gauß-Elimination besteht darin, das Gleichungssystem  $A \cdot x = b$  durch elementare Zeilenumformungen in ein System in oberer Dreiecksgestalt umzuwandeln und dieses dann durch Rückwärtseinsetzen zu lösen. Es ist hierzu sinnvoll, die **erweiterte Matrix**

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

zu definieren.

Subtrahieren wir in der erweiterten Matrix  $(A, b)$  für  $k = 2, \dots, n$  von der  $k$ -ten Zeile das  $l_{k1}$ -fache der ersten Zeile,  $l_{k1} := a_{k1}/a_{11}$ , so verschwinden nacheinander alle Elemente unterhalb des ersten Diagonalelements  $a_{11}$ :

$$(A|b) = \left( \begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & & & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array} \right) \longrightarrow \left( \begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & & & \vdots & \vdots \\ 0 & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} & b_n^{(1)} \end{array} \right) =: (A^{(1)}, b^{(1)}).$$

Entsprechend können wir fortfahren, im System  $(A^{(1)}, b^{(1)})$  geeignete Vielfache der zweiten Zeile von den tiefer liegenden Zeilen zu subtrahieren, um alle Koeffizienten unterhalb des zweiten Diagonalelements auf Null zu setzen. Nach und nach erhalten wir so ein

System der Form

$$\left( \begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & \ddots & & \vdots & \vdots \\ 0 & \cdots & 0 & a_{nn}^{(n-1)} & b_n^{(n-1)} \end{array} \right) =: (A^{(n-1)}, b^{(n-1)}).$$

welches wie in Abschnitt 2.1 durch Rückwärtseinsetzen gelöst werden kann.

**Algorithmus** (Gauß-Elimination ohne Pivotisierung): Im folgenden bezeichne  $a_{ij}$  die Elemente und  $z_j$  die  $j$ -te Zeile der gerade aktuellen erweiterten Matrix.

(S1) Für  $i=1(1)n-1$ :

(S2) Für  $j=i+1(1)n$ :

berechne  $l_{ji} := a_{ji}/a_{ii}$ ;

setze  $z_j := z_j - l_{ji} \cdot z_i$ .

**Beispiel 2:**

$$(A|b) = \left( \begin{array}{ccc|c} 1 & 3 & -2 & 1 \\ 2 & 2 & 1 & 1 \\ -3 & -1 & 1 & 1 \end{array} \right).$$

Es ist  $l_{21} = a_{21}/a_{11} = 2$  und  $l_{31} = a_{31}/a_{11} = -3$ . Die Operationen  $z_j := z_j - l_{j1} \cdot z_1$ ,  $j = 2, 3$ , führen auf das System

$$(A^{(1)}|b^{(1)}) = \left( \begin{array}{ccc|c} 1 & 3 & -2 & 1 \\ 0 & -4 & 5 & -1 \\ 0 & 8 & -5 & 4 \end{array} \right).$$

Es folgt  $l_{32} = a_{32}/a_{22} = -2$ ; das System in Dreiecksform lautet

$$(A^{(2)}|b^{(2)}) = \left( \begin{array}{ccc|c} 1 & 3 & -2 & 1 \\ 0 & -4 & 5 & -1 \\ 0 & 0 & 5 & 2 \end{array} \right).$$

### 2.3 LR-Zerlegung einer Matrix

Soll das Gleichungssystem  $A \cdot x = b$  mehrmals (d.h. mit unterschiedlichen rechten Seiten  $b$ ) gelöst werden, so ist es aus Gründen der Effizienz ratsam, das Gauß-Verfahren leicht zu modifizieren, um unnötige Wiederholungen von Berechnungen zu vermeiden. Hierzu helfen die folgenden Beobachtungen, die hier ohne Beweis wiedergegeben werden.

**Bemerkungen:** (a) Es bezeichne

$$\tilde{L} := \begin{pmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & & & & 1 & \\ & & & & -l_{n,n-1} & 1 \end{pmatrix} \cdots \begin{pmatrix} 1 & & & & & \\ & -l_{21} & 1 & & & \\ & \vdots & & \ddots & & \\ & & & & -l_{n-1,1} & 1 \\ & & & & -l_{n1} & 1 \end{pmatrix}$$

das Produkt der unteren Dreiecksmatrizen, welche als Einträge die im Gauß-Algorithmus berechneten Größen enthalten. Ferner bezeichne  $R$  die obere Dreiecksmatrix, in welche  $A$  als Ergebnis des Eliminationsverfahrens umgewandelt wurde. Dann ist

$$R = \tilde{L} \cdot A. \quad (2.1)$$

(b) Die Inverse von  $\tilde{L}$  ist

$$L := \begin{pmatrix} 1 & 0 & \cdots & 0 \\ l_{21} & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ l_{n1} & \cdots & l_{n,n-1} & 1 \end{pmatrix}.$$

Aus Gleichung (1.1) folgt

$$A = L \cdot R.$$

Damit kann das Gaußsche Eliminationsverfahren interpretiert werden als Verfahren zur Zerlegung von  $A$  in ein Produkt einer unteren Dreiecksmatrix  $L$  mit einer oberen Dreiecksmatrix  $R$  (**LR-Zerlegung**<sup>1</sup>).

<sup>1</sup>links-rechts-Zerlegung; wird in englischsprachiger Literatur gewöhnlich als LU (d.h. *lower-upper*) - Zerlegung bezeichnet.

**Beispiel 2 (Fortsetzung):** Es ist

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 3 & 0 & 1 \end{pmatrix}}_{=\bar{L}} \cdot \begin{pmatrix} 1 & 3 & -2 \\ 2 & 2 & 1 \\ -3 & -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 3 & -2 \\ 0 & -4 & 5 \\ 0 & 0 & 5 \end{pmatrix}$$

und damit

$$\underbrace{\begin{pmatrix} 1 & 3 & -2 \\ 2 & 2 & 1 \\ -3 & -1 & 1 \end{pmatrix}}_A = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -3 & -2 & 1 \end{pmatrix}}_L \cdot \underbrace{\begin{pmatrix} 1 & 3 & -2 \\ 0 & -4 & 5 \\ 0 & 0 & 5 \end{pmatrix}}_R$$

Ist eine LR-Zerlegung von  $A$  bekannt, so kann das Gleichungssystem  $A \cdot x = b$  auf die Lösung zweier gestaffelter Systeme zurückgeführt werden. Ist nämlich  $y$  die Lösung von  $L \cdot y = b$  und  $x$  die Lösung von  $R \cdot x = y$ , so ist  $x$  die gesuchte Lösung, denn

$$A \cdot x = (L \cdot R) \cdot x = L \cdot (R \cdot x) = L \cdot y = b.$$

Dies führt auf die folgende Modifikation der Gauß-Elimination.

**Algorithmus** (Gauß-Elimination) zur Berechnung von  $A \cdot x = b$  :

- S1: Konstruiere die LR-Zerlegung von  $A$  mit Hilfe des Algorithmus (1.6) (ohne rechte Seite);
- S2: Löse  $L \cdot y = b$  durch Vorwärtseinsetzen;
- S3: Löse  $R \cdot x = y$  durch Rückwärtseinsetzen.

Insbesondere zur Lösung großer Gleichungssysteme ist es wichtig zu wissen, wie der Rechenaufwand mit der Größe der Dimension wächst. Als Kennzahl für den numerischen Aufwand benutzen wir hier die Anzahl der Multiplikationen.

**Rechenaufwand:** Für große  $n$  werden für

- den Schritt S1 ca.  $n^3/3$
- jeden der Schritte S2 und S3 ca.  $n^2/2$  Multiplikationen benötigt.

Der Hauptaufwand liegt damit in  $SI$ ; dieser Schritt muss aber bei mehrmaliger Lösung (insbesondere bei der Bestimmung von  $A^{-1}$ ) nur einmal durchgeführt werden.

## 2.4 Spaltenpivotisierung

**Beispiel 3:** Verschwindendes Diagonalelement.

$$(A|b) := \left( \begin{array}{ccc|c} 1 & -1 & 2 & 0 \\ -2 & 2 & 1 & 3 \\ 1 & 3 & 1 & -1 \end{array} \right)$$

Die Elimination der ersten Spalte führt auf die Matrix

$$(A^{(1)}|b^{(1)}) = \left( \begin{array}{ccc|c} 1 & -1 & 2 & 0 \\ 0 & 0 & 5 & 3 \\ 0 & 4 & -1 & -1 \end{array} \right) \quad \begin{array}{l} (1) \\ (2) \\ (3) \end{array}$$

In der zweiten Spalte müsste nun die Elimination des Elements  $a_{32}$  erfolgen. Dies ist wegen  $a_{22} = 0$  nicht möglich. Als Ausweg bietet sich an, die Zeilen (2) und (3) zu vertauschen. (Überlegen Sie sich, warum Zeilenvertauschungen keinen Einfluß auf die gesuchte Lösung  $x$  haben.) Hierdurch erhalten wir das gestaffelte System

$$\left( \begin{array}{ccc|c} 1 & -1 & 2 & 0 \\ 0 & 4 & -1 & 1 \\ 0 & 0 & 5 & 3 \end{array} \right) \quad \begin{array}{l} (1) \\ (3) \\ (2) \end{array}$$

welches wie oben beschrieben gelöst werden kann.

**Beispiel 4:** Das Gleichungssystem

$$(A|b) := \left( \begin{array}{ccc|c} 11 & 44 & 1 & 1 \\ 0.1 & 0.4 & 3 & 1 \\ 0 & 1 & -1 & 1 \end{array} \right)$$

soll auf einem Rechner mit sechsstelliger Genauigkeit gelöst werden. Die exakte Lösung ist

$$x = \frac{1}{329} \begin{pmatrix} -1732 \\ 438 \\ 109 \end{pmatrix} \approx \begin{pmatrix} -5.26444 \\ 1.33131 \\ 0.331307 \end{pmatrix}.$$

Die numerische Gauß-Elimination mit sechsstelliger Genauigkeit führt im ersten Schritt auf

$$(A^{(1)}|b^{(1)}) = \left( \begin{array}{ccc|c} 11 & 44 & 1 & 1 \\ 0 & 2.98023E-08 & 2.99091 & 0.990909 \\ 0 & 1 & -1 & 1 \end{array} \right)$$

und im zweiten auf

$$(A^{(2)}|b^{(2)}) = \left( \begin{array}{ccc|c} 11 & 44 & 1 & 1 \\ 0 & 2.98023E-08 & 2.99091 & 0.990909 \\ 0 & 0 & -1.00358E+08 & -3.3249E+07 \end{array} \right).$$

Die numerische Lösung dieses gestaffelten Systems lautet

$$x_{\text{num},1} = \begin{pmatrix} 0.0607903 \\ 0 \\ 0.331307 \end{pmatrix}.$$

und ist offensichtlich völlig unbrauchbar. Ein besseres Ergebnis erhalten wir, wenn wir nach dem ersten Schritt die zweite und dritte Zeile vertauschen. Dies führt auf die numerische Lösung

$$x_{\text{num},2} = \begin{pmatrix} -5.26444 \\ 1.33131 \\ 0.331307 \end{pmatrix}.$$

Offenbar hat das kleine Diagonalelement  $a_{22}^{(1)}$  im zweiten Schritt zum "Aufschaukeln" von Rundungsfehlern geführt – das System wurde *numerisch instabil*.

Die Beispiele zeigen, dass es aus Gründen der Durchführbarkeit und der *numerischen Stabilität* (d.h. geringer Einfluß von Rundungsfehlern) nötig sein kann, vor Beginn der Elimination einer Spalte eine Zeilenvertauschung vorzunehmen. Aus Stabilitätsgründen ist es dabei ratsam, vor der Elimination der  $j$ -ten Spalte die  $j$ -te Zeile mit derjenigen Zeile  $j+k$  ( $k \geq 0$ ) zu vertauschen, welche an der  $j$ -ten Stelle das betragsgrößte Element enthält. Dies führt zur Modifikation des Gaußschen Eliminationsalgorithmus aus Abschnitt 2.3:

**Algorithmus** (Gauß-Elimination mit Pivotisierung): Führe im Algorithmus zur Gauss-Elimination ohne Pivotisierung in Schleife (S1) vor Beginn der Schleife (S2) die folgenden Schritte durch:

- (i) Bestimme das Maximum  $max$  der Spaltenelemente  $a_{i,i}, a_{i+1,i}, \dots, a_{n,i}$  und die Nummer  $j_0 \geq i$  der Zeile, in der das Maximum auftritt;
- (ii) falls  $j_0 > i$ , vertausche die Zeilen ( $i$ ) und ( $j_0$ ).

Zeilenvertauschungen können durch linksseitige Multiplikation der Matrix mit einer *Permutationsmatrix* beschrieben werden.<sup>2</sup> Beispielsweise ist mit

$$P := \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$PA$  diejenige Matrix, welche aus den Zeilen von  $A$  in der Reihenfolge (2) – (3) – (1) besteht. (Nachprüfen!)

**Bemerkung:** Während das Gauß-Verfahren ohne Pivotisierung eine  $LR$ -Zerlegung von  $A$  liefert, ermöglicht das modifizierte Verfahren lediglich die  $LR$ -Zerlegung einer geeignet permutierten Matrix  $PA$ <sup>3</sup>. So führt die Gauß-Elimination der Matrix aus Beispiel 3 mit Pivotisierung auf die Matrizen

$$L = \begin{pmatrix} 1 & 0 & 0 \\ -1/2 & 1 & 0 \\ -1/2 & 0 & 1 \end{pmatrix}, \quad R = \begin{pmatrix} -2 & 2 & 1 \\ 0 & 4 & 3/2 \\ 0 & 0 & 5/2 \end{pmatrix} \quad \text{und} \quad L \cdot R = \begin{pmatrix} -2 & 2 & 1 \\ 1 & 3 & 1 \\ 1 & -1 & 2 \end{pmatrix}$$

## 2.5 Das Cholesky-Verfahren

Für eine wichtige Klasse von Matrizen – die *symmetrischen, positiv definiten* (kurz: *spd*-) Matrizen – ist die  $LR$ -Zerlegung immer ohne Pivotisierung möglich und läßt sich

<sup>2</sup>Eine Permutationsmatrix  $P$  ist eine  $n \times n$ -Matrix, welche nur 0 und 1 als Koeffizienten hat, wobei in jeder Zeile und in jeder Spalte genau eine 1 steht.

<sup>3</sup>Für Details vgl. Satz 1.8 in Deuffhard/Hohmann, Numerische Mathematik I, de Gruyter, 1993

in kompakter Form darstellen.

**Definition:** (a) Eine  $n \times n$ -Matrix  $A$  heißt **symmetrisch**, falls für alle  $i, j = 1, \dots, n$  gilt:  $a_{ij} = a_{ji}$ .

(b)  $A$  heißt **positiv definit**, wenn für alle Vektoren  $x \neq 0$  gilt

$$x^T A x > 0.$$

(c) Symmetrische, positiv definite Matrizen werden kurz als **spd-Matrizen** bezeichnet.

Das zentrale Ergebnis liefert

**Satz:** Ist  $A$  eine spd-Matrix, so gibt es eine untere Dreiecksmatrix  $L$  derart, dass

$$A = LL^T.$$

Zwischen den Koeffizienten von  $A$  und  $L$  gelten die Beziehungen

$$\begin{aligned} i = j: \quad a_{ii} &= l_{i1}^2 + \dots + l_{ii}^2, \\ i > j: \quad a_{ij} &= l_{i1}l_{j1} + \dots + l_{ij}l_{jj}. \end{aligned}$$

In geschickter Reihenfolge angeordnet, können diese Formeln zur Berechnung von  $L$  verwandt werden. Dies geschieht im

**Algorithmus** (Cholesky-Zerlegung):

Für  $i=1(1)n$ :

$$\text{Berechne } l_{ii} := \sqrt{a_{ii} - \sum_{j=1}^{i-1} l_{ij}^2}$$

Für  $j:=i+1(1)n$

$$\text{Berechne } l_{ji} := (a_{ji} - \sum_{k=1}^{i-1} l_{jk}l_{ik}) / l_{ii}$$

**Beispiel:** Die Matrix

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

ist offensichtlich symmetrisch. Außerdem folgt aus

$$\begin{aligned} \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix} \begin{pmatrix} 2 & 0 & 1 \\ 0 & 2 & 0 \\ 1 & 0 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} &= 2(x_1^2 + x_2^2 + x_3^2 + x_1x_3) \\ &= (x_1 + x_3)^2 + x_1^2 + 2x_2^2 + x_3^2, \end{aligned}$$

dass  $A$  positiv definit ist. Der Cholesky-Algorithmus ergibt

$$\begin{aligned} i=1: \quad j=1: \quad l_{11} &= \sqrt{a_{11}} = \sqrt{2} \\ & \quad j=2: \quad l_{21} = a_{21}/l_{11} = 0 \\ & \quad j=3: \quad l_{31} = a_{31}/l_{11} = 1/\sqrt{2} \\ i=2: \quad j=2: \quad l_{22} &= \sqrt{a_{22} - l_{21}^2} = \sqrt{2} \\ & \quad j=3: \quad l_{32} = (a_{32} - l_{31}l_{21})/l_{22} = 0 \\ i=3: \quad j=3: \quad l_{33} &= \sqrt{a_{33} - l_{31}^2 - l_{32}^2} = \sqrt{3/2} \end{aligned}$$

Als  $LR$ -Zerlegung für  $A$  erhält man

$$A = LL^T = \begin{pmatrix} \sqrt{2} & 0 & 0 \\ 0 & \sqrt{2} & 0 \\ 1/\sqrt{2} & 0 & \sqrt{3/2} \end{pmatrix} \begin{pmatrix} \sqrt{2} & 0 & 1/\sqrt{2} \\ 0 & \sqrt{2} & 0 \\ 0 & 0 & \sqrt{3/2} \end{pmatrix}$$

**Rechenaufwand:** Die numerische Durchführung des Cholesky-Verfahrens erfordert für große  $n$  ca.  $n^3/6$  Multiplikationen und  $n$  Quadratwurzel-Berechnungen.

### 3 Lineare Gleichungssysteme II – Iterationsverfahren

Da der hohe Rechenaufwand des Gaußschen Eliminationsverfahrens für große lineare Gleichungssysteme problematisch ist, wollen wir kurz eine Alternative untersuchen. Hierbei wird die Lösung eines linearen Gleichungssystems nicht durch eine endliche Zahl von Rechenschritten bestimmt, sondern als Limes einer rekursiv definierten Folge von Vektoren.

#### 3.1 Das Jacobi- und das Gauß-Seidel-Verfahren

Gegeben sei ein lineares Gleichungssystem, welches in ausführlicher Darstellung lautet

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n &= b_n \end{aligned}$$

Sind die Diagonalelemente  $a_{ii} \neq 0$ ,  $i = 1, \dots, n$ , so können diese Gleichungen wie folgt umgestellt werden.

$$\begin{aligned} x_1 &= (b_1 - a_{12}x_2 - a_{13}x_3 - \dots - a_{1n}x_n)/a_{11} \\ x_2 &= (b_2 - a_{21}x_1 - a_{23}x_3 - \dots - a_{2n}x_n)/a_{22} \\ &\vdots \\ x_n &= (b_n - a_{n1}x_1 - a_{n2}x_2 - \dots - a_{n,n-1}x_{n-1})/a_{nn} \end{aligned}$$

In Kurzform können diese Gleichungen geschrieben werden als

$$x_i = \left( b_i - \sum_{\substack{j=1 \\ j \neq i}}^n a_{ij}x_j \right) / a_{ii} \quad i = 1, \dots, n \quad (3.1)$$

Beim **Jacobi-Verfahren** wird nun – ausgehend von einem Startvektor  $x^{(0)}$  – durch die Rekursionsvorschrift

$$x_i^{(k+1)} = \left( b_i - \sum_{\substack{j=1 \\ j \neq i}}^n a_{ij}x_j^{(k)} \right) / a_{ii} \quad i = 1, \dots, n \quad (3.2)$$

eine Iterationsfolge  $(x_{k \in \mathbb{N}}^{(k)})$  konstruiert. Man kann leicht zeigen, dass im Falle der Konvergenz dieser Folge der Grenzvektor  $x := \lim_{k \rightarrow \infty} x^{(k)}$  Lösung von Gleichung (3.1), also die Lösung des linearen Gleichungssystems ist.

**Algorithmus** (Jacobi-Verfahren):

$$\begin{aligned}
 (S1) \quad & \text{Für } i=1(1)n: \\
 & \quad x_i^{(k+1)} := b_i \\
 (S2)_J \quad & \text{Für } j=1(1)n: \\
 & \quad \text{Falls } (i \neq j): x_i^{(k+1)} := x_i^{(k+1)} - a_{ij}x_j^{(k)} \\
 (S3) \quad & x_i^{(k+1)} := x_i^{(k+1)} / a_{ii}.
 \end{aligned}$$

Das **Gauß-Seidel-Verfahren** ist eine Variante hiervon. Wir sehen, dass bei der sukzessiven Berechnung der  $x_i^{(k)}$  ( $i = 1(1)n$ ) in (3.2) die Komponenten  $x_j^{(k+1)}$  für  $j < i$  bereits berechnet sind. Diese können also auf der rechten Seite eingesetzt werden. Dies führt auf die Gleichung

$$x_i^{(k+1)} = \left( b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right) / a_{ii} \quad i = 1, \dots, n$$

**Algorithmus** (Gauß-Seidel-Verfahren): Ersetze im Jacobi-Algorithmus den Schritt  $S2_J$  durch

$$\begin{aligned}
 (S2)_{GS} \quad & \text{Für } j=1(1)i-1: \\
 & \quad x_i^{(k+1)} := x_i^{(k+1)} - a_{ij}x_j^{(k+1)} \\
 & \quad \text{Für } j=i+1(1)n: \\
 & \quad x_i^{(k+1)} := x_i^{(k+1)} - a_{ij}x_j^{(k)}
 \end{aligned}$$

**Beispiel 1:** Gelöst werden soll das Gleichungssystem

$$\begin{aligned}
 5x_1 - x_2 - 2x_3 &= 2 \\
 x_1 - 6x_2 + 3x_3 &= -2 \\
 -2x_1 + x_2 + 4x_3 &= 3.
 \end{aligned}$$

$k$	$x_{J,1}^{(k)}$	$x_{J,2}^{(k)}$	$x_{J,2}^{(k)}$	$x_{GS,1}^{(k)}$	$x_{GS,2}^{(k)}$	$x_{GS,3}^{(k)}$
2	0.7667	0.7750	0.8667	0.7667	0.8361	0.9243
4	0.9547	0.9534	0.9772	0.9826	0.9874	0.9980
6	0.9918	0.9914	0.9955	0.9992	0.9992	0.9998
8	0.9977	0.9984	0.9991	0.9999	0.9999	1.0000
10	0.9995	0.9995	0.9999	1.0000	1.0000	1.0000

Tabelle 1: Numerische Lösung des Beispiels 1

Die Iterationsvorschrift für das Jacobi-Verfahren lautet

$$\begin{aligned}x_1^{(k+1)} &= (2 + x_2^{(k)} + 2x_3^{(k)}) / 5 \\x_2^{(k+1)} &= (2 + x_1^{(k)} + 3x_3^{(k)}) / 6 \\x_3^{(k+1)} &= (3 + 2x_1^{(k)} - x_2^{(k)}) / 4.\end{aligned}$$

Beim Gauß-Seidel-Verfahren wird wie folgt modifiziert.

$$\begin{aligned}x_1^{(k+1)} &= (2 + x_2^{(k)} + 2x_3^{(k)}) / 5 \\x_2^{(k+1)} &= (2 + x_1^{(k+1)} + 3x_3^{(k)}) / 6 \\x_3^{(k+1)} &= (3 + 2x_1^{(k+1)} - x_2^{(k+1)}) / 4.\end{aligned}$$

Die Ergebnisse einiger Iterationen zum Startwert  $x_0 = x_2 = x_3 = 0$  sind in Tabelle 1 wiedergegeben. Die exakte Lösung ist  $x_1 = x_2 = x_3 = 1$ .

### 3.2 Lineare Gleichungssysteme als Fixpunktprobleme

Wir leiten einen allgemeinen Rahmen her, in dem die obigen Iterationsverfahren untersucht und verallgemeinert werden können. Hierzu zerlegen wir die Matrix  $A = (a_{ij})_{1 \leq i, j \leq n}$  in eine untere und eine obere Dreiecksmatrix sowie in eine Diagonalmatrix

wie folgt

$$A = \underbrace{\begin{pmatrix} 0 & 0 & \cdots & 0 \\ a_{21} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{pmatrix}}_{A_L} + \underbrace{\begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & a_{nn} \end{pmatrix}}_{A_D} + \underbrace{\begin{pmatrix} 0 & a_{12} & \cdots & a_{1n} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n-1,n} \\ 0 & \cdots & 0 & 0 \end{pmatrix}}_{A_R}$$

Beachten wir, dass

$$A_D^{-1} = \begin{pmatrix} 1/a_{11} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1/a_{nn} \end{pmatrix}$$

so sehen wir leicht, dass (3.1) in Matrix-Vektorschreibweise lautet

$$x = A_D^{-1}(b - (A_L + A_R)x)$$

Dies ist eine **Fixpunktgleichung**, welche äquivalent zum Gleichungssystem  $Ax = b$  ist.

In dieser Schreibweise nimmt das Jacobi-Verfahren die Form an

$$x^{(k+1)} = A_D^{-1}(b - (A_L + A_R)x^{(k)})$$

Eine weitere äquivalente Fixpunktgleichung ist

$$x = (A_L + A_D)^{-1}(b - A_R x)$$

aus welcher wir das Gauß-Seidel-Verfahren ableiten als

$$x^{(k+1)} = (A_L + A_D)^{-1}(b - A_R x^{(k)})$$

Diese äquivalente Umformulierung linearer Gleichungssysteme kann verallgemeinert werden. Zur Herleitung einer Fixpunkt-Formulierung gehen wir aus von einer Zerlegung von  $A$  in der Form  $A = B - C$  mit einer regulären Matrix  $B$ . Man sieht leicht, dass  $x$  genau dann Lösung von  $Ax = b$  ist, wenn  $x$  auch das Fixpunktproblem

$$x = \underbrace{B^{-1}C}_M x + \underbrace{B^{-1}b}_c = Mx + c \quad (*)$$

löst. Wir können nun versuchen, diese Lösung zu approximieren durch die **Fixpunktiteration**

$$x^{(k+1)} = Mx^{(k)} + c$$

Bei der Frage der Konvergenz dieses Verfahrens ist das folgende hinreichende Ergebnis nützlich.

**Satz 1:** Gibt es ein kompatibles Normpaar  $(\|\cdot\|_M, \|\cdot\|_V)$  mit  $\|M\|_M < 1$ , so besitzt die Fixpunktgleichung (\*) eine eindeutige Lösung  $x$  und die Fixpunktiteration konvergiert für beliebigen Startvektor  $x^{(0)}$  gegen  $x$ .

**Beweis:** Ist  $x^*$  die gesuchte Lösung von  $Ax = b$ , so erfüllt  $x^*$  auch die äquivalente Fixpunktgleichung

$$x^* = Mx^* + c$$

Subtrahieren wir diese Gleichung von der Iterationsgleichung

$$x^{(k+1)} = Mx^{(k)} + c$$

so erhalten wir für die Abweichung  $e^{(k)} = x^{(k)} - x^*$  die Rekursion

$$e^{(k+1)} = Me^{(k)}, \quad \text{also} \quad \|e^{(k+1)}\|_V \leq \|M\|_M \|e^{(k)}\|_V$$

Durch vollständige Induktion erhält man leicht für  $k = 1, 2, 3, \dots$

$$\|e^{(k)}\|_V \leq \|M\|_M^k \|e^{(0)}\|_V$$

Wegen  $\|M\|_M < 1$  ist  $e^{(k)}$  eine Nullfolge.  $\circ$

Speziell für das Jacobi- und das Gauß-Seidel-Verfahren geben wir leichter zu überprüfende Kriterien an.

**Satz 2:** (a) Ist für  $i = 1, \dots, n$

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}|$$

(in diesem Fall heißt  $A$  **strikt diagonaldominant**), so konvergieren das Jacobi- und das Gauß-Seidel-Verfahren für beliebige Startvektoren  $x^{(0)}$  gegen die Lösung  $x$ .

(b) Ist  $A$  eine *spd*-Matrix, so konvergiert das Gauß-Seidel-Verfahren für beliebige Startvektoren gegen  $x$ .

**Beispiel:** Auf das Gleichungssystem

$$(A|b) = \left( \begin{array}{ccc|c} 1 & -7 & 5 & 1 \\ 1 & 1 & 12 & 2 \\ 13 & -1 & -1 & -1 \end{array} \right)$$

ist weder das Jacobi- noch das Gauß-Seidel-Verfahren anwendbar. Dagegen führen Zeilenvertauschungen in der Reihenfolge  $z_3 - z_1 - z_2$  auf ein strikt diagonaldominantes System.

**Bemerkung:** Eine wichtige Verallgemeinerung des Gauß-Seidel-Verfahrens, auf welche wir an dieser Stelle nicht eingehen können, ist in der Literatur als **SOR-Verfahren** (“successive overrelaxation”) bekannt.

## 4 Nichtlineare Gleichungen

In diesem Abschnitt sei eine Funktion

$$f : \mathbb{R}^n \longrightarrow \mathbb{R}^n$$

gegeben. Gesucht ist eine Nullstelle. Vorausgesetzt ist, dass  $f$  hinreichend glatt ist – d.h. stetig (Abschnitt 4.1) bzw. stetig differenzierbar (Abschnitt 4.3).

### 4.1 Ableitungsfreie Verfahren bei einer Unbekannten

Die folgenden Verfahren beziehen sich auf Funktionen  $f : \mathbb{R} \rightarrow \mathbb{R}$ , welche stetig sind. Zusätzlich setzen wir zunächst voraus, dass zwei Punkte  $a$  und  $b$  bekannt sind, an denen die Funktionswerte unterschiedliche Vorzeichen haben (solche Punkte kann man häufig durch eine grobe Kurvendiskussion bestimmen), für die also gilt

$$f(a) \cdot f(b) < 0. \tag{4.1}$$

Da  $f$  stetig ist, muß zwischen  $a$  und  $b$  (mindestens) eine Nullstelle liegen<sup>4</sup>. Im folgenden soll versucht werden, durch Einschachtelungsverfahren den Ort einer Nullstelle immer weiter einzugrenzen und so ihre Lage mit beliebiger Genauigkeit zu bestimmen.

#### A – Die Intervallhalbierungsmethode

Bei dieser Methode wird in jedem Iterationsschritt das untersuchte Intervall in zwei gleich große Teilintervalle eingeteilt; in einem dieser beiden Teilintervalle muss sich eine Nullstelle befinden. Im nächsten Schritt wird die Suche auf dieses Teilintervall eingeschränkt.

Ausgangspunkt ist ein Intervall  $[a, b]$ , wobei die Randpunkte  $a$  und  $b$  so gewählt sein müssen, daß  $f(a)$  und  $f(b)$  wechselnde Vorzeichen haben (d.h. die Bedingung (4.1) muss erfüllt sein). Wir bestimmen den Mittelpunkt  $x_0 = (b-a)/2$  und untersuchen die beiden Teilintervalle. Die folgenden drei Fälle sind möglich:

(a) Es ist  $f(x_0) = 0$ ; in diesem Fall ist eine Nullstelle von  $f$  gefunden und unsere Suche

---

<sup>4</sup>Dies folgt aus dem Zwischenwertsatz; vgl. Analysis-Vorlesung

ist beendet.

(b) Es ist  $f(a)f(x_0) < 0$ ; in diesem Fall haben  $f(a)$  und  $f(x_0)$  wechselnde Vorzeichen. Damit hat  $f$  eine Nullstelle in dem kleineren Intervall  $[a, x_0]$ , und es genügt, die Suche auf diesem Teilintervall fortzusetzen. Dies geschieht zweckmäßigerweise dadurch, dass der rechte Randpunkt neu definiert wird durch  $b := x_0$ .

(c) Es ist  $f(b)f(x_0) < 0$ ; in diesem Fall haben  $f(b)$  und  $f(x_0)$  wechselnde Vorzeichen und die Suche wird auf das Teilintervall  $[x_0, b]$  eingeschränkt. Der linke Randpunkt wird neu gesetzt durch die Anweisung  $a := x_0$ .

Für den zugehörigen Algorithmus müssen  $a, b$  als einzugebenden Startparametern sowie ein Fehlerparameter  $\epsilon$  eingegeben werden. Der Algorithmus lautet:

**Algorithmus** zur Intervallhalbierungsmethode: Die Intervallgrenzen  $a$  und  $b$  sind Eingabeparameter.

- S1 Berechne  $x_0 := (b + a)/2$ ;
- S2 Falls  $f(x_0) \neq 0$ :
  - Falls  $f(a) \cdot f(x_0) < 0$ , setze  $b := x_0$ ,
  - andernfalls setze  $a := x_0$
  - andernfalls setzt  $a := x_0$  und  $b := x_0$ .
- S3 Falls  $b - a > 2\epsilon$  gehe zu S1
- andernfalls gib  $x_0 := (b + a)/2$  als Näherungslösung aus und beende.

Der ausgegebene Näherungswert weicht von der gesuchten Nullstelle um höchstens  $\epsilon$  ab. Da sich die Länge des untersuchten Teilintervalls mit jedem Schritt halbiert, müssen die Schritte S1–S 3 etwa  $\log_2(|b_0 - a_0|/\epsilon)$ -mal durchlaufen werden, wenn  $a_0$  und  $b_0$  die Grenzen des Anfangsintervalls sind. Das Verfahren konvergiert damit nur sehr langsam. Beispielsweise sind bei einem Anfangsintervall der Länge 1 und einem maximal erwünschten Fehler  $\epsilon = 10^{-3}$  etwa 10 Durchläufe nötig.

## B – Regula falsi, Sekantenverfahren

Im Unterschied zur Intervallhalbierung wird bei der **Regula falsi** in jedem Schritt nicht

der Intervallmittelpunkt als neuer Referenzpunkt  $x_0$  gewählt wird, sondern die Nullstelle der Geraden  $s(x)$  durch die Punkte  $(a, f(a))$  und  $(b, f(b))$ . Diese Gerade ist gegeben durch

$$s(x) = f(a) + (x - a) \cdot \frac{f(b) - f(a)}{b - a}$$

und hat die Nullstelle

$$x_0 = a - f(a) \cdot \frac{b - a}{f(b) - f(a)} = \frac{a \cdot f(b) - b \cdot f(a)}{f(b) - f(a)}. \quad (4.2)$$

Zur Implementierung dieses Verfahrens ist lediglich im Algorithmus zur Intervallhalbierung die entsprechende Anweisung in den Schritten *S1* und *S3* zu ersetzen durch die Anweisung

$$x_0 := (a \cdot f(b) - b \cdot f(a)) / (f(b) - f(a)).$$

Bei hinreichend glatten Funktionen kann durch diese Modifikation die Konvergenzgeschwindigkeit gegenüber dem Intervallhalbierungsverfahren geringfügig verbessert werden.

Die Forderung wechselnder Vorzeichen der Funktions-Startwerte ist sehr restriktiv und häufig nur schwer zu erfüllen. Beim **Sekantenverfahren** wird die Zuordnung (4.2) ohne Überprüfung dieser Forderung getroffen und das entsprechende Iterationsverfahren durchgeführt. Bei "gutmütigen" Funktionen und nicht zu schlechten Startwerten wird man auch hier zur Approximation einer Nullstelle kommen, es ist jedoch schwer, hinreichend allgemeine Kriterien hierfür anzugeben.

## 4.2 Fixpunktverfahren

In den folgenden zwei Unterabschnitten suchen wir Nullstellen von nichtlinearen Abbildungen  $f : G \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$ .

Eine Möglichkeit besteht in der Umformulierung in ein äquivalentes Fixpunktproblem der Form

$$g(x) = x$$

für eine geeignete Funktion  $g : D \rightarrow D$ , wobei  $D$  eine abgeschlossene Teilmenge von  $G$  ist. Zur numerischen Lösung bietet sich die Fixpunktiteration

$$x^{(k+1)} = g(x^{(k)})$$

an. Ein wichtiges Kriterium zur Konvergenz des Fixpunktverfahrens liefert der folgende Satz.

**Banachscher Fixpunktsatz:** Die Abbildung  $g : D \rightarrow D$  sei **Lipschitz-stetig**, d.h. es gebe eine Konstante  $\gamma \geq 0$  (“Lipschitz-Konstante”) mit der Eigenschaft

$$\|F(x) - F(y)\| \leq \gamma \|x - y\|.$$

Für die Lipschitz-Konstante gelte  $\gamma < 1$ .<sup>5</sup>

Dann besitzt  $g$  genau einen Fixpunkt  $x^* \in D$ , und die Fixpunkt-Iteration konvergiert für jeden beliebigen Startwert  $x^{(0)}$  gegen  $x^*$ . Es gelten die folgenden Abschätzungen.

(a) Für die Abweichung der  $k$ -ten Iterierten  $x^{(k)}$  vom gesuchten Punkt gilt

$$\|x^{(k)} - x^*\| \leq \gamma^k \|x^{(0)} - x^*\|.$$

(b) Der Abstand zum gesuchten Fixpunkt kann durch den Abstand zweier aufeinanderfolgender Iterierter abgeschätzt werden durch

$$\|x^{(k)} - x^*\| \leq \frac{\gamma}{1 - \gamma} \|x^{(k)} - x^{(k-1)}\| \leq \frac{\gamma^2}{1 - \gamma} \|x^{(k-1)} - x^{(k-2)}\| \leq \dots \leq \frac{\gamma^k}{1 - \gamma} \|x^{(1)} - x^{(0)}\|.$$

Den Nachweis, dass eine stetig differenzierbare Funktion eine Kontraktion ist, kann man häufig durch Abschätzen der ersten Ableitung erbringen. Wir zeigen dies für den skalaren Fall.

**Hilfssatz:** Es sei  $I \subseteq \mathbb{R}$  ein Intervall. Ist  $g : I \rightarrow I$  stetig differenzierbar, so ist  $g$  genau dann eine Kontraktion, wenn

$$\gamma := \sup_{x \in I} |g'(x)| < 1. \quad (4.3)$$

---

<sup>5</sup>In diesem Fall heißt  $g$  auch **Kontraktion**.

Vor der Anwendung eines Fixpunktiterationsverfahrens sind folgende Voruntersuchungen sinnvoll.

**Start einer Fixpunktiteration:**  $x^*$  sei die zu berechnende Lösung des Fixpunktproblems  $x = f(x)$ .  $f$  sei stetig differenzierbar. Zu überprüfen ist:

- (i) Ist  $|f'(x^*)| > 1$ , so ist keine Konvergenz der Fixpunktiteration  $x^{(k+1)} = f(x^{(k)})$  gegen  $x^*$  zu erwarten. Stattdessen kann versucht werden, die Iteration auf das äquivalente Problem  $x = f^{-1}(x)$  anzuwenden.
- (ii) Ist  $|f'(x^*)| < 1$ , so konvergiert das Fixpunktverfahren, falls der Startwert  $x^{(0)}$  hinreichend nahe bei  $x^*$  liegt. Ist  $I$  ein Intervall, welches  $x^*$  enthält, und ist  $|f'(x)| < 1$  für alle  $x \in I$ , so gilt  $f : I \rightarrow I$  und als Startwert kann ein beliebiger Wert aus  $I$  gewählt werden.

**Beispiel:** Die Funktion  $f(x) = \exp(x) - 2 - x = 0$  hat genau eine Nullstelle  $x^* > 0$ . Diese soll mit einem geeigneten Fixpunktverfahren approximiert werden.  $x^*$  ist offenbar auch Lösung des Fixpunktproblem  $x = g(x)$  mit  $g(x) = \exp(x) - 2$ . Wegen  $g'(x) = \exp(x) \geq 1$  für  $x \geq 0$  ist allerdings nicht mit Konvergenz der Fixpunktiteration  $x^{(k+1)} = g(x^{(k)})$  gegen  $x^*$  zu rechnen. Anders ist die Situation, wenn wir aus der Gleichung  $\exp(x) = 2 + x$  das Fixpunktproblem  $x = h(x)$  herleiten mit  $h(x) = \ln(x + 2)$ . Für  $x \geq 0$  ist  $h'(x) = 1/(x + 2) \leq 1/2$ . Außerdem gilt  $h : [0, \infty) \rightarrow [0, \infty)$ . Gemäß Hilfssatz sind damit die Voraussetzungen des Banachschen Fixpunktsatzes erfüllt und das zugehörige Fixpunktverfahren  $x^{(k+1)} = \ln(x^{(k)} + 2)$  konvergiert für beliebige Startwerte  $x^{(0)} \geq 0$ .

Das Kriterium des Hilfssatzes kann übrigens auch auf stetig differenzierbare Abbildungen  $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$  für  $n > 1$  übertragen werden.

### 4.3 Das Newton-Verfahren

Das Newton-Verfahren setzt Funktionen voraus, welche nicht nur stetig, sondern mindestens einmal stetig differenzierbar sind.

## A – Der skalare Fall

Im folgenden sei  $f : \mathbb{R} \rightarrow \mathbb{R}$  eine zweimal stetig differenzierbare Funktion und  $x^{(0)}$  eine Näherung einer zu suchenden Nullstelle  $x^*$ . Des weiteren gelte  $f'(x^{(0)}) \neq 0$ . Zur Herleitung des Newton-Verfahrens verwenden wir die Taylorentwicklung von  $f$  um  $x^{(0)}$ . Demzufolge ist für Werte  $x$  in der Nachbarschaft von  $x^{(0)}$

$$f(x) \approx f(x^{(0)}) + (x - x^{(0)}) \cdot f'(x^{(0)}).$$

Setzen wir  $x := x^*$  und  $f(x^*) = 0$  ein, so folgt

$$f(x^{(0)}) + (x^* - x^{(0)}) \cdot f'(x^{(0)}) \approx 0$$

und damit

$$x^* \approx x^{(0)} - f(x^{(0)})/f'(x^{(0)}).$$

Dies führt auf die folgende Iterationsvorschrift.

**Newton-Verfahren:** Der Punkt  $x^{(0)}$  liege hinreichend nahe an der gesuchten Nullstelle  $x^*$ . Ist der Iterationswert  $x^{(k)}$  gegeben, so definiere  $x^{(k+1)}$  durch

$$x^{(k+1)} := x^{(k)} - f(x^{(k)})/f'(x^{(k)}). \quad (4.4)$$

Die geometrische Interpretation ist: Zu gegebenem  $x^{(k)}$  ist  $x^{(k+1)}$  die Nullstelle der Tangente an  $f$  im Punkt  $x^{(k)}$ .

**Beispiel:** Gesucht ist eine numerische Näherung von  $\sqrt{a}$ ,  $a > 0$ . Offenbar ist  $x^* = \sqrt{a}$  eine Nullstelle der Funktion  $f(x) = x^2 - a$ . Das Newton-Verfahren liefert als Iterationsvorschrift:

$$x^{(k+1)} = x^{(k)} - \frac{(x^{(k)})^2 - a}{2x^{(k)}} = \frac{1}{2} \left( x^{(k)} + \frac{a}{x^{(k)}} \right).$$

Ein Zahlenbeispiel für  $a = 0.81$  und  $x^{(0)} = 1.0$  ist in Tabelle 2 angegeben.

## B – Systeme nichtlinearer Gleichungen

$k$	0	1	2	3
$x^{(k)}$	1.0...	0.905...	0.90001...	0.90000000001...

Tabelle 2: Berechnung von  $\sqrt{0.81}$  mit dem Newton-Verfahren

Für Funktionen  $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ ,

$$f(x) = \begin{pmatrix} f_1(x) \\ \vdots \\ f_n(x) \end{pmatrix} \quad \text{mit } x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

gelten die gleichen Argumente wie im skalaren Fall. Ist  $f$  hinreichend regulär, so gilt nach der Taylor-Formel für  $x$  nahe bei  $x^{(0)}$

$$f(x) \approx f(x^{(0)}) + Df(x^{(0)}) \cdot (x - x^{(0)}).$$

Hierbei ist  $Df$  die Jacobi-Matrix, definiert durch

$$Df = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \dots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}.$$

Die Nullstelle der auf der rechten Seite der Taylor-Entwicklung auftretenden linearen Abbildung  $f(x^{(0)}) + Df(x^{(0)}) \cdot (x - x^{(0)})$  ist

$$x^* = x^{(0)} - (Df)^{-1}(x^{(0)}) \cdot f(x^{(0)}).$$

Hieraus ergibt sich – analog zum eindimensionalen Fall – das folgende Iterationsschema.

**Newton-Verfahren**, Iterationsschritt: Gegeben  $x^{(k)}$ , berechne  $x^{(k+1)}$  durch die folgenden Schritte.

S1: Berechne  $f(x^{(k)})$  und Jacobi-Matrix  $Df(x^{(k)})$ .

S2: Berechne die Lösung  $s$  des linearen Gleichungssystems  $Df(x^{(k)}) \cdot s = -f(x^{(k)})$ .

S3: Definiere  $x^{(k+1)} := x^{(k)} + s$ .

**Beispiele:** (a) Gesucht ist eine Nullstelle  $x^*$  der Funktion

$$f(x, y) := \begin{pmatrix} x \exp(y) - 1 \\ \sin(x) - y \end{pmatrix}.$$

Die Jacobi-Matrix von  $f$  ist

$$Df(x, y) = \begin{pmatrix} \exp(y) & x \exp(y) \\ \cos(x) & -1 \end{pmatrix}.$$

Mit dem Startwert

$$\begin{pmatrix} x^{(0)} \\ y^{(0)} \end{pmatrix} := \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

liefert das Newton-Verfahren die folgenden Werte.

$$\begin{aligned} \begin{pmatrix} x^{(1)} \\ y^{(1)} \end{pmatrix} &= \begin{pmatrix} 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 & 0 \\ 1 & -1 \end{pmatrix}^{-1} \begin{pmatrix} -1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \\ \begin{pmatrix} x^{(2)} \\ y^{(2)} \end{pmatrix} &= \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \begin{pmatrix} \exp(1) & \exp(1) \\ \cos(1) & -1 \end{pmatrix}^{-1} \begin{pmatrix} \exp(1) - 1 \\ \sin(1) - 1 \end{pmatrix} = \begin{pmatrix} 0.693\dots \\ 0.675\dots \end{pmatrix}, \\ &\vdots \\ \begin{pmatrix} x^{(5)} \\ y^{(5)} \end{pmatrix} &= \dots = \begin{pmatrix} 0.578713\dots \\ 0.546947\dots \end{pmatrix}. \end{aligned}$$

Dieser letzte Wert hat den Funktionswert  $f(x^{(5)}, y^{(5)}) = (0 \quad -1.15\text{E-}7)^T$  und ist eine sehr gute Näherung der gesuchten Nullstelle.

(b) Gesucht ist das Maximum der Funktion  $H : \mathbb{R}^2 \rightarrow \mathbb{R}$ ,

$$H(x, y) = -5x^4 + 2(x - 1)^3(y + 1) - 100y^2.$$

Eine notwendige Bedingung, dass das Maximum im Punkt  $(x^*, y^*)^T$  angenommen wird, ist, dass die partiellen Ableitungen

$$\frac{\partial H(x^*, y^*)}{\partial x} \quad \text{und} \quad \frac{\partial H(x^*, y^*)}{\partial y}$$

verschwinden. Gesucht ist damit eine Nullstelle der Funktion

$$f(x, y) := \begin{pmatrix} \frac{\partial H(x^*, y^*)}{\partial x} \\ \frac{\partial H(x^*, y^*)}{\partial y} \end{pmatrix} = \begin{pmatrix} -20x^3 + 6(x-1)^2(y+1) \\ 2(x-1)^3 - 200y \end{pmatrix}.$$

Diese kann mit Hilfe des Newton-Verfahrens berechnet werden.

### C – Konvergenz des Newton-Verfahrens

Das Newton-Verfahren ist das in der Praxis wichtigste Verfahren und ist für skalare Probleme im Fall der Konvergenz den in den vorhergehenden Abschnitten beschriebenen Verfahren deutlich überlegen. Voraussetzung sind allerdings ein hinreichend guter Startwert und eine genügend glatte Funktion, sowie eine "reguläre" Nullstelle. Dies wird im folgenden Satz präzisiert<sup>6</sup>.

**Satz:**  $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$  sei eine zweimal stetig differenzierbare Funktion<sup>7</sup>. Gesucht sei eine Nullstelle  $x^*$  von  $f$ ;  $x^*$  sei "regulär" in dem Sinne, dass  $Df(x^*)$  eine reguläre Matrix ist. Dann konvergiert das Newton-Verfahren **lokal quadratisch**; das bedeutet: Es gibt Konstanten  $\gamma > 0$  und  $C > 0$  derart, dass gilt

$$\|x^{(k+1)} - x^*\| \leq C \|x^{(k)} - x^*\|^2,$$

wenn der Anfangsfehler hinreichend klein war in dem Sinne, dass

$$\|x^{(0)} - x^*\| \leq \gamma.$$

Die Aussage der lokal quadratischen Konvergenz muss kurz kommentiert werden.

**Bemerkungen:** (a) Das Newton-Verfahren konvergiert nur, wenn der Startwert hinreichend gut ist; in der Regel ist hierbei nur schwer zu bestimmen, wie groß  $\gamma$  gewählt werden kann. Eine gute Startnäherung kann im skalaren Fall unter Umständen durch

---

<sup>6</sup>Zum Beweis vergleiche man z.B. Satz 3.25 in H.-G. Roos/H. Schwetlick, Numerische Mathematik, Teubner, 1999.

<sup>7</sup>Dies bedeutet, dass alle partiellen Ableitungen von  $f$  bis zur Ordnung 2 existieren und stetig sind.

eines der in Abschnitt 4.1 beschriebenen Verfahren ermittelt werden, im mehrdimensionalen Fall möglicherweise durch Zusatzinformationen oder graphische Hilfsmittel.

**(b)** Um uns die Bedeutung der quadratischen Konvergenz zu veranschaulichen, nehmen wir an, dass für die Konstante  $C$  der Wert 1 gewählt werden kann. Ist in diesem Fall die Näherung  $x^{(k)}$  auf  $r$  Stellen genau, also z.B.  $\|x^{(k)} - x^*\| \leq 10^{-r}$ , so ist  $\|x^{(k+1)} - x^*\| \leq 10^{-2r}$ ; die Anzahl der exakten Stellen verdoppelt sich also in jedem Iterationsschritt.

## 5 Interpolation und Approximation

In diesem Kapitel geht es um die Frage, wie Funktionen rekonstruiert werden können, von welchen nur gewisse Funktionswerte bekannt sind. Anwendungen sind z.B.:

- Aus Messungen im Labor sind nur diskrete Werte  $f(t_1), f(t_2), \dots$  bekannt;
- Funktionswerte einer Funktion sind tabellarisch aufgelistet;
- eine kontinuierliche Funktion soll abgespeichert und später rekonstruiert werden.

### 5.1 Polynominterpolation

Zu den Zeitpunkten  $t_0, \dots, t_n$  seien die Funktionswerte  $f_i := f(t_i), i = 0, \dots, n$ , bekannt. Gesucht ist ein Polynom  $P(t)$  möglichst niedrigen Grades, welches die **Interpolationseigenschaft** erfüllt:

$$P(t_i) = f_i, \quad i = 0, \dots, n.$$

Es gilt das folgende allgemeine Resultat.

**Satz:** Es gibt genau ein Polynom  $P(t)$   $n$ -ten Grades, welches die *Interpolationsbedingungen*

$$P(t_i) = f_i, \quad i = 0, \dots, n \tag{5.1}$$

erfüllt.

Auf den Beweis der *Eindeutigkeit* des Interpolationspolynoms wollen wir hier nicht eingehen. Zum Beweis der *Existenz* werden wir das Polynom konstruieren. Hierzu betrachten wir die folgenden **Lagrange-Polynome**:

$$L_i(t) := \prod_{\substack{j=0 \\ j \neq i}}^n \frac{t - t_j}{t_i - t_j}.$$

Diese Polynome haben die folgenden Eigenschaften.

**Eigenschaften** der Lagrange-Polynome:

(a)  $L_i$  ist Polynom  $n$ -ten Grades.

(b) An den Knoten  $t_j$ ,  $j = 0, \dots, n$ , nimmt  $L_i$  die folgenden Werte an:

$$L_i(t_j) = \delta_{ij} = \begin{cases} 1 & \text{für } j = i \\ 0 & \text{für } j \neq i \end{cases}$$

( $\delta_{ij}$  wird als *Kronecker-Delta* bezeichnet.)

Eine unmittelbare Folge ist

**Folgerung:** Das gesuchte **Interpolationspolynom** ist gegeben durch

$$P(t) = \sum_{i=0}^n f_i L_i(t).$$

**Beispiele:** (a) Die zu den Knoten  $t_0 = 0$ ,  $t_1 = 1$ ,  $t_2 = 3$  und  $t_3 = 4$  gehörigen Lagrange-Polynome sind

$$\begin{aligned} L_0(t) &= -\frac{1}{12}(t-1)(t-3)(t-4) = -\frac{1}{12}(t^3 - 8t^2 + 19t - 12), \\ L_1(t) &= \frac{1}{6}t(t-3)(t-4) = \frac{1}{6}(t^3 - 7t^2 + 12t), \\ L_2(t) &= -\frac{1}{6}t(t-1)(t-4) = -\frac{1}{6}(t^3 - 5t^2 + 4t), \\ L_3(t) &= \frac{1}{12}t(t-1)(t-3) = \frac{1}{12}(t^3 - 4t^2 + 3t). \end{aligned}$$

Das Interpolationspolynom zu den Werten  $f_0 = 1$ ,  $f_1 = 0$ ,  $f_2 = -1$  und  $f_3 = 2$  ist

$$P(t) = L_0(t) - L_2(t) + 2L_3(t) = \frac{1}{12}(3t^3 - 10t^2 - 5t + 12).$$

(b) Der Wert  $\sin(66.3^\circ)$  soll bestimmt werden. In einem Tafelwerk finden wir die tabellierten Werte

$$\sin(60^\circ) = 0.866025$$



wobei die  $\pi_{ik}$  für  $k \geq 1$  aus den Werten der vorhergehenden Spalte bestimmt werden nach der Rekursionsformel

$$\pi_{ik} = \pi_{i,k-1} + \frac{\tau - t_i}{t_i - t_{i-k}} \cdot (\pi_{i,k-1} - \pi_{i-1,k-1}).$$

Dann ist  $P(\tau) = \pi_{nn}$ .<sup>8</sup>

**Beispiel:** Der Wert  $P_3(66.3)$  des obigen Beispiels (b) folgt somit aus dem Schema

$$\begin{aligned} \sin(60^\circ) &= 0.866025 \\ \sin(65^\circ) &= 0.906308 \quad 0.9167816 \\ \sin(70^\circ) &= 0.939693 \quad 0.9149881 \quad 0.9156517 \\ \sin(75^\circ) &= 0.965926 \quad 0.9202806 \quad 0.9156761 \quad 0.9156619. \end{aligned}$$

Aus dem Schema von Neville kann man sich leicht den folgenden Algorithmus herleiten.

**Algorithmus** zum Schema von Neville:

(S1) Für  $k = 0(1)n$ : Setze  $p_k := f_k$ .

(S2) Für  $k = 1(1)n$

Für  $i = n(-1)k$ :

Setze  $p_i := p_i + (t - t_i) \cdot (p_i - p_{i-1}) / (t_i - t_{i-k})$ .

**Bemerkungen** zum Interpolationsfehler: (a)  $f : [a, b] \rightarrow \mathbb{R}$  sei die zu interpolierende Funktion zu den Knoten  $t_0, \dots, t_n$ ;  $f$  sei  $(n+1)$ -mal stetig differenzierbar.  $P(t)$  sei das zugehörige Interpolationspolynom. Das Polynom  $\omega(t)$  sei definiert durch

$$\omega(t) := \prod_{i=0}^n (t - t_i). \quad (5.2)$$

Mittels des Maximums der  $(n+1)$ -ten Ableitung,

$$\|f^{(n+1)}\|_\infty := \max_{a \leq t \leq b} |f^{(n+1)}(t)|$$

---

<sup>8</sup>Die Werte  $\pi_{kk}$  sind die Werte des Interpolationspolynoms zu den Knoten  $t_0, \dots, t_k$  an der Stelle  $\tau$ .

und des Polynoms  $\omega$  lässt sich der Interpolationsfehler wie folgt abschätzen. Es gilt für alle  $t \in [a, b]$ :

$$|P(t) - f(t)| \leq \frac{\|f^{(n+1)}\|_\infty}{(n+1)!} \cdot |\omega(t)|. \quad (5.3)$$

**(b) Für äquidistante Knoten**  $a = t_0 < t_1 < \dots < t_n = b$  und für große  $n$  neigen Interpolationspolynome und das Polynom  $\omega(t)$  zu großen Oszillationen, insbesondere in der Nähe der Randpunkte  $a$  und  $b$ . Interpolationspolynome hohen Grades zu äquidistanten Knoten sind daher praktisch unbrauchbar. (Anders verhält es sich optimal angeordneten Knoten im Fall der Tschebyscheff-Interpolation; vgl. Abschnitt 5.2.)

## 5.2 Tschebyscheff-Interpolation

Wie in Abschnitt 5.1 erwähnt wurde, sind Interpolationspolynome hohen Grades zu äquidistanten Stützstellen praktisch unbrauchbar, da diese starke Oszillationen aufweisen. Die Oszillationen stehen im Zusammenhang mit dem Verhalten des durch (5.2) definierten Polynoms  $\omega(t)$  in der Nähe der Ränder. Die Untersuchung der Frage, wie die Knoten  $t_0, \dots, t_n$  anzuordnen sind, damit

$$\max_{t \in [t_0, t_n]} |\omega(t)|$$

möglichst klein wird, führt auf die *Tschebyscheff-Polynome*.

**Definition:** Die Funktion  $T_n : [-1, 1] \rightarrow \mathbb{R}$ ,

$$T_n(t) = \cos(n \cdot \arccos(t))$$

heißt das **Tschebyscheff-Polynom  $n$ -ten Grades**.

Zunächst einmal ist nicht klar, wieso  $T_n$  ein Polynom sein soll. Dies und weitere Eigenschaften werden im folgenden Satz gezeigt. Zunächst wird eine wichtige Rekursionseigenschaft bewiesen.

**Hilfssatz:** Es ist

$$T_0(t) = 1 \quad \text{und} \quad T_1(t) = t.$$

Für  $n \geq 1$  gilt die **Drei-Term-Rekursion**

$$T_{n+1}(t) = 2tT_n(t) - T_{n-1}(t). \quad (5.4)$$

**Beweis:** Die Aussagen für  $n = 0$  und  $n = 1$  folgen aus den wohlbekanntenen Beziehungen

$$\cos(0) = 1, \quad \cos(\arccos(t)) = t.$$

Zur Herleitung der Drei-Term-Rekursion benötigen wir das Additionstheorem für den Kosinus,

$$\cos(a \pm b) = \cos(a) \cos(b) \mp \sin(a) \sin(b).$$

Mit  $a = n \cdot \arccos(t)$  und  $b = \arccos(t)$  folgt

$$\begin{aligned} \cos((n+1) \arccos(t)) &= \cos(n \cdot \arccos(t)) \cos(\arccos(t)) - \sin(n \cdot \arccos(t)) \sin(\arccos(t)), \\ \cos((n-1) \arccos(t)) &= \cos(n \cdot \arccos(t)) \cos(\arccos(t)) + \sin(n \cdot \arccos(t)) \sin(\arccos(t)). \end{aligned}$$

Aufsummieren der beiden Gleichungen ergibt

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

Hieraus folgt die Drei-Term-Rekursion.  $\square$

**Satz:** Die Tschebyscheff-Polynome erfüllen für  $t \in [-1, 1]$  die folgenden Eigenschaften.

(a)  $T_n$  ist ein Polynom  $n$ -ten Grades und ist damit darstellbar in der Form

$$T_n(t) = a_n^{(n)} t^n + a_{n-1}^{(n)} t^{n-1} + \cdots + a_1^{(n)} t + a_0^{(n)}.$$

(b) Für die führenden Koeffizienten gilt

$$a_n^{(n)} = 2^{n-1} \quad \text{für } n \geq 1.$$

(c)  $T_n$  ist beschränkt durch 1:

$$\max_{t \in [-1, 1]} |T_n(t)| = 1.$$

(d)  $T_n$  hat die  $n$  paarweise verschiedenen Nullstellen

$$t_k^{(n)} = \cos\left(\frac{2k-1}{2n} \cdot \pi\right), \quad k = 1, \dots, n.$$

**Beweis:** Zu (a), (b): Für  $n = 0$  und  $n = 1$  wurden die Aussagen im Hilfssatz gezeigt. Der Beweis für  $n > 1$  folgt durch vollständige Induktion aus der Drei-Term-Rekursion. Zu (c): Die Funktionswerte von  $T_n$  sind Werte der Kosinus-Funktion und daher durch 1 beschränkt. An den Stellen  $\cos(k\pi/n)$  nimmt  $|T_n|$  den Wert 1 an.

Zu (d):

$$T_n(t_k^{(n)}) = \cos\left(n \cdot \frac{(2k-1)\pi}{2n}\right) = \cos\left(\left(k - \frac{1}{2}\right)\pi\right) = 0. \quad \square$$

**Bemerkung:** Mit Hilfe des Hilfssatzes können die Tschebyscheff-Polynome leicht berechnet werden. Es ist

$$\begin{aligned} T_0(t) &= 1 \\ T_1(t) &= t \\ T_2(t) &= 2t^2 - 1 \\ T_3(t) &= 4t^3 - 3t \\ T_4(t) &= 8t^4 - 8t^2 + 1 \\ &\vdots \end{aligned}$$

Werden die Nullstellen  $t_k^{(n)}$  des  $n$ -ten Tschebyscheff-Polynoms als Knoten zur Interpolation gewählt, so können die Interpolationsfehler nicht groß werden; aus dem Satz folgt nämlich

**Folgerung:**  $t_k^{(n)}$  seien die Nullstellen von  $T_n$  aus Teil (d) des Satzes. Dann gilt für

$$\omega(t) = \prod_{k=1}^n (t - t_k^{(n)})$$

die Abschätzung

$$|\omega(t)| \leq \frac{1}{2^{n-1}}.$$

Ohne einen Beweis hierüber führen zu wollen, sei abschließend bemerkt:

Bezüglich der Minimierung des maximalen Interpolationsfehlers durch ein Polynom  $n$ -ten Grades im Intervall  $[-1, 1]$  bilden die Nullstellen des  $(n+1)$ -ten Tschebyscheff-Polynoms die optimale Wahl von Knoten.

## 5.3 Spline-Interpolation

### 5.3.1 Polynom-Splines

Die Idee der Spline-Interpolation ist es, das Interpolationsproblem nicht durch ein einziges Polynom vom Grad  $n$  zu lösen (da – wie wir gesehen haben – Polynome hohen Grades zu starken Oszillationen neigen können), sondern durch eine möglichst glatte Funktion, welche sich stückweise aus Polynomen niedrigeren Grades zusammensetzt. Die Vorgehensweise soll zunächst an einem Beispiel demonstriert werden.

**Beispiel:** Gegeben seien die Punktepaare  $(t_i, f_i)$ ,  $i = 0, \dots, 3$ , mit den Werten  $(t_0, f_0) = (0, 1)$ ,  $(t_1, f_1) = (2, 2)$ ,  $(t_2, f_2) = (3, 0)$  und  $(t_3, f_3) = (5, 0)$ . Diese sollen durch eine stetige zusammengesetzte Funktion  $P_k$  interpoliert werden, welche in jedem Intervall  $[t_i, t_{i+1}]$  ein Polynom  $p_i^{(k)}$   $k$ -ten Grades ist.

(a)  $k = 1$ : Das Polynom  $p_i^{(1)}$  1. Grades durch die Punkte  $(t_i, f_i)$  und  $(t_{i+1}, f_{i+1})$  ist gegeben durch die Gleichung

$$\frac{p_i^{(1)}(t) - f_i}{t - t_i} = \frac{f_{i+1} - f_i}{t_{i+1} - t_i}.$$

Hieraus folgt als interpolierende Funktion

$$P_1(t) = \begin{cases} 0.5t + 1 & \text{für } t \in [0, 2] \\ -2t + 6 & \text{für } t \in [2, 3] \\ 0 & \text{für } t \in [3, 5] \end{cases}$$

Diese Funktion ist stetig, in den Interpolationspunkten aber nicht differenzierbar.

(b)  $k = 2$ : Für das Polynom zweiten Grades in  $[t_i, t_{i+1}]$  machen wir den Ansatz

$$p_i^{(2)}(t) = a_i t^2 + b_i t + c_i.$$

Einsetzen der Interpolationsbedingungen

$$p_i^{(2)}(t_i) = f_i, \quad p_i^{(2)}(t_{i+1}) = f_{i+1}$$

führt auf die 6 linearen Gleichungen für die 9 unbekanntenen Koeffizienten  $a_i$ ,  $b_i$  und  $c_i$ :

$$c_0 = 1, \quad 4a_0 + 2b_0 + c_0 = 2, \quad (5.5)$$

$$4a_1 + 2b_1 + c_1 = 2, \quad 9a_1 + 3b_1 + c_1 = 0, \quad (5.6)$$

$$9a_2 + 3b_2 + c_2 = 0, \quad 25a_2 + 5b_2 + c_2 = 0. \quad (5.7)$$

Für eine eindeutige Lösung benötigen wir 3 weitere Bedingungen. Hiervon verwenden wir zwei Bedingungen, um zu erreichen, dass die Funktion an den Interpolationspunkten stetig differenzierbar ist. Dies wird erreicht durch die Bedingungen

$$\frac{dp_i^{(2)}}{dt}(t_{i+1}) = \frac{dp_{i+1}^{(2)}}{dt}(t_{i+1}) \quad \text{für } i = 1, 0,$$

also durch

$$4a_0 + b_0 = 4a_1 + b_1, \quad 6a_1 + b_1 = 6a_2 + b_2. \quad (5.8)$$

Das immer noch unterbestimmte Gleichungssystem (5.5)  $\cdots$  (5.8) kann nach einem freien Parameter – z.B.  $a_0$  – aufgelöst werden:

$$\begin{aligned} b_0 &= 0.5 - 2a_0, & c_0 &= 1, \\ a_1 &= -2.5 - 2a_0, & b_1 &= 10.5 + 10a_0, & c_1 &= -9 - 12a_0, \\ a_2 &= 2.25 + a_0, & b_2 &= -18 - 8a_0, & c_2 &= 33.75 + 15a_0. \end{aligned}$$

Beispielsweise erhalten wir für  $a_0 = 0$  die Lösung

$$P_2(t) = \begin{cases} 0.5t + 1 & \text{für } t \in [0, 2] \\ -2.5t^2 + 10.5t - 9 & \text{für } t \in [2, 3] \\ 2.25t^2 - 18t + 33.75 & \text{für } t \in [3, 5] \end{cases}$$

während die Wahl  $a_0 = 0.25$  auf die Lösung

$$P_2(t) = \begin{cases} 0.25t^2 + 1 & \text{für } t \in [0, 2] \\ -13t^2 + 13t - 12 & \text{für } t \in [2, 3] \\ 2.5t^2 - 20t + 37.5 & \text{für } t \in [3, 5] \end{cases}$$

führt.

**Definition:**  $a = t_0 < t_1 < \dots < t_n = b$  sei eine aufsteigende Folge von Knoten. Ein **Spline vom Grad  $k$**  ist eine  $(k - 1)$ -mal stetig differenzierbare Funktion, welche auf jedem der Intervalle  $[t_i, t_{i+1}]$  ein Polynom (höchstens)  $k$ -ten Grades ist.

**Bemerkung:** Zu vorgegebenen Datenpaaren  $(t_i, f_i)$ ,  $i = 0, \dots, n$ , gibt es stets einen interpolierenden Spline vom Grad  $k$ . Für  $k \geq 1$  ist das Interpolationsproblem nicht eindeutig, die Lösungen des homogenen Problems (d.h.  $f_i = 0$  für  $i = 0, \dots, n$ ) bilden einen  $(k - 1)$ -dimensionalen Raum. Zur eindeutigen Lösbarkeit sind die Interpolationsbedingungen daher um  $k - 1$  weitere Bedingungen zu ergänzen.

### 5.3.2 Kubische Splines

In der Praxis spielen kubische Splines  $P_3$  eine wichtige Rolle. Zu ihrer Konstruktion wählen wir für die Intervalle  $[t_i, t_{i+1}]$  den Ansatz

$$p_i^{(3)}(t) = \alpha_i(t - t_i)^3 + \beta_i(t - t_i)^2 + \gamma_i(t - t_i) + \delta_i, \quad i = 0, \dots, n - 1.$$

Die Koeffizienten  $\delta_i$  folgen unmittelbar aus den *Interpolationsbedingungen*:

$$\delta_i = f_i.$$

Beziehungen zwischen den übrigen Koeffizienten erhalten wir aus den *Stetigkeitsbedingungen* für  $P_3(t)$ ,  $P_3'(t)$  und  $P_3''(t)$  an den Knotenpunkten  $t_{i+1}$ :

$$\begin{aligned} I & \quad \alpha_i h_i^3 + \beta_i h_i^2 + \gamma_i h_i + f_i = f_{i+1} & (i = 0, \dots, n - 1), \\ II & \quad 3\alpha_i h_i^2 + 2\beta_i h_i + \gamma_i = \gamma_{i+1} & (i = 0, \dots, n - 2), \\ III & \quad 6\alpha_i h_i + 2\beta_i = 2\beta_{i+1} & (i = 0, \dots, n - 2), \end{aligned} \tag{5.9}$$

wobei wir zur Abkürzung definiert haben:

$$h_i = t_{i+1} - t_i.$$

Mit Hilfe der Gleichungen *I* und *III* lassen sich die Größen  $\alpha_i$  und  $\gamma_i$  durch  $\beta_i$  und  $\beta_{i+1}$  ausdrücken gemäß

$$(i) \alpha_i = \frac{\beta_{i+1} - \beta_i}{3h_i}, \quad (ii) \gamma_i = -\frac{h_i}{3}(\beta_{i+1} + 2\beta_i) + \frac{f_{i+1} - f_i}{h_i} \quad (i = 0, \dots, n-2). \quad (5.10)$$

Einsetzen in *II* führt schließlich auf die  $n-2$  Gleichungen

$$r_i\beta_i + 2\beta_{i+1} + (1-r_i)\beta_{i+2} = q_i \quad (i = 0, \dots, n-3) \quad (5.11)$$

für die  $n$  Unbekannten  $\beta_i$ , wobei zur Abkürzung

$$r_i = \frac{h_i}{h_{i+1} + h_i}, \quad q_i = \frac{3}{h_i + h_{i+1}} \cdot \left( \frac{f_{i+2} - f_{i+1}}{h_{i+1}} - \frac{f_{i+1} - f_i}{h_i} \right)$$

verwendet wurde. Beachten Sie, dass auch  $\gamma_{n-1}$  mit Hilfe der Gleichung *II* ( $i = n-2$ ) und der Beziehungen (5.11) durch die Koeffizienten  $\beta_i$  beschrieben werden kann:

$$\gamma_{n-1} = \frac{h_{n-2}}{3}(2\beta_{n-1} + \beta_{n-2}) + \frac{f_{n-1} - f_{n-2}}{h_{n-2}}. \quad (5.12)$$

Durch Einsetzen in *I* ( $i = n-1$ ) folgt außerdem

$$\alpha_{n-1}h_{n-1}^2 = -\beta_{n-1} \left( h_{n-1} + \frac{2}{3}h_{n-2} \right) - \frac{1}{3}\beta_{n-2}h_{n-2} + \frac{h_{n-1} + h_{n-2}}{3}q_{n-2}. \quad (5.13)$$

Damit können alle Koeffizienten berechnet werden, sobald die  $\beta_i$  bestimmt sind.

Für ein vollständiges Gleichungssystem fehlen zwei weitere Bedingungen, welche häufig in Form zusätzlicher Randbedingungen ergänzt werden.

### A – Natürliche Randbedingungen:

Hier wird gefordert, dass die zweiten Ableitungen der Spline-Funktion an den Randpunkten verschwinden.

<b>Natürliche Randbedingungen:</b> (i) $P''(a) = 0$ , (ii) $P''(b) = 0$ .
---

Diese Randbedingungen führen auf das folgende diagonal dominante Tridiagonalsystem.

**Satz:** Unter natürlichen Randbedingungen ist der Vektor  $(\beta_1, \dots, \beta_{n-1})^T$  Lösung des





**Beispiel:** Zu den Knoten  $\tau_0 < \tau_1 < \tau_2 < \tau_3$  seien die Werte  $f(\tau_0)$ ,  $f'(\tau_0)$ ,  $f(\tau_1)$ ,  $f'(\tau_1)$ ,  $f''(\tau_1)$ ,  $f(\tau_2)$ ,  $f(\tau_3)$  und  $f'(\tau_3)$  vorgegeben. Hieraus ergibt sich mit  $t_0 = \tau_0$ ,  $t_2 = \tau_1$ ,  $t_5 = \tau_2$ ,  $t_6 = \tau_3$  die folgende Knotenfolge für die Hermite-Interpolation.

$\tau_i$	$\tau_0$	$\tau_0$	$\tau_1$	$\tau_1$	$\tau_1$	$\tau_2$	$\tau_3$	$\tau_3$
$t_i$	$t_0$	$= t_1$	$< t_2$	$= t_3$	$= t_4$	$< t_5$	$< t_6$	$= t_7$
$d_i$	0	1	0	1	2	0	0	1
$\xi_i$	$f(\tau_0)$	$f'(\tau_0)$	$f(\tau_1)$	$f'(\tau_1)$	$f''(\tau_1)$	$f(\tau_2)$	$f(\tau_3)$	$f'(\tau_3)$

Definieren wir nun  $\xi = (\xi_0, \dots, \xi_n)^T$  mit  $\xi_i := f_i^{(d_i)}$  und die Abbildungen

$$\mu_i : \mathcal{P}_n \longrightarrow \mathbb{R}, \quad \mu_i(P) := P^{(d_i)}(t_i), \quad i = 0, \dots, n$$

so lautet das neue Interpolationsproblem

**Aufgabe der Hermite-Interpolation:** Finde  $P \in \mathcal{P}_n$  mit

$$\mu_i(P) = \xi_i.$$

Die Lösung  $P =: P(\xi|t_0, \dots, t_n)$  heißt **Hermite-Interpolierende** zu den Daten  $\xi$  an den Knoten  $t_i$ .

Ähnlich wie früher kann gezeigt werden

**Satz:** Zu jedem Vektor  $\xi \in \mathbb{R}^{n+1}$  und zu jeder monotonen Knotenfolge

$$a = t_0 \leq t_1 \leq \dots \leq t_n = b$$

gibt es genau eine Hermite-Interpolierende  $P(\xi|t_0, \dots, t_n)$ .

Zur Lösung dieses Interpolationsproblems sind die Lagrange-Polynome nicht geeignet. Stattdessen definieren wir eine andere Klasse von Polynomen – die **Newton-Polynome**

(**Newton-Basis**). Diese sind für eine gegebene Knotenfolge  $\{\tau_0, \dots, \tau_n\}$  gegeben als die Polynome  $\omega_i$ ,  $i = 0, \dots, n$ , definiert durch

$$\omega_0(t) := 1, \quad \omega_i(t) := \prod_{j=0}^{i-1} (t - t_j) \quad (\omega_i \in \mathcal{P}_i)$$

Der führende Koeffizient  $a_n$  des Interpolationspolynoms

$$P(\xi|t_0, \dots, t_n) = a_n t^n + a_{n-1} t^{n-1} + \dots + a_0$$

zu den Knoten  $t_0 \leq t_1 \leq \dots \leq t_n$  heißt *n-te dividierte Differenz* und wird mit

$$\xi[t_0, \dots, t_n] := a_n$$

bezeichnet.

**Satz:** Bezüglich der Newton-Basis besitzt  $P(\xi|t_0, \dots, t_n)$  die Darstellung

$$P_n := P(\xi|t_0, \dots, t_n) = \sum_{i=0}^n \xi[t_0, \dots, t_i] \cdot \omega_i \quad .$$

Ähnlich wie Funktionswerte des Interpolationspolynoms mit Hilfe des Aitken-Neville-Schemas können die dividierten Differenzen rekursiv durch das folgende Schema berechnet werden.

**Berechnung der dividierten Differenzen:** Die Berechnung erfolgt durch das Schema

$$\begin{array}{cccccccc}
 \xi_0 & = & \xi[t_0] & & & & & \\
 & & & \searrow & & & & \\
 \xi_{1-d_1} & = & \xi[t_1] & \rightarrow & \xi[t_0, t_1] & & & \\
 & & \vdots & & \ddots & & & \\
 & & \vdots & & & \ddots & & \\
 & & \vdots & & & & \ddots & \\
 \xi_{n-1-d_{n-1}} & = & \xi[t_{n-1}] & \rightarrow & \xi[t_{n-2}, t_{n-1}] & \rightarrow & \dots & \rightarrow & \xi[t_0, \dots, t_{n-1}] \\
 & & & \searrow & & \searrow & \searrow & & \searrow \\
 \xi_{n-d_n} & = & \xi[t_n] & \rightarrow & \xi[t_{n-1}, t_n] & \rightarrow & \dots & \rightarrow & \xi[t_1, \dots, t_n] & \rightarrow & \xi[t_0, \dots, t_n]
 \end{array}$$

mit den folgenden "Rechenregeln".

(a) Für zusammenfallende Knoten  $t_k = \dots = t_{k+l}$  ist

$$\xi[t_k, \dots, t_{k+l}] = \frac{\xi_{k+l}}{l!} .$$

(b) Ist  $t_i \neq t_j$ , so gilt die Rekursionsformel

$$\xi[t_0, \dots, t_n] = \frac{\xi[t_0, \dots, \hat{t}_i, \dots, t_n] - \xi[t_0, \dots, \hat{t}_j, \dots, t_n]}{t_j - t_i} ,$$

wobei " $\hat{t}_k$ " bedeutet, dass der Knoten  $t_k$  und in der Liste  $\xi = (\xi_0, \dots, \xi_n)^T$  die Komponente  $\xi_k$  gestrichen werden.

**Beispiele:** (a) Gesucht ist das Polynom  $P(t)$  niedrigsten Grades (zur Approximation von  $\sin(t)$ ), welches in den Punkten  $t = 0$  und  $t = \pi$  die Funktionswerte und ersten Ableitungen von  $\sin(t)$  annimmt. Das entsprechende Schema lautet

$$\begin{array}{cccc} [0] & & & \\ [0] & [1] & & \\ [0] & 0 & -1/\pi & \\ [0] & [-1] & -1/\pi & 0 \end{array}$$

wobei die in eckige Klammern gesetzten Werte durch die Regel (a), die anderen durch die Regel (b) berechnet wurden. Damit ist die Hermite-Interpolierende gegeben durch

$$P(\xi|\mathbf{t}) = 0 \cdot 1 + 1 \cdot t - 1/\pi \cdot t^2 + 0 \cdot t^2 \cdot (t - \pi) = -\frac{1}{\pi} \cdot t(t - \pi) .$$

(b) Zu den Vorgaben aus (a) soll hinzugefügt werden:  $P(\pi/6) = P(5 \cdot \pi/6) = 0.5$ . Hieraus ergibt sich das folgende Schema.

$$\begin{array}{l|cccccc} t_0 = 0 & [0] & & & & & \\ t_1 = 0 & [0] & [1] & & & & \\ t_2 = \pi/6 & [0.5] & 3/\pi & -6/\pi + 18/\pi^2 & & & \\ t_3 = 5\pi/6 & [0.5] & 0 & -18/5\pi^2 & 36/5\pi^2 - 648/25\pi^3 & & \\ t_4 = \pi & [0] & -3/\pi & -18/5\pi^2 & 0 & -36/5\pi^3 + 648/25\pi^4 & \\ t_5 = \pi & [0] & [-1] & -6/\pi + 18/\pi^2 & -36/5\pi^2 + 648/25\pi^3 & -36/5\pi^3 + 648/25\pi^4 & 0 \end{array}$$

Damit ist

$$\begin{aligned} P(\xi|\mathbf{t})(t) &= 1 \cdot \omega_1(t) + \frac{6(3-\pi)}{\pi^2} \cdot \omega_2(t) + \frac{36(5\pi-18)}{25\pi^3} \omega_3(t) - \frac{36(5\pi-18)}{25\pi^4} \omega_4(t) \\ &= t + 0.016104 \cdot t^2 - 0.212895 \cdot t^3 + 0.033883 \cdot t^4 \quad . \end{aligned}$$

(c) Die Exponentialfunktion soll in der Nähe von  $t = 0$  durch ein Polynom  $P(t)$  approximiert werden. Dieses Polynom soll möglichst niedrigen Grades sein und definiert durch die Bedingungen  $P(-1) = 1/e$ ,  $P(0) = P'(0) = P''(0) = 1$  sowie  $P(1) = e$ . Aus dem Schema

$$\begin{array}{l|l} t_0 = -1 & [1/e] \\ t_1 = 0 & [1] \quad (e-1)/e \\ t_2 = 0 & [1] \quad [1] \quad 1/e \\ t_3 = 0 & [1] \quad [1] \quad [1/2] \quad (e-2)/2e \\ t_4 = 0 & [1] \quad [1] \quad [1/2] \quad [1/6] \quad (3-e)/3e \\ t_5 = 1 & [e] \quad e-1 \quad e-2 \quad e-5/2 \quad e-8/3 \quad 0.5 \cdot (e-1/e-7/3) \end{array}$$

folgt

$$\begin{aligned} P(t) &= \frac{1}{e} + \left(1 - \frac{1}{e}\right)(t+1) + \frac{1}{e}(t+1)t + \left(\frac{1}{2} - \frac{1}{e}\right)(t+1)t^2 + \left(\frac{1}{e} - \frac{1}{3}\right)(t+1)t^3 \\ &\quad + 0.5 \cdot \left(e - \frac{1}{e} - \frac{7}{3}\right)(t-1)t^4 \\ &= 1 + t + 0.5t^2 + 0.166667t^3 + 0.043081t^4 + 0.008535t^5 \quad . \end{aligned}$$

## 6 Bestimmte Integrale

In diesem Kapitel seien  $a$  und  $b$  fest gewählte reelle Koeffizienten ( $a < b$ ). Das Ziel ist die Herleitung numerischer Verfahren für das Integral

$$I(f) := \int_a^b f(t) dt$$

für hinreichend glatte Funktionen  $f : [a, b] \rightarrow \mathbb{R}$ .

### 6.1 Quadraturformeln

Gesucht sind Näherungsformeln für  $I(f)$  der Form

$$\hat{I}(f) := (b - a) \cdot \sum_{i=0}^n \lambda_i f(t_i) \quad (6.1)$$

mit geeigneten Knoten

$$a \leq t_0 < t_1 < \dots < t_n \leq b$$

und Gewichten  $\lambda_i$ . Gewisse Einschränkungen an die Gewichte und Knoten ergeben sich aus den folgenden Mindestanforderungen.

**Mindestanforderungen:** (a) Konstante Funktionen sollen exakt integriert werden. Das heißt: Für  $f(\cdot) \equiv c$  ist  $I(f) = \hat{I}(f)$ .

(b) Ist  $f(t) \geq 0$  für alle  $t \in [a, b]$ , so ist auch  $\hat{I}(f) \geq 0$ .

Notwendige und hinreichende Bedingungen, welche sich aus den Mindestanforderungen ergeben, lassen sich leicht herleiten. Ist  $f(t) = c \neq 0$  für alle  $t \in [a, b]$ , so ist  $I(f) = c \cdot (b - a)$  und

$$\hat{I}(f) = (b - a) \cdot c \cdot \sum_{i=0}^n \lambda_i.$$

Die Forderung (a) ist daher äquivalent zur Forderung  $\sum_{i=0}^n \lambda_i = 1$ .

Die Forderung (b) führt zur Bedingung  $\lambda_i \geq 0$ ,  $i = 0, \dots, n$ . Um dies zu sehen, nehmen wir an, dass für ein  $j \in \{0, \dots, n\}$  gilt  $\lambda_j < 0$ . Wir konstruieren die lineare Spline-Funktion  $f$ , welche eindeutig bestimmt ist durch

$$f(t_i) = \begin{cases} 0 & \text{falls } i \neq j \\ 1 & \text{für } i = j. \end{cases}$$

Diese Funktion ist sicherlich nichtnegativ. Für die Integralnäherung gilt

$$\hat{I}(f) = (b - a)\lambda_j < 0.$$

Damit ist die Forderung (b) nicht erfüllt.

Im folgenden betrachten wir nur Näherungsformeln, für welche die Mindestanforderungen erfüllt sind und welche durch die folgende Definition charakterisiert sind.

**Definition:** Eine Formel der Form (6.1) heißt **Quadraturformel für  $f$** , wenn gilt

$$\begin{aligned} \text{(a)} \quad & \sum_{i=0}^n \lambda_i = 1, \\ \text{(b)} \quad & \lambda_i \geq 0, \quad i = 0, \dots, n. \end{aligned}$$

Quadraturformeln können leicht auf heuristischem Weg hergeleitet werden, wie die folgenden Beispiele zeigen.

**Beispiel: (a) Mittelpunktsregel.** Durch die Wahl der Knoten

$$t_i := a + i \cdot \frac{b - a}{N}, \quad i = 0, \dots, N,$$

zerlegen wir das Intervall  $[a, b]$  in  $N$  gleich große Teilintervalle der Länge  $h = (b - a)/N$ ; das Integral über dem  $i$ -ten Teilintervall approximieren wir durch die Fläche des Rechtecks mit Höhe  $f(\tau_i)$ , wobei

$$\tau_i = a + (i - 0.5)h, \quad i = 1, \dots, N$$

der Mittelpunkt des Teilintervalls ist. Dies führt auf die Näherungsformel

$$\hat{I}(f) = h \cdot \sum_{i=1}^N f(\tau_i) = (b - a) \cdot \sum_{i=1}^N \frac{1}{N} f(\tau_i).$$

Dies ist eine Quadraturformel mit den Gewichten  $\lambda_i = 1/N$  ( $i = 1, \dots, N$ ).

**(b) Trapezregel.** Das Integral  $[a, b]$  wird zerlegt wie in (a). Über jedem Teilintervall  $[t_i, t_{i+1}]$  wird die Funktion  $f$  approximiert durch das lineare Interpolationspolynom zu den Knoten  $t_i$  und  $t_{i+1}$ . Die Fläche über dem Teilintervall wird approximiert durch das

hierdurch entstehende Trapez mit der Fläche  $0.5 \cdot h \cdot (f(t_i) + f(t_{i+1}))$ . Damit erhalten wir als Approximation des Integrals  $I(f)$

$$\begin{aligned}\hat{I}(f) &= 0.5 \cdot h \cdot \sum_{i=0}^N (f(t_i) + f(t_{i+1})) \\ &= 0.5 \cdot h \cdot (f(a) + f(b)) + h \cdot \sum_{i=1}^{N-1} f(t_i).\end{aligned}$$

Dies ist eine Quadraturformel mit den Knoten  $t_i$ ,  $i = 0, \dots, N$ , und den Gewichten  $\lambda_0 = \lambda_N = 1/(2N)$  und  $\lambda_1 = \dots = \lambda_{N-1} = 1/N$ .

## 6.2 Newton-Cotes-Formeln

Die den Newton-Cotes-Formeln zugrunde liegende Idee ist es, die Funktion  $f$  durch das Interpolationspolynom zu geeigneten Knoten  $t_0, \dots, t_n$  zu approximieren und als Näherungsformel für  $I(f)$  das exakte Integral des Interpolationspolynoms zu berechnen.

### A – Einfache Newton-Cotes-Formeln

Ist  $L_{in}$  das  $i$ -te Lagrange-Polynom zu den Knoten  $t_0, \dots, t_n$  (vgl. Abschnitt 5.1), so ist das Interpolationspolynom gegeben durch

$$P(t) = \sum_{i=0}^n f(t_i) \cdot L_{in}(t);$$

das Integral von  $P(t)$  erfüllt die Gleichung

$$\int_a^b P(t) dt = (b-a) \cdot \sum_{i=0}^n \lambda_i f(t_i)$$

Dies ist eine Approximationsformel der Form (6.1) mit den Gewichten

$$\lambda_i = \frac{1}{b-a} \int_a^b L_{in}(t) dt. \quad (6.2)$$

**Definition:** Im Falle äquidistanter Knoten

$$t_i = a + i \cdot h, \quad i = 0, \dots, n \quad \text{mit} \quad h = \frac{b-a}{n} \quad (6.3)$$

heißen die Näherungsformeln (6.1) mit den Gewichten (6.2) (**einfache**) **Newton-Cotes-Formeln**.

**Beispiele:** (a)  $n = 1$  : Die Lagrange-Polynome zu den Knoten  $t_0 = a$  und  $t_1 = b$  sind

$$L_{01}(t) = \frac{b-t}{b-a} \quad \text{und} \quad L_{11}(t) = \frac{t-a}{b-a}.$$

Durch Integration folgt

$$\lambda_0 = \lambda_1 = \frac{1}{2}.$$

In Analogie zum Beispiel (b) aus Abschnitt 6.1 wird dieses Verfahren als (**einfache**) **Trapezregel** bezeichnet.

(b)  $n = 2$  : Die Lagrange-Polynome zu den Knoten  $t_0 = a$ ,  $t_1 = 0.5(a+b)$  und  $t_2 = b$  sind

$$\begin{aligned} L_{02}(t) &= \frac{1}{(b-a)^2} \cdot (2t^2 - (a+3b)t + (a+b)b), \\ L_{12}(t) &= \frac{-4}{(b-a)^2} \cdot (t^2 - (a+b)t + ab), \\ L_{22}(t) &= \frac{1}{(b-a)^2} \cdot (2t^2 - (3a+b)t + a(a+b)). \end{aligned}$$

Für die Gewichte folgt

$$\lambda_0 = \lambda_2 = \frac{1}{6}, \quad \lambda_1 = \frac{4}{6}.$$

Das entsprechende Verfahren heißt (**einfache**) **Simpson-Regel**.

Die Gewichte für die Newton-Cotes-Formeln sind für  $n = 1, \dots, 4$  in Tabelle 3 zusammengestellt. Man überzeugt sich leicht, dass dies Quadraturformeln im Sinne der Definition sind. Dies ändert sich ab  $n = 8$ , da hier auch negative Gewichte auftreten. Einfache Newton-Cotes-Formeln sind für große  $n$  nicht sehr sinnvoll.

## B – Fehlerrechnung

Um den Fehler bei der Integration mit Hilfe der Newton-Cotes-Formeln abzuschätzen, erinnern wir uns an die Fehlerabschätzungen bei der Polynom-Interpolation. Ist  $P(t)$  das Interpolationspolynom zu den Knoten  $t_0 < \dots < t_n$  und ist  $f$  mindestens  $(n+1)$ -mal

$n$	Gewichte				Verfahren	
1		$\frac{1}{2}$	$\frac{1}{2}$		Trapezregel	
2		$\frac{1}{6}$	$\frac{4}{6}$	$\frac{1}{6}$	Simpson-Regel	
3	$\frac{1}{8}$		$\frac{3}{8}$	$\frac{3}{8}$	$\frac{1}{8}$	Newtonsche 3/8-Regel
4	$\frac{7}{90}$	$\frac{32}{90}$	$\frac{12}{90}$	$\frac{32}{90}$	$\frac{7}{90}$	Milne-Regel

Tabelle 3: Newton-Cotes-Formeln

differenzierbar, so gibt es zu jedem  $t \in [a, b]$  einen Zwischenwert  $\tau = \tau(t) \in [a, b]$  derart, dass

$$P(t) - f(t) = \frac{f^{(n+1)}(\tau)}{(n+1)!} \cdot \omega(t). \quad (6.4)$$

Für den Fehler bei der numerischen Integration mit einfachen Newton-Cotes-Formeln ergibt sich

$$\hat{I}(f) - I(f) = \int_a^b P(t)dt - \int_a^b f(t)dt = \int_a^b \frac{f^{(n+1)}(\tau(t))}{(n+1)!} \cdot \omega(t)dt. \quad (6.5)$$

Eine weitere Auswertung dieser Ausdrücke ergibt die folgenden Abschätzungen.

#### Fehlerabschätzungen: (a) Trapezregel

$$T(f) = \frac{b-a}{2} (f(a) + f(b))$$

Ist  $f$  zweimal stetig differenzierbar, so gilt mit einem geeigneten  $\tau \in [a, b]$

$$T(f) - \int_a^b f(t)dt = \frac{(b-a)^3}{12} \cdot f''(\tau).$$

#### (b) Simpsonregel

$$S(f) = \frac{b-a}{6} \left( f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right)$$

Ist  $f$  viermal stetig differenzierbar, so gilt mit einem geeigneten  $\tau \in [a, b]$

$$S(f) - \int_a^b f(t)dt = \frac{(b-a)^5}{90} \cdot f^{(4)}(\tau).$$

**(c) Newtons 3/8-Regel**

$$N(f) := \frac{b-a}{8} \left( f(a) + 3f\left(a + \frac{b-a}{3}\right) + 3f\left(a + 2\frac{b-a}{3}\right) + f(b) \right)$$

Bei mindestens viermal stetig differenzierbaren Funktionen  $f$  erfüllt der Fehler der Integrationsformel eine Gleichung der Form

$$N(f) - \int_a^b f(t)dt = \frac{3(b-a)^5}{80} \cdot f^{(4)}(\tau)$$

mit einem geeigneten  $\tau \in [a, b]$ .

**(d) Milne-Regel**

$$M(f) := \frac{b-a}{90} \left( 7f(a) + 32f\left(a + \frac{b-a}{4}\right) + 12f\left(a + \frac{b-a}{2}\right) + 32f\left(a + 3\frac{b-a}{4}\right) + 7f(b) \right)$$

Für mindestens sechsmal differenzierbare Funktionen  $f$  ist der Integrationsfehler von der Form

$$M(f) - \int_a^b f(t)dt = \frac{8(b-a)^7}{945} \cdot f^{(6)}(\tau).$$

mit einem geeigneten  $\tau \in [a, b]$ .

**C – Summierte Newton-Cotes-Formeln**

Anstelle der Anwendung der Newton-Cotes-Formeln auf das ganze Intervall  $[a, b]$  ist es in der Regel sinnvoller, das Intervall in  $M$  gleich große Teilintervalle zu teilen und die Quadraturformeln auf jedes Teilintervall anzuwenden. Wir demonstrieren dies am Beispiel von 13-Knoten-Integrationsformeln ( $N = 12$  Teilintervalle)

$$\hat{I}(f) = (b-a) \sum_{i=0}^{12} \lambda_i f(t_i)$$

mit den Knoten  $t_i = a + ih$ ,  $h = (b-a)/12$ .

**(a) Summierte Trapezregel.** Über jedem der  $M = N = 12$  Teilintervalle  $[t_i, t_{i+1}]$  wenden wir die Trapezregel an und erhalten als Flächeninhalt der Teilintervalle

$$T_i = \frac{h}{2} (f(t_i) + f(t_{i+1})).$$

$(2M)^{-1}$ .	1	2	2	...	...	...	...	...	...	2	1	Trapezregel
$(6M)^{-1}$ .	1	4	2	4	2	...	...	...	...	4	1	Simpson-Regel
$(8M)^{-1}$ .	1	3	3	2	3	3	2	...	...	3	3	Newtonsche 3/8-Regel
$(90M)^{-1}$ .	7	32	12	32	14	...	...	...	32	12	32	Milne-Regel

Tabelle 4: Gewichte für summierte Newton-Cotes-Formeln

Durch Summation der Teilflächen erhalten wir die summierte Trapezregel

$$\hat{I}_T(f) = \sum_{i=0}^{11} T_i = \frac{b-a}{12} \left( \frac{1}{2} (f(a) + f(b)) + \sum_{i=1}^{11} f(t_i) \right).$$

Dies entspricht der Trapezregel des Beispiels (b) aus Abschnitt 6.1. Der Fehler für jedes Teilintervall ist gleich

$$T_i(f) - \int_{t_i}^{t_{i+1}} f(t) dt = \frac{h^3}{12} f''(\tau_i)$$

mit geeigneten  $\tau_i \in [t_i, t_{i+1}]$ . Da nach dem Mittelwertsatz gilt

$$\sum_{i=0}^{11} f''(\tau_i) = 12 \cdot f''(\tau)$$

für ein  $\tau \in [a, b]$ , kann der Gesamtfehler geschrieben werden in der Form

$$\hat{I}_T(f) - I(f) = h^2 \cdot \frac{b-a}{12} f''(\tau) = \mathcal{O}(h^2).$$

**(b) Summierte Simpson-Regel.** Wir teilen das Gesamtintervall in  $M = N/2 = 6$  Teilintervalle der Länge  $2h$  und wenden auf jedes der Teilintervalle  $[t_{2i}, t_{2i+2}]$  die Simpsonregel zu den Knoten  $t_{2i}, t_{2i+1}$  und  $t_{2i+2}$  an:

$$S_i(f) = \frac{2h}{6} (f(t_{2i}) + 4f(t_{2i+1}) + f(t_{2i+2})) = \frac{b-a}{36} (f(t_{2i}) + 4f(t_{2i+1}) + f(t_{2i+2})).$$

Durch Aufsummieren der 6 Teilflächen entsteht eine Summationsformel  $\hat{I}_S(f)$  mit den in Tabelle 4 angegebenen Gewichten. Für die Einzelfehler gilt nach Abschnitt 6.2 B

$$S_i(f) - \int_{t_{2i}}^{t_{2i+2}} f(t) dt = \frac{(2h)^5}{90} \cdot f^{(4)}(\tau_i).$$

Den Gesamtfehler erhält man mit Hilfe des Mittelwertsatzes gemäß

$$\hat{I}_S(f) - I(f) = h^4 \cdot \frac{8(b-a)}{45} f^{(4)}(\tau) = \mathcal{O}(h^4).$$

**(c) Summierte Newtonsche 3/8-Regel.** Durch Einteilung von  $[a, b]$  in  $M = N/3 = 4$  Teilintervalle und Anwendung der 3/8-Regel erhält man eine Quadraturformel  $\hat{I}_N(f)$  mit den in Tabelle 4 angegebenen Gewichten. Der Fehler ist

$$\hat{I}_N(f) - I(f) = h^4 \cdot \frac{243(b-a)}{80} f^{(4)}(\tau) = \mathcal{O}(h^4).$$

**(d) Summierte Milne-Regel.** Die summierte Milne-Regel  $\hat{I}_M(f)$  zu  $M = N/4 = 3$  Teilintervallen hat die in Tabelle 4 angegebenen Gewichte und den Fehler

$$\hat{I}_M(f) - I(f) = h^6 \cdot \frac{32768(b-a)}{945} f^{(6)}(\tau) = \mathcal{O}(h^6).$$

Die in diesem Spezialfall hergeleiteten Formeln für die Fehlerordnung in Abhängigkeit von der Schrittweite  $h$  gelten allgemein.

**Fehlerordnungen:** Für summierte Newton-Cotes-Formeln zur Integration hinreichend glatter Funktionen mit der Schrittweite  $h$  gelten die Fehlerordnungen

$$\begin{aligned} \mathcal{O}(h^2) & \text{ für die Trapezregel,} \\ \mathcal{O}(h^4) & \text{ für die Simpsonregel,} \\ \mathcal{O}(h^4) & \text{ für Newtons 3/8-Regel,} \\ \mathcal{O}(h^6) & \text{ für die Milneregel.} \end{aligned}$$

### 6.3 Extrapolationsverfahren

Extrapolationsverfahren dienen der Erhöhung der Fehlerordnung von Integrationsverfahren und beruhen auf asymptotischen Fehlerentwicklungen von Quadraturformeln. Wir demonstrieren dies am Beispiel der summierten Trapezregel und leiten hiermit die klassische Romberg-Quadratur her.

#### A – Asymptotische Fehlerentwicklungen

Wie in Abschnitt 6.2C bezeichne  $\hat{I}_T(f)$  die summierte Trapezregel für das Integral  $I(f)$  zu einer vorgegebenen Schrittweite  $h$ . Das im folgenden vorzustellende Extrapolationsverfahren beruht auf der folgenden Beobachtung, welche wir hier ohne Beweis

angeben.<sup>9</sup>

**Satz:** Ist  $f$  mindestens  $2m + 1$ -mal differenzierbar, so lässt sich  $\hat{I}_T(f)$  in Abhängigkeit von  $h$  beschreiben durch die asymptotische Entwicklung

$$\hat{I}_T(f) = \mathcal{T}(h^2),$$

wobei  $\mathcal{T}$  eine Funktion ist der Form

$$\mathcal{T}(\eta) = I(f) + \tau_1\eta + \tau_2\eta^2 + \cdots + \tau_m\eta^m + R_{m+1}(\eta)\eta^{m+1}$$

mit Konstanten  $\tau_i$  und dem beschränkten Restglied  $R_{m+1}$ .

Aus dem Satz folgt, dass  $\hat{I}_T(f)$  und  $\mathcal{T}$  für  $h = 0$  mit dem gesuchten Integralwert  $I(f)$  übereinstimmen. Die Koeffizienten  $\tau_i$  und das Restglied sind von  $f$  abhängig und in der Regel unbekannt. Die Idee der Extrapolation ist es,  $\mathcal{T}$  durch ein Polynom zu approximieren und dieses an der Stelle  $h = 0$  auszuwerten.

## B – Die Idee der Extrapolation

Die Idee der Extrapolation soll am einfachsten Fall der linearen Interpolation von  $\mathcal{T}$  demonstriert werden. Hierzu nehmen wir an, dass  $I(f)$  für die zwei Schrittweiten

$$h_0 = \frac{b-a}{n} \quad \text{und} \quad h_1 = \frac{h_0}{2} = \frac{b-a}{2n}$$

nach Auswertung der summierten Trapezregel durch die Werte  $\mathcal{T}(h_0^2)$  und  $\mathcal{T}(h_1^2)$  approximiert wird. Für kleine Schrittweiten  $h$  ist nach obigem Satz  $\mathcal{T}(h^2) \approx I(f) + \tau_1 h^2$ . Hieraus folgt

$$\begin{aligned} \mathcal{T}(h_0^2) &\approx I(f) + \tau_1 h_0^2, \\ \mathcal{T}(h_1^2) &\approx I(f) + \tau_1 h_1^2 = I(f) + \tau_1 \frac{h_0^2}{4}. \end{aligned}$$

Durch Kombination der Gleichungen kann die Unbekannte  $\tau_1$  eliminiert werden und wir erhalten

$$I(f) \approx \frac{4\mathcal{T}(h_1^2) - \mathcal{T}(h_0^2)}{3}. \quad (6.6)$$

---

<sup>9</sup>Der Beweis beruht auf der Euler-MacLaurinschen Summenformel; eine formale Begründung und Literaturhinweise für einen streng mathematischen Beweis sind zu finden in Abschnitt 8.1.3 von H. R. Schwarz, Numerische Mathematik, Teubner, 1993.

Um diese Formel zu interpretieren, betrachten wir erneut die summierte Trapezregel. Seien

$$t_i := a + i \cdot h_1, \quad i = 0, \dots, 2n$$

die äquidistanten Knoten zur Schrittweite  $h_1$ . Nach den Ergebnissen des Abschnitts 6.2 gilt

$$\begin{aligned} \mathcal{T}(h_0^2) &= 2h_1 \left( \frac{1}{2} (f(a) + f(b)) + \sum_{i=0}^n f(t_{2i}) \right), \\ \mathcal{T}(h_1^2) &= h_1 \left( \frac{1}{2} (f(a) + f(b)) + \sum_{i=0}^{2n} f(t_i) \right), \end{aligned}$$

und damit

$$\frac{4\mathcal{T}(h_1^2) - \mathcal{T}(h_0^2)}{3} = \frac{h_1}{3} [f(a) + 4f(t_1) + 2f(t_2) + \dots + 2f(t_{2n-2}) + 4f(t_{2n-1}) + f(b)].$$

Dies entspricht aber gerade der summierten Simpson-Formel zur Schrittweite  $h_1$ . Damit konnten die  $\mathcal{O}(h^2)$ -Ergebnisse der Trapezregel kombiniert werden zu einer  $\mathcal{O}(h^4)$ -Quadraturformel. Wir halten als Beobachtung fest:

Durch Kombination der Ergebnisse einer Quadraturformel der Ordnung  $\mathcal{O}(h^2)$  zu unterschiedlichen Schrittweiten  $h$  ist es möglich, Formeln höherer Ordnung herzuleiten.

## C – Das Romberg-Verfahren

Das klassische Romberg-Quadraturverfahren lässt sich durch konsequente Fortsetzung dieser Extrapolationsidee herleiten. Hierzu stellen wir fest, dass sich nach dem Satz über die asymptotische Fehlerentwicklung das Ergebnis  $\mathcal{T}(h^2)$  der summierten Trapezregel zur Schrittweite  $h$  gut durch ein Polynom  $m$ -ten Grades in  $\eta = h^2$  beschreiben lässt, falls die zu integrierende Funktion  $f$  genügend glatt ist. Beispielsweise kann die Trapezregel zu  $m+1$  verschiedenen Knoten  $h_i$  berechnet werden und als approximierendes Polynom das Interpolationspolynom  $\mathcal{P}(\eta)$  zu den Knoten  $\eta_i = h_i^2$ ,  $i = 0, \dots, m$  bestimmt werden. Da nur eine Approximation des gesuchten Integralwerts  $I(f) = \mathcal{T}(0)$  gesucht ist, ist eine

explizite Bestimmung der Koeffizienten von  $\mathcal{P}$  nicht nötig. Stattdessen kann der Wert  $\mathcal{P}(0)$  mit Hilfe des Neville-Schemas aus Abschnitt 5.1 bestimmt werden. Dies führt auf das folgende **Extrapolationstableau**.

$$\begin{array}{ccccccc}
 & & & & & & \\
 & & & & & & \\
 \mathcal{T}_{11} & & & & & & \\
 & \searrow & & & & & \\
 \mathcal{T}_{21} & \rightarrow & \mathcal{T}_{22} & & & & \\
 \vdots & & & \ddots & & & \\
 \vdots & & & & \ddots & & \\
 \vdots & & & & & \ddots & \\
 \mathcal{T}_{m-1,1} & \rightarrow & \mathcal{T}_{m-1,2} & \rightarrow & \cdots & \rightarrow & \mathcal{T}_{m-1,m-1} \\
 & \searrow & & \searrow & & \searrow & \\
 \mathcal{T}_{m1} & \rightarrow & \mathcal{T}_{m2} & \rightarrow & \cdots & \rightarrow & \mathcal{T}_{m,m-1} \rightarrow \mathcal{T}_{mm}
 \end{array}$$

Hierbei sind

$$\mathcal{T}_{i1} = \mathcal{T}(h_i^2)$$

die aus der summierten Trapezregel berechneten Werte zu den Schrittweiten  $h_i$ ; für  $j > 1$  wird  $\mathcal{T}_{ij}$  berechnet durch die Rekursionsformel

$$\mathcal{T}_{ij} = \mathcal{T}_{i,j-1} + \frac{\eta_i}{\eta_{i-j+1} - \eta_i} \cdot (\mathcal{T}_{i,j-1} - \mathcal{T}_{i-1,j-1}).$$

Das Romberg-Verfahren erhält man nun über die zwei zusätzlichen Festlegungen:

- (1) Die Trapezregel wird ausgewertet für eine Folge  $h_1, h_2, \dots$  mit  $h_i = 0.5 \cdot h_{i-1}$ .
- (2) Die Tiefe des Extrapolationstableaus wird nicht von vornherein festgelegt, sondern ergibt sich durch Auswertung eines geeigneten **Abbruchkriteriums**.

Dies führt auf den folgenden Algorithmus zur Berechnung von  $I(f)$ .

**Romberg-Verfahren:**

$N$	$ \hat{I}_T(f_1) - I(f_1) $	Romberg-Korrekturen		
2	5.2E-2			
3	2.3E-2	1.3E-4		
4	3.2E-3	8.3E-6	1.2E-7	
5	8.0E-4	5.2E-7	1.9E-9	<5E-11

Tabelle 5: Integrationsfehler für  $f_1$ 

S1: Bestimme eine hinreichend kleine Grundschriftweite  $H = (b - a)/N$  ( $N \in \mathbb{N}$ );

setze  $i := 1$  sowie  $h_1 := H$ .

S2: Bestimme nach der summierten Trapezregel zur Schrittweite  $h_i$  den Wert

$$\mathcal{T}_{i1} := \mathcal{T}(h_i^2).$$

S3: Für  $j = 2, \dots, i$  berechne

$$\mathcal{T}_{ij} := \mathcal{T}_{i,j-1} + (\mathcal{T}_{i,j-1} - \mathcal{T}_{i-1,j-1}) / (2^{2(j-1)} - 1).$$

S4: Ist das Abbruchkriterium erfüllt, so wähle  $\mathcal{T}_{ii}$  als Approximation für  $I(f)$ ; andernfalls setze  $i := i + 1$  und  $h_i := h_{i-1}/2$  und gehe zu S2.

**Beispiel:** Für die Funktionen  $f_i : [0, \pi/2] \rightarrow \mathbb{R}$  ( $i = 1, 2, 3$ ) mit  $f_1(t) = \cos(t)$ ,  $f_2(t) = \sqrt{\sin(t)} \cos(t)$  und

$$f_3(t) = \begin{cases} 2/\pi & \text{für } t < \pi/2 \\ 0 & \text{sonst} \end{cases}$$

sollen die Integrale  $\int_0^{\pi/2} f_i(t) dt$  numerisch berechnet werden. Die exakten Integralwerte sind jeweils 1. Hierfür definieren wir für  $N = 1, 2, 3, 4$  die Schrittweiten  $h = \pi \cdot 2^{-N}$  und wenden zunächst die Trapezregel an. Die Fehler  $|\hat{I}_T(f_i) - I(f_i)|$  sind in den ersten Spalten der Tabellen 5 bis 7 zu finden. Korrigieren wir diese Ergebnisse nach durch die Romberg-Extrapolation, so erhalten wir die Ergebnisse in den folgenden Spalten. Die qualitativen Unterschiede der Fehler liegen an der unterschiedlichen Glattheit der Integranden.

$N$	$ \hat{I}_T(f_2) - I(f_2) $	Romberg-Korrekturen		
2	3.0E-1			
3	9.6E-2	2.9E-2		
4	3.2E-2	1.1E-2	9.3E-3	
5	1.1E-2	3.7E-3	3.3E-3	6.2E-3

Tabelle 6: Integrationsfehler für  $f_2$

$N$	$ \hat{I}_T(f_3) - I(f_3) $	Romberg-Korrekturen		
2	5.0E-1			
3	2.5E-1	1.7E-1		
4	1.3E-1	8.3E-2	7.8E-2	
5	6.3E-2	4.2E-2	3.9E-2	3.8E-2

Tabelle 7: Integrationsfehler für  $f_3$

## 7 Numerik gewöhnlicher Differentialgleichungen

### 7.1 Anfangswertprobleme

**Definition:** (a) Gegeben sei eine Funktion  $f : [0, T] \times \mathbb{R} \rightarrow \mathbb{R}$ . Eine Gleichung der Form

$$x'(t) = f(t, x(t))$$

heißt *gewöhnliche Differentialgleichung* (kurz: *DGL*) auf dem Intervall  $[0, T]$ .

(b) Eine differenzierbare Funktion  $x(t)$  auf  $[0, T]$  heißt *Lösung der DGL*, falls für alle  $t \in [0, T]$  gilt:  $x'(t) = f(t, x(t))$ .

(c) Eine DGL in Verbindung mit einer Bedingung der Form  $x(0) = c$  für ein festes  $c \in \mathbb{R}$  heißt *Anfangswertproblem (AWP)*.

Aus der Mathematikvorlesung ist bekannt: Erfüllt  $f(., .)$  eine *Lipschitzbedingung* der Form

$$|f(t, x) - f(t, y)| \leq L \cdot |x - y| \quad \text{für ein } L > 0 \quad \text{und für alle } t \in [0, T], x \in \mathbb{R}$$

so hat jedes AWP eine eindeutige Lösung  $x(t) =: \Phi^{t,0}c$ .

**Definition:** Die Abbildung  $R : [0, T] \times \mathbb{R} \rightarrow \mathbb{R}^2$ ,

$$R(t, x) := \frac{1}{\sqrt{1 + f^2(t, x)}} \begin{pmatrix} 1 \\ f(t, x) \end{pmatrix}$$

heißt *Richtungsfeld* der Differentialgleichung  $x' = f(t, x)$ .

Durch das Richtungsfeld können wir einen ersten Eindruck über den Verlauf der Lösung eines AWP's gewinnen, indem wir graphisch eine Linie konstruieren, welche dem Verlauf des Richtungsfelds folgt.

**Beispiel:** (a)  $f(t, x) = x$ . Das Richtungsfeld hängt nicht von  $t$  ab.

(b)  $f(t, x) = t + x$ . Das Richtungsfeld ist konstant entlang der Linien  $x = \alpha - t$ .

**Idee zur numerischen Lösung:** Gegeben sei das AWP  $x' = f(t, x)$ ,  $x(0) = c$  mit der

exakten Lösung  $x(t)$ . Wähle einen Zeitschritt  $\Delta t = T/N$  und definiere die diskreten Zeitschritte  $t_i := i \cdot \Delta t$ ,  $i = 0 \dots N$ . Zu diesen Zeitschritten definiere rekursiv die Approximationen  $\eta_i$  von  $x(t_i)$  durch

- (i)  $\eta_0 = c$
- (ii)  $\eta_{i+1} = \eta_i + \Delta t \cdot \Psi$  mit einer geeigneten Steigung  $\Psi = \Psi(t_i, \eta_i, \Delta t)$ .

Rekursive Vorschriften dieser Art bezeichnet man als **Einschrittverfahren** (ESV).

**Beispiel:** Das **Eulersche Polygonzugverfahren (EPZV)** erhält man durch Setzen von

$$\Psi(t_i, \eta_i, \Delta t) := f(t_i, \eta_i)$$

## 7.2 Konsistenz und Konvergenz von ESV

Gegeben seien ein AWP und ein zugehöriges ESV wie oben beschrieben. Gegeben sei  $\eta_i$ . Dann wird  $\eta_{i+1}$  berechnet durch

$$\eta_{i+1} = \eta_i + \Delta t \cdot \Psi$$

**Definition:**  $z(t)$  sei die (eindeutige) Lösung der Differentialgleichung  $x' = f(t, x)$ , welche die Zusatzbedingung  $z(t_i) = \eta_i$  erfüllt. Der Ausdruck

$$\epsilon(t_i, \eta_i, \Delta t) := \frac{1}{\Delta t} (z(t_{i+1}) - \eta_{i+1})$$

heißt **lokaler Diskretisierungsfehler** des ESV. Das Verfahren hat die (**Konsistenz-**) **Ordnung**  $p$  ( $p \in \mathbb{N}$ ), falls

$$\epsilon(t_i, \eta_i, \Delta t) = \mathcal{O}((\Delta t)^p)$$

Ist die Ordnung  $\geq 1$ , so heißt das Verfahren **konsistent**.

Zur Bestimmung der Ordnung ist die Technik der Taylorreihenentwicklung hilfreich. Entwickeln wir zunächst  $z(t)$  um den Entwicklungspunkt  $t_i$ . Es ist

$$z(t_{i+1}) = z(t_i + \Delta t) = z(t_i) + \Delta t \cdot z'(t_i) + \frac{(\Delta t)^2}{2} z''(t_i) + \text{Terme höherer Ordnung}$$

Wegen

$$\begin{aligned} z(t_i) &= \eta_i \\ z'(t_i) &= f(t_i, z_i) \\ z''(t_i) &= \frac{d}{dt} f(t, z(t))|_{t=t_i} = \partial_t f(t_i, \eta_i) + \partial_x f(t_i, \eta_i) \cdot f(t_i, \eta_i) \end{aligned}$$

folgt

$$z(t_{i+1}) = \eta_i + \Delta t \cdot f(t_i, \eta_i) + \frac{(\Delta t)^2}{2} [\partial_t f(t_i, \eta_i) + \partial_x f(t_i, \eta_i) \cdot f(t_i, \eta_i)] + \mathcal{O}((\Delta t)^3)$$

**Beispiele:** (a) Für das EPZV folgt mit  $\eta_{i+1} = \eta_i + \Delta t \cdot f(t_i, \eta_i)$

$$\epsilon(t_i, \eta_i, \Delta t) = \frac{\Delta t}{2} [\partial_t f(t_i, \eta_i) + \partial_x f(t_i, \eta_i) \cdot f(t_i, \eta_i)] + \mathcal{O}((\Delta t)^2) = \mathcal{O}(\Delta t)$$

Damit hat das Verfahren die Ordnung 1.

(b) Das **Heun-Verfahren** berechnet zunächst mit Hilfe des EPZV die Hilfsgröße

$$\overline{\eta}_{i+1} = \eta_i + \Delta t \cdot f(t_i, \eta_i)$$

und dann  $\Psi$  als Mittelwert von  $f(t_i, \eta_i)$  und  $f(t_{i+1}, \overline{\eta}_{i+1})$ :

$$\eta_{i+1} = \eta_i + \frac{\Delta t}{2} \cdot [f(t_i, \eta_i) + f(t_{i+1}, \eta_i + \Delta t \cdot f(t_i, \eta_i))]$$

Zur Bestimmung der Ordnung entwickeln wir  $f(t_{i+1}, \eta_i + \Delta t \cdot f(t_i, \eta_i))$  um den Entwicklungspunkt  $(t_i, \eta_i)$ :

$$f(t_{i+1}, \eta_i + \Delta t \cdot f(t_i, \eta_i)) = f(t_i, \eta_i) + \Delta t \cdot \partial_t f(t_i, \eta_i) + \Delta t \cdot f(t_i, \eta_i) \cdot \partial_x f(t_i, \eta_i) + \mathcal{O}((\Delta t)^2)$$

Durch Einsetzen erhalten wir, dass das Verfahren die Ordnung 2 hat.

(c) Das **modifizierte Euler-Verfahren**: Hier ist

$$\Psi = f\left(t_i + \frac{\Delta t}{2}, \eta_i + \frac{\Delta t}{2} f(t_i, \eta_i)\right)$$

Auch dieses Verfahren hat die Ordnung 2.

Aus der Konsistenz eines Verfahrens folgt die Konvergenz. Wir wählen einen beliebigen

Punkt  $\bar{t} \in (0, T]$  und definieren die Schrittweite  $\Delta t = \Delta t_{\bar{t}, N} := \bar{t}/N$ .

**Satz:** Es sei  $x(t)$  die exakte Lösung des AWP  $x' = f(t, x)$ ,  $x(0) = c$ .  $\eta_i$  sei die Lösung des ESV zur Schrittweite  $\Delta t_{\bar{t}, N}$ , d.h.

$$\eta_{i+1} = \eta_i + \Delta t_{\bar{t}, N} \cdot \Psi$$

Hat das ESV die Ordnung  $p$  und ist

$$|\Psi(t, x, \Delta t) - \Psi(t, \hat{x}, \Delta t)| \leq M \cdot |x - \hat{x}| \quad \text{für alle } t, x, \hat{x}, \Delta t$$

so gibt es eine Konstante  $C > 0$  mit

$$|\eta_N - x(\bar{t})| \leq C \cdot (\Delta t)^p \cdot \left( \frac{\exp(\bar{t}M) - 1}{M} \right)$$

### 7.3 Runge-Kutta-Verfahren

**Beispiel:** Für ein ESV werde der Ansatz gewählt

$$\eta_{i+1} = \eta_i + \Delta t \cdot \Psi(t_i, \eta_i, \Delta t)$$

wobei  $\Psi$  definiert ist durch

$$\begin{aligned} k_1 &:= f(t_i, \eta_i) \\ k_2 &:= f(t_i + c \cdot \Delta t, \eta_i + a \cdot \Delta t \cdot k_1) \\ \Psi(t_i, \eta_i, \Delta t) &:= b_1 k_1 + b_2 k_2 \end{aligned}$$

Ausführlich ergibt sich also für  $\Psi$  die Darstellung

$$\Psi(t_i, \eta_i, \Delta t) = b_1 f(t_i, \eta_i) + b_2 f(t_i + c \cdot \Delta t, \eta_i + a \cdot \Delta t \cdot f(t_i, \eta_i))$$

Die Koeffizienten  $a$ ,  $b_1$ ,  $b_2$  und  $c$  sollen dabei so gewählt werden, dass ein Verfahren möglichst hoher Ordnung entsteht. Durch Taylorreihenentwicklung erhalten wir

$$\Psi(t_i, \eta_i, \Delta t) = (b_1 + b_2) f(t_i, \eta_i) + \Delta t \cdot b_2 \cdot [c \partial_t f(t_i, \eta_i) + a \partial_x f(t_i, \eta_i) \cdot f(t_i, \eta_i)] + \mathcal{O}(\Delta t^2)$$

Wir erkennen, dass das Verfahren

- (i) (mindestens) Ordnung 1 hat, wenn

$$b_1 + b_2 = 1$$

- (ii) (mindestens) die Ordnung 2 hat, wenn zusätzlich gilt

$$b_2a = b_2c = \frac{1}{2}$$

Das Beispiel repräsentiert ein sog. *zweistufiges Runge-Kutta-Verfahren* und kann wie folgt verallgemeinert werden.

**Definition:** Ein *s-stufiges Runge-Kutta-Verfahren (RKV)* ist definiert durch

$$\Psi(t_i, \eta_i, \Delta t) = \sum_{r=0}^s b_r k_r$$

mit

$$\begin{aligned} k_1 &= f(t_i, \eta_i) \\ k_2 &= f(t_i + c_2 \Delta t, \eta_i + \Delta t \cdot a_{21} k_1) \\ k_3 &= f(t_i + c_3 \Delta t, \eta_i + \Delta t \cdot [a_{31} k_1 + a_{32} k_2]) \\ &\vdots \\ k_s &= f(t_i + c_s \Delta t, \eta_i + \Delta t \cdot [a_{s1} k_1 + \cdots + a_{s,s-1} k_{s-1}]) \end{aligned}$$

Ein RKV wird eindeutig beschrieben durch zwei Spaltenvektoren  $\mathbf{b}$ ,  $\mathbf{c}$  und eine Matrix  $\mathcal{A}$  der Form

$$\mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_s \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} 0 \\ c_2 \\ \vdots \\ c_s \end{pmatrix}, \quad \mathcal{A} = \begin{pmatrix} 0 & \cdots & \cdots & 0 \\ a_{21} & 0 & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ a_{s1} & \cdots & a_{s,s-1} & 0 \end{pmatrix}$$

Ähnliche Rechnungen wie im obigen Beispiel ergeben

**Satz:** Ein  $s$ -stufiges RKV, welches die **Bedingung**

$$\sum_{q=1}^s a_{rq} = c_r \quad \text{für } r = 1, \dots, s$$

erfüllt, hat (mindestens) die Ordnung

- 1, wenn  $\sum_{r=1}^s b_r = 1$
- 2, wenn zusätzlich  $\sum_{r=1}^s b_r c_r = 1/2$
- 3, wenn zusätzlich  $\sum_{r=1}^s b_r c_r^2 = 1/3$  und  $\sum_{q,r=1}^s b_q a_{qr} c_r^2 = 1/6$
- ⋮

**Beispiele:** (a) Das dreistufige RKV

$$\mathbf{b} = \frac{1}{3} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} 0 \\ 1/2 \\ 1 \end{pmatrix}, \quad \mathcal{A} = \begin{pmatrix} 0 & 0 & 0 \\ 1/2 & 0 & 0 \\ 1/3 & 2/3 & 0 \end{pmatrix}$$

lautet ausführlich geschrieben

$$\begin{aligned} k_1 &= f(t_i, \eta_i) \\ k_2 &= f(t_i + \Delta t/2, \eta_i + \Delta t/2 \cdot k_1) \\ k_3 &= f(t_i + \Delta t, \eta_i + \Delta t/3 \cdot [k_1 + 2k_2]) \\ \Psi(t_i, \eta_i, \Delta t) &= \frac{1}{3} \cdot (k_1 + k_2 + k_3) \end{aligned}$$

Man überzeugt sich leicht, dass die Bedingung des Satzes erfüllt ist, und rechnet ebenso leicht die Ordnung 2 nach.

(b) Das **klassische RKV** ist vierstufig und gegeben durch

$$\mathbf{b} = \frac{1}{6} \cdot \begin{pmatrix} 1 \\ 2 \\ 2 \\ 1 \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} 0 \\ 1/2 \\ 1/2 \\ 1 \end{pmatrix}, \quad \mathcal{A} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Die Ordnung ist 4. Es ist ein in der Praxis häufig benutztes Verfahren, da es einen guten Kompromiss zwischen Ordnung und Rechenaufwand (vier Funktionsauswertungen von  $f(.,.)$  pro Zeitschritt) bietet.

## 7.4 Die Stabilität von ESV

Im Folgenden wollen wir untersuchen, wie sich die **Wahl des Zeitschritts**  $\Delta t$  auf die Anwendung eines ESV's auf das **Testproblem**

$$x' = \lambda x, \quad x(0) = 1$$

für  $\lambda < 0$  auswirkt. Die exakte Lösung des Testproblems lautet  $x(t) = \exp(\lambda t)$ . Wegen  $\lambda < 0$  gilt  $\lim_{t \rightarrow \infty} x(t) = 0$ .

**Beispiel:** In einem numerischen Experiment wählen wir  $\lambda = -80$  und wenden das EPZV zunächst mit der Schrittweite  $\Delta t = 0.01$  an. Als Approximationen für  $x(t)$  für  $t = 0.9, 1.0, 1.1$  berechnen wir  $\eta_{90} = 1.238 \cdot 10^{-63}$ ,  $\eta_{100} = 1.268 \cdot 10^{-70}$ ,  $\eta_{110} = 1.298 \cdot 10^{-77}$ . Die numerische Lösung zeigt das erwünschte qualitative Verhalten  $\lim_{i \rightarrow \infty} \eta_i = 0$ .

Wählen wir dagegen  $\Delta t = 0.1$ , so erhalten wir für die selben Zeitpunkte  $\eta_9 = -4.035 \cdot 10^7$ ,  $\eta_{10} = 2.825 \cdot 10^8$ ,  $\eta_{11} = -1.977 \cdot 10^9$ . Wir sehen ein alternierendes Verhalten mit stark ansteigenden Beträgen – ein ziemlich eindeutiges Zeichen für eine numerische Instabilität. Dieses Verhalten wollen wir genauer untersuchen.

In allen bisher besprochenen ESV führt die Anwendung auf das Testproblem auf eine Rekursionsvorschrift der Form

$$\eta_{i+1} = F(\lambda \Delta t) \cdot \eta_i$$

woraus wir leicht mit vollständiger Induktion herleiten können

$$\eta_i = F(\lambda \Delta t)^i \cdot \eta_0$$

Das erwünschte qualitative Abklingverhalten

$$\eta_i \rightarrow 0 \quad \text{für} \quad i \rightarrow \infty$$

erhalten wir offenbar genau dann, wenn

$$|F(\lambda \Delta t)| < 1$$

**Definition:** Eine Schrittweite  $\Delta t$  heißt **dem Testproblem angemessen**, wenn  $|F(\lambda\Delta t)| < 1$  gilt.

**Beispiele:** (a) Beim EPZV ist

$$\eta_{i+1} = (1 + \lambda\Delta t) \cdot \eta_i$$

also  $F(\lambda\Delta t) = 1 + \lambda\Delta t$ .  $\Delta t$  ist damit genau dann angemessen, wenn  $\Delta t < 2/|\lambda|$  ist. In obigem Beispiel  $\lambda = -80$  entspricht das dem Schwellwert 0.025.

(b) Beim Heun-Verfahren ist

$$F(\lambda\Delta t) = 1 + \lambda\Delta t + 0.5 \cdot (\lambda\Delta t)^2 = \frac{1}{2} [1 + (1 + \lambda\Delta t)^2]$$

Auch hier ist die Schrittweitenbeschränkung gegeben durch  $|\lambda|\Delta t < 2$ .

(c) Beim klassischen RKV ist

$$F(\lambda\Delta t) = 1 + \lambda\Delta t + \frac{1}{2}\lambda\Delta t^2 + \frac{1}{6}\lambda\Delta t^3 + \frac{1}{24}\lambda\Delta t^4$$

Die Schrittweitenbeschränkung lautet  $|\lambda|\Delta t < 2.7854$ .

**Bemerkung:** Bei allen bisher besprochenen (expliziten) ESV ist die Funktion  $F$  ein nicht-konstantes Polynom. Für solche Polynome  $F$  gilt

$$|F(z)| \rightarrow \infty \quad \text{für} \quad |z| \rightarrow \infty.$$

Bei diesen Verfahren wird es daher *immer* eine Schrittweitenbeschränkung geben.

Um die Schrittweitenbeschränkung zu umgehen, führt man **implizite** ESV ein. Das sind Verfahren, bei denen  $\Psi$  auch von  $\eta_{i+1}$  abhängt. Wir beschränken uns hier auf ein einfaches Beispiel.

**Beispiel:** Das **implizite Euler-Verfahren** ist gegeben durch

$$\eta_{i+1} = \eta_i + \Delta t f(t_{i+1}, \eta_{i+1}) \quad (*)$$

Durch Anwendung auf das Testproblem errechnet man leicht

$$F(\lambda\Delta t) = \frac{1}{1 - \lambda\Delta t}$$

Wie man sieht, ist die Bedingung  $|F(\lambda\Delta t)| < 1$  für  $\lambda < 0$  und beliebige  $\Delta t > 0$  erfüllt. Allerdings ist es im Allgemeinen schwierig, die Gleichung (\*) nach  $\eta_{i+1}$  aufzulösen.